Food Waste Redistribution Platform

Overview

This platform aims to reduce food wastage by connecting restaurants, event organizers, and households who have surplus edible food with nearby NGOs, shelters, or individuals in need. The system will allow donors to post food availability details, and receivers to request or claim donations based on time, freshness, and location proximity. It promotes sustainability and community welfare by redistributing excess food efficiently.

Requirement 1: Donors can post surplus food details for redistribution

Features:

- The system should allow registered donors to upload details such as food type, quantity, expiry time, and pickup location.
- The system should support image uploads of food items to ensure transparency and trust.
- The system should automatically mark expired or unclaimed food as unavailable after a set time.
- The system should allow donors to edit or remove a listing before it's claimed.

Requirement 2: Receivers can browse and claim available food donations

Features:

- The system should display available food donations sorted by distance or freshness.
- The system should allow receivers to filter food based on categories (e.g., cooked, uncooked, packaged).
- The system should allow receivers to claim a food item and notify the donor automatically.
- The system should prevent multiple claims on the same donation once it's reserved.

Requirement 3: The system facilitates real-time communication between donors and receivers

Features:

- The system should include a simple in-app chat or message feature for coordination after a claim.
- The system should send instant notifications to both parties when a donation is claimed or updated.
- The system should store message history for each donation for future reference.
- The system should allow users to report inappropriate behavior or misuse directly from the chat.

Requirement 4: The platform tracks and displays community impact statistics

Features:

- The system should maintain data on total food donated, received, and waste reduced.
- The system should visualize impact through charts (e.g., total meals saved per month).
- The system should display a donor's personal contribution stats on their profile.
- The system should reward active users with digital badges or recognition titles.

Requirement 5: The platform supports coordination and pickup management

Features:

- The system should allow donors to set preferred pickup time slots for each donation.
- The system should send pickup reminders to both donor and receiver before the scheduled time.

- The system should allow receivers to mark a donation as “collected” once the food has been received.

- The system should notify admins automatically if a claimed donation remains uncollected after the pickup window.

Login and Registration Module

Features:

- The system should allow new users to register as either a donor or receiver.

- The system should validate credentials before account creation (unique email, password strength).

- The system should allow registered users to log in securely using encrypted authentication.

- The system should provide a password reset option through email verification.

- The system should maintain session-based login for persistent access until logout.

Non-Functional Requirements

1. **Performance:** The system should load any main page (home, food list, or profile) within 3 seconds under normal network conditions.

2. **Usability:** New users should be able to understand the basic workflow (posting or claiming food) within 10 minutes of use.

3. **Security:** All user credentials and sensitive data must be stored using encryption (e.g., bcrypt for passwords).

4. **Reliability:** The system should recover from server crashes within 60 seconds using automatic restarts.

5. **Scalability:** The system should support up to 10,000 concurrent users without significant performance degradation.

6. **Availability:** The service should be available 99% of the time, excluding maintenance windows.

7. **Maintainability:** The codebase should follow modular structure and naming conventions for ease of updates.

8. **Portability:** The platform should be deployable on both cloud-based and local Node.js environments.

9. **Backup & Recovery:** Database backups should occur daily and be retained for at least 30 days.

10. **Accessibility:** The website should be usable on both desktop and mobile devices with responsive UI design.