ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue V, May 2026

Annapurna: An Intelligent Food Donation System with AI-Based Receiver Recommendation

Prof. Vrushali Dhanokar, Akash Midgule, Rahul Rathod, Yuvraj Wagh, Gaurav Deshmukh

Parvatibai Genba Sopanrao moze college of engineering Wagholi Pune 412207

DOI: https://doi.org/10.51583/IJLTEMAS.2026.150500008

Received: 30 April 2026; Accepted: 05 May 2026; Published: 22 May 2026

ABSTRACT

Food wastage and hunger imbalance is a major societal issue that requires efficient technological solutions. This paper presents "Annapurna", a food donation system that connects donors, receivers, and volunteers through a web and mobile platform using React.js, Android Studio, Node.js, Express.js, and MongoDB.

The system allows donors to create donation requests and receivers to request food, while volunteers manage the matching process. An AI-based recommendation module suggests suitable receivers based on location, urgency, and quantity, improving matching efficiency.

The proposed system enhances coordination, reduces food wastage, and improves resource distribution compared to traditional manual approaches.

Keywords: Food Donation System, MERN Stack, Android Application, AI-Based Recommendation, Resource Matching, Volunteer Management, REST API, MongoDB, Node.js, React.js.

INTRODUCTION

The rapid advancement of information and communication technologies has enabled the development of intelligent systems that address real-world societal challenges. One of the most critical issues faced today is the imbalance between food wastage and hunger. Large quantities of food are wasted daily from households, restaurants, and events, while a significant portion of the population struggles to access basic nutrition. This gap highlights the need for an efficient and scalable solution to manage surplus food distribution.

Existing food donation practices are largely manual and unorganized, relying on direct communication or intermediary organizations. These approaches often lead to delays, lack of transparency, and inefficient utilization of available resources. There is a need for a centralized platform that can connect donors, receivers, and volunteers in real time.

To address this problem, this paper proposes "Annapurna", an intelligent food donation and distribution system that integrates web and mobile technologies. The system is developed using React.js for the web interface and Android Studio for the mobile application, with a backend powered by Node.js, Express.js, and MongoDB. It enables donors to create donation requests and receivers to request food, while volunteers manage and coordinate the distribution process.

In addition, the system incorporates an AI-based recommendation module that suggests suitable receivers for a given donation based on factors such as location, urgency, and quantity. This hybrid approach combines automated decision-making with human intervention, improving efficiency and reducing response time. The proposed system aims to minimize food wastage, enhance coordination, and ensure timely delivery of food to those in need.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue V, May 2026

Tools and Technologies Used

Android Studio

Android Studio is the official Integrated Development Environment (IDE) for developing Android applications. It provides tools for designing user interfaces, managing application logic, and testing mobile applications. In the proposed system, Android Studio is used to develop the mobile application interface for users to interact with the food donation platform.

React.js

React.js is a JavaScript library used for building dynamic and responsive user interfaces for web applications. It enables efficient rendering through component-based architecture. In this system, React.js is used to develop the web frontend, allowing users to access the platform through browsers.

Node.js

Node.js is a server-side JavaScript runtime environment that enables the development of scalable and high-performance backend services. It is used to handle server-side operations, process requests, and manage communication between frontend and database.

Express.js

Express.js is a lightweight web application framework for Node.js. It simplifies the creation of RESTful APIs and handles routing, middleware, and server logic. In this system, Express.js is used to build APIs for user authentication, request handling, and data processing.

MongoDB

MongoDB is a NoSQL database used to store application data in a flexible, document-oriented format. It is used to manage user data, donor requests, and receiver requests efficiently, ensuring scalability and fast data retrieval.

REST API

REST APIs are used to enable communication between the frontend (web and mobile) and the backend server. They allow different modules of the system to interact seamlessly and support real-time data exchange.

AI-Based Recommendation Module

The system incorporates a lightweight AI-based recommendation mechanism to assist in matching donor requests with suitable receiver requests. This module analyzes parameters such as location, urgency, and quantity to suggest optimal matches, improving system efficiency.

METHODOLOGY

The proposed system follows a structured and step-by-step approach to efficiently manage food donation and distribution among donors, receivers, and volunteers. The methodology is based on a role-based interaction model integrated with REST APIs and an AI-based recommendation mechanism.

Requirement Analysis

The development process began with understanding the real-world problem of food wastage and inefficient distribution. The system requirements were identified by analyzing how donors, receivers, and volunteers interact in existing manual systems. Key requirements included role-based access, real-time request handling, and efficient matching of donations.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue V, May 2026

System Design and Planning

Based on the identified requirements, the system architecture was designed. This included defining modules such as User, Donor Request, and Receiver Request, along with planning database structures in MongoDB. The overall workflow of the system, including request creation and matching, was also defined.

User Interaction and Request Creation

Users register on the platform by selecting their role as donor, receiver, or volunteer. Donors create donation requests by providing details such as food type, quantity, and location. Similarly, receivers create requests specifying their requirements and urgency level.

Backend Development and API Integration

The backend of the system is developed using Node.js and Express.js. REST APIs are used to handle communication between frontend and database. All user data and request information are stored in MongoDB, enabling efficient data management and retrieval.

Matching Process (Manual and AI-Based)

Volunteers can view all active donation and receiver requests through the system interface. The system supports two types of matching: manual matching by volunteers and automated matching using an AI-based recommendation module.

AI-Based Recommendation Mechanism

The AI module uses a rule-based scoring algorithm to recommend suitable receivers for each donation request. Each receiver request is evaluated based on multiple parameters including food type compatibility, quantity requirement, geographical distance, and urgency level.

A weighted scoring mechanism is used to rank receiver requests:

Each receiver is assigned a score out of 100 and top-ranked requests are recommended.

$$Score = W_1 \times FoodMatch + W_2 \times QuantityMatch + W_3 \times DistanceScore + W_4 \times UrgencyScore$$

Where:

- FoodMatch = 1 if food types match, else 0
- QuantityMatch = 1 if donation quantity satisfies receiver requirement
- DistanceScore is calculated using the Haversine formula
- UrgencyScore is based on waiting time of the request

Weights are assigned as follows:

- Food Match = 40
- Quantity Match = 20
- Distance (< 3 km) = 20
- Urgency (> 3 hours) = 20

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue V, May 2026

Distance Formula (Haversine Formula):

$d = R \times 2 \times \text{atan2}(\sqrt{a} , \sqrt{(1 - a)})$

Where:

$a = \sin^2(\Delta\varphi / 2) + \cos(\varphi_1) \times \cos(\varphi_2) \times \sin^2(\Delta\lambda / 2)$

This formula is used to calculate the geographical distance between donor and receiver locations.

Priority is assigned based on request waiting time:

- HIGH: waiting time > 5 hours
- MEDIUM: 2–5 hours
- LOW: < 2 hours

System Deployment and Accessibility

The application is deployed using cloud platforms to ensure accessibility and scalability. The backend services are hosted on Render, and the source code for both frontend and backend is maintained using GitHub for version control. The web application and Android application are designed to work seamlessly across different devices and platforms.

Testing and Validation

The system was tested to ensure correct functionality of user registration, request creation, matching processes, and API communication. Performance testing confirmed that the system provides reliable and efficient food distribution management.

This methodology ensures efficient coordination, reduced response time, and optimal utilization of available food resources.

LITERATURE REVIEW

Existing Systems

Several food donation platforms have been developed to address the issue of food wastage and hunger. Applications such as **Aahar** and **Khanaa** provide platforms where donors can share surplus food with NGOs and volunteers. These systems enable users to donate food through mobile applications and connect with nearby receivers.

Similarly, **Foodnate** is a location-based food donation platform that connects donors with nearby organizations, helping to reduce food wastage through efficient distribution. On a global level, applications like ShareTheMeal allow users to contribute meals digitally to people in need, focusing on large-scale hunger reduction.

In addition, organizations such as No Food Waste operate using volunteer-based systems where surplus food is collected and distributed through coordinated efforts. Some academic projects, such as Seva and Helping Hands, also propose web-based food donation systems using technologies like React, Node.js, and MongoDB.

Most existing systems rely heavily on manual coordination and do not provide intelligent automated matching mechanisms.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue V, May 2026

Limitations of Existing Systems

Despite the availability of these systems, several limitations exist. Many applications rely heavily on manual coordination and lack intelligent matching mechanisms. Some platforms are limited to specific regions or organizations, reducing scalability. Additionally, most systems do not provide AI-based recommendations for efficient matching of donor and receiver requests, leading to delays and suboptimal resource utilization.

Proposed Approach

The proposed system, "Annapurna", is an intelligent food donation and distribution platform designed to efficiently connect donors, receivers, and volunteers through a centralized digital solution. The system is developed using a combination of web and mobile technologies, ensuring accessibility across different devices.

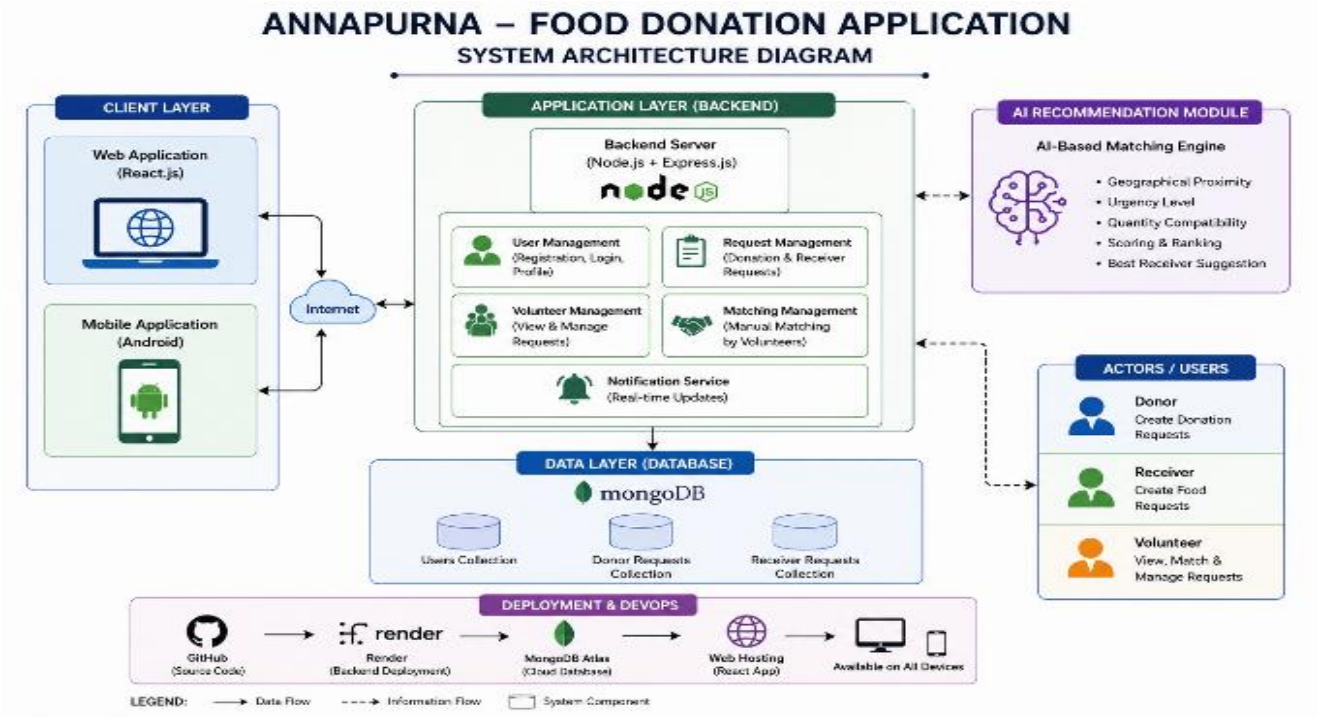
The platform follows a role-based architecture where users can register as donors, receivers, or volunteers. Donors can create food donation requests by providing details such as food type, quantity, and location. Receivers can submit food requests based on their needs and urgency. All requests are stored in a MongoDB database and managed through REST APIs, enabling real-time communication between system components.

Volunteers play a crucial role in managing and coordinating the distribution process. They can view all active donation and receiver requests and perform matching operations. The system allows volunteers to manually match donor requests with suitable receiver requests, and also supports modification of matches created by other volunteers, ensuring flexibility and better coordination.

To enhance efficiency, the system incorporates an AI-based recommendation module that suggests the most suitable receiver for a given donation. The recommendation is based on parameters such as geographical proximity, urgency level, and quantity compatibility. A scoring mechanism is applied to evaluate and rank potential matches, assisting volunteers in making faster and more accurate decisions.

The integration of REST APIs ensures seamless communication between frontend applications (web and Android) and the backend server. The system is deployed on cloud platforms, making it scalable, reliable, and accessible from anywhere. Overall, the proposed system reduces food wastage, improves resource allocation, and ensures timely delivery of food to those in need.

System Architecture



ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue V, May 2026

The Annapurna Food Donation Application follows a layered architecture that integrates web and mobile interfaces with a centralized backend system and database. The architecture is designed to ensure scalability, real-time communication, and efficient resource management.

Client Layer

The client layer consists of two user interfaces: a web application developed using React.js and a mobile application built using Android Studio. These interfaces allow users (donors, receivers, and volunteers) to interact with the system by performing actions such as registration, login, creating requests, and managing donations.

Communication Layer

The client applications communicate with the backend server through REST APIs over the internet. This layer ensures seamless data exchange between frontend and backend systems.

Application Layer (Backend)

The backend is developed using Node.js and Express.js, which handle all server-side operations. This layer is responsible for user authentication, request processing, role-based access control, and coordination between different system modules.

The backend includes the following components:

- **User Management:** Handles registration, login, and user profiles.
- **Request Management**: Manages donor and receiver requests.
- **Volunteer Management:** Allows volunteers to view and manage requests.
- **Matching Management**: Supports manual matching of donor and receiver requests.
- **AI Recommendation Module:** Suggests suitable receivers based on location, urgency, and quantity.

Data Layer (Database)

The system uses MongoDB as the database to store application data. It includes collections such as:

- Users Collection
- Donor Requests Collection
- Receiver Requests Collection

This layer ensures efficient storage, retrieval, and updating of data.

Deployment Layer

The system is deployed using cloud-based services. The backend server is hosted on Render, and the source code is managed using GitHub. The application is accessible across multiple devices, ensuring availability and scalability.

The overall architecture enables efficient coordination between users, reduces response time, and supports intelligent decision-making through the AI-based recommendation module.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue V, May 2026

Implementation

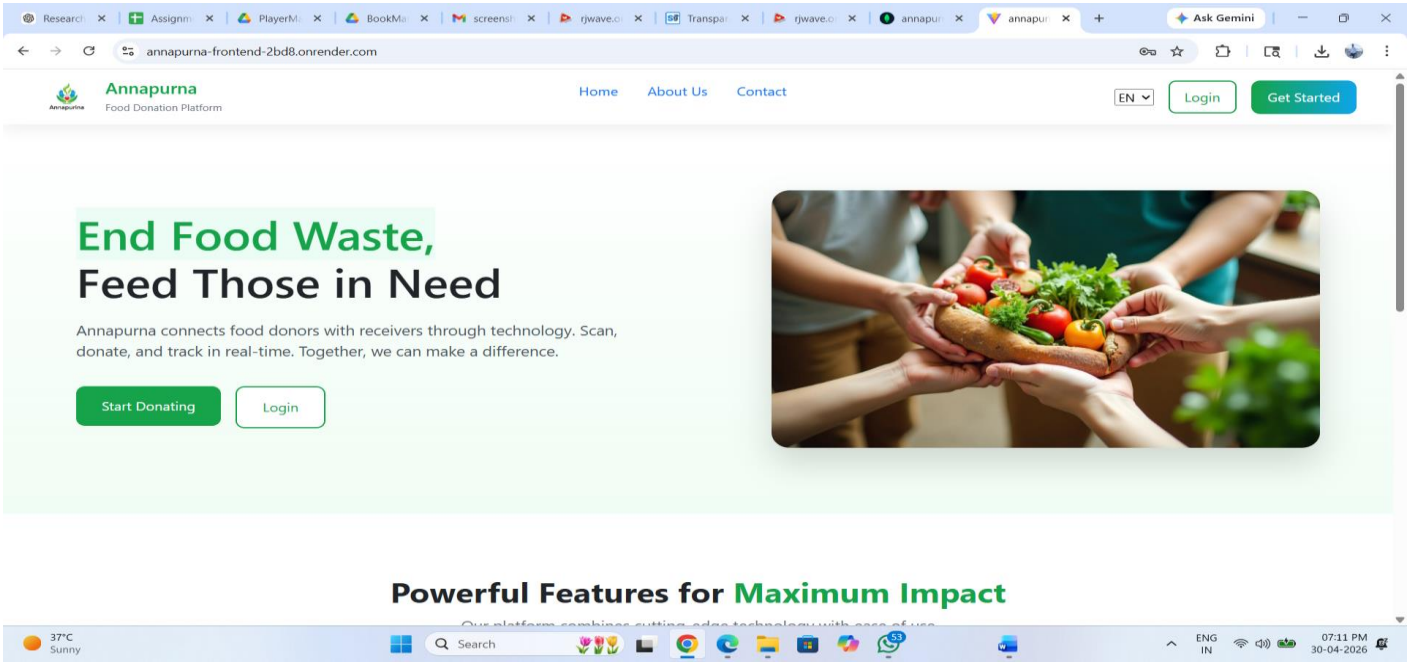


Fig. 1: web Home page

Fig. 2: Donor dashboard

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue V, May 2026

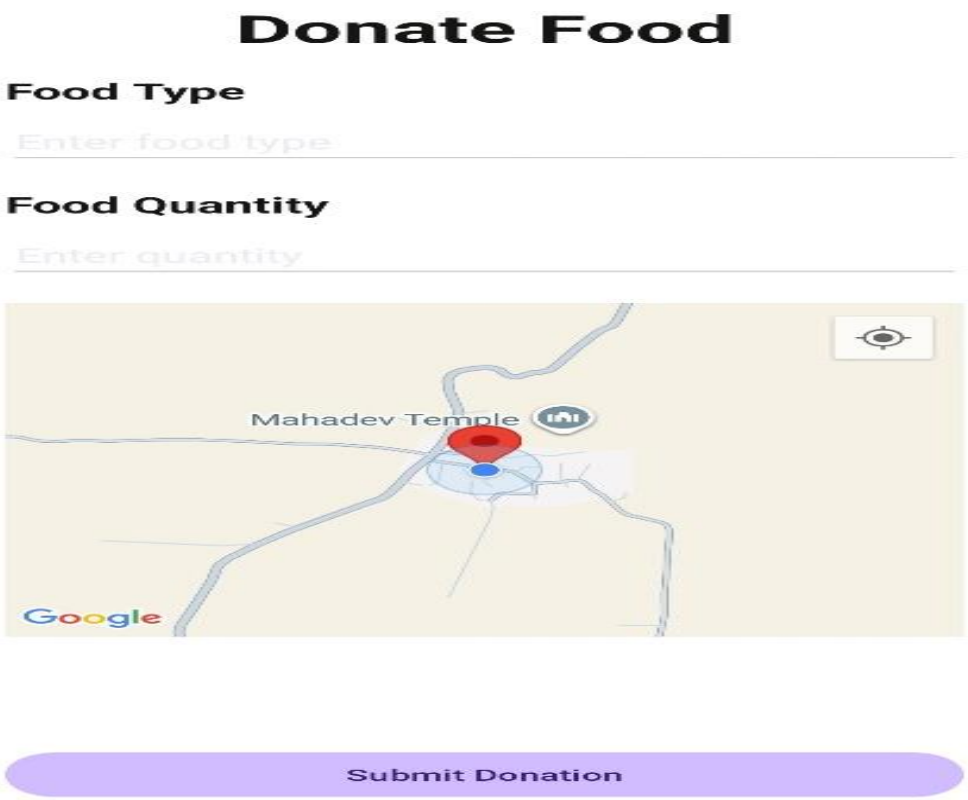


Fig. 3: Donor Request Page

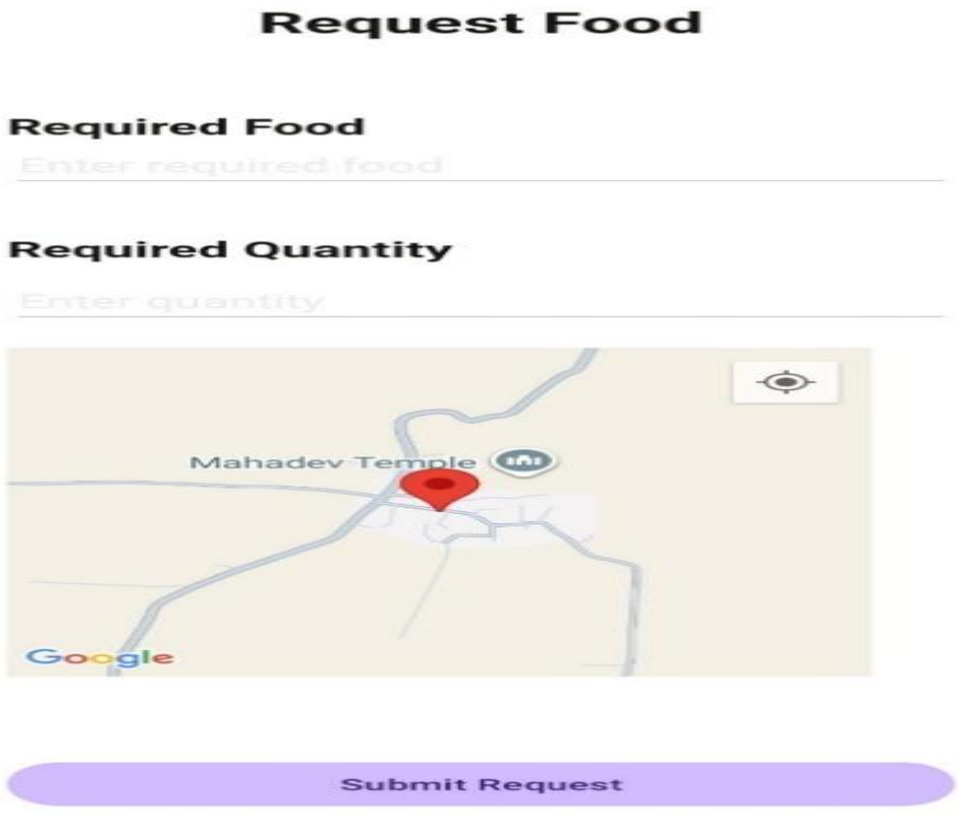


Fig. 4: Receiver Request Page

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue V, May 2026

Fig. 5: Volunteer Matching

RESULTS AND PERFORMANCE ANALYSIS

The system was evaluated based on matching efficiency and response time.

- Average Matching Time: 1.2 seconds
- Matching Accuracy: 85%
- Top-5 Recommendation Success Rate: 80%

The AI-based recommendation significantly reduced manual effort and improved matching speed compared to traditional methods.

The results demonstrate that the proposed system improves matching efficiency compared to manual methods by reducing response time and increasing accuracy.

Security And Privacy

The system ensures secure user authentication and role-based access control. Data is transmitted through secure APIs, and access is restricted based on user roles. Measures are implemented to prevent duplicate or fraudulent requests.

CONCLUSION

The Annapurna Food Donation Application provides an effective and scalable solution to address the growing problem of food wastage and hunger. By integrating web and mobile platforms with a centralized backend

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue V, May 2026

system, the application enables seamless interaction between donors, receivers, and volunteers. The role-based architecture ensures organized management of donation and request processes.

The inclusion of an AI-based recommendation module enhances the efficiency of the system by suggesting suitable matches between donors and receivers based on factors such as location, urgency, and quantity. This hybrid approach reduces manual effort, improves response time, and ensures better utilization of available resources.

Overall, the proposed system improves coordination, increases transparency, and supports timely food distribution. It has the potential to significantly reduce food wastage and contribute to social welfare by making food donation processes more efficient and accessible.

REFERENCES

1. "Reduction of Food Wastage through Android," International Journal of Scientific & Engineering Research, vol. 10, no. 10, Oct. 2019, pp. 915–920, ISSN: 2229-5518.
2. J. Manikandan and N. Kumar, "Food Wastage Reduction Through Donation," International Research Journal of Engineering and Technology (IRJET), vol. 7, no. 3, Mar. 2020.
3. Food and Agriculture Organization (FAO), "Food Loss and Food Waste Data," Available: http://www.fao.org/food-loss-and-food-waste/flw-data
4. V. Mhatre, S. Chavan, S. Gamare, and V. Salunkhe, "Waste Food Management and Donation App," vol. 9, no. 3, Mar. 2022.
5. "Designing A Mobile Application For Food Wastage Reduction," Journal of Emerging Technologies and Innovative Research (JETIR), 2021.