## **Project Report: Local Food Wastage Management System**

**Author:** Tushar Garg

**Date:** August 15, 2025

**Project Title:** Local Food Wastage Management System

### **1. Executive Summary**

This report documents the development of the Local Food Wastage Management System, a data-driven web application designed to combat food waste and insecurity at a community level. The project successfully connects food providers (restaurants, grocery stores) with surplus food to receivers (NGOs, individuals) through an intuitive, interactive platform.

Leveraging **Python**, **SQL**, and **Streamlit**, the system provides a robust solution for listing available food, analyzing donation trends, and managing data records. The final application features an analytics dashboard with 15 distinct SQL queries, a dynamic food listing page with multi-criteria filtering, and full CRUD (Create, Read, Update, Delete) functionality. The project overcame initial technical hurdles related to database management by adopting a flexible and resilient **SQLite** backend, ensuring a successful and functional outcome.

### **2. Introduction & Problem Statement**

Food wastage is a critical global issue with significant social, economic, and environmental impacts. While a substantial amount of edible food is discarded daily by households and businesses, a large segment of the population struggles with food insecurity. This project addresses this disconnect by creating a centralized platform to facilitate the efficient redistribution of surplus food.

The primary goal was to develop a system where:

* Restaurants and individuals can easily list surplus food items.
* NGOs or individuals in need can find and claim this food.
* Data on food donations and claims can be analyzed to derive actionable insights.

### **3. Methodology & Development Process**

The project was executed in a structured, step-by-step manner, from setting up the environment to deploying the final application.

#### **3.1. Initial Setup & Tools**

* **Code Editor:** Visual Studio Code (VSCode) was used as the primary integrated development environment (IDE).
* **Programming Language:** Python was the core language for backend logic and application development.
* **Virtual Environment:** A Python virtual environment (venv) was established to manage project-specific dependencies and avoid library conflicts.
* **Key Libraries:**
  + **Streamlit:** For building and running the interactive web application.
  + **Pandas:** For efficient data manipulation and reading CSV files.
  + **SQLite3:** Python's built-in library for interacting with the SQLite database.

#### **3.2. Database Creation & Data Preparation**

The foundation of the project is a structured SQL database.

* **Initial Challenge:** The initial approach using MySQL encountered persistent "list index out of range" errors during the CSV data import process via the MySQL Workbench wizard. This indicated formatting inconsistencies within the source CSV files that were difficult to resolve directly.
* **Solution - Adopting SQLite:** To overcome this roadblock and simplify the overall architecture, a decision was made to switch to **SQLite**. This serverless, file-based database system eliminated the need for a complex server setup and provided a more robust method for data ingestion.
* **Automated Data Loading:** A Python script (load\_data.py) was created to automate the entire data preparation and database creation process. This script performs the following actions:
  1. Reads each of the four provided CSV files (providers\_data.csv, receivers\_data.csv, food\_listings\_data.csv, claims\_data.csv) into a Pandas DataFrame.
  2. Creates a new SQLite database file named food\_wastage.db.
  3. Loads the data from each DataFrame into a corresponding table within the SQLite database, automatically handling schema creation.

This automated approach proved to be highly effective, completely bypassing the manual import errors and creating a reliable, self-contained database.

### **4. System Architecture & Data Flow**

The system's architecture is designed for simplicity and efficiency.

1. **Data Storage:** The raw data resides in four CSV files.
2. **Processing Pipeline:** The load\_data.py script acts as the processing pipeline. It reads the CSVs, cleans the data implicitly using Pandas, and populates the food\_wastage.db SQLite database.
3. **Backend Logic:** The main app.py script contains all the backend logic. It connects to the SQLite database, runs SQL queries to fetch or modify data, and processes the results.
4. **Frontend Interface:** Streamlit renders the user interface in the web browser. It takes user input (e.g., filter selections, form submissions) and communicates with the backend to display data and execute actions.

This creates a clear and linear data flow from the source files to the end-user.

### **5. SQL-Powered Data Analysis**

A core requirement of the project was to perform in-depth data analysis using SQL. The "Analytics Dashboard" page of the application was built specifically for this purpose. It allows users to select from a dropdown of 15 predefined analytical questions.

For each selected question, the application:

1. **Displays the full SQL query** used for the analysis, providing transparency and educational value.
2. **Executes the query** against the SQLite database.
3. **Presents the results** in a clean, tabular format.
4. **Generates a bar chart** for visual comparison when the data is suitable.

Key analytical questions answered include:

* Distribution of food providers and receivers by city.
* Top contributing provider types by food quantity.
* Percentage distribution of claim statuses (Completed, Pending, Cancelled).
* Busiest days of the week for food claims.
* Listings of food items nearing their expiry date.

### **6. Application Features & Functionality**

The final Streamlit application is a comprehensive tool that meets all project objectives. It is organized into three main pages:

#### **6.1. 📈 Analytics Dashboard**

This page serves as the central hub for data insights, featuring the 15 SQL query analyses as described above.

#### **6.2. 🔍 Find Food & View Listings**

This is the primary interface for users seeking food. It features:

* **A full, searchable table** of all available food listings.
* **Contact and location details** for each provider to facilitate direct coordination.
* **A powerful filtering system** in the sidebar, allowing users to narrow down listings by:
  + City
  + Provider Type (e.g., Restaurant, Grocery Store)
  + Food Category (e.g., Vegetarian, Vegan)
  + Meal Type (e.g., Lunch, Dinner)

#### **6.3. 📝 Manage Data (CRUD)**

This page provides administrative capabilities for managing the database records directly from the user interface. It is organized into tabs for each operation:

* **Add Listing (Create):** A form to add new food items to the food\_listings\_data table.
* **Update Claim (Update):** A tool to change the status of a pending claim to "Completed" or "Cancelled".
* **Delete Listing (Delete):** An option to permanently remove a food listing and its associated claims from the database.

### **7. Conclusion & Future Scope**

This project successfully delivered a fully functional Local Food Wastage Management System. By adopting a pragmatic approach and pivoting to SQLite to overcome initial technical challenges, all core requirements were met and exceeded. The final application is a powerful tool for data analysis, resource management, and social good.

**Future Enhancements could include:**

* **User Authentication:** Creating login systems for providers and receivers.
* **Live Map Integration:** Using geolocation to display food availability on an interactive map.
* **Automated Notifications:** Sending email or SMS alerts to receivers when new food is listed in their area.
* **Deployment:** Deploying the application to a cloud service like Streamlit Community Cloud to make it publicly accessible.