# SD Food Connect: Leveraging Technology to Address Food Insecurity in San Diego

Department of Information Systems, San Diego State University

Instructor: Carlos D. Paternina, Ph.D.

April 18, 2025

# ‘Methodology’

## 4.1 Requirements Gathering

The first phase involved defining system objectives and gathering requirements from stakeholders including food donors (e.g., restaurants, grocers), recipient organizations (e.g., food banks), and volunteer drivers. Consultations was conducted via interviews and foc groups. These discussions revealed inefficiencies in real-time availability tracking and route planning. In parallel, this project analyzed individually existing food donation platforms and reviewed academic literature to identify best practices in system design, data security, and user interface development. The findings led to a formal requirements document that detailed performance benchmarks, user roles, and core system functionalities.

## 4.2 ‘System Design’

The architecture of SD Food Connect was intentionally modular and scalable to meet real-time operational demands while ensuring data security and user accessibility. Figure 1 illustrates the end-to-end system flow, which enables seamless interaction between food donors, recipient organizations, and volunteer drivers.

Core Components:

1. Frontend: Built with React, featuring role-based dashboards tailored to donors, recipients, and volunteers. React-Leaflet was integrated for interactive maps, allowing users to visualize donation pickups and deliveries.
2. Backend: A Flask API facilitates communication between frontend and backend services. It handles user authentication, routing logic, donation matching, and transaction logging.
3. Database: PostgreSQL manages relational data including donation metadata (type, volume, perishability), user credentials (securely hashed), delivery logs, and matching records. Geo-indexing features support fast spatial queries.
4. Mapping & Routing: OpenStreetMap (OSM) and Nominatim API are used for address geocoding and route planning. This supports future integration with real-time traffic APIs like Mapbox Directions or Google Maps.
5. Matching Engine: Built on rules-based logic prioritizing perishability (40%), proximity (35%), and quantity (25%). Donations are dynamically scored to identify optimal recipients and drivers.
6. BI Dashboards: Operational insights are visualized using BI tools such as Tableau and Power BI. Metrics include delivery latency, donation types, and match success rates.
7. Security Features: Includes JWT-based session management, encrypted credentials, and anonymized storage of personal data. Secure login prevents unauthorized access to sensitive information.

A diagram of a system architecture

AI-generated content may be incorrect.

**Figure 1: System Architecture**

## 4.3 Implementation

Backend services was implemented using RESTful Flask endpoints with SQLAlchemy for ORM. The database schema supported donor, recipient, and volunteer roles with encrypted user data. The frontend employed role-based dashboards with mapping integrations through OSM and Nominatim API.

## 4.4 Validation

A pilot test was conducted using synthetic data mimicking real-world donations and delivery conditions.

The Validation steps included:  
1. Performance Metrics: Matching time and delivery duration were recorded  
2. Usability Testing: Users tested interfaces for clarity and navigation  
3. Load Testing: Simulated concurrent actions to test server performance  
4. Privacy Checks: Data was anonymized and encryption validated

## 4.5 Iterative Refinement

Feedback from stakeholders resulted in enhanced search filters, UI improvements, and backend logic updates. Iterative cycles ensured that SD Food Connect evolved with user needs and system performance insights.

## 4.6 Application Optimization Techniques

To improve the responsiveness and efficiency of the system:  
1. Asynchronous Processing: Background tasks were handled using Celery + Redis.  
2. Caching: Frequently accessed data were cached using Flask-Caching.  
3. Progressive Enhancement: Mobile-first design ensured accessibility on low-bandwidth devices.  
4. Database Indexing: Indexes were created on geospatial fields and timestamped logs.

# ‘Results and Analysis’

## 5.1 Overview of Analytical Techniques

The system was assessed through visual and statistical data analysis techniques. Data included donation records, delivery times, and matching attempts. Python (Matplotlib, Seaborn, Scikit-learn) and BI tools were used for visualization and modeling.

## 5.2 Statistical Methods and Tools

1. Descriptive Stats: To summarize donation volumes and delivery timelines  
2. Machine Learning Models: To predict match outcomes and optimize donor classification  
3. Visualization Tools: Tableau and Python plots for clarity  
4. Validation Methods: Cross-validation for models, simulated delivery trials

## 5.3 Key Visualizations and Interpretations

The analytical dashboard for SD Food Connect reveals multiple performance insights. Each visualization was designed to inform stakeholder decisions, monitor system efficiency, and guide iterative system improvements. The following visualizations were central to evaluating donation behavior and distribution logistics:

**Figure 1: Donation Types by Frequency**  
A graph of different colored squares

AI-generated content may be incorrect.  
**Interpretation:**  
This bar chart highlights the most common categories of donated food. Bread, canned goods, and produce dominate the dataset, suggesting partnerships with bakeries and grocers were particularly effective. Monitoring this distribution assists recipient organizations in planning storage and refrigeration needs.

**Figure 2: Total Quantity Donated by Type**  
A pie chart with different colored circles

AI-generated content may be incorrect.  
**Interpretation:**  
While donation frequency offers one lens, this visualization emphasizes total weight. For instance, though produce might appear less frequently, its bulk volume is considerable. This metric is critical for assessing transportation needs and volunteer capacity planning.

**Figure 3: Delivery Times (Boxplot)**  
A blue box diagram with white text

AI-generated content may be incorrect.  
**Interpretation:**  
This figure shows the variance in delivery duration, with a typical range between 30 to 90 minutes. Outliers beyond 120 minutes indicate potential mismatches in routing or traffic delays. Adjustments to volunteer routing logic were informed by these findings.

**Figure 4: Delivery Times**   
A graph of a distribution of delivery times

AI-generated content may be incorrect.  
**Interpretation:**  
A concentration of deliveries within the 60–80 minutes window supports the reliability of the matching algorithm under average load. This distribution aligns with operational benchmarks and guides volunteer scheduling strategies.

**Figure 5: Spatial Distribution of Deliveries**  
A map of the world

AI-generated content may be incorrect.  
**Interpretation:**  
Geospatial clustering around metro San Diego validates the pilot area choice. Denser delivery zones also correlate with higher donor participation. These insights support future clustering logic in the expansion strategy.

**Figure 6: Matching Success Over Time**  
A graph of a graph

AI-generated content may be incorrect.  
**Interpretation:**  
The upward trend in successful matches demonstrates the algorithm’s effectiveness. Iterative backend improvements and UI refinements contributed to increased user engagement and better system responsiveness.

**Figure 7: Delivery Time Over Time**  
A graph of a delivery

AI-generated content may be incorrect.  
**Interpretation:**  
Fluctuations in delivery time across days reflect external variables such as volunteer availability and donation surges. These patterns inform future development of time-aware route planning, incorporating traffic data and peak hour scheduling.

## 5.4 Machine Learning Models Utilized

1. **Logistic Regression**: Predicted match success using donation type, quantity, and distance.

A logistic regression model was developed to predict whether a food donation would be successfully matched with a nearby food bank or recipient. This binary classification model considered features such as Donation Type (categorical), Quantity (kg), and Distance (km) between donor and recipient. One-hot encoding and standardization were applied in preprocessing. The model output provided probabilistic predictions, allowing for threshold-based decision-making. This model was chosen for its interpretability and efficiency in producing real-time predictions for operational use (Pedregosa et al., 2011).

2. **Random Forest Classifier:** Incorporated perishability and revealed its importance in match probability.

To improve prediction accuracy and understand feature contributions, a Random Forest Classifier was implemented. This model included the additional variable Perishability, mapped from food categories on a scale from 1 (non-perishable) to 4 (highly perishable). The classifier provided not only high prediction accuracy but also feature importance scores, which indicated that perishability and distance were the most influential factors in determining match success. These insights support operational strategies, such as prioritizing matches for highly perishable and nearby food items (Breiman, 2001).

3. **K-Means Clustering**: Segmented donors by geolocation and donation volume for route optimization.

An unsupervised K-Means clustering model was used to segment donors based on their geolocation (latitude and longitude) and donation volume. Standardized values were used to identify natural groupings in the data, such as urban donors with frequent small donations versus suburban donors with larger but less frequent contributions. The clustering output is valuable for planning volunteer pickup routes, optimizing donor engagement, and resource allocation across regions (MacQueen, 1967).

**Integration and Application.**  
These three models collectively support the Food Connect platform by enabling intelligent matchmaking, operational planning, and donor segmentation. Logistic regression serves real-time prediction needs; Random Forest provides deeper insights and interpretability; and K-Means aids in geographic clustering for logistical optimization. Together, they form the foundation of the platform's decision-support system, helping to enhance efficiency and reduce food waste through data-driven strategies.

## 5.5 Assumptions and Justification

1. Synthetic Data: Simulated realistic donation and delivery conditions  
2. Data Privacy: All identifying information was anonymized  
3. Tool Selection: Chosen tools ensured efficiency, interpretability, and scalability

## 5.6 Scalability and Risk Considerations

Though SD Food Connect performed well in metro San Diego, expansion efforts surfaced key challenges:  
1. Volunteer Shortages in Suburban/Rural Areas  
2. Internet Access Limitations for low-bandwidth devices  
3. Routing Complexity in low-density areas

## 5.7 Matching Algorithm Scoring Formula

Each donation is scored using a weighted composite of perishability (40%), proximity (35%), and quantity (25%) to determine priority in the matching algorithm.

Score = (0.4 × Perishability) + (0.35 × Proximity) + (0.25 × Quantity)

# ‘References’

1. Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32. https://doi.org/10.1023/A:1010933404324
2. MacQueen, J. (1967, June). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability* (Vol. 1, No. 14, pp. 281–297).
3. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.