CS 383 – Group Project

Software Design Document

SDD

for:

MealBridge

Presented to:

Dr. Rafaa Aljarbua

Table of Contents

1. Introduction

1. Purpose
2. Product scope
3. References
4. Structure

2. System Overview

3. Architecture Design

1. Architecture description
2. Decomposition description
3. Design rationale

4. Data Design

1. Database description
2. Data Structure

5. Component Design

1. Class diagrams
2. State diagrams
3. Activity diagrams
4. Sequence diagrams

6. Human Interface Design

1. Overview of user interface
2. Detail design of user interface

Introduction

This document presents the Software Design Document (SDD) for MealBridge, a cross-platform application designed to connect surplus food donors with beneficiaries to reduce food waste and support communities in need. The following sections describe the system architecture, detailed component designs, data structures, and interface specifications, providing a foundation for implementation, testing, and maintenance.

1.1. Purpose

The Software Design Document (SDD) describes the architectural and detailed design of the MealBridge system. Its purpose is to:

- Translate all system requirements (as defined in the SRS and meeting documentation) into a clear, implementable technical design.
- Describe system components, system architecture, database schema, APIs, data flows, and UI structure.
- Provide the development team with UML diagrams (Sequence, Activity, State) that guide consistent and correct implementation.
- Support maintenance, enhancement, and future scalability of the system.

1.2. Product Scope

MealBridge is a digital platform designed to manage, organize, and facilitate operations related to food services or meal resources. The system includes:

- User management with multiple roles (user, provider, administrator).
- Creation and management of items/resources (meals, food items, requests, or any relevant objects).
- A reservation/booking or request-handling system (Reservation/Loan) with status tracking.
- Documentation of meetings (Meeting Minutes) and storage of all project files in a version-controlled repository (e.g., GitHub).
- Integration with RESTful APIs for external systems or internal modules.
- Security and privacy measures to ensure confidentiality, data quality, and secure access.
- An administrative dashboard for monitoring system status and generating reports.

1.3. References

This document is based on:

- MealBridge SRS
- https://www.lucidchart.com/pages/uml-sequence-diagram
- https://www.lucidchart.com/pages/uml-class-diagram
- https://www.lucidchart.com/pages/tutorial/uml-activity-diagram

1.4. Structure

This SDD is organized into the following sections:

1. Introduction – Explains the purpose of the SDD, the overall scope of the MealBridge system, the references used, and how the rest of the document is organized.
2. System Overview – Provides an explanation of what MealBridge is, its goals, major features, and the context in which it operates.

3. Architecture Design – Describes the system's architectural style, the major modules and how they interact, and the reasoning behind key design choices.
4. Data Design – Outlines the structure of the database, key entities, relationships, and the data formats used throughout the system.
5. Component Design – Contains detailed design models such as class diagrams, activity diagrams, state diagrams, and sequence diagrams that show how system components behave and interact.
6. Human Interface Design – Presents the layout, navigation flow, and visual structure of the user interface, including mockups and UI behavior.

System Overview

- MealBridge is a cross-platform mobile and web system designed to connect surplus food donors with recipients in an efficient and secure way.
- The system follows a multi-tier client–server architecture, where the client handles user interaction, the server processes business logic, and the database manages all stored information.
- Donors can create meal donations by entering food details, pickup information, and uploading photos through camera or gallery integration.
- Recipients can browse available donations using list or map views, apply filters, and submit claims for suitable meals.
- The backend manages donation status updates, automated matching, notifications, and secure data handling across all user roles.
- Administrators have access to a dedicated dashboard for monitoring system activity, generating reports, verifying users, and managing platform operations.
- The system integrates essential external services such as location APIs, cloud-based push notifications, and secure authentication to enhance usability and reliability.
- Overall, MealBridge provides a streamlined, scalable, and user-friendly platform that reduces food waste and improves food distribution efficiency within the community.

Software Process Model

1. Selected Process Model

MealBridge will follow the **Agile software process model**.

2. Rationale for Selection

Agile is suitable for MealBridge because the system contains multiple evolving components such as multi-role user interfaces (donor, recipient, admin), external integrations (maps, authentication, cloud storage), real-time communication, and GPS/camera hardware usage. These requirements may change or require refinement as the system is designed and tested.

Agile supports iterative development, allowing the team to gradually design, implement, and validate features while accommodating feedback from stakeholders and adapting to changes.

3. Alignment with Project Requirements

- Agile enables frequent increments that fit well with MealBridge's needs, including:
- UI and UX refinement for **Arabic and English mobile/web interfaces** (as defined in the SRS).
- Progressive integration of external services such as **maps showing nearby donations, camera access for meal photos, and gallery upload functionality** (as defined in the SRS).
- Incremental testing of secure communication protocols and real-time updates (as defined in the SRS).
- Stepwise development of admin and partner interfaces (as defined in the SRS).

These features benefit from short development cycles where functionality can be reviewed and improved regularly.

4. Key Process Activities

The project will follow core Agile activities:

- Backlog creation and refinement based on the **SRS document**.
- Iteration planning to select features for short development cycles.
- Incremental implementation and testing of UI, backend, and integration modules.
- Review and feedback at the end of each iteration to validate progress.
- Continuous improvement through reflection and adjustment.

Architecture Design

1. Architecture description

The MealBridge application will utilize a **Multi-Tier Client-Server Architecture** design pattern. This approach is suitable for a distribution system that needs to support multiple client types (mobile and web) while maintaining security, scalability, and centralized data management (pp. 4-5).

This architectural style logically separates the system into distinct, independent tiers that interact over a network:

- **Presentation Tier (Client Side):** Handles the user interface and interaction logic.
- **Application/Business Tier (Server Side):** Processes business logic, orchestrates workflows, and manages interactions with external services.
- **Data Tier (Backend Services):** Manages data storage, retrieval, and persistence.

This structure ensures that changes in one tier (e.g., updating the mobile app UI) do not necessitate changes in another (e.g., the core database structure), promoting modularity and maintainability. The system will operate as a digital platform relying on secure, cloud-based data management and external services for location matching and push notifications.

2.Decomposition description

2.1. Core Components and Interaction

The system is organized into the following layers, as illustrated in the accompanying block diagram:

1. User Interface (UI):

Tier Role: Represents the Presentation Tier (Client Side).
Function: Handles all user inputs and displays the application interfaces for donors and recipients. It sends all processing requests directly to the Business Logic layer.

2. Business Logic:

Tier Role: Represents the Application Business Tier (Server Side).
Function: Serves as the system's core processing engine. It executes all operational rules (e.g., safety verification, automated matching, and managing donation status transitions). It orchestrates interactions between the UI and Data Access layers.

3. Data Access:

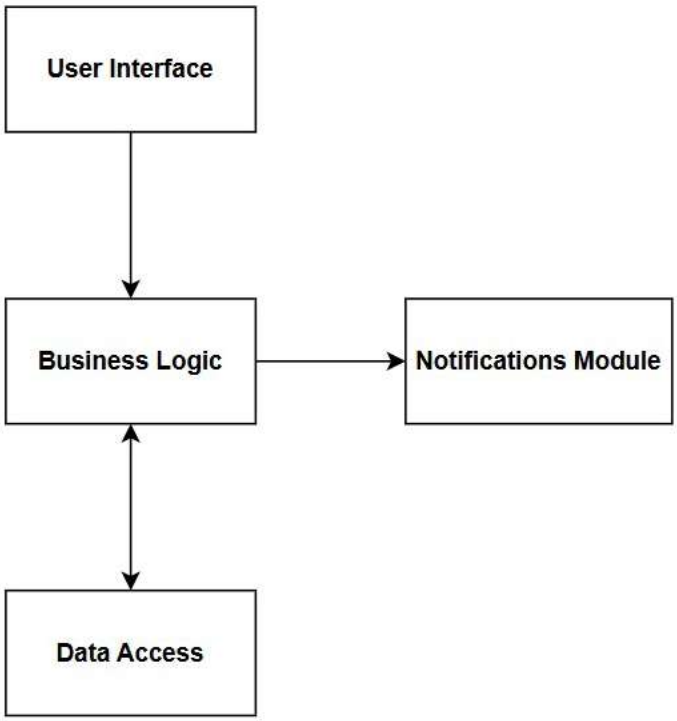
Tier Role: Represents the Data Tier (Backend Services).
Function: Exclusively handles secure storage and retrieval of all data (users, donations, history) from the central database. It only receives commands from the Business Logic layer.

4. Notifications Module:

Tier Role: Functions as an integral External Service.
Function: Triggered by the Business Logic layer to send instant alerts and reminders to users (e.g., reservation confirmations or availability updates).

2.2. Control Flow Summary

The interaction follows a vertical path: requests move from the UI to Business Logic, which then communicates with Data Access for persistence. The Business Logic is the central point for initiating any external actions, such as triggering the Notifications Module.



3.3 Design Rationale

The architecture and design of the MealBridge system were developed to meet the practical, social, and operational needs of users involved in food donation. The following key factors explain the major design decisions:

1. Compatibility and Efficiency

MealBridge was built using the Flutter framework to ensure full compatibility with both Android and iOS platforms. By relying on a single codebase, the development process becomes faster and maintenance becomes easier, allowing the system to reach a wider audience including donors, volunteers, and charity organizations. This approach ensures consistent performance and a unified user experience across all supported devices.

2. Seamless and Accessible User Experience

A simple, clean, and intuitive interface was prioritized to make the application easy to use for all user types whether they are individuals donating meals, volunteer teams, or charity staff. The system guides users step-by-step through actions such as creating a donation request, reviewing available donations, and updating statuses. The organized structure and responsive design reduce user effort and enhance overall satisfaction.

3. Secure and Reliable Data Management

Since the system handles sensitive information (user accounts, location data, donation history), strong security measures were included. Cloud-based storage was selected to ensure real-time access, scalability, and safe data backup. Additionally, encryption and secure API communication were chosen to protect user privacy and maintain trust among all system participants.

4. Automated Matching and Efficient Workflow

An automated matching engine was integrated to intelligently pair donations with the nearest eligible charity organizations. This decision helps reduce manual communication, improves delivery speed, and ensures that meals reach beneficiaries before expiration. This automated workflow supports the project's mission to reduce food waste effectively.

5. Clear Role-Based System Design

MealBridge separates users into roles: donors, charity organizations, and system administrators. Each role has different access permissions and interface elements. This design choice enhances security, prevents misuse, and ensures that each user only interacts with the features they need. It also makes maintenance easier and supports future scalability.

6. Integration with Essential External Services

The system relies on map services and real-time location APIs to match donors and charities efficiently. These integrations were selected to ensure accurate distance calculations and optimal routing. Cloud push-notification services were also included to allow instant updates, reminders, and alerts. Such external integrations help create a smooth, connected, and reliable user experience.

Data Design

4.1. Database description

The MealBridge system uses a relational database to store and manage all core application data. The database is designed around five main entities: User, Donation, Match, Notification, and Feedback. Primary keys uniquely identify each record, while foreign keys enforce referential integrity between related tables.

User:
Stores information about all registered users in the system, including donors, charities/volunteers, and administrators.
Typical attributes include: UserID, Name, Email, Password, Phone, Role, and Region.
Each user can create multiple donations, receive many notifications, and write feedback after completed donations.

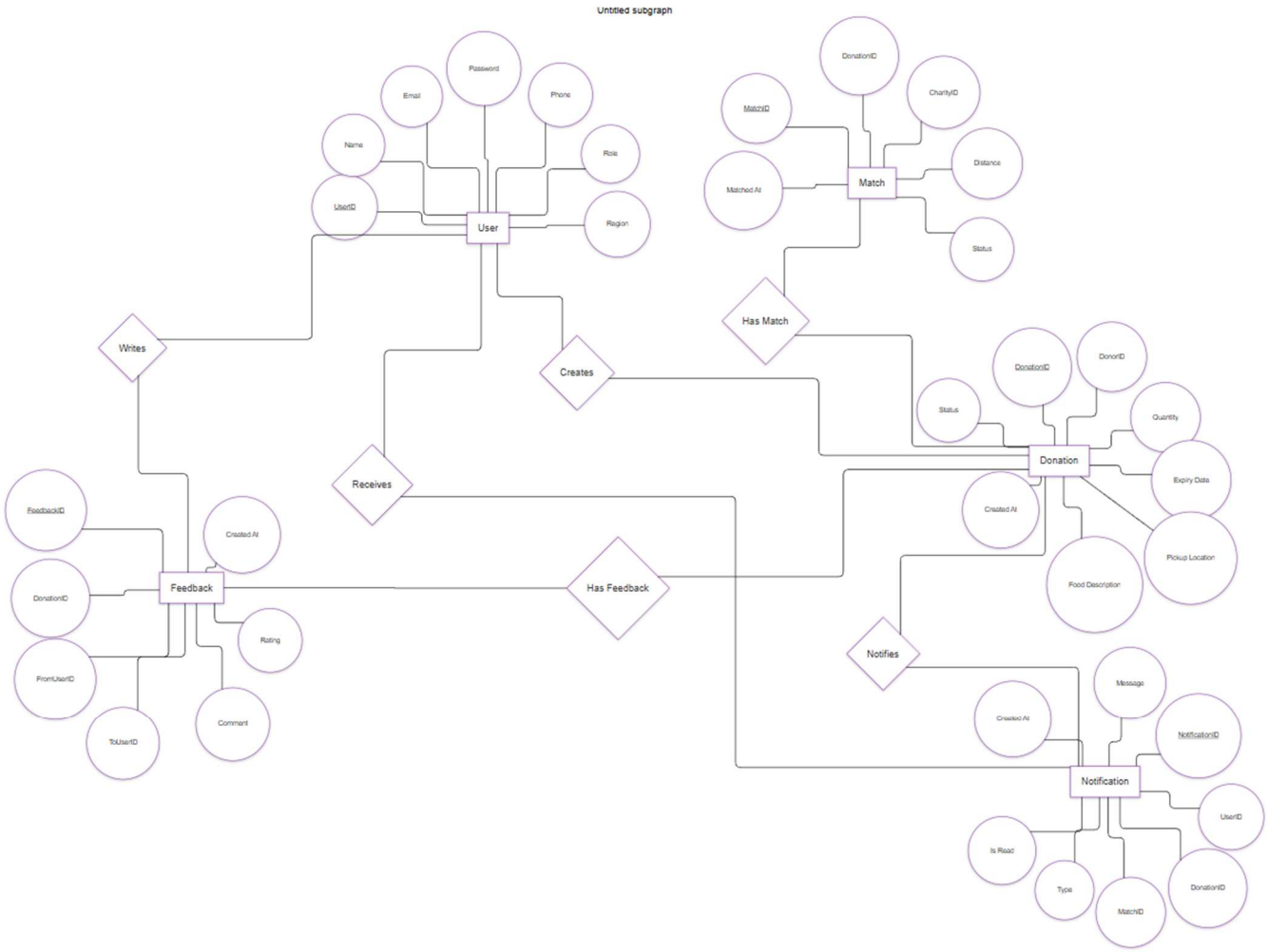
Donation:
Represents each food donation created by a donor.
Key attributes include: DonationID, DonorID, FoodDescription, Quantity, ExpiryDate, PickupLocation, Status, and CreatedAt.
DonorID is a foreign key referencing User(UserID), linking each donation to its owner.
A donation can be matched to one or more charities, can generate several notifications, and can have one or more feedback records.

Match:
Stores the result of the matching process between a donation and a charity/beneficiary.
Main attributes: MatchID, DonationID, CharityID, Distance, Status, and MatchedAt.
DonationID references Donation(DonationID) and CharityID references User(UserID), indicating which charity is assigned to which donation.

Notification:
Contains all system notifications sent to users.
Attributes include: NotificationID, UserID, DonationID, MatchID, Message, Type, IsRead, and CreatedAt.
UserID links the notification to its receiver, while DonationID and MatchID (when applicable) link the notification back to the related donation or match event.

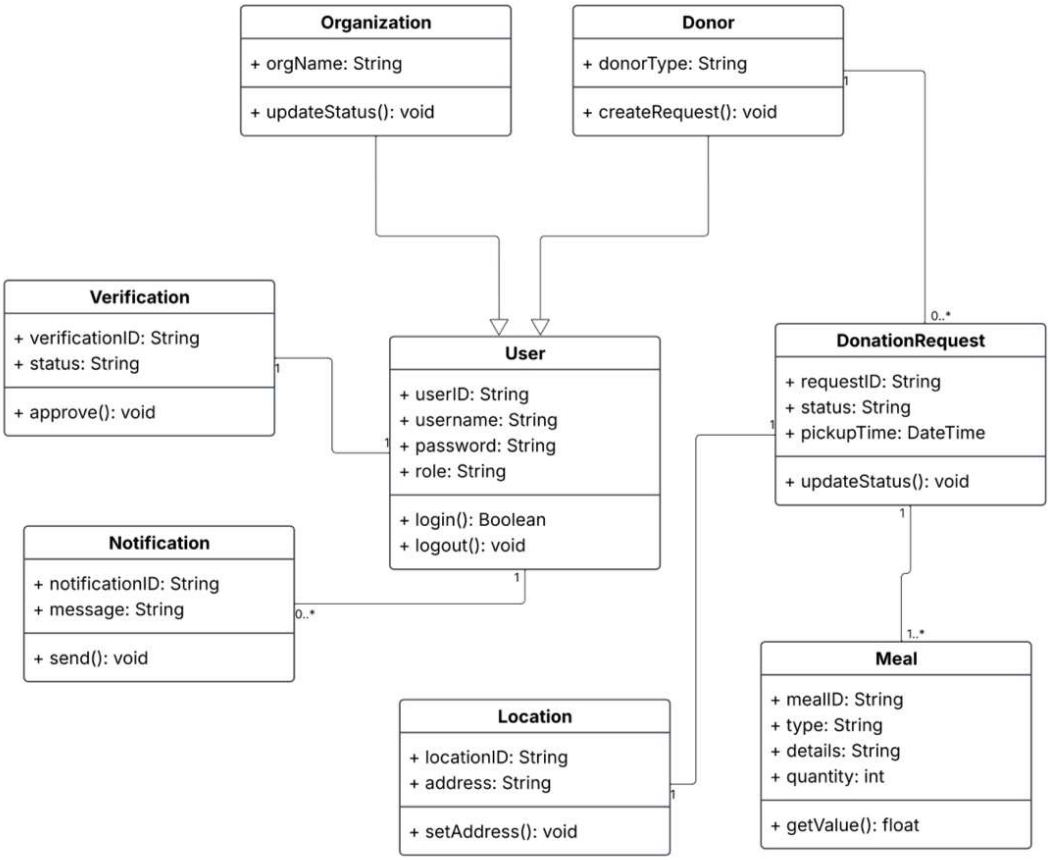
Feedback:
Stores user ratings and comments after a donation has been completed.
Attributes: FeedbackID, DonationID, FromUserID, ToUserID, Rating, Comment, and CreatedAt.
DonationID references the donation being evaluated, while FromUserID and ToUserID reference the users who give and receive the feedback.
These tables are connected through well-defined primary and foreign key relationships (e.g., User–Creates–Donation, Donation–HasMatch–Match, User–Receives–Notification, User–Writes–Feedback, Donation–HasFeedback–Feedback).
This structure ensures data consistency, supports traceability of each donation from creation to completion, and enables reliable reporting and auditing within the MealBridge system.

4.2. Data Structure

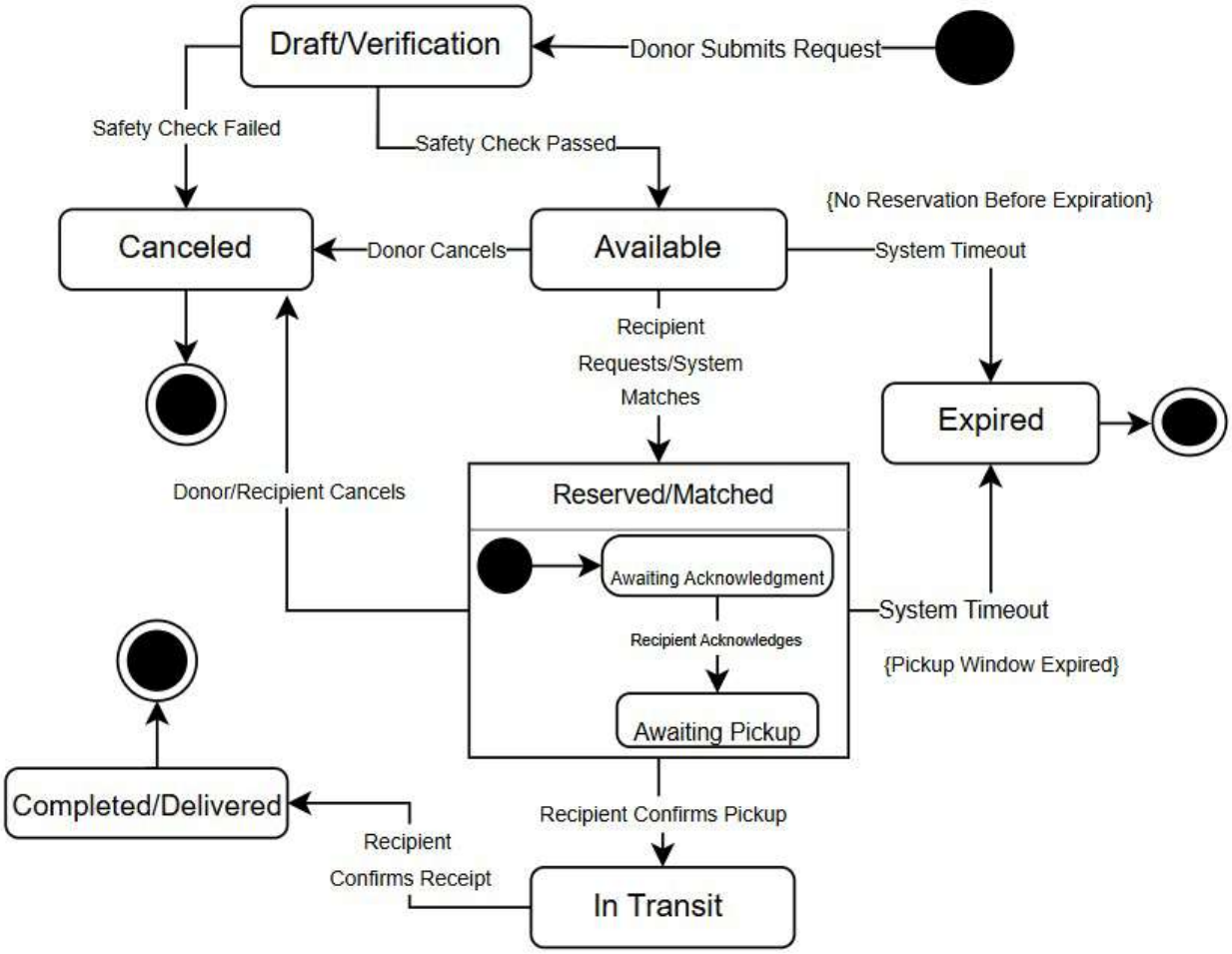


Component Design

5.1. Class diagrams



5.2. State Diagram



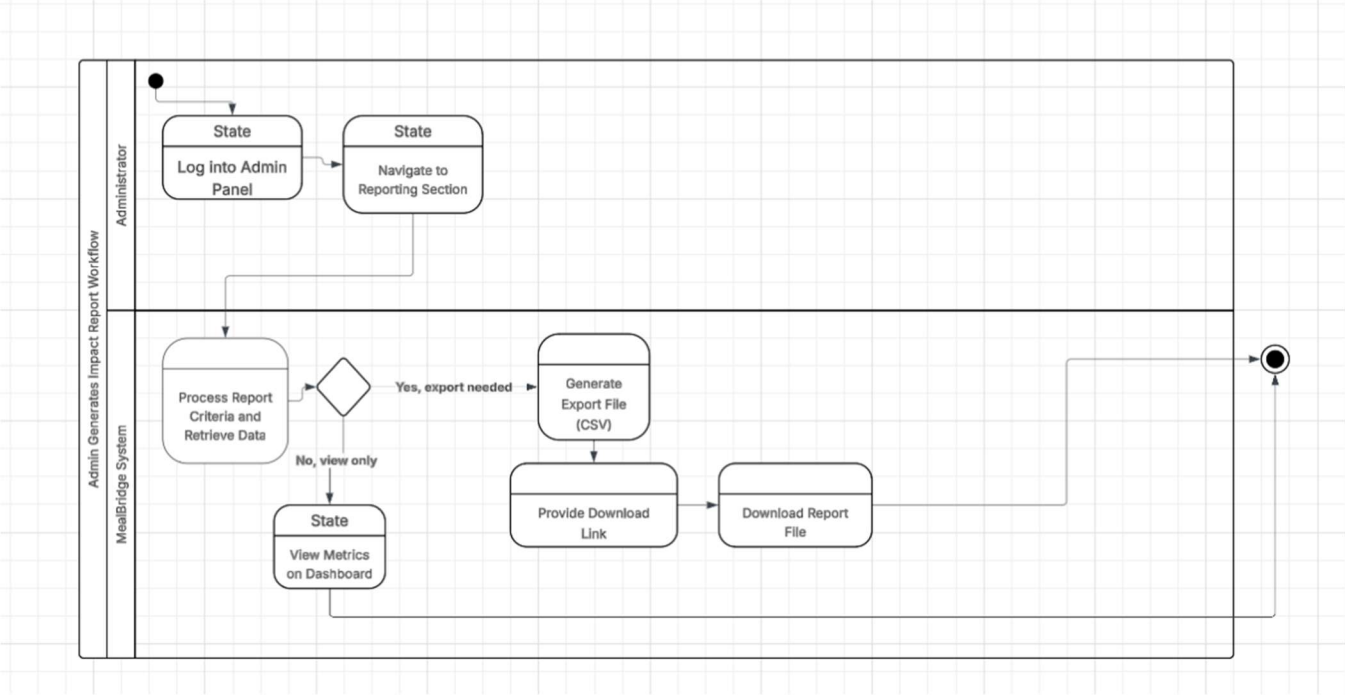
5.3. Activity Diagram

This activity diagram illustrates the workflow for the Administrator generating an impact report within the MealBridge system.

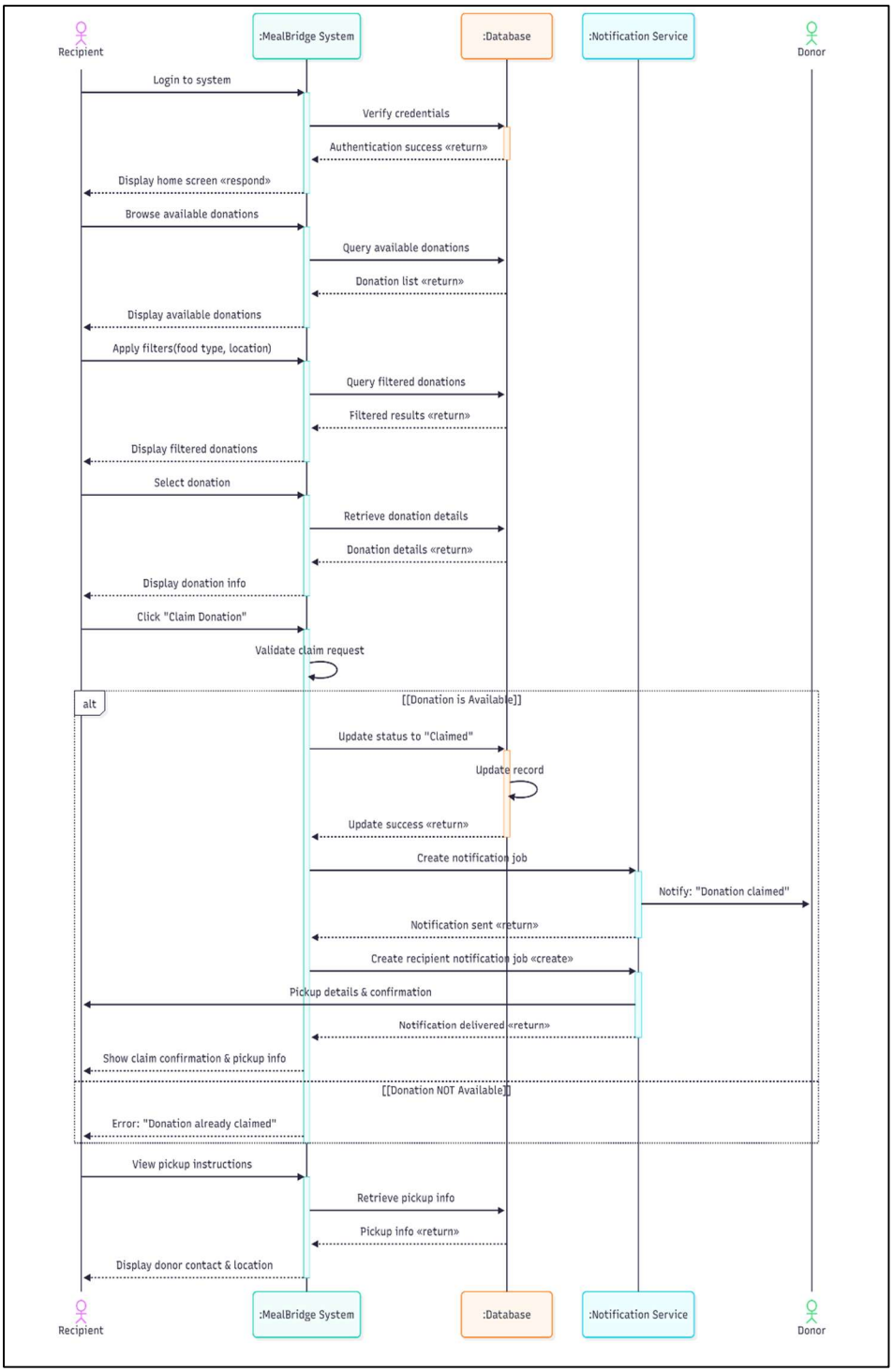
The diagram was created using **Lucidchart** and follows **UML (Unified Modeling Language)** conventions.

It maps the specific steps an administrator takes to fulfill the ***R-9: Reporting and Analytics functional requirement***. It uses lanes to distinguish actions taken by the Administrator user versus automated processes within the MealBridge System.

The flow highlights a key decision point where the administrator chooses between simply viewing metrics on a dashboard or generating and downloading a formal export file (PDF/CSV) for stakeholders.



5.4. Sequence diagrams



A charity organization recipient logs into the **MealBridge** application and browses available food donations in their area. They use the search and filter functionality to find suitable donations based on food type, Location and Expiration date. After selecting an appropriate donation, they submit a claim request. The system verifies the donation's availability, updates its status to "claimed," and automatically notifies both the donor and recipient about the successful match. The recipient then receives confirmation with pickup details and can track the donation status until collection.

Human Interface Design

6.1 Overview of User Interface

MealBridge features a mobile app-style interface optimized for iOS and Android devices. The application supports bilingual functionality (English/Arabic), it provides three distinct role-based dashboards. While the Software Design Document primarily describes the mobile app interface, a web version is also available with minor adjustments for web-specific layout and navigation.

Authentication Layer: Users access the system through a streamlined landing page leading to registration and login screens. Social authentication is supported via Google and Apple sign-in, alongside traditional email/password authentication. Role selection (Donor, Recipient, or Administrator) occurs during registration.

Navigation Pattern: The interface utilizes a bottom tab navigation bar for primary navigation between Dashboard, Browse/Create, Claimed/History, and Account sections, while the web version adapts these sections to a sidebar or top navigation layout for better desktop usability.

Donor Interface: Donors access a dashboard displaying donation statistics, active donations, and quick actions. A multi-step donation creation flow guides users through food details, photo upload, and pickup information. Post-donation feedback collection includes star ratings and text reviews. The history view tracks all past donations with status indicators.

Recipient Interface: Recipients browse available donations through both list and map views. Advanced filtering by food type (prepared meals, fresh produce, baked goods) and sorting options (nearest location, expiring soon) enhance discoverability. A claim dialog confirms pickup details before finalizing, followed by a feedback system for rating the donation experience.

Administrator Interface: Admins manage user verification, monitor system statistics, generate reports (Weekly Volume, User Growth, Geographic Distribution, Hygiene & Safety, Impact Assessment), and oversee platform activity through a centralized dashboard.

Visual Design: The interface maintains brand consistency, compact typography optimized for mobile screens.

App Version Preview (this version will be shown in the following pages)**:**

https://mealbridgeapp.netlify.app/

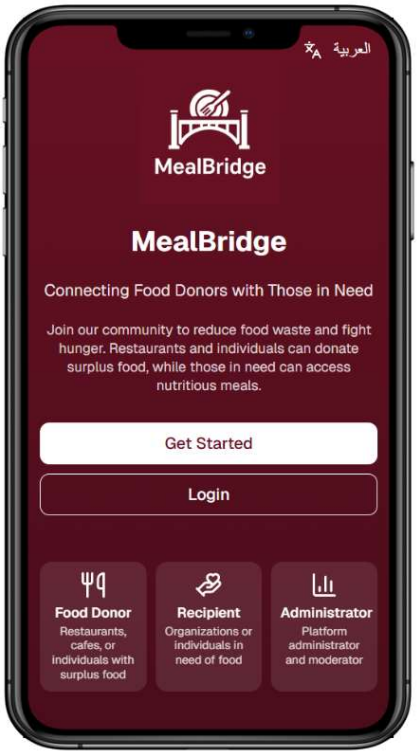
Web Version Preview (This version is the same as the app but adjusted for a web-friendly experience)**:**

https://mealbridgeweb.netlify.app/

6.2 Detail design of User Interface

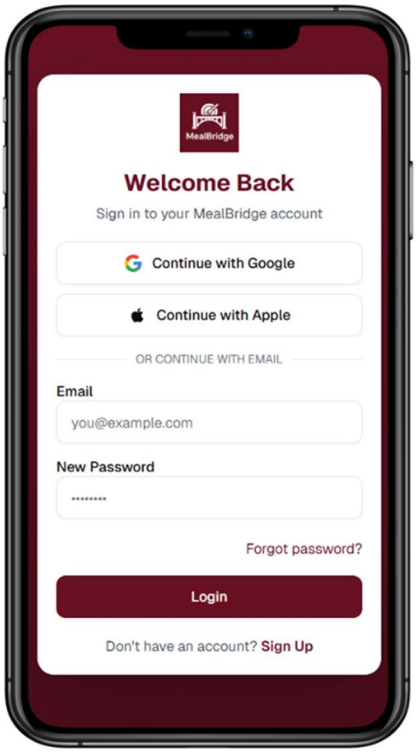
Registration:

Start page



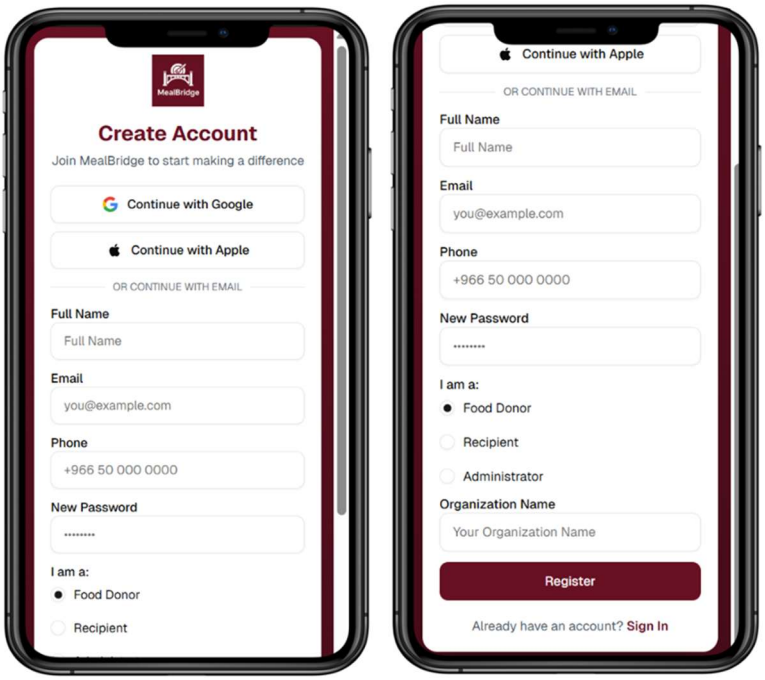
Welcome page offering Sign Up or Login options, with a brief app overview.

Login page



Login page for registered users, with options to sign in via Google or Apple.

Sign Up page

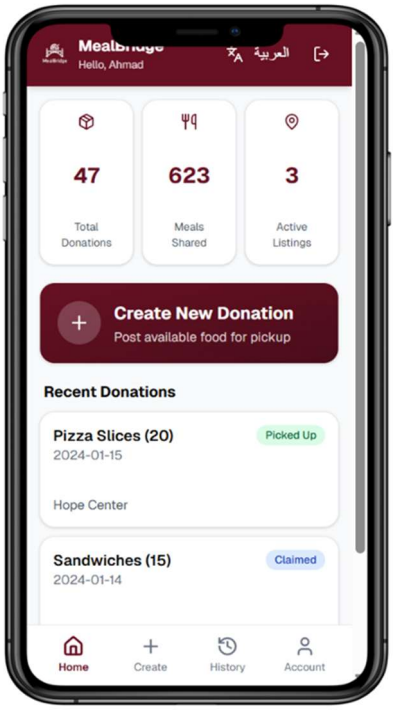


Sign-up page for new users, with options to register via Google or Apple, select a role, and provide full name, phone, and organization (for donors or recipients).

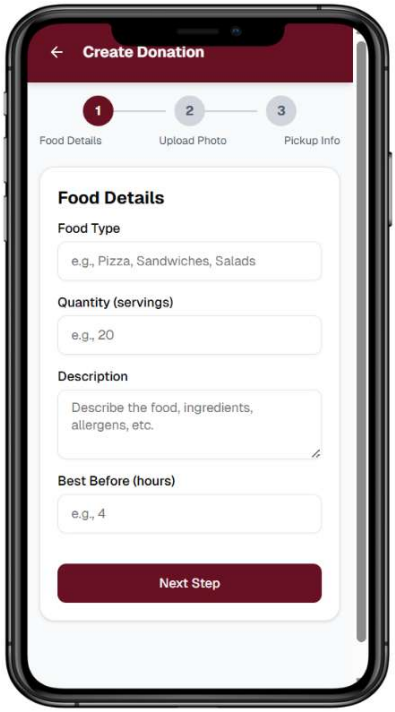
Donor Interface:

Home page

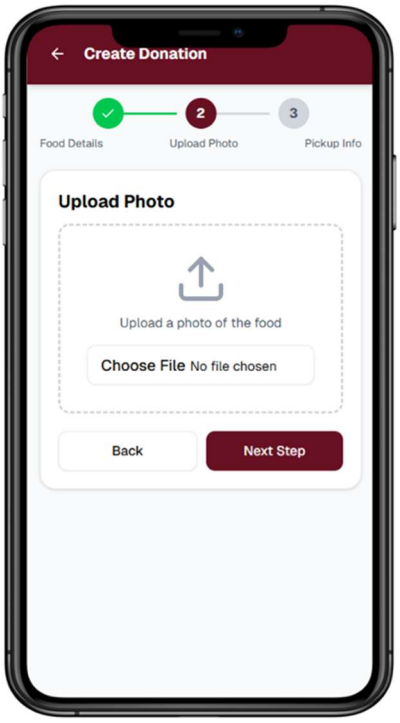
Donor dashboard showing total donations, meals shared, active listings, and a button for quick donation creation.



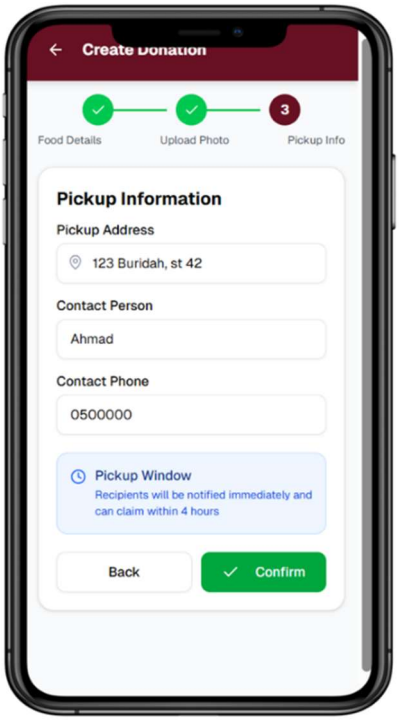
Donation Creation page



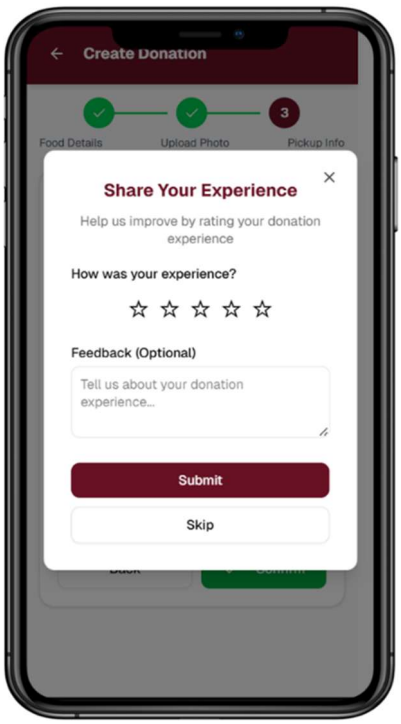
Selecting Quick Donation navigates to the donation creation page "Create", beginning with food details.



Next step is to upload a photo of the food.

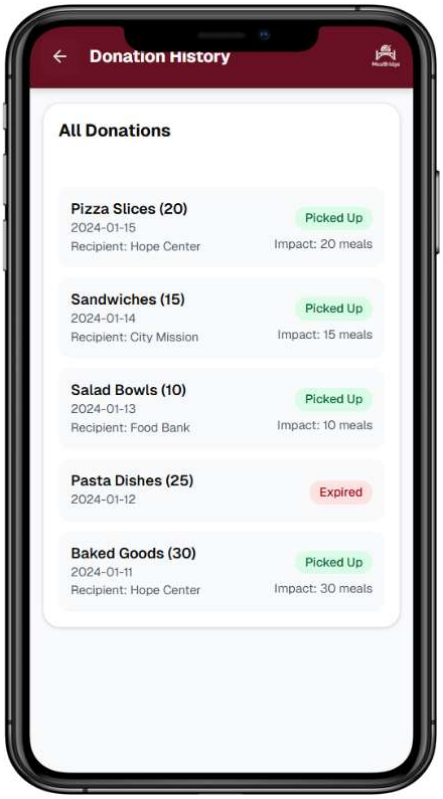


Then they provide the location and contact information.



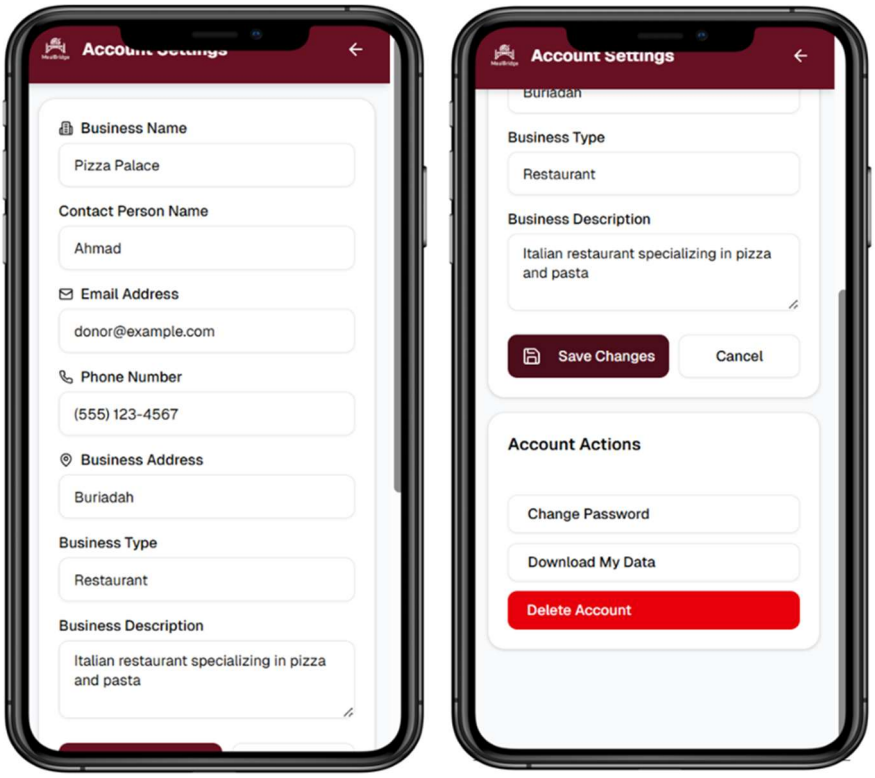
After confirming, a rating and feedback popup appears, allowing users to share their experience.

History page



The History page allows donors to track all their past donations, providing detailed information and status updates for each entry.

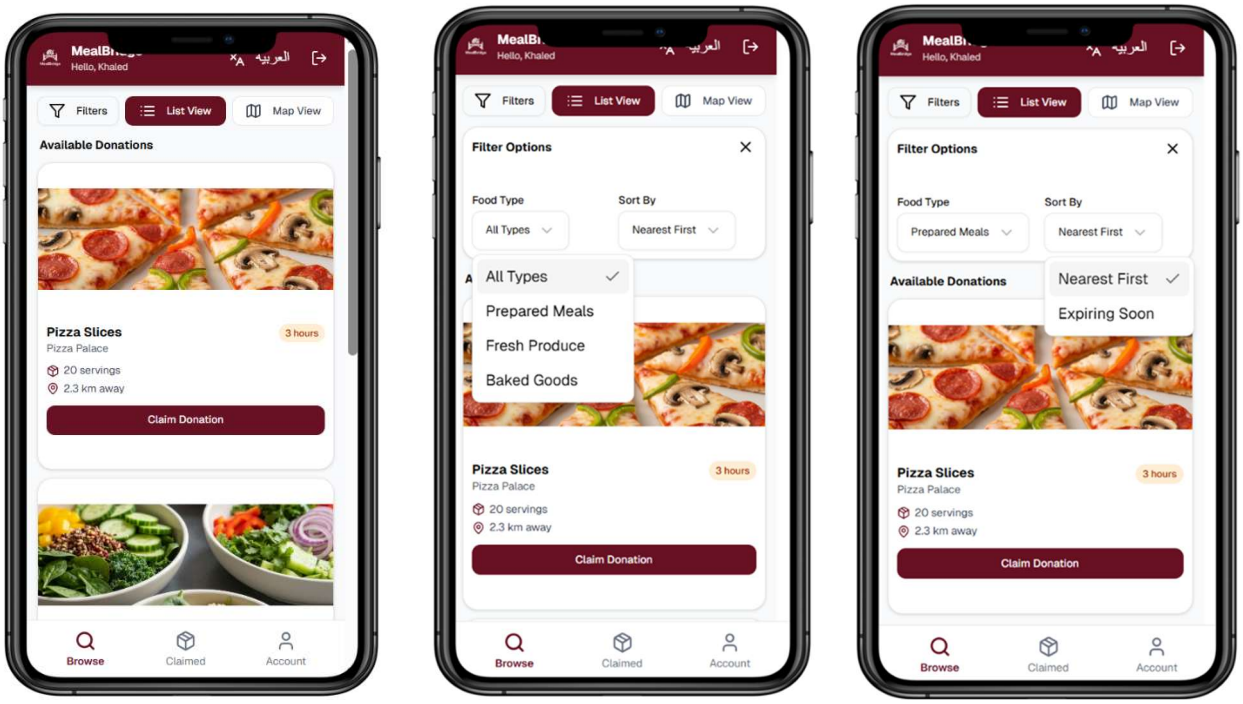
Account page



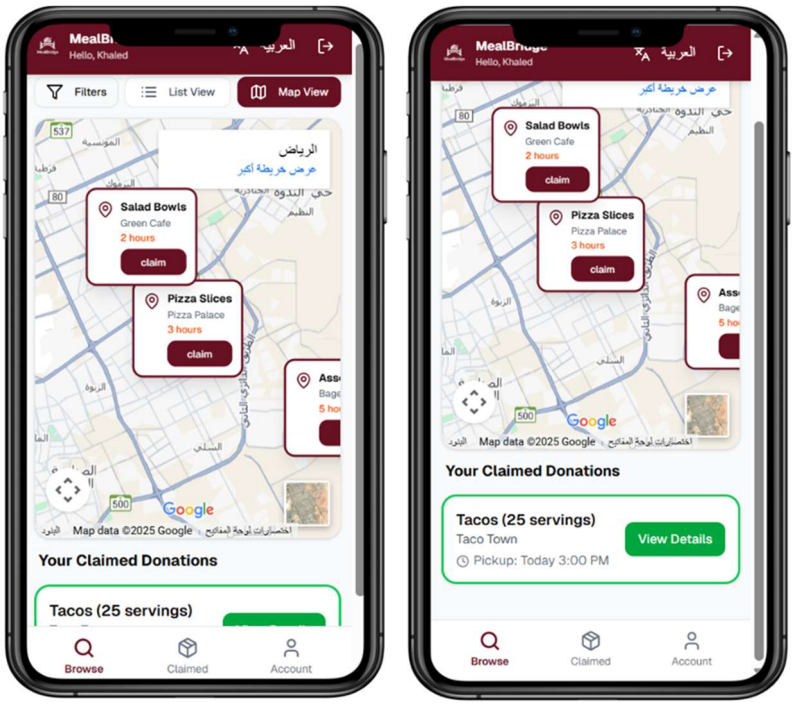
The Account Settings page lets users update their information, change passwords, delete their account, or download their data.

Recipient Interface:

Browse page

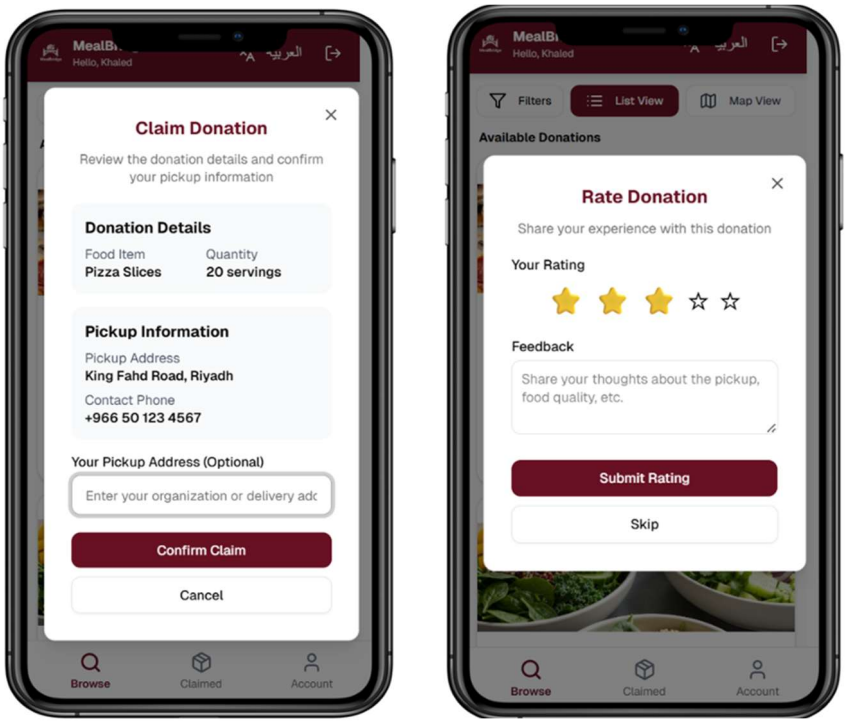


When registering as a recipient, the user is taken to the Browse page, where they can apply filters and view available donations based on Food Type, Location and Expiration date.



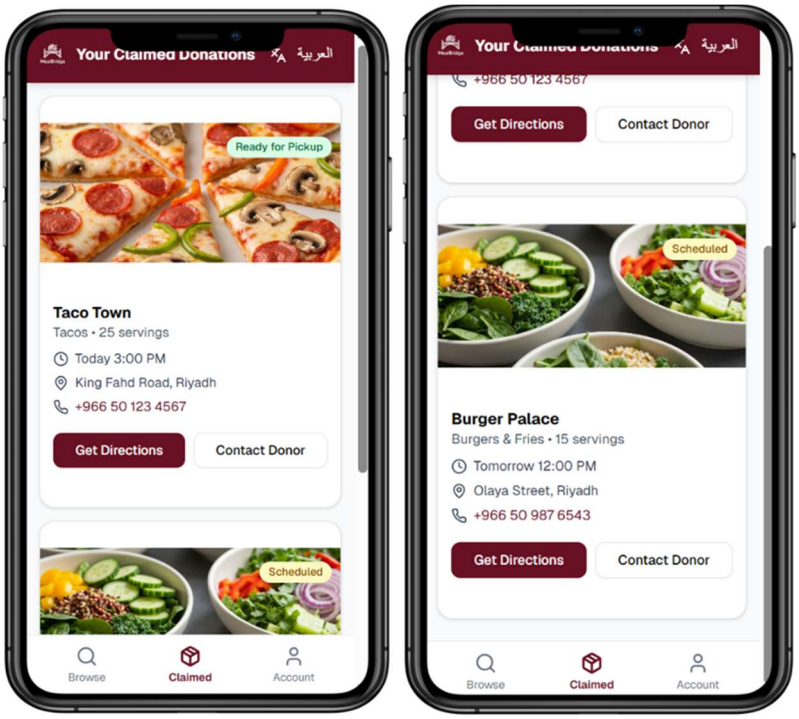
In Map View, nearby donations are displayed through Google Maps, allowing recipients to see locations clearly and get directions. The Browse page also shows recently claimed donations, with an option to view more details.

Claiming Donations process:



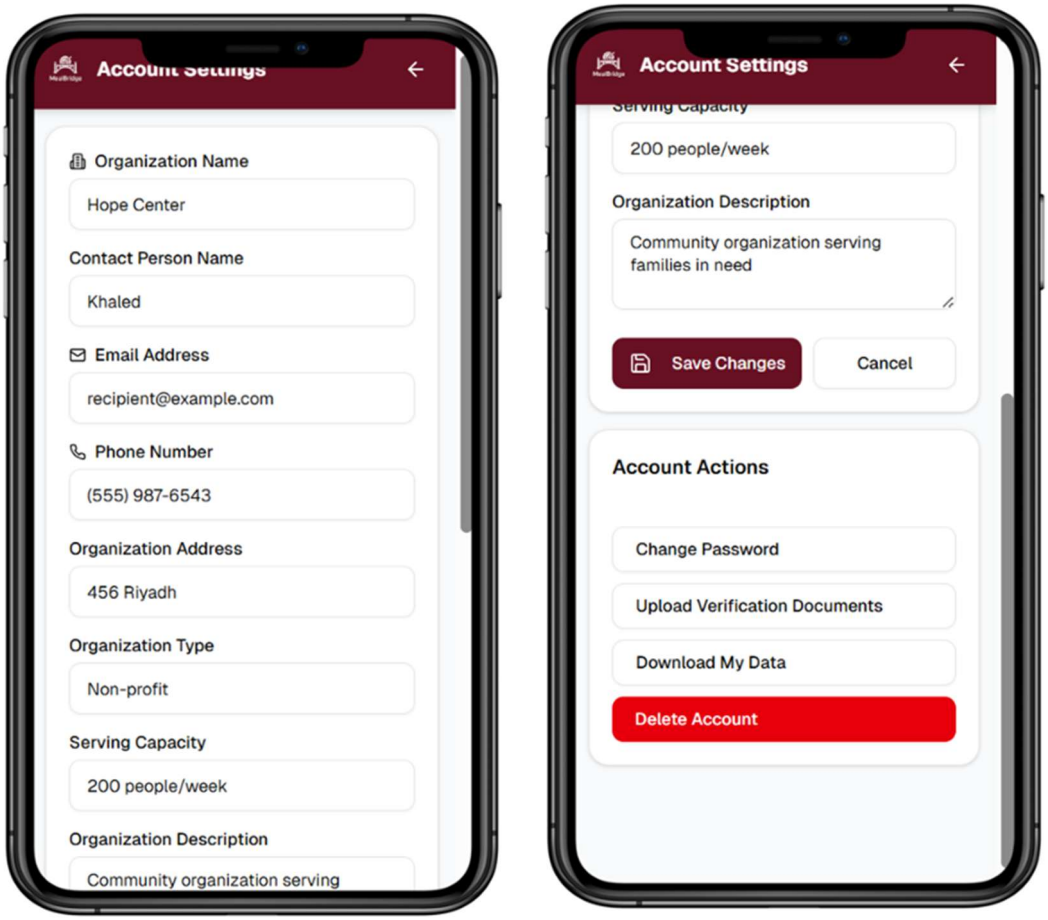
When selecting a donation from the List or Map View, a details window appears where the recipient can review the donation and enter a pickup address. After confirming the claim, the user can rate the donation and provide feedback.

Claimed page



The Claimed page lets users view their claimed donations and check their status Ready for Pickup or Scheduled, with options to contact the donor or get directions.

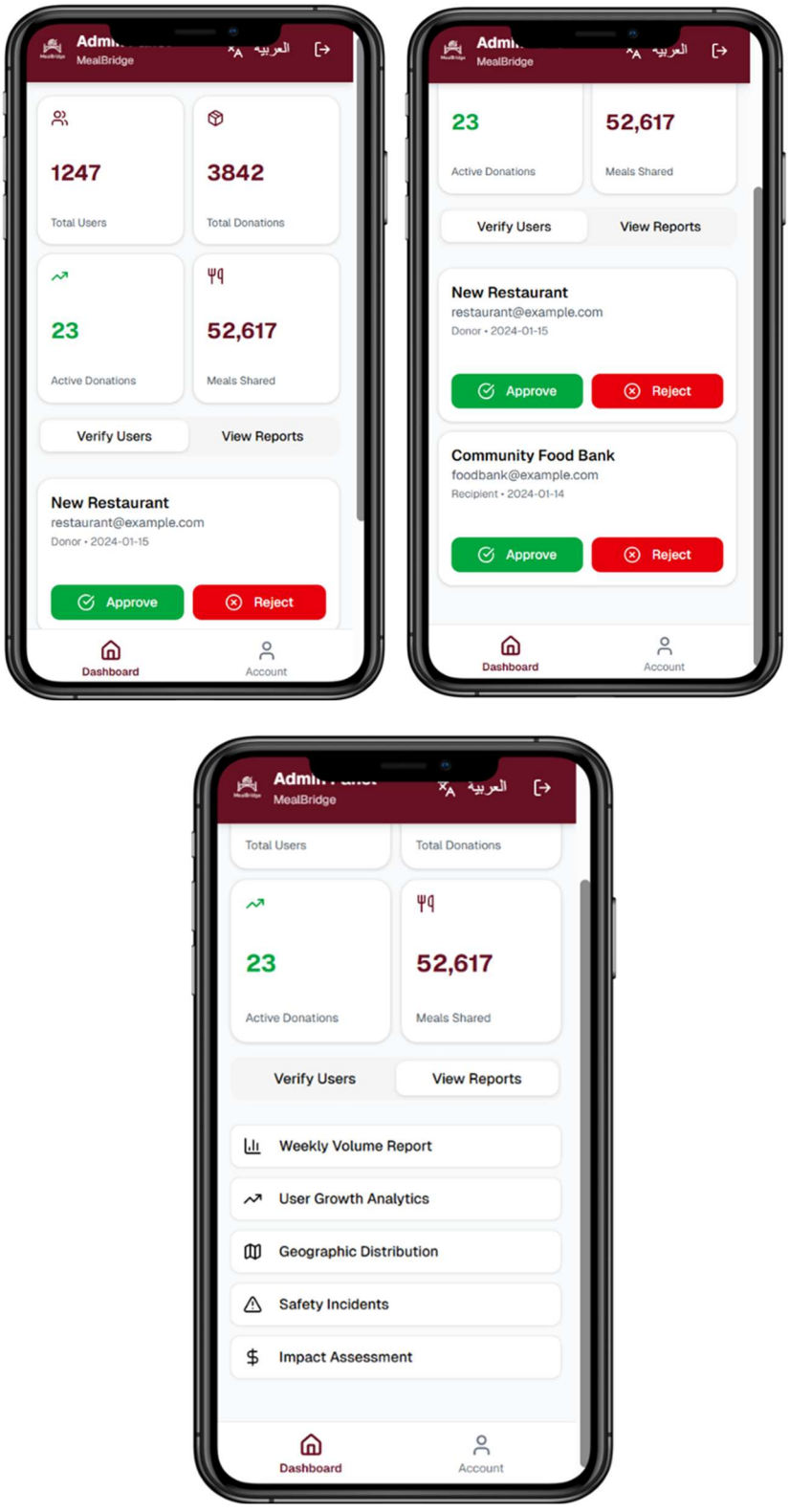
Account page



The Account Settings page lets users update their information, change passwords, delete their account, upload verification documents or download their data.

Admin Interface:

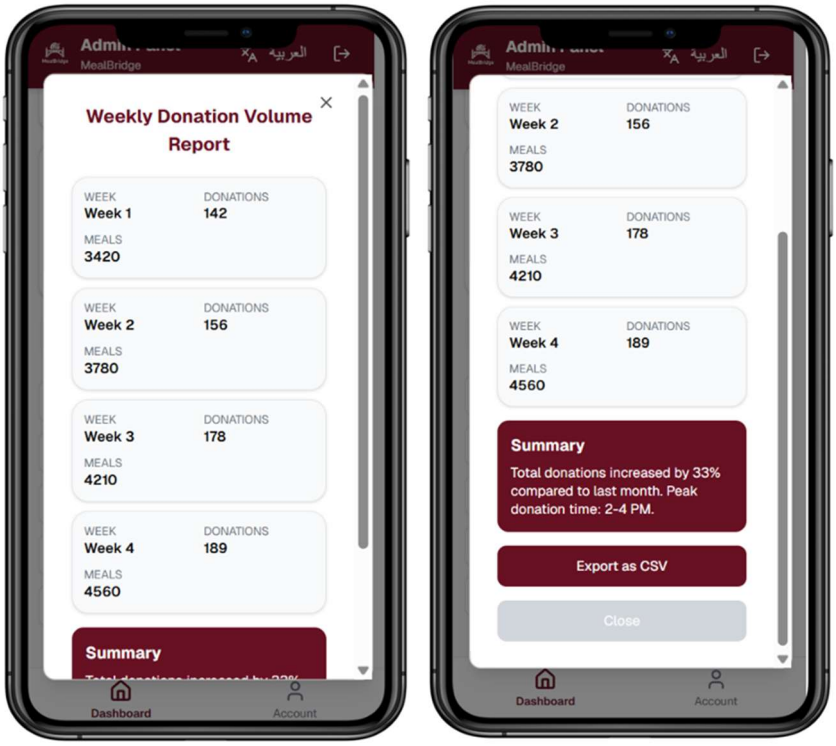
Dashboard page



They also can view different reports from view reports tab.

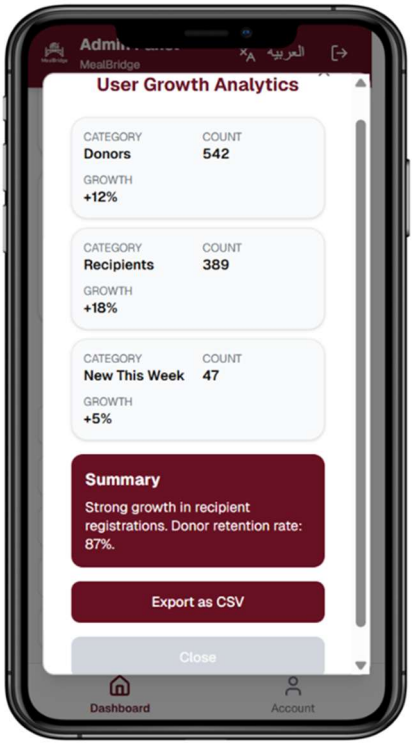
Reports on Admin Dashboard:

Weekly Volume Report



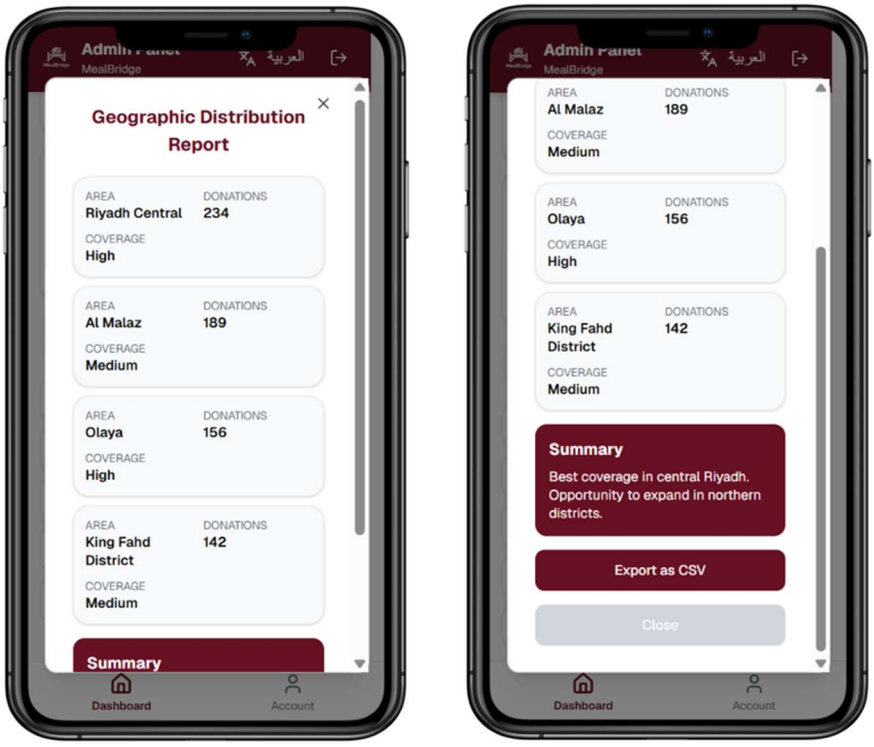
This report shows weekly totals for donations and meals, along with a summary of the figures, with the option to export the report as CSV.

User Growth Report



This report shows user growth with a summary of the figures with the option to export the report as CSV.

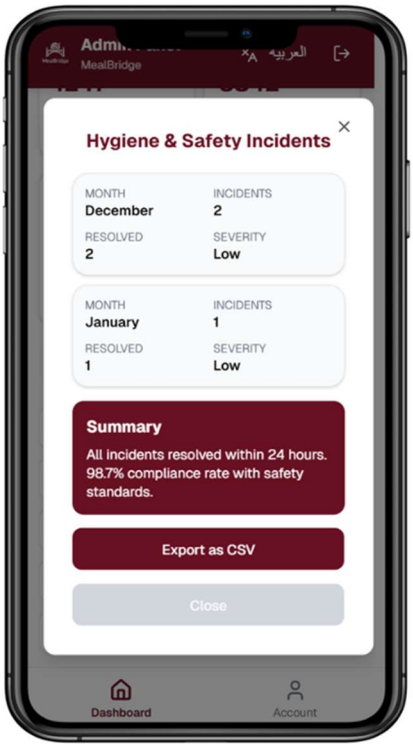
Geographic Distribution Report



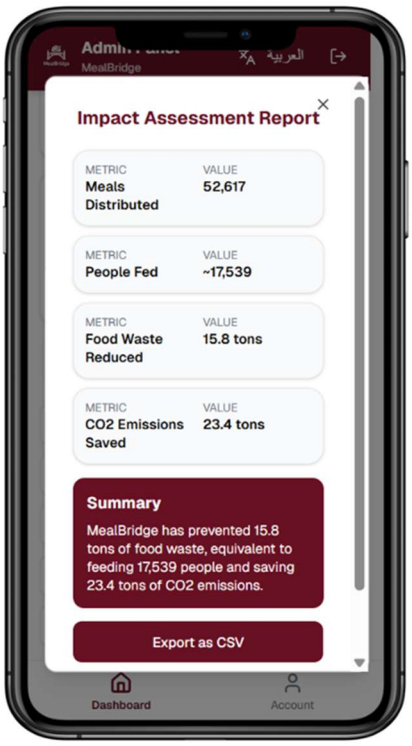
This report shows the geographic distribution of donations, detailing numbers and coverage for each area, with a summary of the figures and the option to export the report as CSV.

Safety Incidents Reports

This report shows safety incidents by month, providing details and a summary of each month's incidents with the option to export the report as CSV.

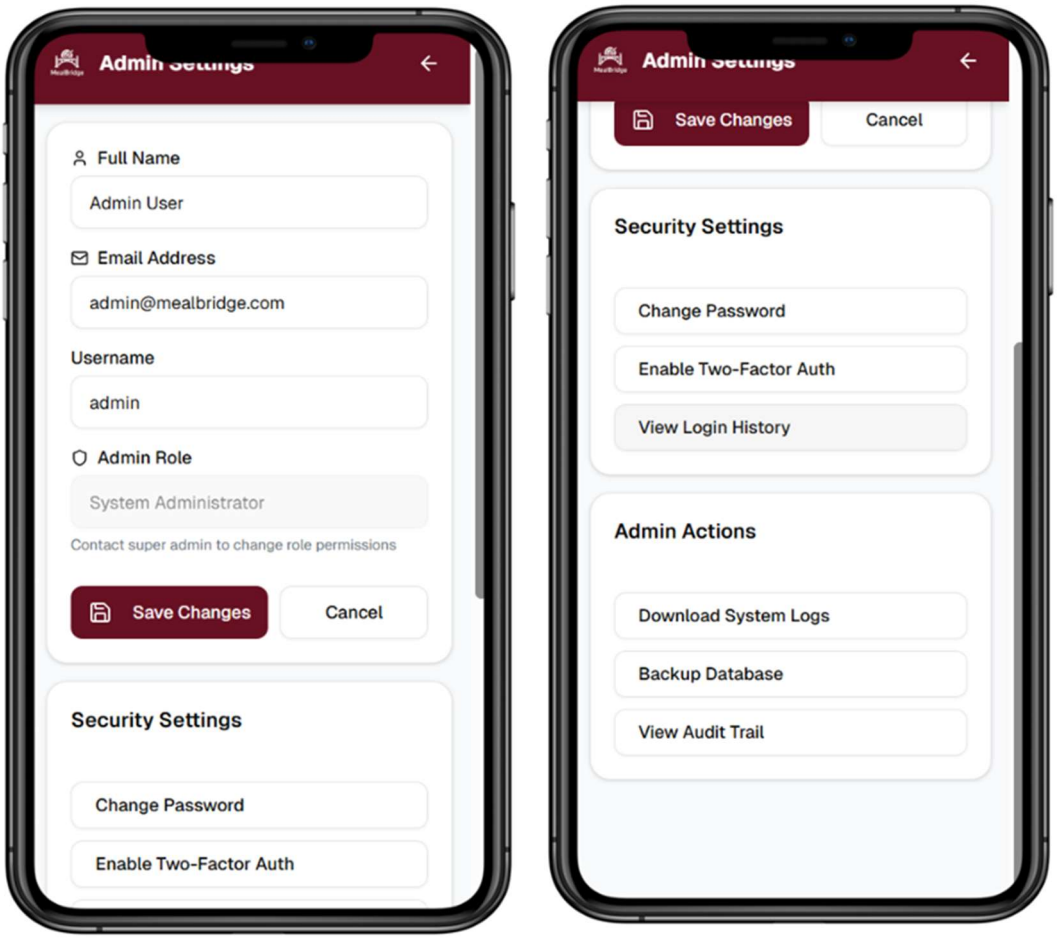


Impact Assessment Reports



This report shows the app's impact across metrics food waste, people fed, meals distributed, and CO_2 emissions saved along with a summary and the option to export the report as CSV.

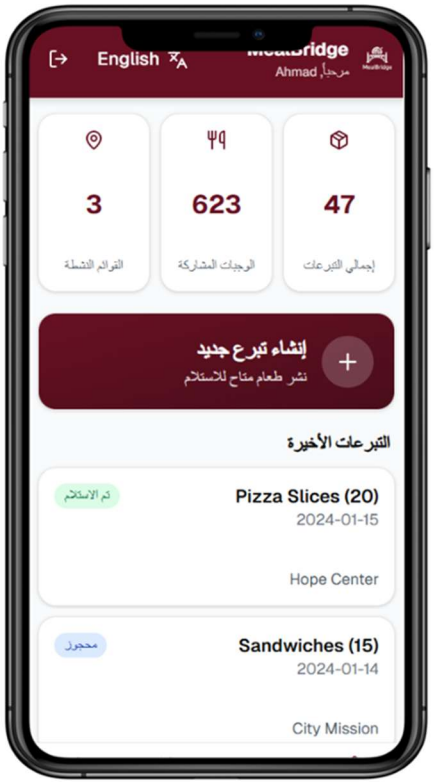
Admin Account Page

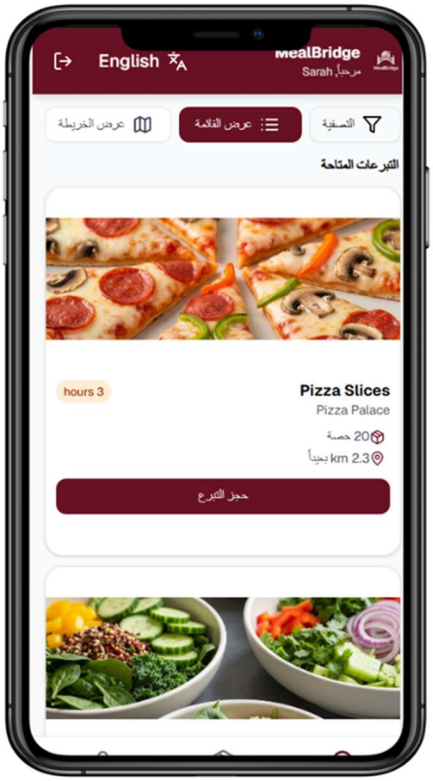


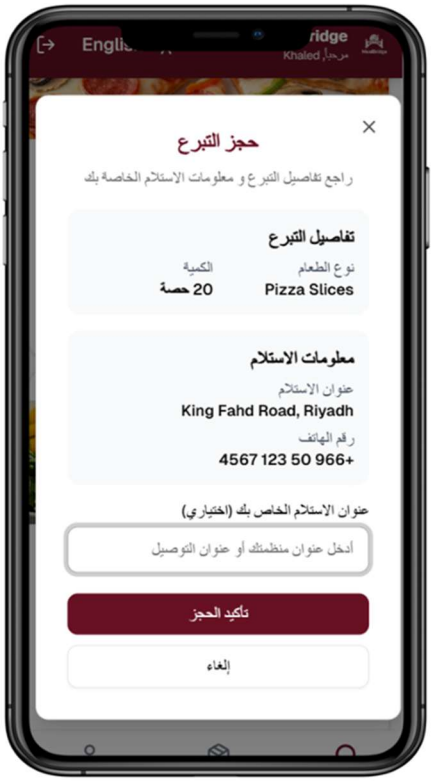
In Admin Account Settings, admins can update their account (except role, which requires a super admin), change passwords, and perform admin actions such as downloading system logs and backing up the database.

Arabic Language:

Our system supports Arabic; the following pages are shown in Arabic.







Contribution of each member:

Section	Student
Introduction	**Lena Alswed**
System Overview	**Lena Alswed**
Software Process Model	**Bedor Alharbi**
Architecture Design	**3.1 – Muzna Adelgadir** **3.2 – Maha Alrashidi** **3.3 – Balqees Almohsen**
Data Design	**Aroob Altuwajiri**
Component Design	**5.1 – Balqees Almohsen** **5.2 – Maha Alrashidi** **5.3 – Muzna Abdelgadir** **5.4 – Bedor Alharbi**
Human Interface Design	**Bedor Alharbi**