CS 383 – Group Project

MealBridge

Prepared by:

Bedor Alharbi - 432205469

Muzna Abdelgadir - 441211827

Maha Alrashidi - 441203676

Balqees Almohesn - 441203100

Aroob Altuwajiri - 441203113

Lena Alswed - 432205329

Presented to:

Dr. Rafaa Aljarbua

Section: 5220

Semester: 471

Submission Date: 30/11/2025

Table of Contents

Part 1

Project Management

This section documents the meeting schedule, agendas, and outcomes that demonstrate the team's project management and communication processes throughout the project lifecycle which guided its development.

MEETING AGENDA I

Date	12th of September, 2025
Time	11:30 – 11:40
Location	WhatsApp
Time Allocated	10 mins

Attendees

Present	**Absence**
Bedor Alharbi	-
Muzna Abdelgadir	-
Maha Alrashidi	-
Balqees Almohesn	-
Aroob Altuwajiri	-
Lena Alswed	-

Meeting Goal

Getting to know the team and nominating a leader.

Agenda Items

#	Topic	Description	Time
1	Member Introductions	Members introduced themselves and got to know each other.	5 mins
2	Leader Nomination	Members discussed and nominated a leader.	5 mins

Meeting Notes:

- Each member shared a brief introduction about themselves.
- Bedor Alharbi was nominated as leader.
- Next meeting will focus on sharing and discussing ideas.

Written by: Bedor Alharbi

MEETING AGENDA II

Date	26th of September, 2025
Time	16:45 – 17:15
Location	Discord server
Time Allocated	30 mins

Attendees

Present	Absence
Bedor Alharbi	-
Muzna Abdelgadir	-
Maha Alrashidi	-
Balqees Almohesn	-
Aroob Altuwajiri	-
Lena Alswed	-

Meeting Goal

Discuss project ideas and select the final idea to develop.

Agenda Items

#	Topic	Description	Time
1	Idea Discussion	Members suggested potential ideas to develop: Maha: MealBridge unites individuals, restaurants and cafes with charities and volunteers to donate leftover food and reduce waste. Balqees: Reservation management application designed to simplify booking processes for users.	20 mins
2	Idea Selection	After Discussion the team agreed to work with Maha's idea: MealBridge	10 mins

Meeting Notes:

- The team discussed the feasibility, creativity and alignment of both ideas with the project requirements.
- After evaluation, the team voted for Maha's idea.
- The next meeting will focus on assigning roles and discussing the project structure.

Written by: Muzna Abdelgadir

MEETING AGENDA III

Date	27th of October, 2025
Time	18:00 – 18:40
Location	Discord server
Time Allocated	40 mins

Attendees

Present	**Absence**
Bedor Alharbi	**-**
Muzna Abdelgadir	**-**
Maha Alrashidi	**-**
Balqees Almohesn	**-**
Aroob Altuwajiri	**-**
Lena Alswed	**-**

Meeting Goal

Discuss the SRS document structure and distribute sections among team members.

Agenda Items

#	Topic	Description	Time
1	SRS Discussion	The team discussed the required SRS structure and clarified the project requirements.	20 mins
2	Task Distribution	The work was divided as follows: Muzna: Introduction + Functional Requirements Aroob & Maha: Non-functional Requirements Balqees & Lena: Functional Requirements Bedor: External Interface Requirements	20 min

Meeting Notes:

- The team reviewed the SRS template and discussed each section's purpose.
- Workload was assigned equally based on team members' roles.
- The next meeting will focus on reviewing the written sections and ensuring consistency.

Written by: Balqees Almohsen

MEETING AGENDA IV

Date	4th of November, 2025
Time	18:00 – 19:00
Location	Discord server
Time Allocated	1 hour

Attendees

Present	**Absence**
Bedor Alharbi	**Lena Alswed**
Muzna Abdelgadir	**-**
Maha Alrashidi	**-**
Balqees Almohesn	**-**
Aroob Altuwajiri	**-**

Meeting Goal

Finalizing SRS Document.

Agenda Items

#	Topic	Description	Time
1	Project Status	Each team member presented their individual contributions, followed by a collaborative discussion to review and integrate the work.	15 mins
2	SRS Discussion	The team finalized the discussion on the Software Requirements Specification (SRS).	25 mins
3	Task Distribution	The work was divided as the following: Aroob: Product perspective & Product features Balqees: User roles and characteristics & Operating environment Maha: Design and implementation constraints & Assumptions and dependencies Muzna and Bedor: Requirements elicitation, Requirements analysis and Requirements validation Lena: UML use case diagrams	20 min

Meeting Notes:

- The team reviewed each member's work and discussed the finalized SRS document structure.
- Each section was assigned to the members responsible based on their role and progress.
- The next meeting will focus on reviewing the written sections, ensuring consistency in format and terminology, in addition to working on SDD.

Written by: Aroob Altuwajiri

MEETING AGENDA V

Date	15th of November, 2025
Time	18:00 – 18:45
Location	Discord server
Time Allocated	45 mins

Attendees

Present	Absence
Bedor Alharbi	**Lena Alswed**
Muzna Abdelgadir	**-**
Maha Alrashidi	**-**
Balqees Almohesn	**-**
Aroob Altuwajiri	**-**

Meeting Goal

Discuss the SDD document structure and distribute sections among team members.

Agenda Items

#	Topic	Description	Time
1	SDD Discussion	The team discussed the required SDD structure and clarified the project requirements.	25 mins
2	Task Distribution	The SDD work distribution was as follows: Lena: Introduction & System overview. Muzna: Architecture description & Activity diagrams. Maha: Decomposition description & State diagrams. Balqees: Design rationale & Class diagrams. Aroob: Data Design. Bedor: Human Interface Design & Sequence diagrams.	20 min

Meeting Notes:

- The team reviewed the SDD template and discussed the purpose of each design section.
- Workload was assigned to team members following a discussion and mutual agreement on the distribution of SDD sections.
- The next meeting will focus on finalizing the SDD document after all required revisions and adjustments are complete, and discussion on the project presentation.

Written by: Maha Alrashidi

MEETING AGENDA VI

Date	25th of November, 2025
Time	16:30 – 17:00
Location	Discord server
Time Allocated	20 min

Attendees

Present	Absence
Bedor Alharbi	-
Muzna Abdelgadir	-
Maha Alrashidi	-
Balqees Almohesn	-
Aroob Altuwajiri	-
Lena Alswed	-

Meeting Goal

Review the entire project progress and agree on final deadlines for the next meeting, the presentation, and the final project file.

Agenda Items

#	Topic	Description	Time
1	Full Work Review	The team reviewed all completed sections of the project and ensured overall consistency.	15 mins
2	Presentation Planning	The team discussed when and how the project presentation will be prepared.	10 mins
3	Final File Deadline	A final deadline was set for completing and submitting the final project file.	5 mins

Meeting Notes:

- The team reviewed the entire project and confirmed overall progress.
- The next meeting will focus on preparing and practicing the final presentation, as well as reviewing the final project file.
- Team members will bring any remaining questions or unclear parts to be resolved in the next meeting, and each team member will review another member's work.

Written by: Lena Alswed

MEETING AGENDA VII

Date	28th of November, 2025
Time	18:00– 18:25
Location	Discord server
Time Allocated	25 mins

Attendees

Present	Absence
Bedor Alharbi	-
Muzna Abdelgadir	-
Maha Alrashidi	-
Balqees Almohesn	-
Aroob Altuwajiri	-
Lena Alswed	-

Meeting Goal

Review the presentation and final project file as well as practicing the project's presentation in preparation for the final submission.

Agenda Items

#	Topic	Description	Time
1	Presentation Review	The team reviewed the entire presentation to ensure clarity, flow, and consistency.	10 min
2	Final Project File Check	The team went through the final project file and confirmed that all required sections are complete and aligned.	10 mins
3	Presentation Practice	The team practiced presenting the slides and discussed improvements for the final delivery.	5 mins

Meeting Notes:

- The team reviewed the full presentation and made necessary adjustments to improve quality and organization.
- All sections of the final project file were checked and approved.
- The team practiced delivering the presentation and agreed on the speaking order and key points for each member.
- Final corrections will be completed before the next meeting, where the final run-through of the presentation will take place.

Written by: Lena Alswed

MEETING AGENDA VIII

Date	30th of November, 2025
Time	16:00– 16:40
Location	Discord server
Time Allocated	40 mins

Attendees

Present	Absence
Bedor Alharbi	-
Muzna Abdelgadir	-
Maha Alrashidi	-
Balqees Almohesn	-
Aroob Altuwajiri	-
Lena Alswed	-

Meeting Goal

Practice presenting the project for the last time and ensure readiness for the discussion.

Agenda Items

#	Topic	Description	Time
1	Final Presentation Run-through	Each member practiced their part of the presentation, ensuring timing, clarity, and smooth transitions.	20 min
2	Final Adjustments & Q&A Preparation	The team made final modifications, reviewed potential discussion questions, and aligned on key talking points for the project defense.	20 mins

Meeting Notes:

- The team completed a full practice of the project presentation.
- Final improvements were made to slide transitions, speaking order, and delivery.
- Members discussed potential questions that could be asked during the project discussion and agreed on clear, consistent answers.
- Everyone confirmed they are ready for the final project discussion.

Written by: Bedor Alharbi

Part 2

Software Engineering

Ethics & Responsibility

This section outlines the ethical considerations and responsible practices adopted by the team throughout the "MealBridge" project, ensuring adherence to professional codes of conduct, specifically the principles of the Software Engineering Code of Ethics and Professional Practice (SEEPP) developed by the ACM/IEEE-CS Joint Task Force.

Before we begin, here is a small list of abbreviations that will be used in this section:

Abbreviation	Full Term
ACM	Association for Computing Machinery
EIR	External Interface Requirement
IEEE-CS	Institute of Electrical and Electronics Engineers Computer Society
NFR	Non-Functional Requirement
R-#	Requirement Number (Functional Requirement)
SEEPP	Software Engineering Code of Ethics and Professional Practice

Application of the SEEPP Principles

1. Public

The team's primary ethical obligation was to the well-being of the public, which guided the core mission of "MealBridge" to **connect people in need to donors, bridging the gap between surplus food and those in need.**

- **Prioritizing Safety and Welfare:** We implemented mandatory verification steps (R-10, Food Safety Assurance Constraint) to ensure all shared meals comply with food safety standards, directly protecting public health.
- **Accessibility and Reach:** The requirement for multi-language support (R-12) and compatibility with major browsers and device types (NFR 5.7, EIR-1) ensures maximum accessibility for a diverse user base.
- **Data Privacy for Vulnerable Users:** We prioritized the privacy of recipients, mandating encryption for sensitive information (addresses, phone numbers) when stored in the database (NFR 5.6).

2. Client and Employer

We maintained transparency and honesty with our stakeholders (project management and instructors).

- **Transparency of Limitations:** We documented clear dependencies, such as the reliance on external map services (EIR-3) and prompt recipient response times (Assumption 2.6.2), managing expectations about system capabilities.
- **Accurate Documentation:** The clear definition of requirements, from functional R-1 through R-12 to all EIRs, provided a precise and verifiable project scope to our client/employer.

3. Product

We aimed for a reliable, secure, and high-quality product suitable for its sensitive operational environment.

- **Security by Design:** We adhered to NFR 5.6 and EIR-4, mandating the use of HTTPS for all data in transit and encryption at rest, ensuring the confidentiality and integrity of user information.
- **Reliability and Performance:** The Performance Constraint (NFR 5.4), requiring actions within 3 seconds, was deemed an ethical requirement as well as a technical one, as speed is crucial for perishable food donations to reach recipients quickly before expiry.

4. Judgment

Sound ethical and technical judgment was exercised in all development decisions.

- **Balancing Speed and Safety:** A key judgment call was balancing quick donation matching (R-3) with necessary safety checks (R-10). We prioritized safety as a non-negotiable requirement, even if it added slight complexity to the donation flow (EIR-1).
- **Role-Based Access:** We used judgment to implement strict role-based access control to ensure data security and prevent misuse of the platform by unauthorized users (R-1, EIR-1).

5. Management

Our Agile management approach fostered an ethical and efficient working environment for the team.

- **Fair Resource Allocation:** Iteration planning ensured that technical challenges (e.g., integrating device hardware interfaces like GPS and camera, EIR-2) were assigned equitably to team members with appropriate expertise.
- **Responsible Monitoring:** The Admin interface (EIR-5) was designed with clear ethical boundaries for monitoring activity, focusing purely on verifying organization accounts and handling reports responsibly (R-9).

6. Profession

We upheld the integrity and reputation of the software engineering profession through adherence to standards and continuous learning.

- **Adherence to Standards:** The team adhered strictly to structured requirements engineering processes and used standardized communication protocols (HTTPS, WebSockets, EIR-4) to ensure professional rigor in system development.
- **Promoting Best Practices:** We adopted industry best practices for security and robust error handling (NFR 5.5) to build a resilient system.

7. Colleagues

We fostered a supportive and collaborative environment among team members.

- **Knowledge Sharing:** When navigating complex third-party software integrations (e.g., Google Maps API, Firebase, EIR-3), team members openly shared insights, ensuring the entire team benefited from shared knowledge.
- **Constructive Feedback:** Peer review sessions focused on improving the quality of the product, such as refining the 3-step donation flow (EIR-1) or implementing multi-language support (R-12), without personal conflict.

8. Self

Each team member committed to personal growth and maintaining high ethical standards.

- **Continuous Improvement:** The Agile framework's continuous improvement cycles allowed for regular self-reflection on the ethical implications of our design choices, such as how the automated matching system (R-3) impacts different user groups.
- **Accountability:** Team members took ownership of their functional requirements (R-1 through R-12) and external interface requirements (EIR-1 through EIR-5), ensuring all technical aspects were handled responsibly.

Part 3

Software Process Activities & Models

This section documents the specific software development life cycle (SDLC) model chosen for the project, detailing how its associated activities such as planning, design, implementation, testing, and deployment were applied to manage the project effectively.

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.

Part 4

Software Requirements

This section details the functional and non-functional requirements gathered and specified for the application. It outlines the methodologies used to define stakeholder needs and translate them into a clear, measurable set of requirements that guided the development process.

Introduction

1.1. Purpose

The purpose of this document is to provide a detailed Software Requirements Specification (SRS) for the MealBridge application. This document outlines the functional and non-functional requirements, serving as a comprehensive guide for the development, testing, and deployment teams. The MealBridge application aims to address the issue of food waste by creating a cross-platform solution that efficiently connects surplus food sources with those in need.

1.2. Product scope

MealBridge is a cross-platform software that will serve as a technological bridge to connect surplus food donors with beneficiaries (e.g., charities, non-profit organizations, and volunteers). The system will enable donors, including restaurants, bakeries, and individuals, to post available surplus food. It will also allow beneficiaries, such as charities and volunteers, to access this information to organize and facilitate the donation process. The primary objective is to reduce food waste and support communities by providing an effective tool for managing food donations.

1.3. References

- **Benchmarking of Existing Food Donation Applications**:
  Analysis of similar regional applications including *بركة*, *لتدوم*, and *حفظ النعمة* to understand common features, user flows, and best practices in donation and distribution systems.
- **https://www.lucidchart.com/pages/tutorial/uml-use-case-diagram**

1.4. Structure

This document is organized into five main sections:

- Introduction: Provides an overview of the project, its purpose, and the scope of the application.
- System Overview: Provides a high-level description of the product, its features, user classes, and operating environment.
- Requirements Engineering: Details of the methodology used to gather and analyze requirements.
- Functional Requirements: Specifies what the system must do.
- Non-functional Requirements: Defines quality attributes and constraints.
- External User Interface: Describes the user, hardware, software, and communication interfaces.

All requirements are uniquely identified for traceability.

System Overview

MealBridge is a comprehensive digital platform designed to tackle the dual challenges of food waste and food insecurity within communities. The system serves as a critical link between food donors such as restaurants, businesses, and individuals and verified charitable organizations. By leveraging technology to automate the donation matching and management process, MealBridge aims to ensure that surplus food is redirected quickly, safely, and efficiently to those in need. This section provides a detailed overview of the MealBridge system.

2.1. Product perspective

MealBridge is a digital donation management system connecting food donors with charitable organizations. It operates as an independent platform accessible across different devices and relies on external services for location matching and secure cloud-based data management. The system automates the donation process, including matching, communication, and tracking, to ensure efficiency, transparency, and safety. It forms part of a broader social initiative aimed at reducing food waste and promoting community welfare through technology. MealBridge is designed to be scalable, reliable, secure, and easy to use for all users.

2.2. Product features

The main features of MealBridge include the following:

1. User Registration and Authentication: Secure sign-up and login for donors, charities, and administrators. Role-based access to ensure appropriate permissions for each user type.

2. Donation Creation and Management: Donors can create donation posts including meal details, quantity, expiry date, and pickup location. Option to upload food photos and verify expiry for safety.

3. Automated Matching System: the system automatically connects donors with nearby charities based on real-time location data. o Notifications are sent instantly to both parties when a match occurs.

4. Donation Tracking and Confirmation: Real-time status updates for donation progress (pending, matched, collected, completed). o Charities confirm pickup and receipt through the app.

5. User Profiles and Ratings: Each user can view and update their profile. Feedback and rating system for transparency and trust building.

2.3. User roles and characteristics

Donors:

Individuals, families, or restaurants that have extra meals to donate. They generally have basic technical skills and require a simple and intuitive interface. Their primary goal is to submit donation requests easily and quickly.

Charity Organizations / Volunteer Teams:

Authorized organizations responsible for receiving and distributing donated meals. They typically have moderate technical experience and require features that support reviewing requests, scheduling pickup times, and updating donation statuses.

System Administrator:

A technically experienced user responsible for managing the application's backend activities. This includes verifying organization accounts, monitoring system performance, handling reports, and managing data integrity.

2.4. Operating environment

- The system will operate as a mobile application available on both Android and iOS platforms.
- Users are required to have internet connectivity to submit and receive real-time updates on donation requests.
- The application will rely on cloud-based storage to store user accounts, donation details, and communication logs securely.
- The system will run on devices with standard hardware specifications and will support a wide range of smartphone models.

2.5. Design and implementation constraints

Mobile Platform Requirement
The system must be developed as dedicated, high-performance applications for mobile devices (smartphones). This is a mandatory constraint because the app needs guaranteed, fast access to device features like GPS and the camera to support core functions such as location matching and photo uploads.

Performance Constraint
All essential user actions, including user login, new donation creation, and the automated matching process, must be completed within 3 seconds. The system must also be capable of supporting at least 100 simultaneous users without performance degradation.

Security and Data Protection Constraint
The system is strictly required to use secure, encrypted connections (like HTTPS) for all data sent over the internet. Furthermore, sensitive user information, specifically recipient addresses and phone numbers, must be protected by encryption when stored in the database.

Food Safety Assurance Constraint
The system design must include mandatory verification steps (such as required photo uploads and confirmation of expiry dates) during the donation process to ensure the quality and safety of the donated food.

2.6. Assumptions and dependencies

2.6.1. Assumptions

User Willingness and Participation: It is assumed that there is a sufficient and active user base of corporate/individual donors and verified charity organizations willing to use the application in the initial target operating area.

User Technical Proficiency: It is assumed that users possess a smartphone with an active internet connection and have the basic ability to use mobile applications to create or claim donations.

Data Integrity and Honesty: It is assumed that donors will accurately and truthfully provide information regarding the food safety, preparation time, and expiry dates of the donated meals.

2.6.2. Dependencies

Reliance on Recipient Response: The fast delivery of donated food depends heavily on the charity organizations or volunteers responding quickly to the match notification and being ready to pick up the food immediately. If they are slow, the system's goal of reducing food waste will be hard to achieve.

Need for Reliable Map Services: The core functions of the app, such as finding the closest charity to a donation and calculating the travel distance, entirely rely on the accurate performance of external map services. If the map service fails or provides bad data, the matching system will not work correctly.

External Verification for Charities: To keep the platform trustworthy, the system needs an official, external method to check and confirm that the charity organizations joining the app are legitimate.

Requirements Engineering

This section outlines the process of identifying, analyzing, and validating the requirements for the MealBridge application. By using a combination of structured techniques and user-centered methods, the team ensured that the system's features align with real user needs and the project's overall objectives. The following subsections describe how requirements were gathered, refined, and confirmed to form a solid foundation for the development of MealBridge.

3.1 Requirements Elicitation

To collect and define the requirements of *MealBridge*, several techniques were used:

1. **Brainstorming Sessions:**
   The project team held multiple brainstorming sessions to generate ideas about the app's main features and functionality. During these sessions, we discussed how the system can effectively connect food donors (restaurants, bakeries, individuals) with receivers (charities and volunteers). The main goal was to identify what each user type needs and how to make the process simple and efficient.
2. **Observation of Similar Applications:**
   We observed existing applications and websites that focus on food donation and waste reduction (e.g., *لتدوم, بركة, حفظ النعمة*) to understand their strengths, weaknesses, and user experience designs. This helped us identify common features such as donor listings, pickup scheduling, and notification systems, and notice areas where we could improve, such as simplifying the registration process and supporting Arabic language.
3. **Interviews and Informal Discussions (4–6 participants):**
   Short interviews and informal discussions were conducted with 4–6 potential users to gather insights about their expectations and challenges regarding food donation and collection. These conversations helped the team understand the basic needs of both donors and receivers, as well as identify key features that would make the app simple and practical to use.

These methods allowed us to gather both functional and non-functional requirements to guide the development of *MealBridge*.

3.2 Requirements Analysis

After collecting the requirements, the team analyzed and categorized them to ensure clarity and consistency:

- **Classification:** Requirements were divided into functional and non-functional categories.
- **Feasibility Check:** Each requirement was reviewed to determine whether it could realistically be implemented within the project's scope, time, and technical constraints.
- **Conflict Resolution:** Any overlapping or conflicting requirements from different stakeholders were discussed and refined during team meetings.
- **Prioritization:** The most essential features such as donor registration, listing donations, and notifying nearby charities were given higher priority, while secondary features (like donation history) were marked for future implementation.

3.3 Requirements Validation

To confirm that the gathered requirements accurately represent user needs:

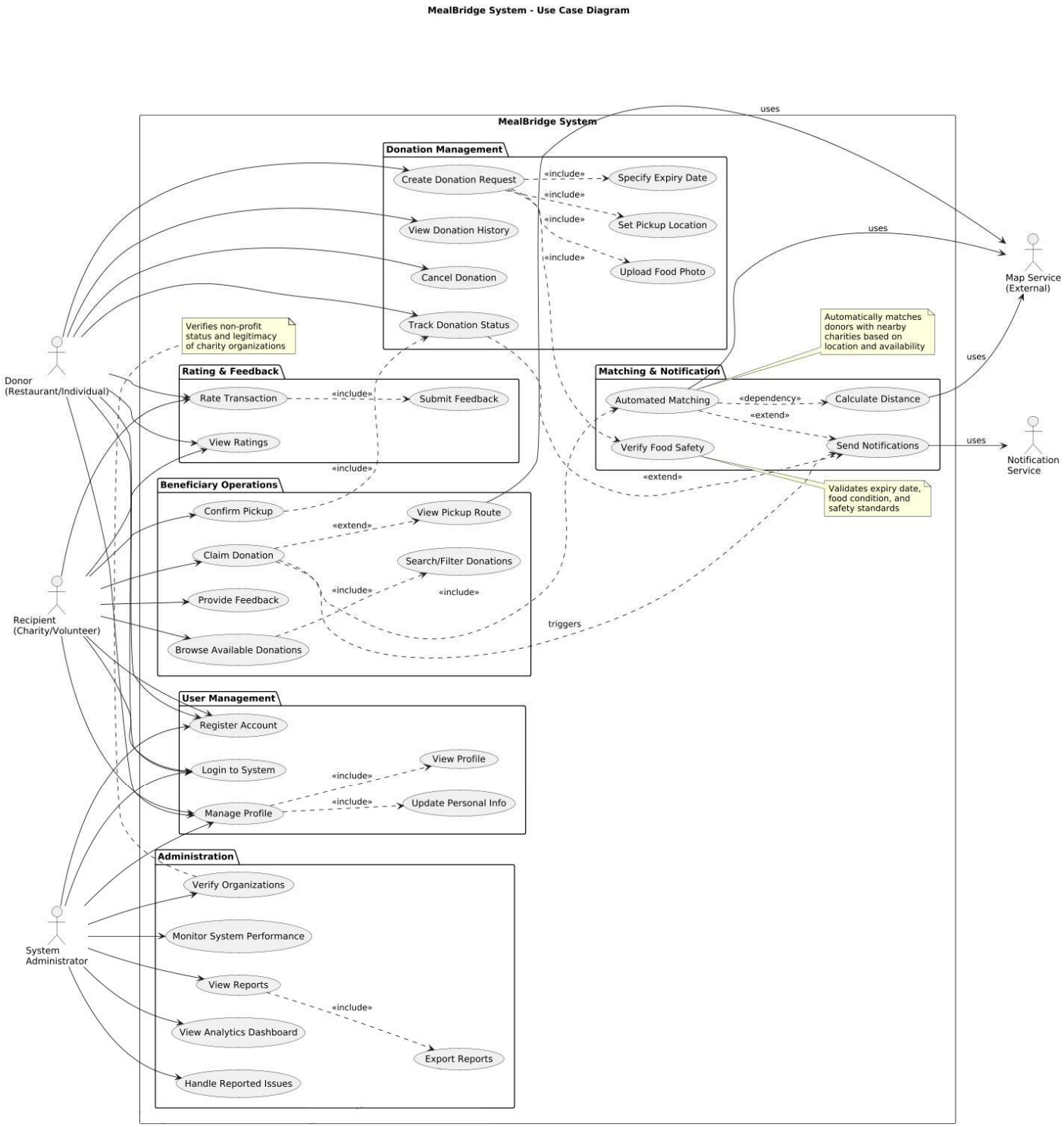
- **Team Review:**
  The team collaboratively examined all requirements to ensure they were understandable, complete, and aligned with the project scope.

- **Requirements Checklist Review:**
  The team validated requirements using a checklist focusing on completeness, clarity, consistency, testability, and feasibility.

- **Cross-Verification with Similar Systems:**
  Requirements were compared with features from existing food donation platforms to ensure they align with industry norms and avoid missing essential functions.

- **Goal Alignment Check:**
  Each requirement was evaluated against MealBridge's main objective reducing food waste and efficiently connecting donors with recipients to confirm it contributes directly to the system's purpose.

UML Use Case Diagram



Functional Requirements

R-1: User Registration and Login

Description: The system shall allow users to register and log in using their credentials.
Rationale: To ensure authorized access and protect user data.
Priority: High.

R-2: Donation Request Creation

Description: The system shall enable donors to create donation requests by specifying meal details, quantity, and pickup location.
Rationale: To allow donors to share excess meals easily and efficiently.
Priority: High.

R-3: Automated Matching System

Description: The system shall automatically match donation requests with nearby charity organizations and notify both parties.
Rationale: To reduce manual coordination and ensure efficient food distribution.
Priority: Medium.

R-4: Donation Tracking and Updates

Description: The system shall provide real-time tracking for donation status and send updates to both donor and receiver.
Rationale: To enable users to easily track the donation process and improve the experience for both parties.
Priority: Medium.

R-5: User Profile Management

Description: The system shall allow users to create, view, and edit their profiles. For donors (businesses/individuals), this shall include organization name, address, and food types typically offered. For beneficiaries (charities/volunteers), this shall include organization name, address, proof of non-profit status (optional but encouraged), and types of food needed.
Rationale: To build user identity and trust within the platform, and to enable better, more relevant matching.
Priority: High.

R-6: Beneficiary Request Management

Description: The system shall allow beneficiaries to view available donations and make claim donation requests.
Rationale: To ensure a smooth and verifiable food handover process.
Priority: High.

R-7: Search and Filter Donations

Description: The system shall allow beneficiaries to search and filter available donation requests based on criteria such as food type, quantity, location proximity, and pickup time.
Rationale: To help beneficiaries quickly find suitable donations that meet their specific needs and capacity.
Priority: High.

R-8: Notification Management

Description: The system shall send push notifications and/or email alerts for new donation matches, pickup reminders, status updates, and system announcements. Users shall be able to customize notification preferences.
Rationale: To keep users informed in real-time and reduce missed donations while allowing users to control notification frequency.
Priority: Medium.

R-9: Reporting and Analytics

Description: The system shall provide an admin dashboard with reports showing key metrics such as total meals donated, top donors, active beneficiaries, and food waste reduced (in meals).
Rationale: To measure the impact of the platform, provide valuable insights to stakeholders, and identify areas for improvement.
Priority: Low.

R-10: Food Safety and Expiry Verification

Description: The system shall allow donors to input the expiry date and food condition details when submitting a donation. The system shall automatically verify that the donation meets predefined safety criteria (e.g., not expired, prepared within safe time limits).
Rationale: To ensure all shared meals comply with food safety standards and protect beneficiaries' health.
Priority: High.

R-11: Feedback and Rating System

Description: The system shall allow both donors and beneficiaries to provide feedback and rate their experience after each completed donation transaction.
Rationale: To promote accountability, transparency, and continuous improvement in the donation process.
Priority: Medium.

Non-Functional Requirements

5.4. Performance Requirements

Description: The system shall process user requests (e.g., login, donation creation, and matching) within an average of **3 seconds**. It shall support at least **100 concurrent users** without noticeable performance degradation.
Rationale: To ensure fast and responsive interaction for all users, improving user satisfaction.
Priority: High.

5.5. Safety Requirements

Description:
The system shall include automatic **data backup** and recovery mechanisms to prevent data loss in case of server failure. It shall also prevent accidental deletion of important records by requiring confirmation before removal.
Rationale: To maintain data integrity and prevent data loss due to errors or system crashes.
Priority: Medium.

Description: The system shall detect common runtime errors (e.g., failed form submissions, invalid inputs) and provide clear, user-friendly error messages. It shall also automatically retry critical operations when safe to do so.
Rationale: To prevent user frustration, reduce data loss, and ensure a smooth recovery from unexpected failures.
Priority: High.

5.6. Security Requirements

Description: The system must protect sensitive recipient information (such as addresses and phone numbers) by encrypting it when stored in the database. It must also use a secure connection (like HTTPS) when transferring data over the internet.
Rationale: To protect user privacy and prevent unauthorized access to personal information.
Priority: Critical.

5.7. Software Quality Attributes

Description: The system shall be compatible with major web browsers (Google Chrome, Safari and Edge) and support both desktop and mobile devices. The web interface shall be responsive and maintain full functionality on various screen sizes.
Rationale: To ensure accessibility and consistent experience for all users regardless of their device or platform.
Priority: Medium.

Description: The system shall support Arabic and English languages for both the interface and notifications, allowing users to switch their preferred language at any time.
Rationale: To increase accessibility and usability for users with different language preferences.
Priority: Medium.

External Interface Requirements

1.1. EIR-1: User Interface Requirements

Description:
MealBridge shall provide a responsive, user-friendly interface for mobile and web users with role-specific views for Donors, Recipients (charities/volunteers), and Administrators.
Requirements:

- The system shall provide native mobile apps (Android & iOS) and a responsive web interface that adapts to screen widths from 320px to 1920px.
- The UI shall support Arabic and English; users must be able to switch language.
- The UI shall present role-based navigation: Donor flows (Create Donation, View History), Recipient flows (Browse Donations, Claim), and Admin flows (Verify Users, Reports).
- The donation reporting flow (create donation → upload photo → set pickup info → confirm) shall be completable within three main steps or screens for usability.

1.2. EIR-2: Hardware Interface Requirements

Description:
MealBridge shall interface with mobile device hardware to capture location, images, and scan/confirm donation pickups.
Requirements:

- The mobile app shall request and use GPS/location services to detect user location (with user consent) for proximity matching and route suggestions.
- The mobile apps shall access the device camera and photo library to allow donors to attach images of donations.
- Works on Android and iOS devices with internet access.
- The app shall use device notification hardware (sound, vibration) to alert users of urgent pickup requests or confirmations.

1.3. EIR-3: Software Interface

Description:
MealBridge will integrate with third-party services for maps, authentication, storage, and notifications to provide core functionality.
Requirements:

- The backend shall be integrated with a mapping service (e.g., Google Maps API) to display maps, compute distances, and estimate routes and ETA.
- The system shall use a secure authentication provider (e.g., Firebase Authentication, OAuth 2.0 with Google/Apple sign-in) for account management.
- The system shall store images and large objects in cloud storage and record metadata in a cloud database service (e.g., Firestore)
- The system shall support push notifications using platform services.

1.4. EIR-4: Communication Interface Requirements

Description:
MealBridge shall use secure, standardized communication protocols between clients, backend services, and external APIs.
Requirements:

- All communication between the app and the server, and between the server and other services, shall be secure using standard internet encryption (HTTPS).
- The system shall provide a way for the app to send and receive information from the server using standard web requests (e.g., sending donation details, updating profiles, or performing admin tasks).
- Real-time updates (e.g., live donation availability, delivery tracking) shall be implemented via WebSocket or Firebase Realtime mechanisms.
- The system shall make sure that only authorized users can access their accounts and limit excessive requests to prevent misuse.

1.5. EIR-5: System/Partner Interfaces Requirements

Description:
MealBridge shall provide mechanisms for partner charities, admin staff, and third-party systems to exchange data for reporting, verification, and bulk integration.
Requirements:

- The system shall provide an Admin web panel that interfaces with the backend for user verification, donation approvals, and analytics dashboards.
- Authorized partner organizations shall be able to pull or push data via a secured API (endpoints for donation status, pickup confirmations, and distribution results) using API keys and role-based access.
- The system shall support exporting reports in CSV formats (e.g., weekly donation volume, source breakdown, hygiene incidents) for partner review and regulatory needs.

Part 5

Software Modeling

This section presents the various models and diagrams utilized to visualize the system's design, architecture, and behavior. These models served as essential blueprints for understanding the system's structure and facilitating the implementation phase.

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.

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.

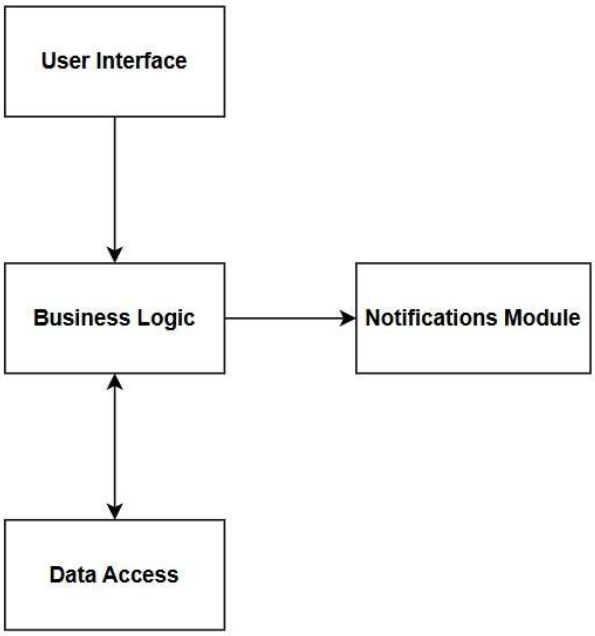
4. Notifications Module:

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. Business Logic is the central point for initiating any external actions, such as triggering the Notifications Module.



2.3. 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.

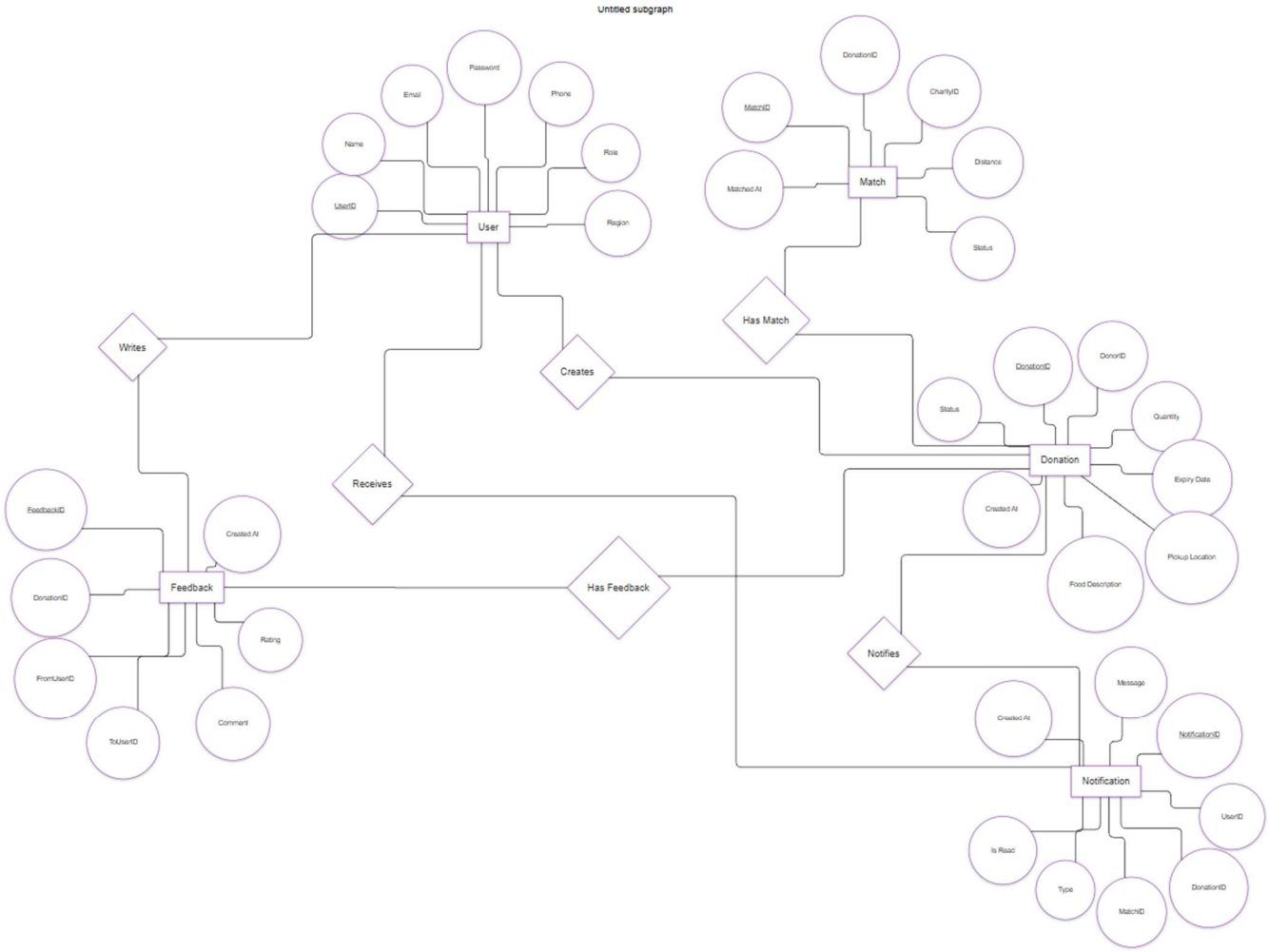
Donation:
Represents each food donation created by a donor.
Key attributes include: DonationID, DonorID, FoodDescription, Quantity, ExpiryDate, PickupLocation, Status, and CreatedAt.
DonorID is a foreign key referencing User(UserID), linking each donation to its owner.
A donation can be matched to one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

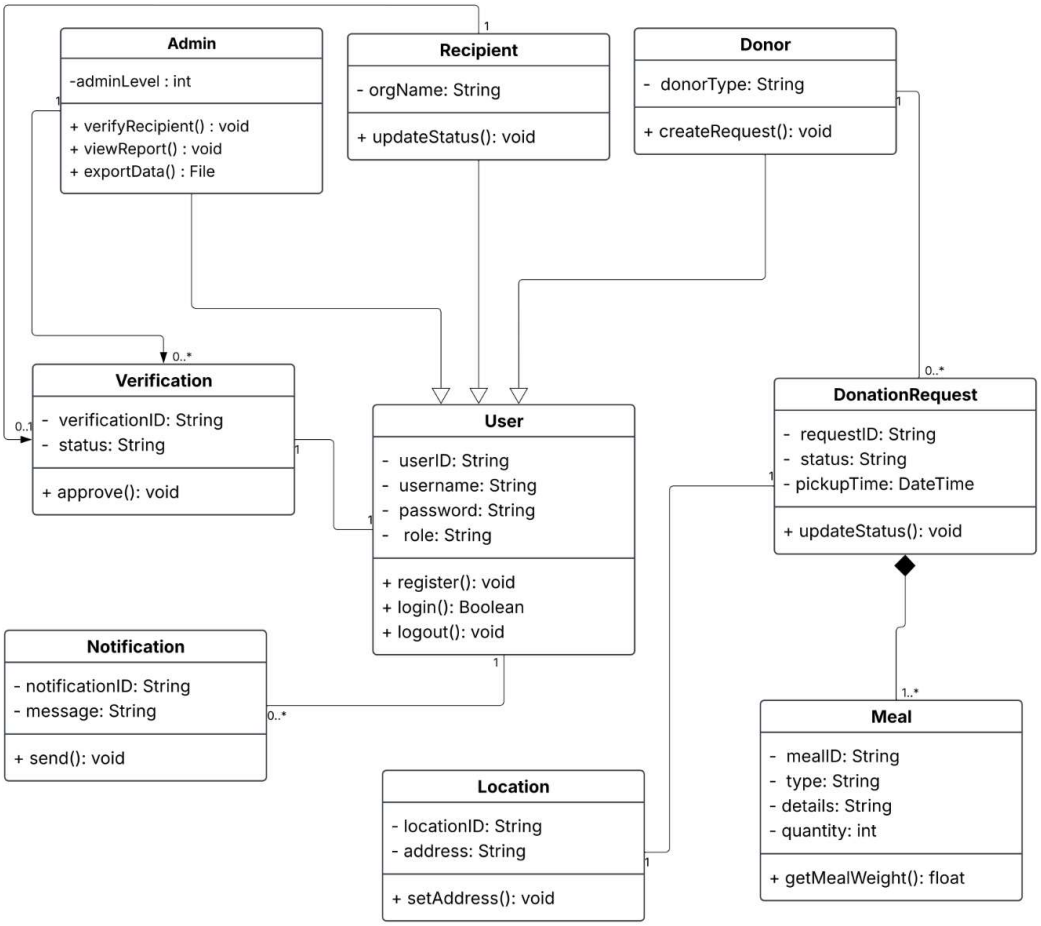
Feedback:
Stores user ratings and comments after a donation has been completed.
Attributes: FeedbackID, DonationID, FromUserID, ToUserID, Rating, Comment, and CreatedAt.
DonationID references the donation being evaluated, while FromUserID and ToUserID reference the users who give and receive the feedback.
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



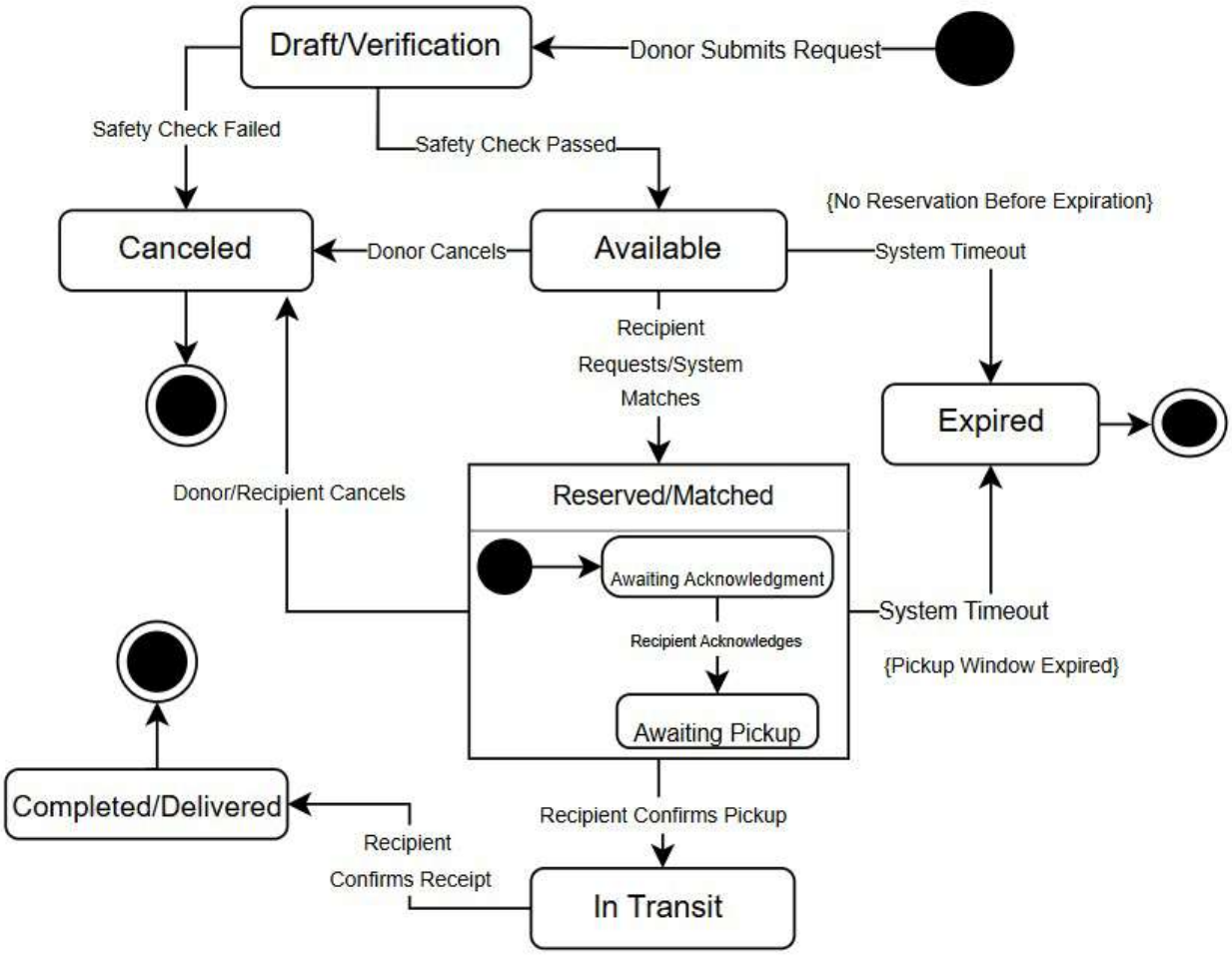
Component Design

5.1. Class diagrams



5.2. State Diagram

Creating a donation



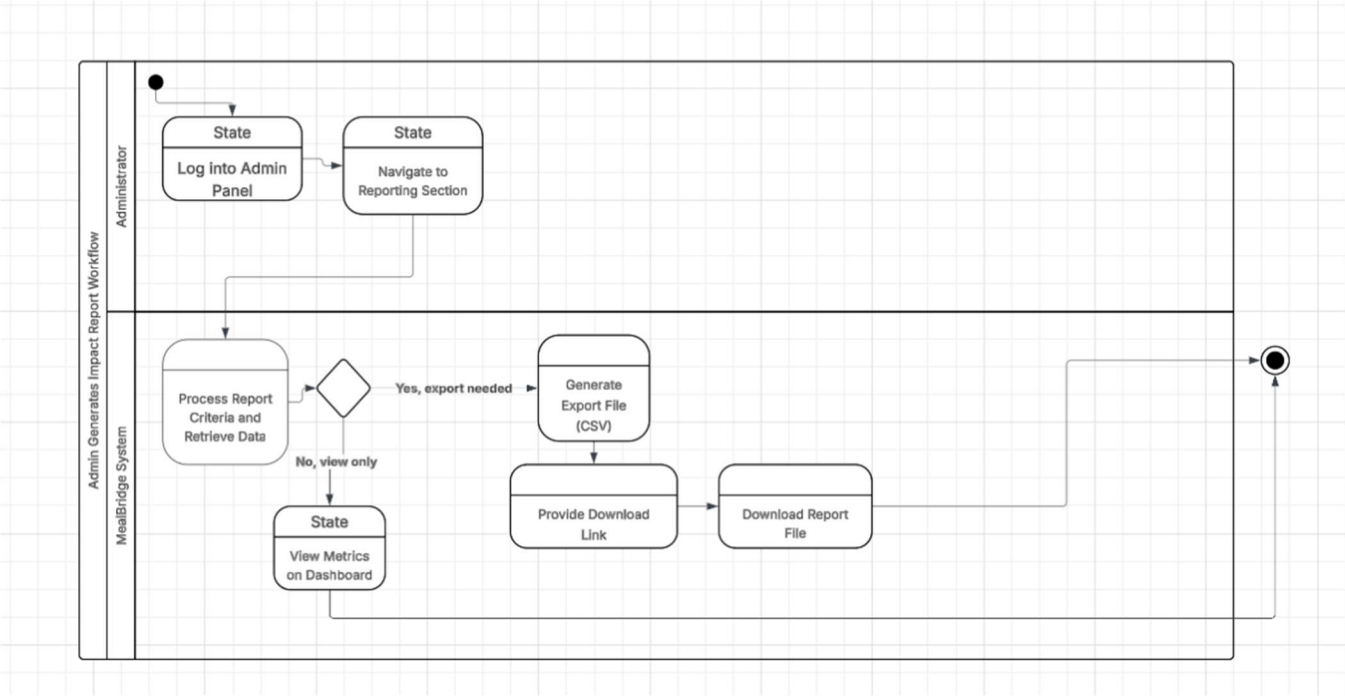
5.3. Activity Diagram

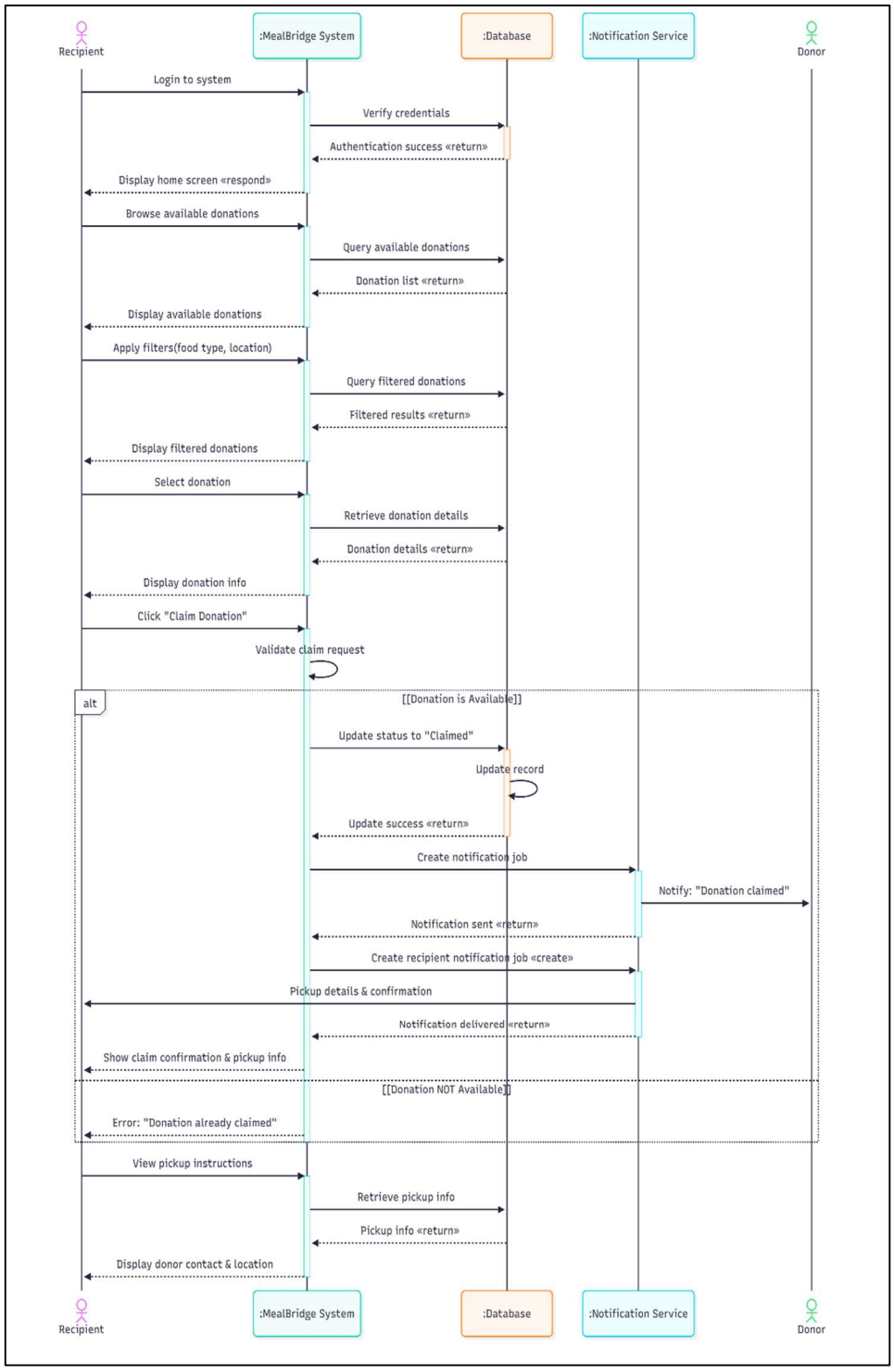
This activity diagram illustrates the workflow for the Administrator generating an impact report within the MealBridge system.

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 (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://mealbridgeappver.netlify.app/

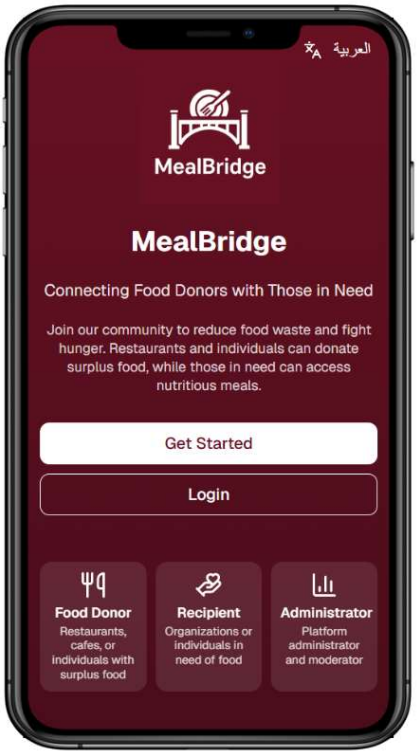
Web Version Preview (This version is the same as the app but adjusted for a web-friendly experience)**:**

https://mealbridgewebver.netlify.app/

6.2. Detail design of User Interface

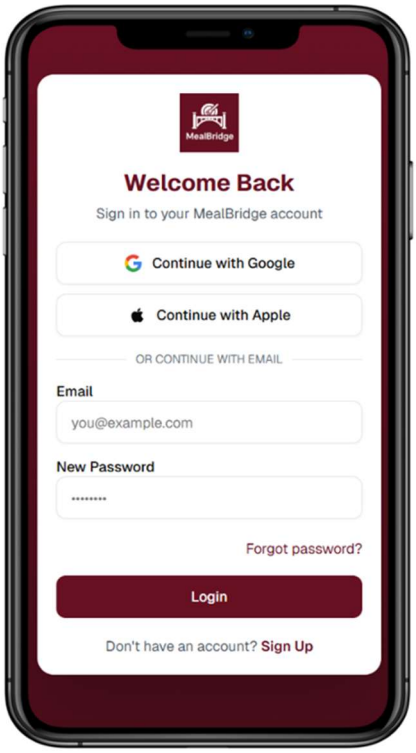
Registration:

Start page



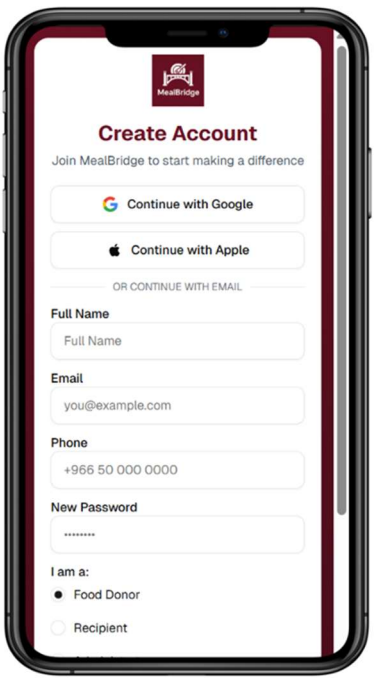
Welcome page offering Sign Up or Login options, with a brief app overview.

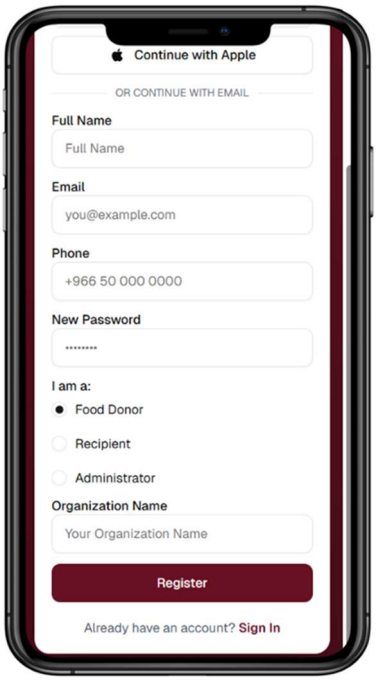
Login page



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

Sign Up page

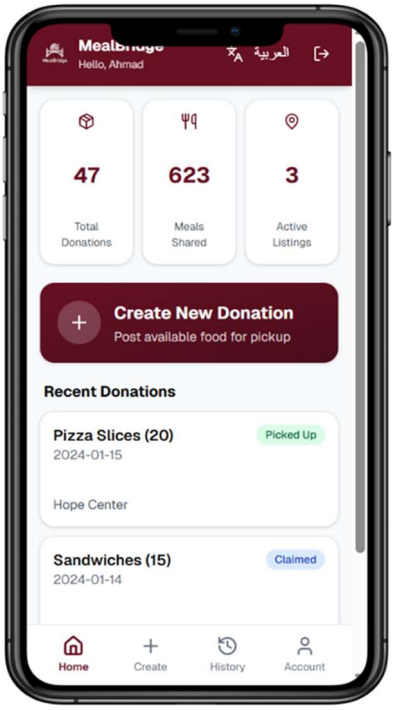




Sign-up page for new users, with options to register via Google or Apple, select a role, and provide full name, phone, and organization (for donors or recipients).

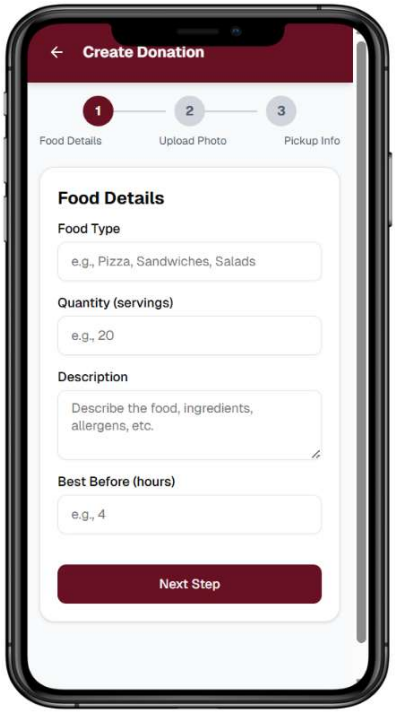
Donor Interface:

Home page

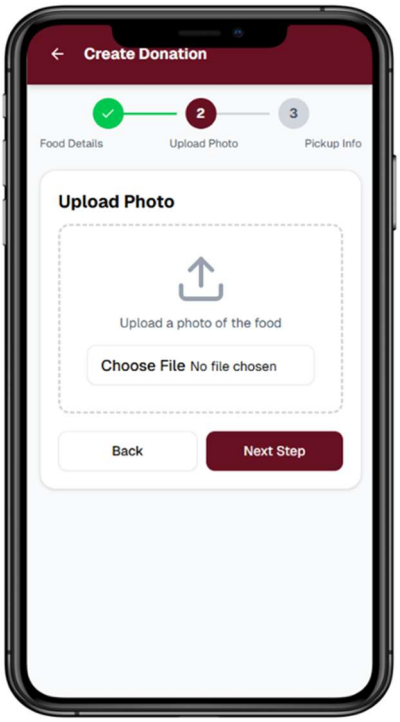


Donor dashboard showing total donations, meals shared, active listings, and a button for quick donation creation.

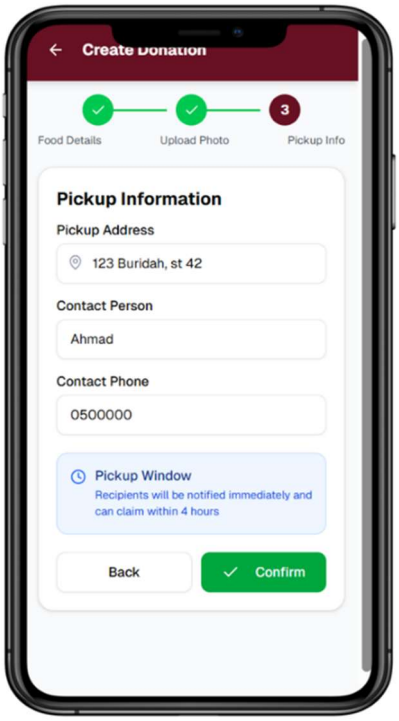
Donation Creation page



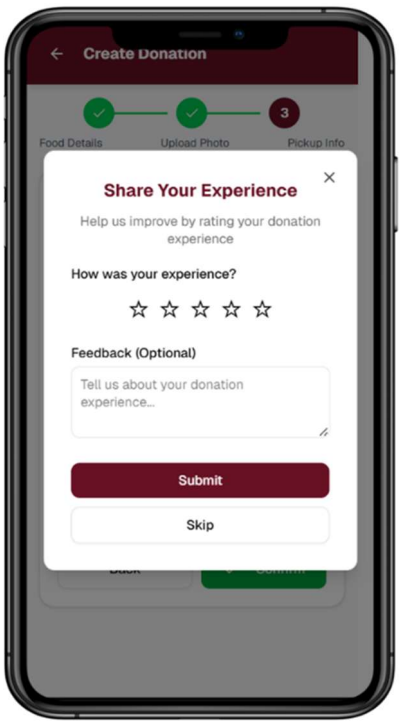
Selecting Quick Donation navigates to the donation creation page "Create", beginning with food details.



Next step is to upload a photo of the food.

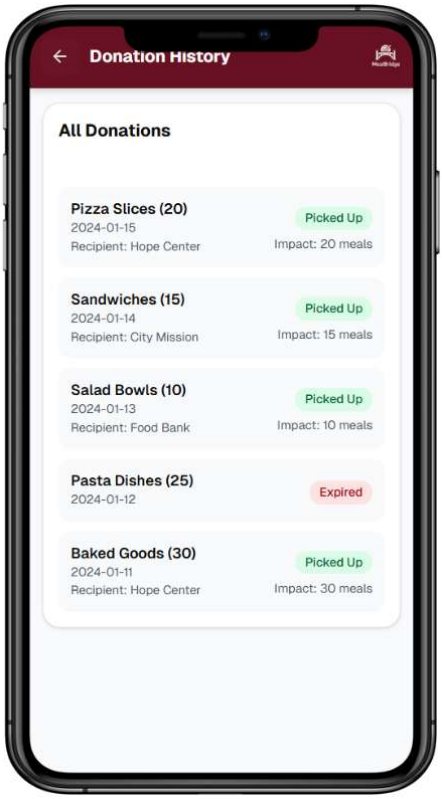


Then they provide the location and contact information.



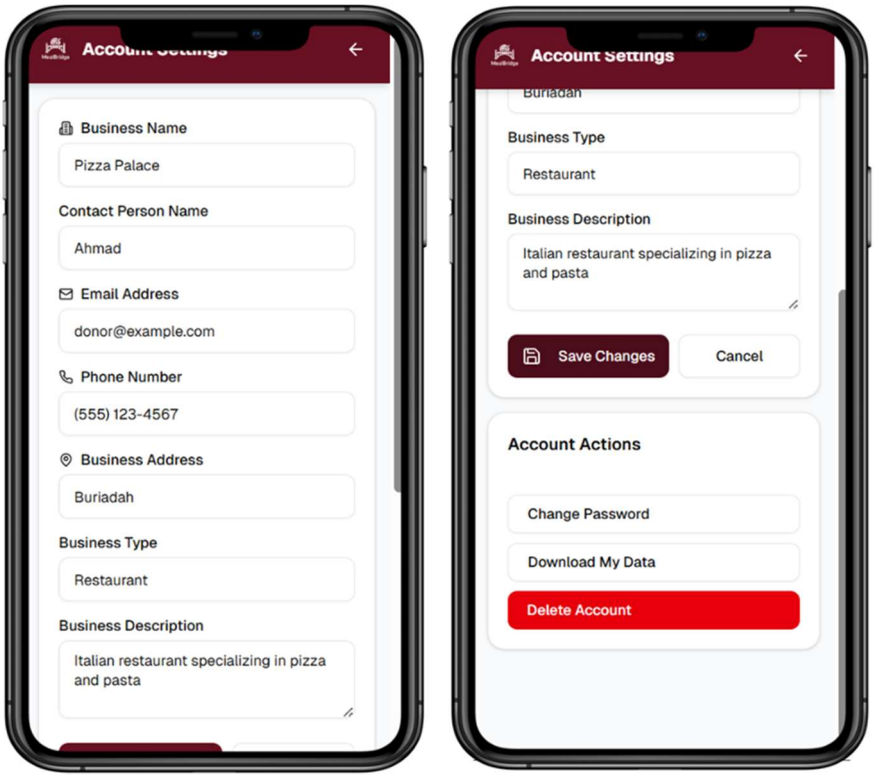
After confirming, a rating and feedback popup appears, allowing users to share their experience.

History page



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

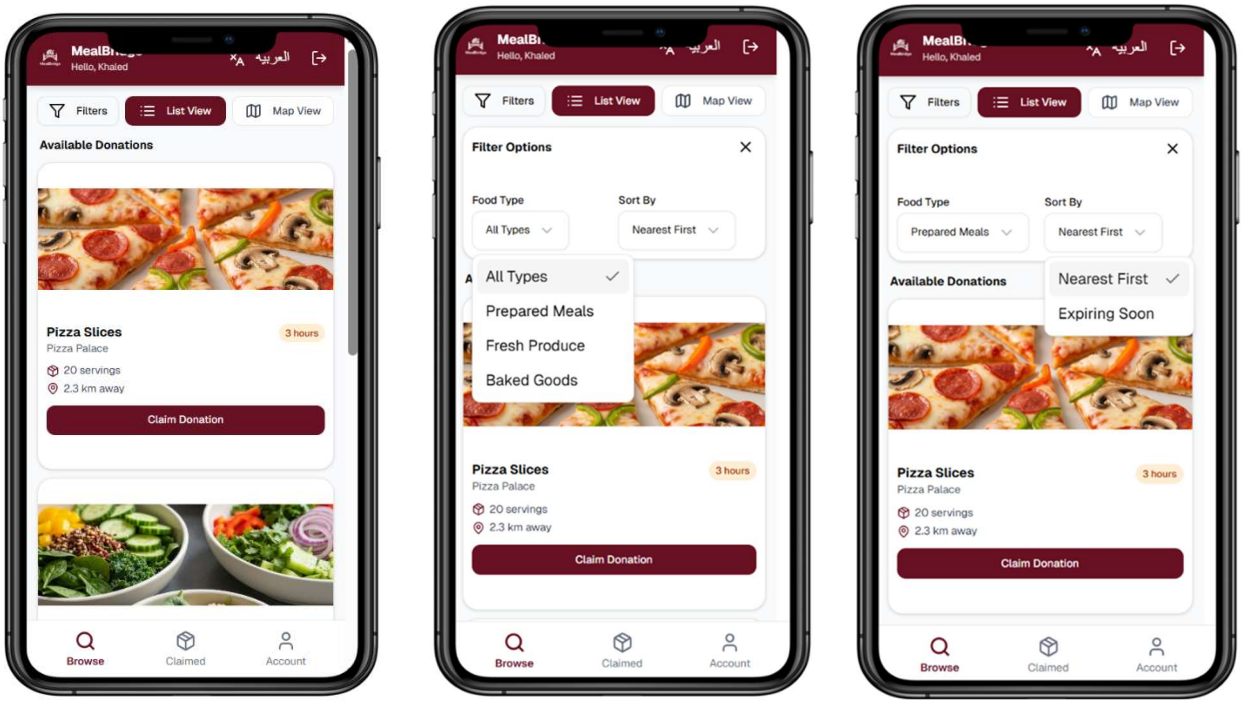
Account page



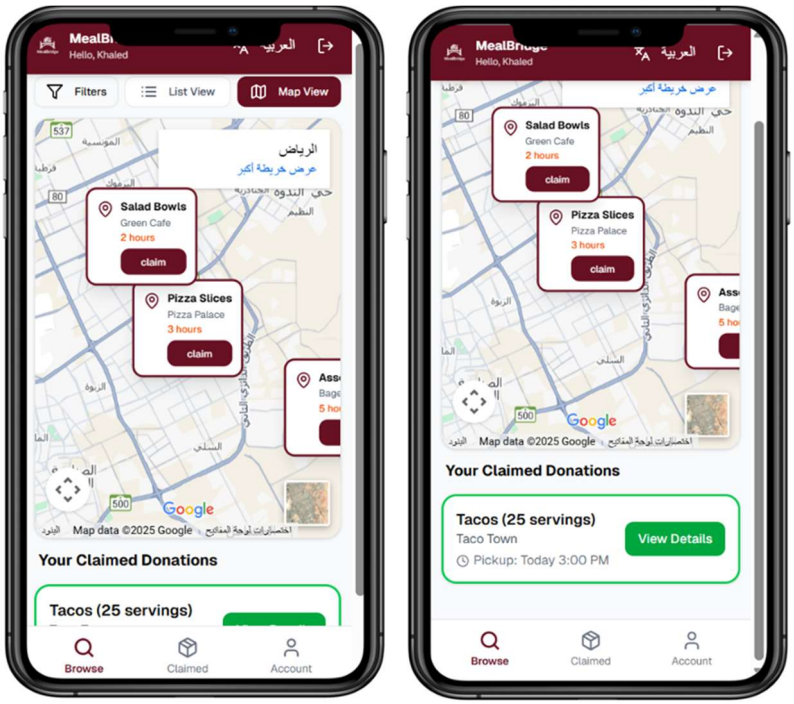
The Account Settings page lets users update their information, change passwords, delete their account, or download their data.

Recipient Interface:

Browse page

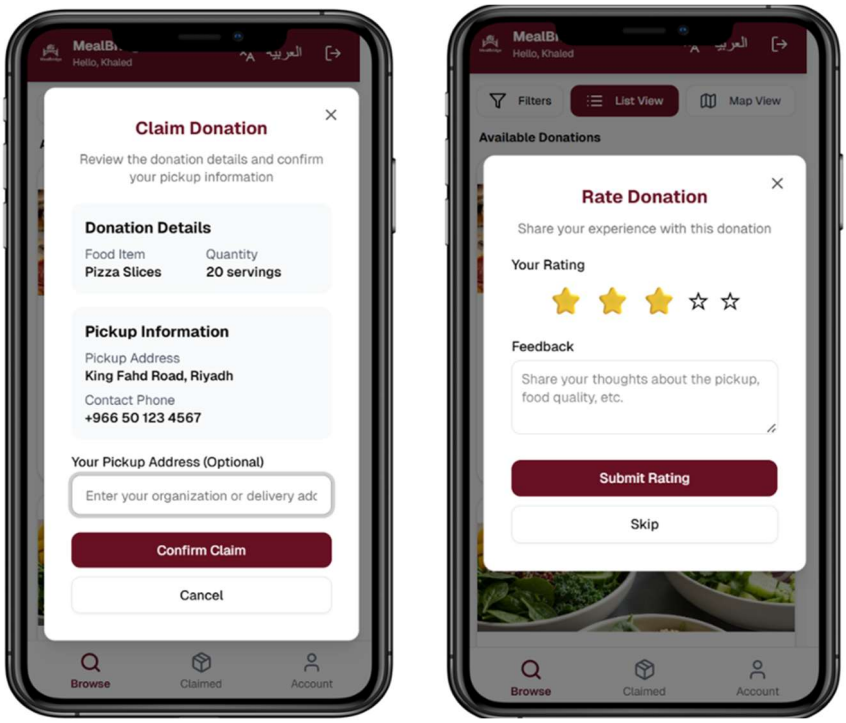


When registering as a recipient, the user is taken to the Browse page, where they can apply filters and view available donations based on Food Type, Location and Expiration date.



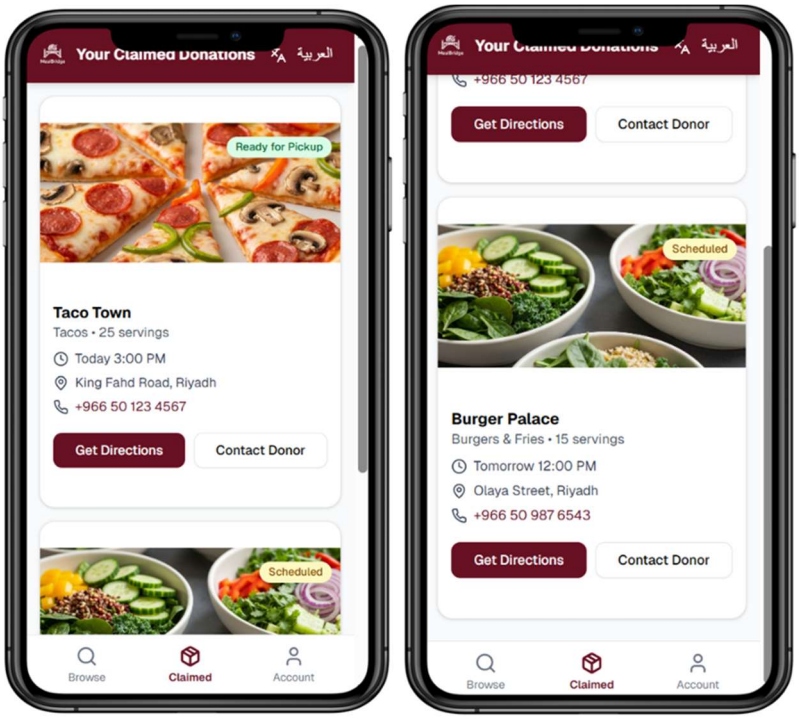
In Map View, nearby donations are displayed through Google Maps, allowing recipients to see locations clearly and get directions. The Browse page also shows recently claimed donations, with an option to view more details.

Claiming Donations process:



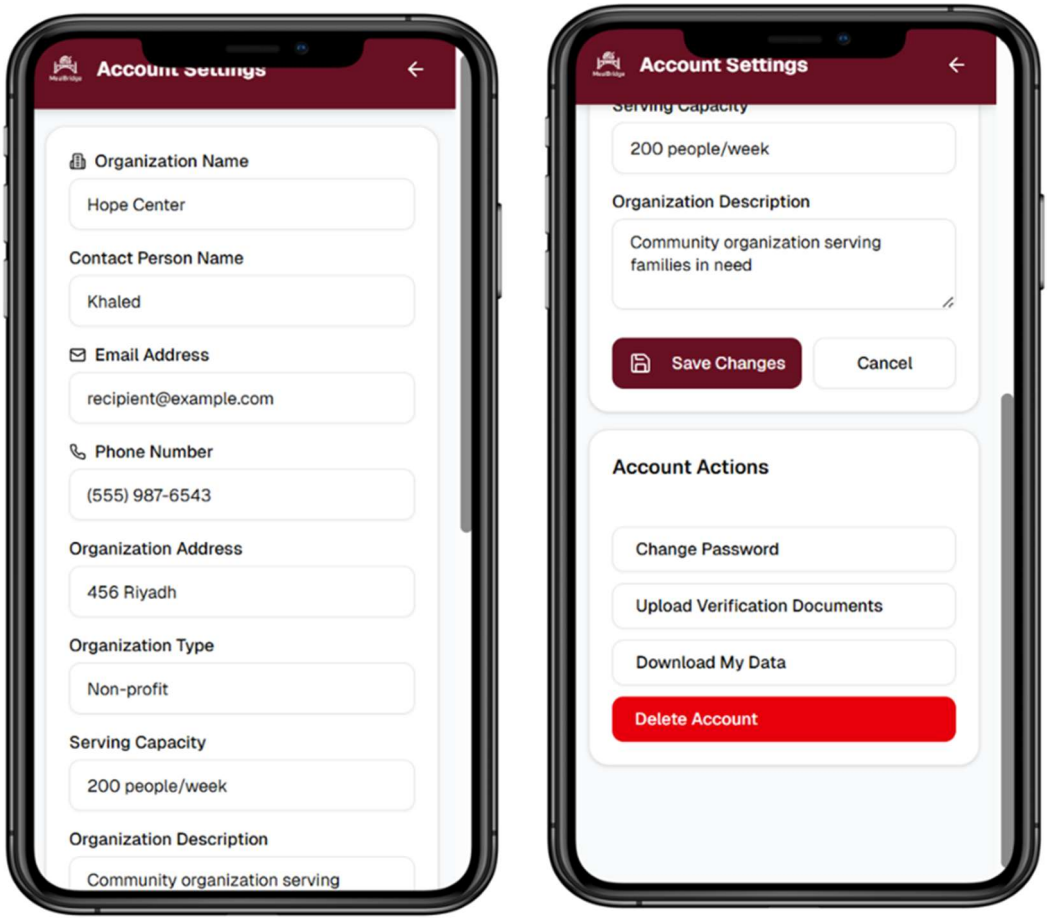
When selecting a donation from the List or Map View, a details window appears where the recipient can review the donation and enter a pickup address. After confirming the claim, the user can rate the donation and provide feedback.

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.

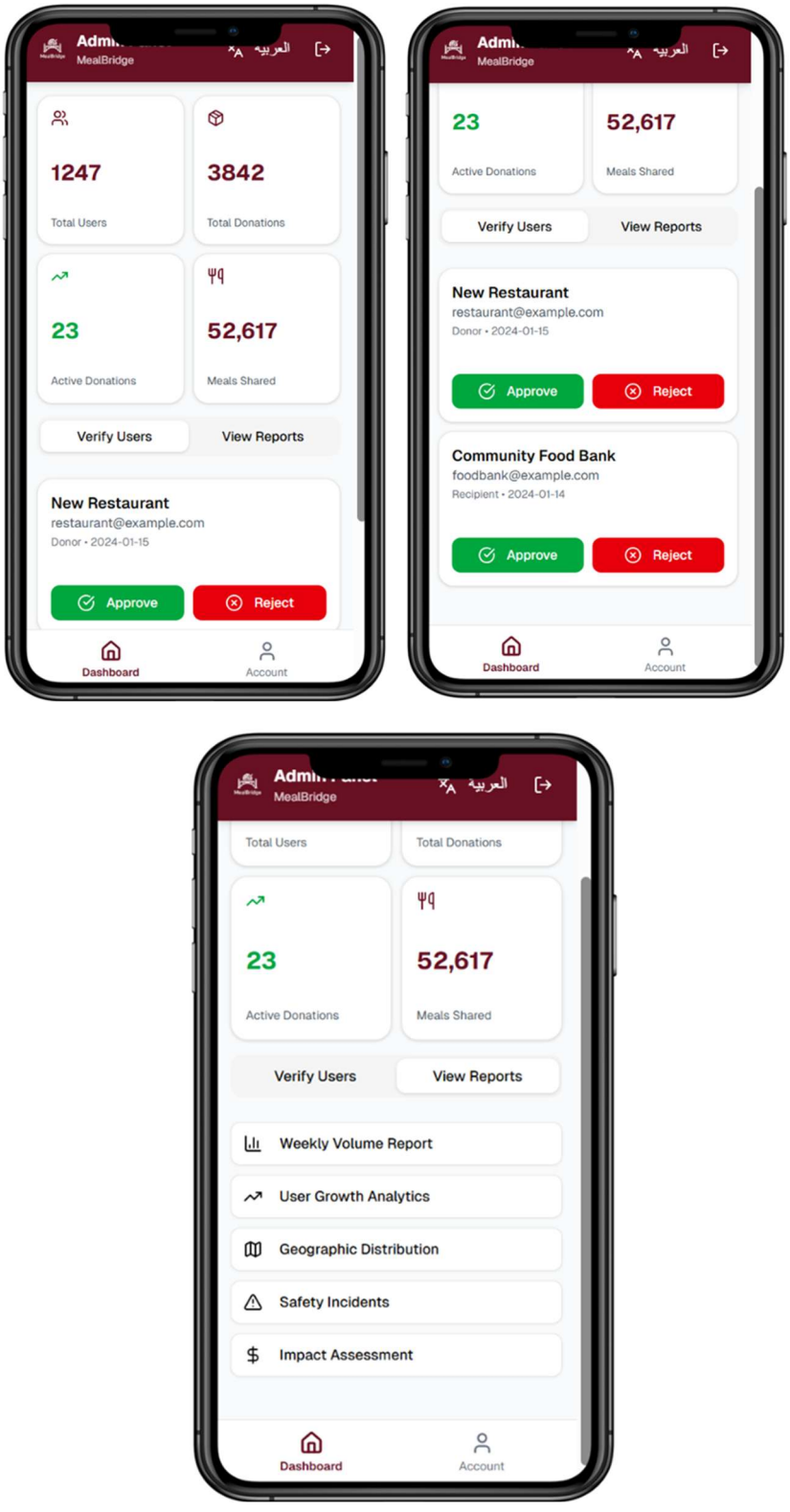
Account page



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

Admin Interface:

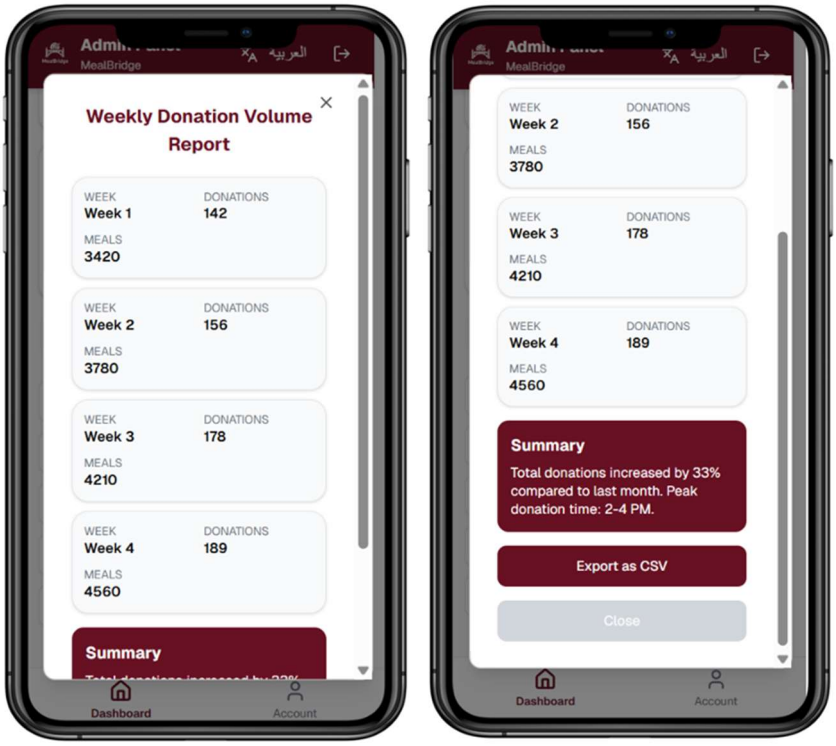
Dashboard page



They also can view different reports from view reports tab.

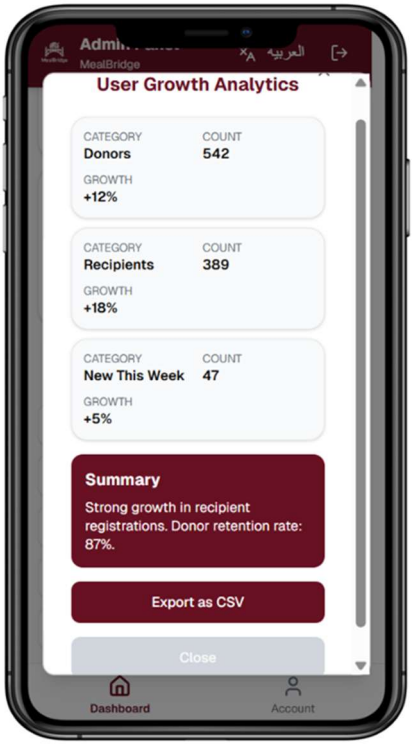
Reports on Admin Dashboard:

Weekly Volume Report



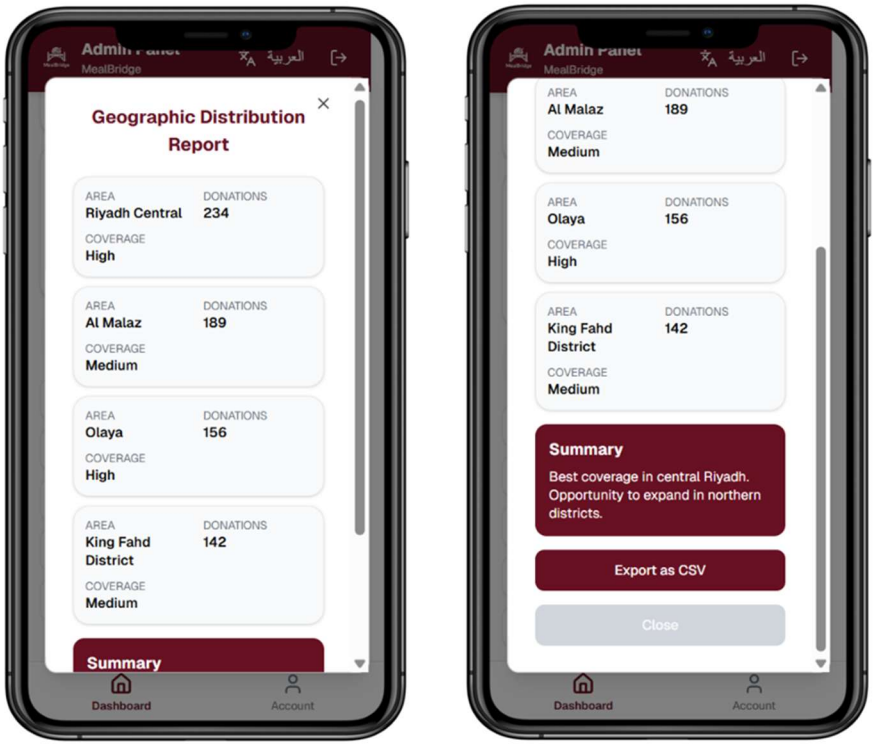
This report shows weekly totals for donations and meals, along with a summary of the figures, with the option to export the report as CSV.

User Growth Report



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

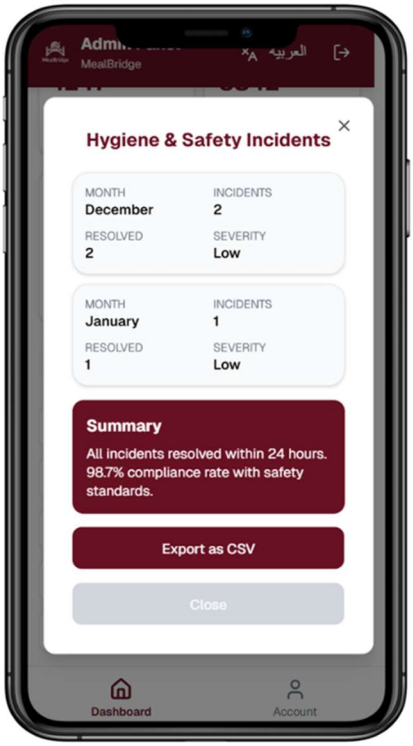
Geographic Distribution Report



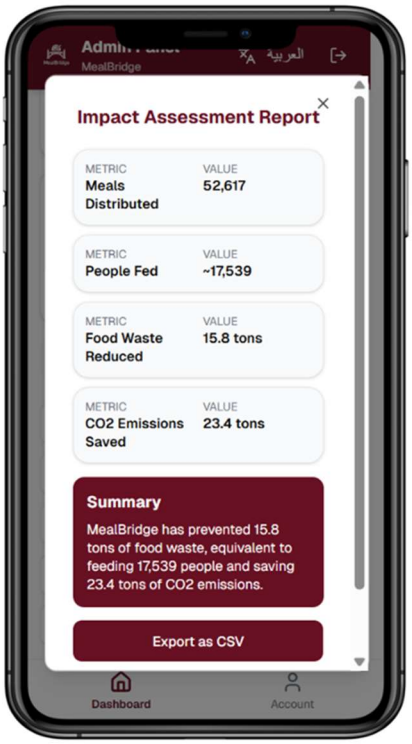
This report shows the geographic distribution of donations, detailing numbers and coverage for each area, with a summary of the figures and the option to export the report as CSV.

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.

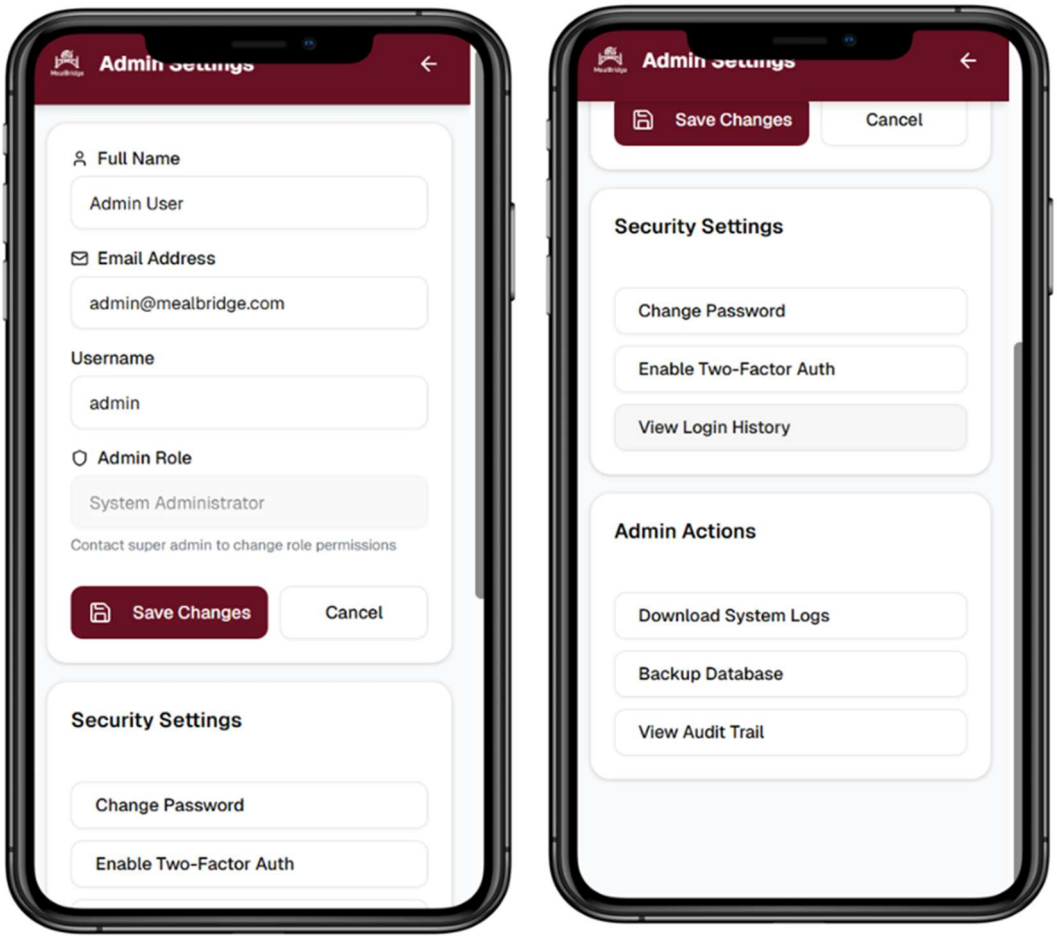


Impact Assessment Reports



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.

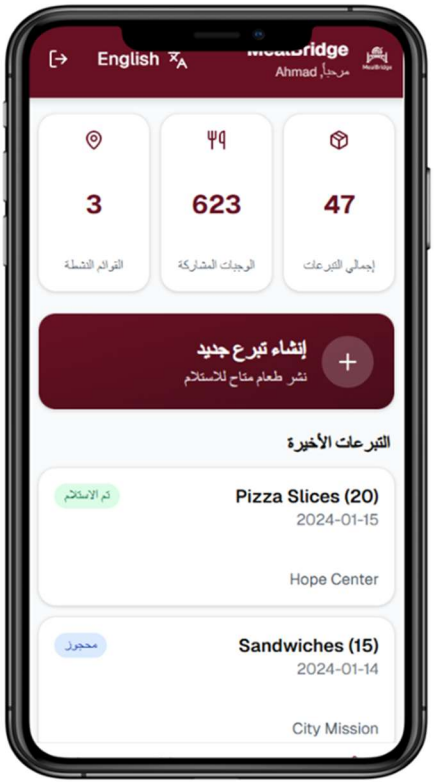
Admin Account Page

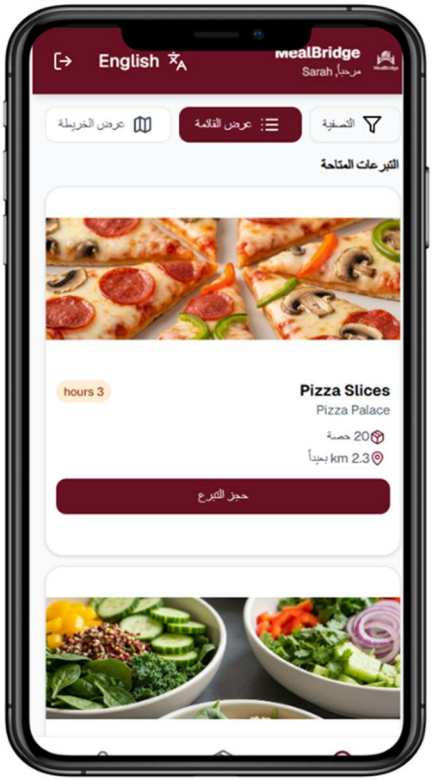


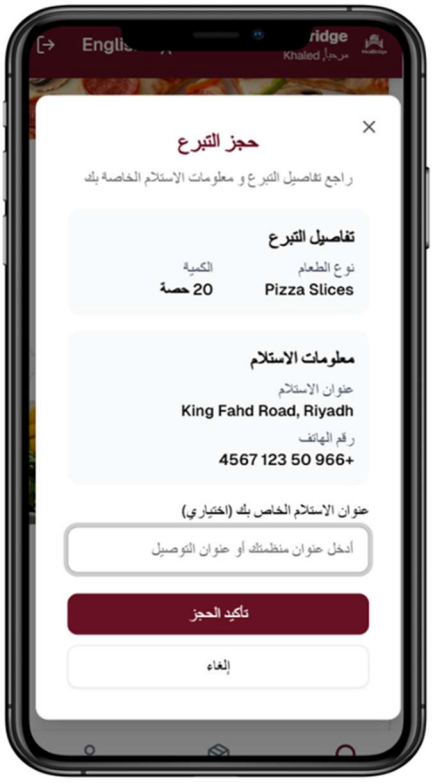
In Admin Account Settings, admins can update their account (except role, which requires a super admin), change passwords, and perform admin actions such as downloading system logs and backing up the database.

Arabic Language:

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







(SRS) Contribution of each member:

Section	Student
Introduction	**Muzna Abdelgadir**
System Overview	**2.1, 2.2 – Aroob Altuwajiri** **2.3, 2.4 – Balqees Almohesn** **2.5, 2.6 – Maha Alrashidi**
Requirement Engineering	**Bedor Alharbi & Muzna Abdelgadir**
UML Use Case Diagram	**Bedor Alharbi**
Functional Requirements	**Balqees Almohsen & Muzna Abdelgadir & Lena Alswed**
Non-Functional Requirements	**Aroob Altuwajiri & Maha Alrashidi**
External Interface Requirements	**Bedor Alharbi**

(SDD) 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**

For full project documentation and more details, please visit our GitHub repository:

https://github.com/Fullmoons17/SWE-MealBridge