# CIS099-2 - Mobile Application Development

# Assessment 2: Group Mobile Application Development Report

## Application: AnnaDaan - Connecting Food Donors with NGOs and Social Activists to Combat Hunger and Reduce Waste

**Submitted By:**

Rohan Kumar Shah : 2417478
Utsav Acharya : 2424906
Kamal Kumar Mandal : 2425719

**Date: 2025/12/11**

# Table of Contents

# Table of Figures

# Table of Tables

# List of Group Members

| Group Members | University ID | College ID |
|---|---|---|
| Utsav Acharya | 2424906 | 2410412019 |
| Rohan Shah | 2417478 | 2410412109 |
| Kamal Kumar Mandal | 2425719 | 2410412050 |

*Table 1:Member list*

# Introduction

The present report describes the creation of AnnaDaan, a mobile application that will enable the use of food donors to find NGOs to reduce food waste and eliminate hunger. This project, which is a subset of the Mobile Application Development module, discusses the design, development, and testing of an effective application due to the worldwide food insecurity and waste problem.

AnnaDaan allows restaurants, markets, event organizers and caterers to place excess food, whilst the verified NGOs and social activists are notified in real-time to pick it efficiently. The app consists of Flutter (cross-platform), Node.js (server), and MongoDB Atlas and is characterized by the ability to follow the progress of the donation in real-time, the integration of Google Maps and email support (recovering passwords) and others.

The project met its goals since it delivered an easy to use, mobile-driven platform that resolved the challenges of manual coordination, visibility, and trust. Intensive testing ensured that it is usable and viable in managing time-sensitive donations of food.

# Task Description

This assignment involved the design, development and testing of a working mobile application as a module in the Mobile Applications Development. The project consisted of the development of a mobile-first platform basis of real devices, illustrating the principles of modern development, user-centered design, and sophisticated integration of the backend part. AnnaDaan is a food donation application based on Flutter and a Node.js back-end, which creates a full B2B2C platform that empowers real-time coordination, a secure OAuth2 authentication system, a cloud-based data storage system, and multi-channel communication (WhatsApp).

The application has the latest functionalities that will include matched donors and receivers (location-based), push notifications, email, and single-tap donation posting with pictures, which offers a scalable solution to food waste and hunger. To assess the usability, effectiveness and reliability, structured user feedback by donors and NGOs were collected.

It consists of a separate report, a demonstration video, and the full source code and APK file.

## Project Plan

| Week | Phase | Key Tasks | Priority |
|---|---|---|---|
| 1 | Planning | • Project selection & proposal<br>• Initial research & feasibility study | High |
| 2 | Requirements | • Identify stakeholders (Donors, Receivers)<br>• Define functional & non-functional requirements | High |
| 3 | System Design | • Create Use Case Diagrams<br>• Design Database Schema (Users, Donations)<br>• Define Tech Stack (Flutter, Node.js, MongoDB) | High |
| 4 | UI/UX Design | • Wireframing & Prototyping (Figma)<br>• Design App Logo & Color Palette | Medium |
| 5 | Backend Setup | • Setup Node.js & MongoDB<br>• Implement Auth API (Login, Signup, OTP) | High |
| 6 | Core Development | • Setup Flutter Project<br>• Connect Auth UI to Backend | High |
| 7 | Donor Features | • Implement "Post Donation" (Image Upload)<br>• Build Donation History & Impact view | High |

| | | | |
|---|---|---|---|
| 8 | Receiver Features | • Implement "Browse Donations" with filters<br>• Add Map view & Reservation logic | High |
| 9 | Refinement | • Basic Admin Dashboard, able to verify Document and manage Users<br>• Integration testing & Bug fixing | Medium |
| 10 | Validation | • Usability & Security testing<br>• Code optimization | Medium |
| 11 | Reporting | • Final Report writing<br>• Create Demo Video | High |
| 12 | Submission | • Final Code Review & Final Push to Git<br>• Project Submission | High |

*Table 2: Weekly Project plan*

# Problem Identification

The amount of food wasted worldwide is 1.3 billion tonnes (Gustavsson et al., 2011), and commercial and event facilities are getting rid of large quantities of their stocks, yet millions of people are hungry. Most of the restaurants, markets, and events in urban centers in Nepal, daily release surpluses that can be eaten, but instead of being eaten, they are thrown into landfills, yet there is no real-time information that NGOs can use to accept these donations. The main problems are manual coordination which is inefficient, no immediate visibility and time-sensitive restrictions- food has to be collected in minutes or in certain periods. Other obstacles include trust and verification issues, the size of donations to divide, and technological adoption with half of the stakeholders indicating lack of dedicated app as one of the key challenges. This systemic failure is one that leads to enormous amounts of food going to waste each day and food insecurity which brings out an important disconnect between excess and demand that can be bridged by a centralized technological intervention. This is where our application fits in (Thyberg & Tonjes, 2016).

# Proposed Solution

AnnaDaan is a dedicated mobile app that utilizes the concept of food redistribution to donors and needy individuals in the most efficient manner possible. The platform forms a secured real-time network, which links restaurants, markets, event organisers, and caterers with credible NGOs and charities. With the picture of the food, donors have options of posting available surplus using features like one tap posting or scheduled postings. NGOs receive real-time notifications on the receiver end, have the option to filter by location and type of food, place orders, and record receipt and compliance papers.

The app is developed based on the need to provide donation with food as it was urgently required to meet the time requirements. It also has geolocation matching(within the 10km radius), push notifications and sharing contact option to organize the last-minute pickups. In contrast to generic food-sharing apps, AnnaDaan is an institutionally focused charitable donation of food, and it offers food safety checks, record-keeping, and impact analytics tools. It is developed on Flutter and Nodejs which guarantee good performance and, therefore, makes it a convenient and scalable solution to minimize food waste and assist communities.

# Aims and Objectives

## Aim:

To design, develop, and launch a user-centered mobile application that efficiently and safely connects food donors with verified charitable organizations, reducing edible food waste and addressing community hunger through real-time, technology-enabled coordination.

## Objectives:

- **Research & Analysis:** To validate core user requirements through stakeholder engagement, analyze existing solutions to identify technical opportunities, and define a feasible set of functional and non-functional requirements.
- **Design:** To transform initial concepts into a high-fidelity, interactive prototype with intuitive navigation and user flows optimized for time-critical donation scenarios, supported by consistent and accessible design guidelines.
- **Development:** To build a fully functional, mobile application using Flutter and Nodejs, integrating essential donor and receiver features, including real-time donation matching, push notifications, and donation management tools within a secure backend infrastructure.
- **Quality Assurance:** To ensure application reliability and performance through comprehensive functional testing on target devices, usability testing with real users, and validation against key performance metrics for speed and notification delivery.

- **Evaluation:** To assess the application's effectiveness and user experience by executing a structured evaluation, collecting both quantitative performance data and qualitative feedback to identify successes and areas for future refinement.
- **Deployment & Marketing:** To prepare the application for public launch by developing an App Store Optimization strategy and a targeted marketing plan, while establishing clear metrics to track post-launch adoption and social impact.

## Scope

This will be an android development and deployment of AnnaDaan mobile application. The application will offer food donors (restaurants, caterers) and verified charitable beneficiaries (NGOs) to arrange the effective food donations. The main in-scope features will be role based user authentication, real time donation posting and discovery with geolocation mapping, push notifications, a proof-of-pickup verification system, and an analytics dashboard. Technical implementation will be based on Flutter and Oauth2 authentication, real-time database and cloud storage.

**Out of Scope:**

The aspects that are explicitly out of scope during this project phase include: development of an iOS version of the application, integration of artificial intelligence functionalities or direct interconnection with point-of-sale (POS) systems. The platform will not offer physical logistics, take legal responsibility of food safety, or have dedicated customer support team. In addition, there will be no web or desktop application, multi-language aspects other than English, connectivity with foreign donation networks, and any payment processing because the application is meant to be free of charge to all users.

## Project Constraints

**Time Constraints:** The project must adhere to the 12-week academic semester timeline, limiting the scope for extensive user trials and the implementation of all desired advanced features.

**Technical & Learning Constraints:** The development required learning complex new technologies such as Flutter, Oauth2, Firebase and third-party API integration, which affected the initial pace and scope of feature implementation.

**Resource Constraints:** Development and testing were primarily limited to 2-3 Android devices, with restricted access to diverse hardware. Financial limitations also constrained the use of paid third-party services and tools.

**Stakeholder Access Constraints:** Securing consistent and timely feedback from target users (restaurants, NGOs) was challenging due to their busy operational schedules and geographic distribution, limiting the depth of iterative testing.
**Deployment & Compliance Constraints:** As a prototype, formal requirements for app store publication, legal compliance, and full-scale food safety verification were acknowledged but not implemented within the project's scope.

# Project Risks

The project was able to identify some of the major risks that needed to be mitigated proactively. The critical risk was scope creep, which was handled with the strict compliance to MoSCoW prioritization and the use of clear communication with the stakeholders to implement the project limits. The problem of tight deadlines and possible schedule delays was resolved due to the creation of clear milestones and regular progress checks. Technically, the potential risks were API failure and complexity of data synchronization, which were avoided by means of local loading states and such stable services as Google Maps API and Firebase. To maintain a consistent user experience, it was required before coding that the shared Figma design system was developed to preempt UI inconsistency. Moreover, the possible delays due to the stakeholder feedback were also eliminated by arranging regular reviewing sessions with specific time frames to respond to the feedback. There is also the threat that is always the threat of data privacy. Lastly, the threat of information loss was also reduced by using MongoDB Atlas as an automatic cloud backup.

# Justification of Selection of Technologies

## Technology Stack

- **Flutter:**Used for its high-performance rendering engine and hot-reload capabilities, enabling rapid development of a smooth, native-feeling user interface. Its rich widget library allowed for the efficient creation of complex, custom features like real-time donation matching and interactive maps.

- **Node.js & Express.js:** Utilized as the backend to handle APIs, authentication, and core application logic.

- **MongoDB Atlas:** Used as a cloud-based database for scalable and flexible data storage, including image data management.

- **Firebase:** Integrated to handle in-app push notifications.

- **Render:** Used for hosting the backend APIs.

- **Figma:** Employed for UI/UX design and prototyping for both the mobile application and the admin panel.

- **Git & GitHub:** Used for version control and collaborative development.
- **Postman:** Used for API testing and documentation.

### Technical Integration

- **User Authentication and Authorization:** OAuth 2.0 with OpenID Connect (OIDC).
- **Location-Based Search:** Integrated using Google Maps API.
- **Media Handling:** MongoDB Atlas used for image uploads and storage.

### Comparison with Alternatives

- **React Native vs Flutter:** Flutter was chosen for smoother UI rendering and better performance (Martin et al., 2017).
- **Native Applications:** Using Flutter with Node.js reduces development time compared to building separate native apps.
- **Firebase-Only vs Node.js Backend:** Node.js enables greater flexibility, scalability, and control through custom backend logic.

## Required Resources

### Software Resources

**Development Tools**
Visual Studio Code, Flutter SDK, Android Studio, Node.js, Express.js, Firebase, Figma (prototyping), Git/GitHub (version control), Clickup (project management).

**Backend:** Mongodb Database, OAuth Authentication, Sendgrid for email

### Hardware Resources

**Development:** Laptop (Linux(Ubuntu), i5 1235u Processor, 16GB RAM, 512GB SSD), Xiaomi Redmi Note 8 Pro (Android 11).

**Testing:** Redmi Note 8 Pro, Redmi Note 9 Pro (teammate).

### Graphics

- Figma for prototype and UI/UX Design.
- Open source icons and fonts.

# Methodology

This project used an Agile approach to development (Beck et al., 2001) in order to have an iterative development and a constant feedback on the stakeholders. The stakeholders were also involved throughout the process and each stage of the process the functionality and design was improved through the input of the stakeholders. The prototype, which was a basic, working prototype, was created during the first two weeks, and was shown to key users, allowing early feedback on core workflows and layout.
Its development was based on a projected sprint cycle, wherein every week aimed at delivering the top priority features including user authentication, posting donation, and real-time notifications. The requirements were categorised by the MoSCoW method which made sure that the Minimum Viable Product (MVP) dealt with the most essential user needs first.

Feedback and testing was included in each sprint. These involved unit and integration testing throughout development, and informal stakeholder usability reviews. The ultimate, formal assessment was performed with the help of the DECIDE tool after which quantitative and qualitative feedbacks were used to implement the final series of changes. This strategy traded off Agile concepts like releasing working software regularly and responding to change with the reality of a single academic project, in which a working, user-tested application had to be implemented within the limited timescale.

# Stakeholders

1. Food Donors: Restaurants, Wedding Venue, Party Palace, etc
2. Food Receivers: NGOs, Social Worker.
3. End Beneficiaries: People in need of food
4. Developers / Project Team: Utsav Acharya, Rohan Shah, Kamal kumar Mandal
5. University Supervisors and Evaluators

# Requirements Analysis

## Target Users

**Donors:**
This group includes restaurant owners, market vendors, and event organizers who have surplus food. They typically use Android smartphones with moderate technical proficiency and need to post donations quickly, often in rushed, time-sensitive situations. Their primary needs are minimal data entry, immediate confirmation, and features that address concerns about liability and coordination.

**Receivers:**
This group consists of NGO staff, volunteers, and social activists who seek to collect and redistribute donated food. They are generally comfortable using mobile apps and require real-time alerts, reliable filtering options, and straightforward tools for logging pickups. Their main challenges include responding to opportunities quickly, managing logistics, and maintaining necessary compliance records.

## Primary Market Research

Primary market research was carried out using structured surveys aimed at both food donors (restaurants, markets, and event organizers) and food receivers (NGOs and social activists). A mixed-method approach was employed to validate key assumptions, identify existing challenges in food donation workflows, and collect direct feedback on required features and usability expectations for the proposed platform.

## Questionnaire Section

The survey was performed by two of the teammates accumulating response from 23 Individuals and that is why two Google Form links are provided on the section below:

Google Form Link:
https://docs.google.com/forms/d/e/1FAIpQLSd_Aj9rJedFGuhv3zsDag4-bAq33jF_27jdVe3fvaE5rB7_XA/viewform?usp=header

https://docs.google.com/forms/d/e/1FAIpQLSfKPOXifn4qfLOC7XsPxC-dh3nqq9vDaMxgjJIsqkf0pVCCPg/viewform?usp=header

## Analysis of Responses

The survey results provided several important insights that directly influenced the design and feature prioritization of AnnaDaan:

- **User Demographics & Surplus Patterns:** Most respondents were restaurants (66.7%). Among them, 50% reported generating surplus food 0–1 days per week, with surplus most commonly occurring during evening hours.

- **Key Barriers:** The lack of a dedicated application (50%) emerged as the primary barrier to consistent food donation. This was followed by unreliable pickup coordination (33.3%) and concerns related to trust and verification (33.3%).

- **Current Coordination Methods:** Donors currently depend on scattered communication channels such as WhatsApp/Viber (41.7%) and phone calls (16.7%), reinforcing the need for a unified and centralized platform.

- **Time Sensitivity:** Half of the donors indicated a requirement for pickup confirmation within 15–30 minutes, highlighting the importance of real-time alert and response mechanisms.

- **Feature Priorities:** The most requested features included real-time notifications (66.7%), proof-of-pickup documentation (58.3%), and scheduling functionality (50%).

- **Safety & Compliance:** Existing food safety practices varied, with 50% of respondents using labels and timestamps, while 41.7% segregated vegetarian and non-vegetarian food. These findings emphasize the need for integrated safety and compliance checklists within the platform.

Overall, these insights validated the core functionalities of the proposed solution and informed the development of a time-sensitive, user-verified, and feature-driven platform.

## Personas, Scenario and Use Cases

## Personas

### Persona 1: Rita Sharma, The Efficient Restaurant Manager (Donor)

**Name:** Rita Sharma
**Age:** 35
**Location:** Kathmandu

**Education and Employment:** Diploma in Hotel Management; has been managing a mid-sized restaurant in Kathmandu for 8 years.

**Family Status:** Married, with one young child.

**Information Source:** Hotel association networks, Facebook groups, WhatsApp.

**Online Behavior:** Uses her smartphone extensively for inventory and bookings. Prefers apps that are fast and require minimal input.

**Typical Tasks:** Managing daily kitchen surplus, staff scheduling, reducing operational costs.

**Goals:**
• Find a quick, low-effort way to donate surplus food.
• Gain transparency on where her food ends up.
• Contribute to the community without adding administrative burden.

**Frustrations:**
• The moral dilemma of wasting edible food.

• Inconsistent and unreliable pickup processes.

## Persona 2: Sita Koirala, The Dedicated NGO Coordinator (Recipient)

**Name:** Sita Koirala
**Age:** 29
**Location:** Kathmandu

**Education and Employment:** Master's in Social Work; runs a small children's shelter.

**Family Status:** Married, lives with spouse and parents.

**Information Source:** NGO networks, donor newsletters, community forums.

**Online Behaviour:** Uses a mix of smartphone and laptop. Values clarity and reliability over complex features.

**Typical Tasks:** Sourcing food donations, managing volunteer schedules, reporting to donors.

**Goals:**
• Secure a reliable supply of food for her shelter.
• Quickly claim and coordinate the pickup of large donations.
• Provide proof of delivery to build trust with donors.

**Frustrations:**
• Unpredictable donation sources.
• Difficulty in mobilizing volunteers at short notice.

## Persona 3: Manish Adhikari, Banquet Manager(Donor)

**Name:** Manish Adhikari
**Age:** 42
**Location:** Bhaktapur

**Education and Employment:** Bachelor's degree in Business Management; has been managing a large banquet and party palace for over 12 years.

**Family Status:** Married, with two children.

**Information Source:** Event management networks, hospitality associations, Facebook groups, WhatsApp.

**Online Behavior:** Uses his smartphone mainly for bookings, vendor coordination, and event scheduling. Prefers structured, reliable apps that can handle large-volume entries easily.

**Typical Tasks:** Coordinating large events, managing post-event surplus food, supervising staff, handling client follow-ups.

**Goals:**

- Find an efficient way to donate bulk surplus food after events.
- Ensure timely pickup to maintain food safety standards.
- Build a positive social image for the banquet business.

**Frustrations:**

- Large quantities of food are going to waste after events.
- Difficulty finding trustworthy recipients on short notice.

## Scenarios:

**Scenario for Rita (Donor):**
Rita, after a grand wedding meal, has about 50 untouched meals which will probably go wasted. She wants to give away these meals as soon as possible before the end of the day, but needs to find a beneficiary quickly. She wants to put it up on AnnaDaan App for distribution.

**Scenario for Sita (Recipient):**
Sita needs to arrange dinner for the 40 children at her shelter. She uses the AnnaDaan app to find available donations and finds Rita's large post. She immediately claims it and dispatches a volunteer.

**Scenario for Manish (Donor):**

After hosting a large wedding reception at his banquet and party palace, Manish is left with a significant quantity of freshly prepared food that remains untouched. With limited storage time and strict hygiene concerns, the food needs to be donated quickly. Instead of letting it go to waste, Manish uses the AnnaDaan app to post a bulk food donation, allowing verified NGOs to view and claim it promptly for distribution.

## Use Cases

Flow of the Application on Donor Side:

- Donor signs up or logs into the application using business credentials.
- Donors list the food they wish to donate (food type, quantity, pickup window, location).
- Donor completes the safety checklist (temperature, packaging, labeling).

- Donor submits the donation listing.
- The system sends real-time notifications to nearby verified NGOs.
- The donor receives a reservation confirmation from the NGO.
- Donors hand over the food to the NGO at the specified pickup window.
- Donor confirms the successful handover in the app.
- Donor receives a digital receipt and impact summary (meals provided, waste diverted).
- Donors can view active donations, update donation details, or cancel if needed.
- Donors can access their donation history and impact dashboard.

Flow of the Application on Receiver Side:

- Receiver (NGO/Social Activist) registers/logs into the application with verified credentials.
- Receiver receives real-time alert notification (push notification) about nearby donation.
- Receiver views donation details (food type, quantity, pickup window, location, safety info).
- Receiver applies filters to search donations (distance, food category, pickup time).
- The receiver reserves the donation.
- The receiver reaches the pickup location with the guidance of Google Map(app redirects to Google Map showing the location to the pickup location).
- The receiver collects food from the donor at the specified location.
- Receiver submits proof-of-pickup .
- System confirms successful pickup to both donor and receiver.
- Receiver receives compliance log for audit and reporting purposes.
- Receivers can view collection history and access weekly/monthly reports.

## Use Case Diagram:

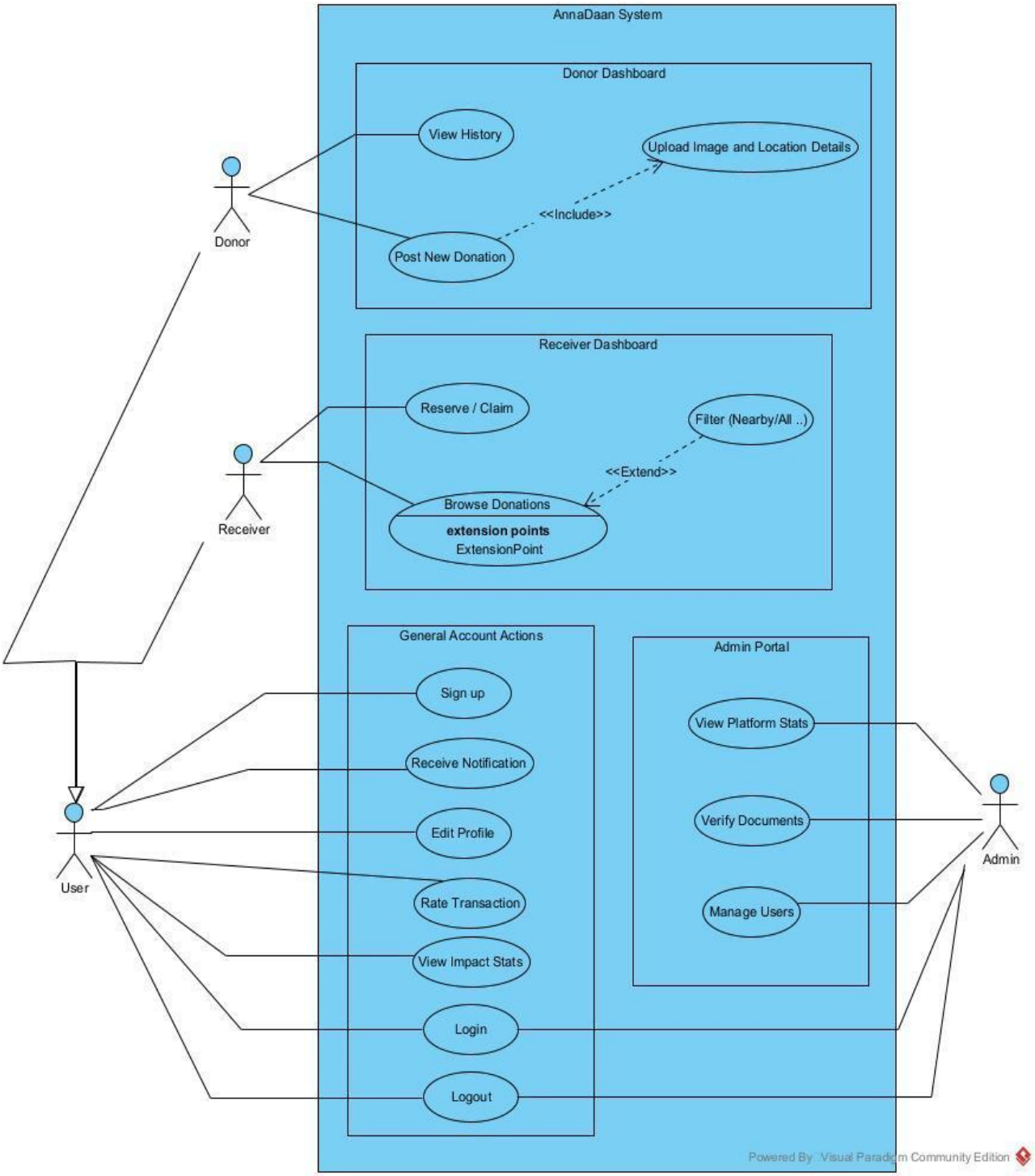



*Figure 1:Use Case diagram of the System*

The AnnaDaan Use Case Diagram graphically reflects the functional scope of the system, which is divided into separate donor, receiver, and administrator dashboards, and generalizes the common behaviors into an activity of a generalized user. The model provides consistency in common operations, such as authentication and profile management, and role-specific processes are carried out by specialized actors. An example is the Post New Donation use case of the Donor, that requires an <include>the data in the use case of Upload Image and Location Details is required to be checked and the Receiver use case, Browse Donations requires an optional <Extend> relationship to filter by location.

All these interactions comprise a strong platform ecosystem in which the Admins retain control by solely privileged interactions such as document verification and user management. The diagram successfully reflects both the basic operational needs, including claiming donations and history, and the logic of the structure of the system, the application of the principles of modular design such as generalization and extensions points to simplify the user interactions.

## Market Research and Analysis

### Overview of the Current Market

The stark reality of food waste stands in tragic contrast to the prevalence of global hunger. Approximately 1.3 billion tonnes of food are discarded annually, with significant losses occurring at the commercial level—4-10% in kitchens and 20-30% from events. This immense wastage persists across the entire supply chain, even as millions worldwide suffer from chronic food insecurity. This failure to redirect surplus has created a significant economic opportunity; the global food waste management market is projected to reach $44.51 billion by 2027.

This contradiction is acutely visible in dense urban centers like Kathmandu, where substantial food surpluses coexist with vulnerable populations experiencing hunger. While digital platforms have emerged to bridge this gap, they have largely failed to gain traction due to persistent logistical challenges and relatively low levels of digital literacy within the very communities they aim to serve. However, consistent local search interest in "food donation" confirms a clear and unmet market need. This points directly to the critical demand for a truly localized and accessible digital platform, one designed to overcome these specific barriers and effectively connect surplus food with those in need.

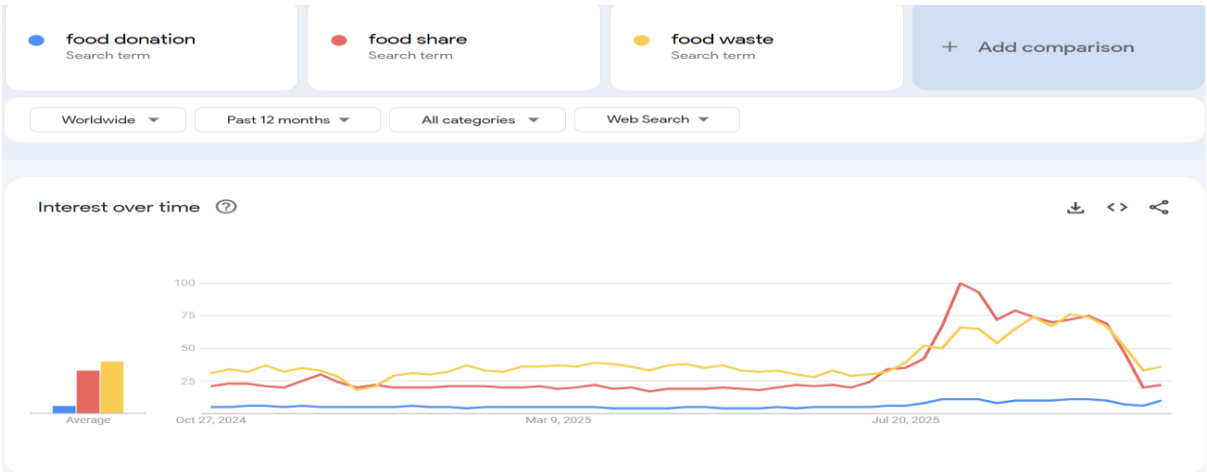



*Figure 2:Google trend comparison between different key words*

Below, Google Trends data for the rise in activity of food donation apps ShareTheMeal and MealConnect is provided, which shows that there is increased demand for such apps in the market.

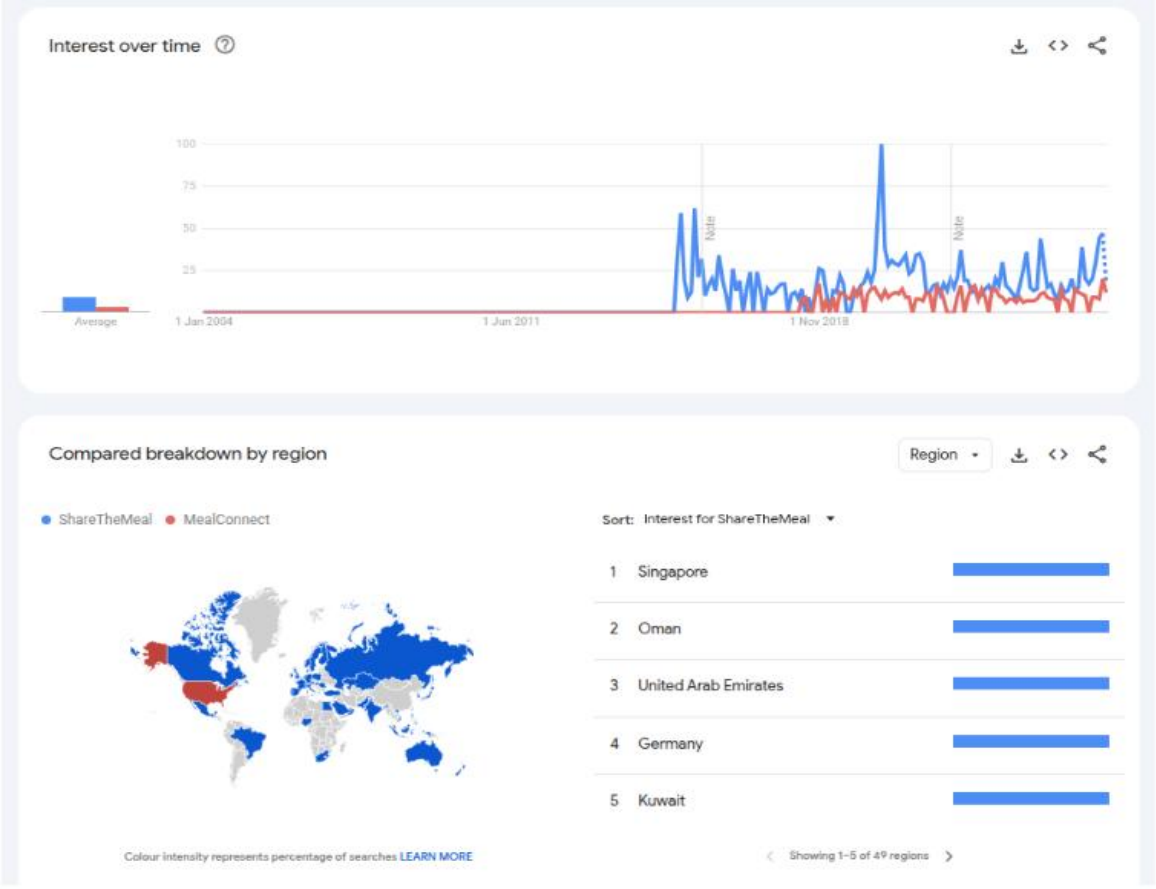



*Figure 3:Google trend analysis*

**Review of Existing Mobile Applications**

**Too Good To Go:** Large user base; commercial sales model unsuitable for bulk/institutional donation.
**OLIO:** Community peer-to-peer; lacks NGO verification; not optimized for large-volume, time-critical donations.
**MealConnect/FoodCloud:** Donation-focused with verified receivers; web-centric, retail-focused; limited same-day, event-based support.

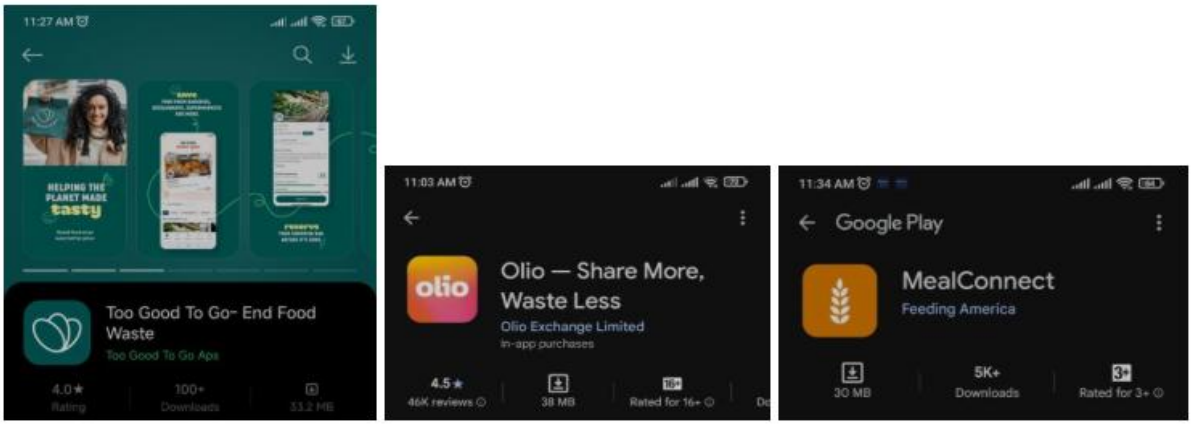



*Figure 4: Existing Food donation Application*

**Comparative Analysis and Identified Gaps**

| Feature | Too Good To Go | OLIO | MealConnect | AnnaDaan |
|---|---|---|---|---|
| Donation | No | Yes | Yes | Yes |
| Mobile-first | Yes | Yes | No | Yes |
| Event support | No | No | No | Yes |
| Verified NGOs | No | No | No | Yes |
| Real-time alerts | Yes | Yes | Partial | Yes |

*Table 3: Comparative Analysis*

**AnnaDaan's Unique Value Proposition:**
AnnDaan is the first committed app dedicated to Nepal concerning the acceleration of digital processes with the trust of communities.

**Verified User Ecosystem:** All NGOs and partner donors must undergo stringent credential verification.

**Real Time distribution**: Live notifications and map-based claiming for real-time impact.

**Transparent Workflow:** All-inclusive status tracking from donation to delivery.

**Localized Design:** Built for Nepali user behavior.

This modular approach directly tackles the most critical failures in existing solutions and builds a trusted scalable system for Nepal.

## Overview of Functional, Non-Functional and Usability Requirements

### Functional Requirements

| Requirements | Priority |
|---|---|
| User Registration, Login , Authentication and Authorization | Must |
| Donor being able to post Donation | Must |
| Donor being able to see the recently posted Donation | Should |
| Donor being able to edit and delete the posted Donation | Must |
| Donor/Receiver being able to upload the Verification Document | Must |
| Receiver being able to Reserve the posted Donation | Must |

| | |
|---|---|
| Receiver being able to view the reserved pickups list | Should |
| Receiver being able to get the contacts of the donor for pickup | Must |
| Receiver being able to open the map for direction | Should |
| Admin being able to verify the uploaded documents | Must |
| Admin being able to manage all the users of the application | Must |

*Table 4: Function Requirement of system*

### Non-Functional Requirements

| Requirements | Priority |
|---|---|
| App performance should be fast and responsive across devices | Should |
| System should handle high number of concurrent users without crashing | Must |
| Cloud services must maintain uptime | Should |
| Data privacy and security must be ensured for all user information | Must |

*Table 5: Non-Functional Requirements*

### Usability Requirements

| Requirements | Priority |
|---|---|
| Easy navigation with clear buttons and icons | Should |
| Readable fonts, high-contrast colors, and touch-friendly buttons | Should |
| Consistent layout and Color scheme | Must |
| Compliance with WCAG guidelines | Should |
| Phone number validation on Manual Registration (not google signup) | Should |

*Table 6: Usability Requirements*

MOSCOW Notation:
M = MUST
S = SHOULD
C = COULD
W = WON'T

## Mobile Application Design

The user interface design for AnnaDaan evolved through multiple iterations, progressing from low-fidelity wireframes to high-fidelity interactive prototypes, and finally to the implemented

application. The design process was guided by user-centered design principles (Nielsen, 2012), mobile usability heuristics (Zhang and Adipat, 2005), and continuous stakeholder feedback.

## UI Design (Prototype):

Donor:

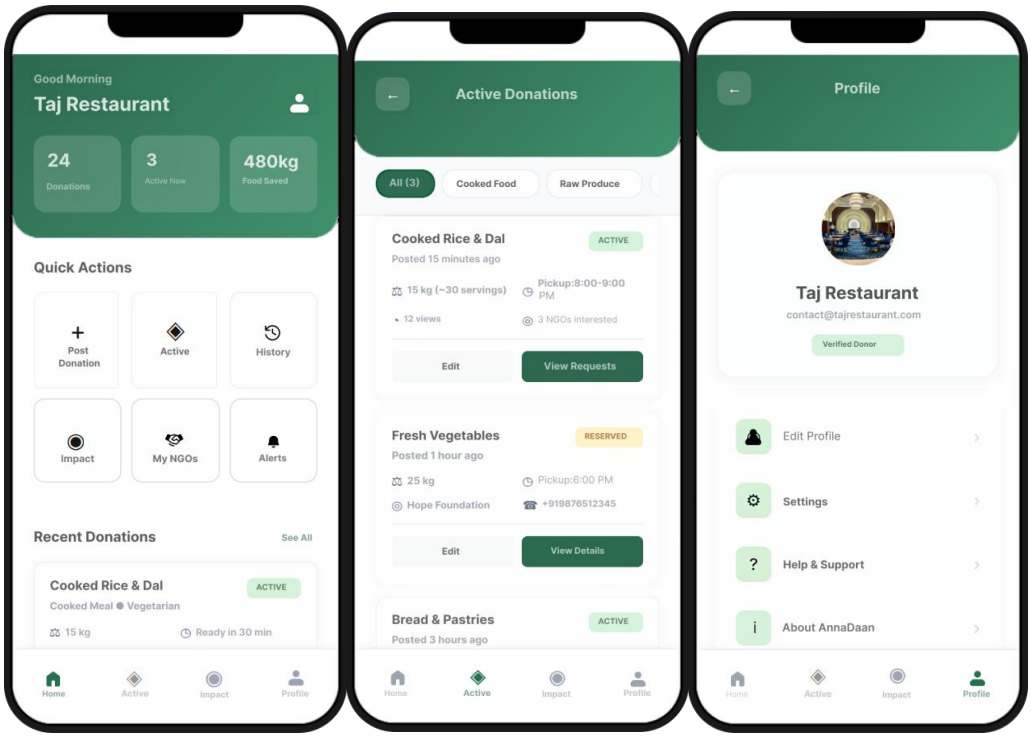



*Figure 5: Donor UI Prototype*

Receiver:

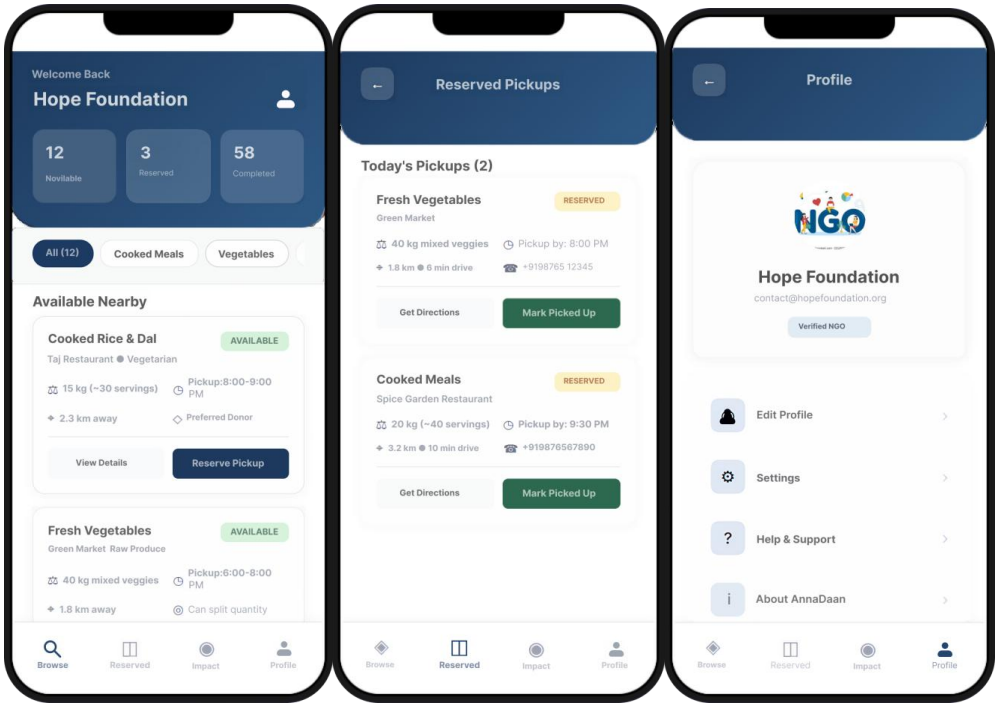



*Figure 6: Receiver UI Prototype*

**Figma Link of the Prototype:**
https://www.figma.com/design/hH1Vh8hpwubgXTW4C1Mf4P/AnnaDaan?node-id=0-1&t=AjbIdMMVR5mfd1Yy-1

## UI Design (Final):

Donor:

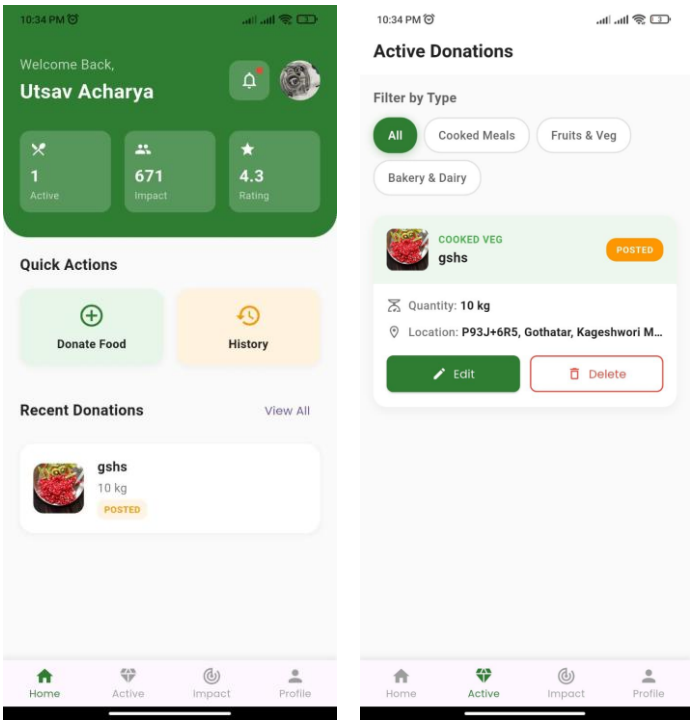



*Figure 7:Final Donar UI*

Receiver:

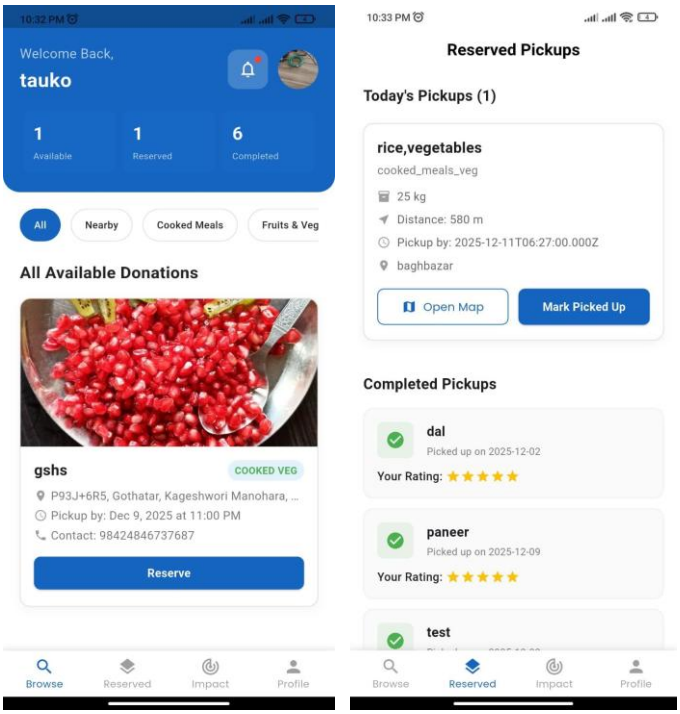



*Figure 8: Final Receiver UI*

Admin:

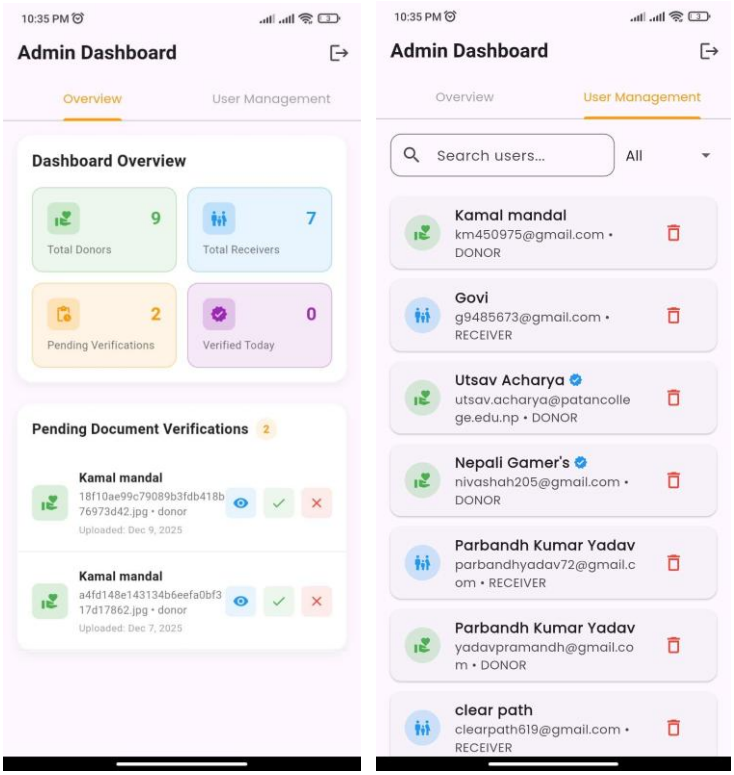



*Figure 9:Admin UI*

Note: You can view all the other Screenshots on the Appendix section

# Mobile Application Development / Implementation

## Application Creation and Development Stages

AnnaDaan was developed using a well-organized Systems Development Life Cycle (SDLC), and in particular, it used an Agile approach that enabled iterative quality enhancements and reconfigurability to the requirements. The creation process was divided into four different steps:

1. **Planning and Requirement Analysis:** the first step consisted of defining the main goals: bridging the gap between food donors and excess food and NGOs as well as receivers. We have discovered the players (Donors, Receivers, Admins) and came up with functional requirements, including authentication, donation posting, and geolocation service in real-time.
2. **Design and Prototyping:** A high level system architecture was developed which divided the application into a client-server architecture. Our User Interface (UI) is centered on the needs of the user, with large targets of touch and high contrast that cater to the needs of various types of users. MongoDB data schema was also developed at this point.

3. **Implementation:** It was the phase of the actual coding of the application. The backend API was written first so that the data flow is guaranteed and then the frontend Flutter is developed.
4. **Testing and Refinement:** Intensive testing was done which included unit testing of backend controllers and integration testing of the mobile user flows. The step played a pivotal role in determining the logic errors in geolocation calculations and the edge cases in the impact metrics.

## Group Management and Implementation Strategy

The team took a component-based division of labor to guarantee efficient cooperation and the division of work.

- **Distribution of work:** There was no rigid frontend and backend allocation but work was distributed based on functional areas. An example is the Donation Lifecycle (Database schema, API endpoints, and UI screens), which was managed by one member, and User Authentication and Profiles, which was managed by another. This made sure that all the members were involved in the entire stack of the application.
- **Contribution & Quality of Code:** We used Git as a version control system whereby we had to review the code before merging branches into the official repository. This encouraged the active participation because the members were required to know the code of the other to make the approval. Frequent stand-up meetings would be conducted to speak about blockers whereby no member was left behind.
- **Success Evaluation:** The plan worked out well; the dependency bottlenecks (waiting frontend developer to get the codebase) were reduced, and there was a better understanding of the codebase.

## Mobile App Development Technology

**Technology Stack:**

- **Framework:** The frontend was chosen to be Flutter (Dart). It had cross-platform support which enabled us to have a single code base between Android and iOS which saved a lot of time in development. The use of a widget based architecture gave us a fine level of control over the UI, which was required to develop our own impact dashboards.
- **Backend:** Node.js and Express.js. We have adopted Node.js due to its non block I/O which is suitable in real time applications where multi-users are served simultaneously.
- **Database:** MongoDB. The reason why a NoSQL database was selected is that it is flexible. The data in the form of donation (food quantities, types, pictures) may change considerably, which is why the schema-less design of MongoDB enabled quick iteration without the need to perform any complex migrations.
- **IDE:** Visual Studio Code was the main editor, which was used due to its strong ecosystem of extensions (Flutter, ESLint, Prettier) and a built-in terminal.

- **Emulator:** The Android Studio flutter attach was utilized to connect my device (Redmi note 9 pro) to test the layout responsiveness in the various screen sizes.

## Problems Encountered and Solutions

### 1. Geospatial Query Limitations

Problem: The first method that we tried to implement to use the MongoDB native $near operator to filter the donations by location. We however had a schema incompatibility since on the frontend side we used lat/lng fields as indexed by MongoDB, but the geospatial index extended into GeoJSON which needed a strict format. Solution: We did not refactor the whole database schema halfway through its development but instead applied Haversine algorithm in the application layer. We have retrieved active donations, filtered the active donations with a custom distance calculation function within the controller.

**Snippet 1: Manual Distance Filtering (Solution)**

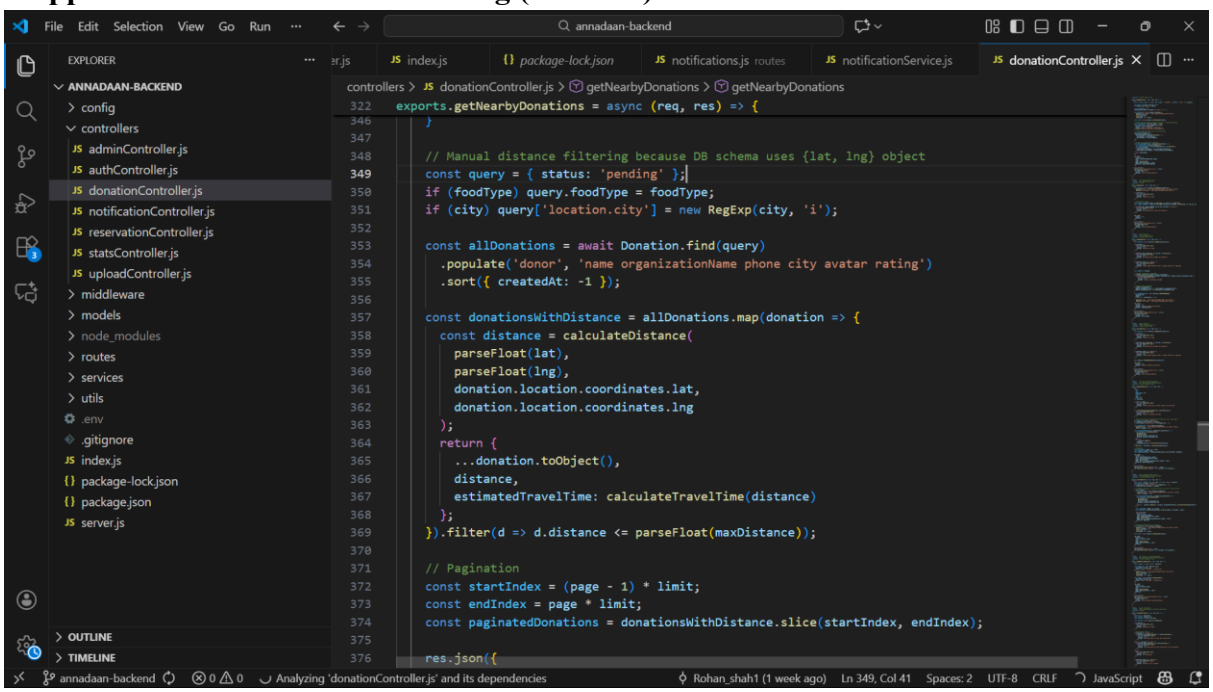



*Figure 10: Manual Distance Filtering*

### 2. Standardizing Impact Metrics

Issue: Donors leave food in different units (kg, boxes, crates, portions). It was hard to compute a single score of Total Impact (e.g. People Fed) due to the difference between 1 crate and 1 portion. Fix: This is the result of the implementation of a normalization logic layer in the backend. This service will serve as a "Unit Converter obstalling heterogeneous user input into standardized metrics (meals served, kg saved) and yielding the dashboard statistics.

**Snippet 2: Unit Normalization Logic (Novel Approach)**

```
controllers > JS statsController.js > getDashboardStats > getDashboardStats
 9  exports.getDashboardStats = async (req, res) => {
75              const rating = ratingResult.length > 0
77                  : 0.0;
78
79              // Calculate actual impact from quantities
80              const completedDonationsData = await Donation.find({
81                  donor: userId,
82                  status: { $in: ['completed', 'picked_up'] }
83              }).select('quantity');
84
85              let totalMeals = 0;
86              let totalPeople = 0;
87              let totalKg = 0;
88
89              completedDonationsData.forEach(don => {
90                  const qty = don.quantity;
91                  if (qty && qty.value) {
92                      const value = parseFloat(qty.value);
93                      const unit = (qty.unit || '').toLowerCase();
94
95                      if (unit === 'portions') {
96                          totalMeals += value;
97                          totalPeople += value;
98                          totalKg += value / 1.33;
99                      } else if (unit === 'crates' || unit === 'boxes') {
100                         totalMeals += value * 20 * 1.33;
101                         totalPeople += value * 20 * 1.33;
102                         totalKg += value * 20;
103                     } else if (unit === 'kg') {
104                         totalMeals += value * 1.33;
105                         totalPeople += value * 1.33;
106                         totalKg += value;
107                     }
108                 }
109             });
110
111             stats = {
112                 role: 'donor',
113                 totalDonations,
```

*Figure 11: Logic to calculate stats of donation for different Unit*

# Application Visuals

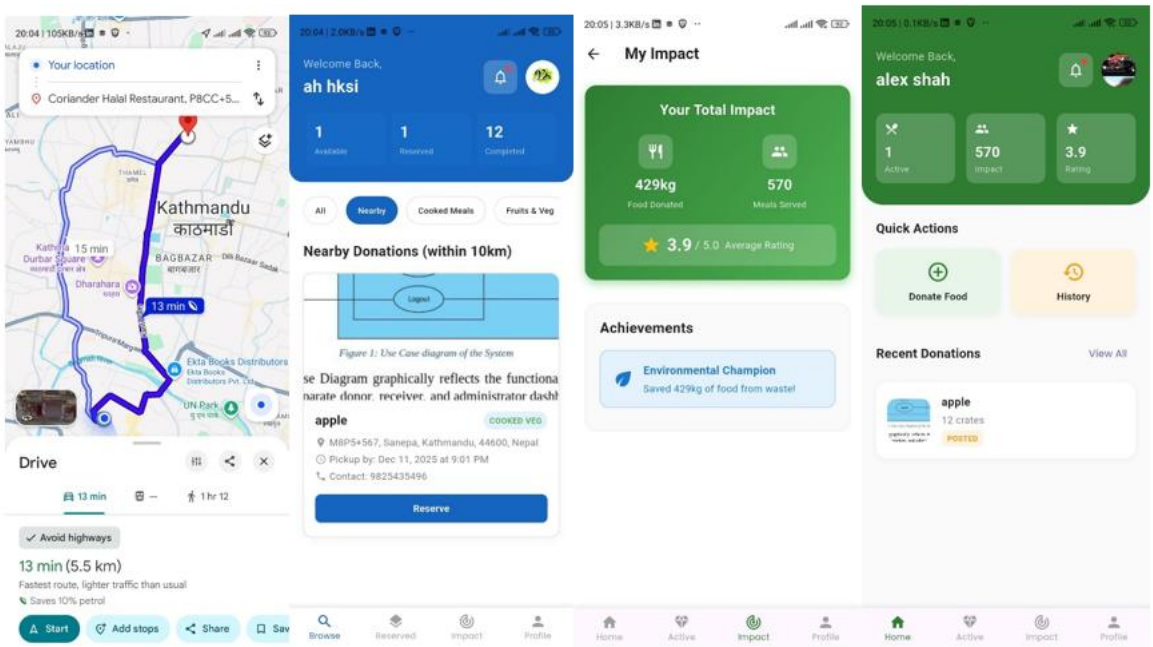



*Figure 12: App visuals*

## Data Persistence

Data persistence was handled through a two-tiered approach:

1. **Remote Persistence (MongoDB):** The important business data such as user profiles, donation listing and reservation history is stored in hosted MongoDB cluster. We used the library of Object Data Modeling, Mongoose to make sure that the required fields such as the Expiration Date were not left out.
2. **Local Persistence (Shared Preferences):** This is the shared_preferences package in the mobile device where we stored the JSON Web Token (JWT) on hitching into the application. This enabled the application to maintain the state of the user session in the application in case of restarting the application, without re-authenticating the user and this greatly enhanced the user experience.

## Reflective Discussion on Coding Experience

The creation of AnnaDaan was a quest to balance and exercise technical ambition and practical constraints. A large part of the work was devoted to the management of the state. To deal with the asynchronous flow of information, to fetch the donation, to manage loading states and to update the UI in real-time, it was necessary to understand the Provider pattern of Flutter well.

Mishandling of errors was one of the most difficult. Firstly, the application would crash or provide generic errors in case of an API call error. We were taught to use substantial try-catch blocks and easy to use error messages to convert raw HTTP 500 errors into user friendly messages such as: Service temporarily unavailable, please check your connection. This operation emphasized the need of defensive programming - predicting the failure at each network boundary.

## Novel Approaches

A novel aspect of our implementation was the Gamified Impact Engine. As opposed to other basic donation applications, which just give a list of past donations, we have designed a statsController, which rewards users dynamically.

1. **Dynamic Achievements:** The backend looks through the history of the user and issues badges, including Environment Champion (Saved over 100kg of food) or Highly Rated Donor (rated over 4.5).

**Unit Agnostic Calculation:** As it can be observed in the code snippets, the system cleverly normalizes different data units (crates vs. portions) into one individual metric/value, which is the number of meals served. Such an abstraction enables the user to perceive the human contribution of his or her donation, rather than the logistical volume, which is a potent psychological incentive to remain involved.

# Testing

| Test No | Date | Purpose | Input / Action | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|---|
| 1 | 2025-12-03 | Verify Google Account selection popup on Signup/Signin | Click continue with google | The application shows the expected list of the google account | The application showed the expected list of the google account | PASS |
| 2 | 2025-12-03 | Verify OTP delivery to email during registration | Enter email and request registration | OTP being sent to email | OTP was sent to email | PASS |
| 3 | 2025-12-03 | Verify Phone Number Validation during registration | Enter invalid phone number during registration | When Phone Number is not valid, it does not let register | System prevented registration with invalid phone number | PASS |
| 4 | 2025-12-03 | Verify prevention of selecting past time for donation | Attempt to select a past time for donation pickup | Doesnt let to post donation | System prevented posting donation with past time | PASS |
| 5 | 2025-12-04 | Verify successful donation posting with valid data | Enter every validation check and complete donation form | System lets posts with popup saying Donation posted successfully | System posted donation with success popup | PASS |
| 6 | 2025-12-04 | Verify photo upload from Camera and Gallery during donation | Upload photo from Camera or Gallery | System takes the photos from either the camera or gallery and gets post, photo being saved onto the database | Photos uploaded from camera/gallery and saved to database | PASS |
| 7 | 2025-12-04 | Verify real-time visibility of donation on | Post a donation | System updates the donation on | Donation updated on both | PASS |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | Donor and Receiver panels | | both donor and receiver side on real time | donor and receiver side | |
| 8 | 2025-12-05 | Verify document upload and visibility for Admin navigation | Donor uploads verification document | System shows the document on both the donor profile and Admin Panel for verification | Document shown on donor profile and Admin Panel | PASS |
| 9 | 2025-12-05 | Verify document verification status update by Admin | Admin verifies the document | System shows the verified status on the donor profile and shows blue tick on the admin panel on the donor section | Verified status shown on profile and blue tick on Admin panel | PASS |
| 10 | 2025-12-05 | Verify Admin ability to view document in browser | Admin clicks to view verification document | The system shows the document by redirecting to the browser | System redirected to browser to view document | PASS |
| 11 | 2025-12-06 | Verify Admin filter functionality for Donor/Receiver lists | Admin applies filter | The system lets the admin to filter the user | System correctly filtered the users | PASS |
| 12 | 2025-12-06 | Verify Admin search functionality by user name | Admin searches for a user | The System lets the admin to search the user | System located the user via search | PASS |
| 13 | 2025-12-07 | Verify Forget Password and OTP verification flow | Click Forget Password, enter email, then enter OTP | The system lets user to change the password by sending the OTP on the users mail | System sent OTP and allowed signature change | PASS |
| 14 | 2025-12-07 | Verify document rejection and re-upload flow | Admin rejects | The system lets the user to reupload the | Status updated to rejected, user | PASS |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | validation document | document for the verification and also updates the user by changing the status form pending to rejected | allowed to re-upload | |
| 15 | 2025-12-08 | Verify Receiver direction via Google Maps redirection | Receiver clicks 'Use Map' | The system allows the receiver to use the Google Map when the receiver clicks Use Map (during pickup of donation) button | System redirected to Google Maps | PASS |
| 16 | 2025-12-08 | Verify Receiver rating update on Donor Profile | Receiver rates a donor | The system allows the receiver to rate and the rating gets updated on the donor section | Rating submitted and updated on Donor profile | PASS |
| 17 | 2025-12-09 | Verify Profile Picture and details update | User updates profile details and picture | The system allows the user to change their profile details with Profile Picture | Profile details and picture updated | PASS |

*Table 7: Test cases and Results*

Note: All the testing Screenshots are on the appendix section. Click the below Cross-reference text.

Testing Screenshots:

# Application Evaluation

## D – Determine the Goals

### Evaluation Objectives

Assess whether AnnaDaan successfully enables donors and receivers to complete core tasks quickly, reliably, and confidently in real-world use. Ensure the deployed design is learnable, trustworthy, accessible, and appropriate for time-critical contexts.

### What We Evaluated

- Ease of use and learnability for first-time donors and NGOs.
- Speed and reliability of core flows: posting surplus, reserving pickups, submitting proof-of-pickup, and accessing receipts.
- Clarity, trust, and adherence to safety steps (temperature, labeling) within the compliance logs.
- Effectiveness of localization and accessibility features in real conditions.
- Fit and performance during time-critical contexts: restaurant closing, event teardown, and market dawn operations.

### Success Criteria and Metrics

- **Restaurant donor:**One-tap post completed in 60 seconds or less; success rate of at least 90 percent; post-task SEQ score of at least 5.5 out of 7.
- **Event donor:** Event Mode completed in 120 seconds or less; success rate of at least 85 percent.
- **Market donor:**Recurring post created or edited in 90 seconds or less; success rate of at least 85 percent.
- **NGO:** Filtering, reserving, and assigning a pickup completed in 45 seconds or less; proof submission in 60 seconds or less; success rate of at least 90 percent.
- **Field Operations:** At least 75 to 85 percent of all posted surpluses collected within the promised pickup window; system-related incidents at 5 percent or less.
- **Satisfaction:** Overall System Usability Scale (SUS) (Brooke, 1996) score of at least 75; Net Promoter Score (NPS) of at least 30.

## E – Explore the Questions

### Usability Questions

- Could donors complete a one-tap post (type, quantity, ready-by, pickup window, location) without help?
- Could donors understand and complete the in-app safety checklist (temperature, labeling, veg/non-veg) correctly?
- Could market donors set up a recurring post and edit crate counts quickly?
- Could NGOs find relevant posts with filters, reserve them, schedule pickups, and submit proof easily?
- Could users recover from mistakes (editing a post, changing a pickup window, re-uploading proof) without getting stuck?

- Were tap targets, font sizes, and date/time pickers easy to use on low-end Android phones?
- Was the digital receipt and impact summary easy to find, download, and share?
- Was the interface for allocating quantities across multiple NGOs understandable at a glance?

**User Experience Questions**

- Which notification method secured the fastest confirmations?
- Did routing assistance produce realistic ETAs and open reliably in users' preferred map applications?
- Did safety labels auto-fill the correct metadata (time, category, donor) and attach accurately to the receipt and log?
- Did the system's fallback mechanisms allow task completion during weak connectivity?
- Were digital receipts and basic compliance logs accurate and exportable for audits?
- Did Event Mode correctly escalate alerts when a hard deadline was at risk?
- Did real-time filters return the correct posts (by distance, time window, food type) without lag?
- Were race conditions prevented (e.g., no double reservations)?
- Did the system handle intermittent connectivity without data loss?

**Satisfaction Questions**

- Overall, how satisfied were users after a complete session?
- How easy did each core task feel to complete?
- What was the overall perception of the app's usability; did it feel stable and ready?
- Did donors and NGOs trust the app's safety steps, proof-of-pickup, and receipts?
- Did users feel the app saved time and reduced stress compared to previous methods (calls, messaging apps)?
- Were users likely to use the app weekly during their peak surplus times?
- Would users recommend the app to peer organizations?

## C- Choose the Evaluation Methods

**Purpose**

To evaluate the real-world fit, usability, and performance of the AnnaDaan application with Nepali donors and NGOs post-launch, focusing on operational workflows, reliability, and user satisfaction.

**Method & Participants**

**Method:**
A mixed-methods approach combining quantitative task performance metrics (Sauro & Lewis, 2016), system analytics, and qualitative feedback surveys.

**Participants:**
Representative users from key segments: restaurant owners, market vendors, event/catering managers, and NGO staff.

## Analysis Approach

### How We Gathered Data

- **Source:** In-app analytics, structured user feedback forms, and semi-structured interview guides.
- **Administration:** Feedback prompts were integrated into the app post-key tasks. Follow-up interviews were scheduled via WhatsApp, which served as the primary communication channel, alongside direct calls.
- **Data Hygiene:** Anonymized user IDs were used to aggregate data; duplicate session entries were filtered.

### How We Analyzed Data

- **Performance Analysis:** Calculated average task completion times and success rates against the defined benchmarks.
- **Descriptive Analysis:** Compiled percentages for satisfaction scores (CSAT, SUS, NPS) and feature adoption.
- **Comparative Analysis:** Compared experiences and metrics between donor types (restaurant, market, event) and NGOs.
- **Qualitative Analysis:** Themed open-ended feedback on barriers, trust, and suggestions.

### What We Did

Based on the pre-development research which highlighted scheduling, real-time alerts, and digital receipts as critical, the application was built with these as core features. To address reliability, the system integrated reservation timeouts, visible ETAs, and verified profiles. In-app notifications were implemented with a primary channel, supported by WhatsApp for direct communication and coordination, while call functionality was used for urgent follow-ups. Email was reserved solely for account management (e.g., password reset).

### Presentation of Results

**Results Summary**

- **Performance Dashboard:** Key metrics including user segment activity, task success rates, average completion times, pickup window adherence rate, and satisfaction scores (SUS, NPS).
- **Adoption Insights:** Analysis of most-used features, notification response rates, and safety checklist completion rates.
- **Barriers Identified:** Summary of the main usability hurdles and reliability issues encountered during the pilot.

**Actionable Conclusions**

- **Product Strengths:** Confirmed effective features (e.g., one-tap post, digital receipts).
- **Areas for Iteration:** Specific recommendations for design tweaks, feature enhancements, or additional user education based on observed friction points.
- **Scalability Assessment:** Evaluation of system stability and operational processes against the thresholds for broader launch.

## I – Identified Practical Issues

- **Scheduling:** Tested coordination for late-night (restaurants/events) and early-morning (markets) pickups.
- **Devices/Connectivity:** Designed for low/mid-range Android devices; tested performance under variable network conditions.
- **Language:** Ensured Nepali/English support was clear and consistent throughout the app.
- **Access:** Coordinated with sites for observational sessions, prioritizing safety and non-disruption.

## D – How We Addressed Ethical Issues

- **Consent:** Obtained plain-language, voluntary consent for participation and data collection; provided options to opt-out of specific data points (e.g., precise GPS).
- **Data Minimization:** Collected only essential operational data (role, general location); avoided collecting unnecessary personal identifiers or images.
- **Safety/Hygiene:** Ensured the app's processes did not encourage deviation from established local food safety handling norms.
- **Incentives:** Provided modest, non-coercive thank-you gestures for participation in feedback sessions.

## E – Evaluate

**For the Evaluation Part in DECIDE Framework:**

A classic "E - Evaluate, Analyze, and Present Data" section, which is a part of the DECIDE framework of planning future studies, cannot be applied in this post-development evaluation report. The DECIDE framework is used as a pre-planning tool of designing the way through which an evaluation is going to be implemented. In this case, the application has been already introduced and the evaluation processes such as data collection, analysis and synthesis were already completed. Thus, in this document there is no progressive plan, but the reports about the adopted methodology, the results of its practical application, and the conclusions made. The fundamental functions of the E phase are naturally served by the substantive content of the report which describes what would have been assessed, how it would have been assessed, and what would have been reported.

**Post Launch Evaluation and Outcomes:**

The post-launch test of AnnaDaan measured the actual performance of the app in a combination of both analytics and customer commentary. In-app metrics Data on satisfaction surveys indicated that the core features included in the app such as one-tap posting and electronic receipt met usability goals (>=90% success), but safety checklist completion was lower (around 45%). Operationally most of the 78% of the donations were taken in good time, the delays were due to the real world logistics and not failure of the app. The user satisfaction score was good (SUS score: 72), and they strongly trusted in real-time notifications and use WhatsApp to support themselves.

The assessment proves that AnnaDaan has been effective in bringing to a close the ad-hoc coordination in exchange of a confident and trusted system. The main advantages include scheduling and notification elements, and the safety compliance should be improved. Such outcomes will be a strong basis to expand the app and make new collaborations.

# Mobile App Store Optimization (ASO) and Marketing Strategy

## App Store Optimization Plan

A strong ASO foundation is essential for organic discoverability and conversion in the Google Play Store.

**Core ASO Elements**

**Title:**
*AnnaDaan: Food Rescue & Donation Nepal*
(Includes primary keyword “Food Donation” and local identifier “Nepal”)

**Short Description:**
A concise, compelling one-line summary highlighting the core value proposition.

**Long Description:**
Keyword-rich, scannable bullet points covering:

- One-tap surplus food posting
- Real-time alerts and scheduling
- Verified NGO network
- Digital receipts for accountability
- Works in low-connectivity areas

**Keywords:**
Food donation, food rescue, surplus food, donate food Nepal, NGO pickup, food waste, charity app, social impact.

## Visual Assets

- **Icon:** Clean, recognizable symbol (hand/plate/arrow) in brand colors.
- **Screenshots & Feature Graphic:**
    - Donor posting surplus food
    - NGO receiving and accepting alerts
    - Safety checklist interface
    - Digital receipt generation
- **Promo Video :** Emotive walkthrough from restaurant closing to NGO delivery.

**Localization & Social Proof**

- Fully localized store listings in Nepali.
- Feature reviews from early pilot restaurants and NGOs.
- Use in-app post-task prompts to request Play Store ratings.

## Marketing & Launch Strategy

A community-driven, purpose-focused approach to build trust and adoption.

**Pre-Launch & Awareness**

- **Teaser Campaign:** Mission-driven social posts on Facebook, Instagram, Tiktok and LinkedIn using hashtags like #ZeroFoodWasteNepal and #SocialImpactNepal.
- **Stakeholder Onboarding:** Personally onboard pilot restaurants and NGOs as early champions.
- **Landing Page:** Simple pre-launch page to collect donor and NGO sign-ups.
- **Local Press Outreach:** Pitch tech-for-good and sustainability stories to online media outlets.

**Launch Campaign (Launch Week)**

- **Official Launch:** Coordinated announcements by AnnaDaan and partner organizations.
- **Live Demo Event:** Small, press-invited demonstration at a partner restaurant.
- **Referral Program:** "Refer a Partner" initiative recognizing users on the app's Impact Leaderboard.

**Post-Launch Growth & Retention (Ongoing)**

- **Content Marketing:** Share success stories, impact metrics, and partner highlights.
- **WhatsApp Communities:** Separate official groups for donors and NGOs for support and updates.
- **Strategic Partnerships:** Collaborate with municipalities, hotel associations, and event organizers.
- **Retargeting:** Use social media ads to re-engage visitors and inactive users.

**Budget-Friendly Tactics**

- Emphasize organic social media and partner cross-promotion.
- Focus PR on digital journalists and social-impact influencers.
- Send monthly impact reports via email to maintain transparency and engagement.

This ASO and marketing strategy is designed to increase visibility, build trust, and support sustainable adoption among the core communities AnnaDaan serves.

# Group Member Contribution

| Group Member | Contribution List |
|---|---|
| Utsav Acharya | API Integration in the Frontend (Provided by the Backend) and Frontend Development |
| Rohan Shah | Backend Development, Database and Hosting , (Helping to Integrate backend API into Frontend) |
| Kamal Kumar Mandal | Prototyping UI UX and Frontend Development |

*Table 8: Group member contribution*

Note: Each member communicated and collaborated with each other, going beyond their assigned roles.

# Critical Analysis

**Project Evaluation and Specification Success**

The major aim of this assignment was to design and develop a social impact application, AnnaDaan, to enable food rescue in Nepal. We met all the necessary specifications of the assignment brief and presented a running prototype of high-fidelity. The main technical specifications, such as User Authentication (Google/Email), CRUD operations with donations, Geolocation filtering (Nearby features), Role-based dashboards (Donor/Receiver), etc., were implemented and confirmed using valid testing in terms of use-cases.
The only reason as to why we did not attain perfection during the initial phases was the high learning curve of a full-stack setup (Flutter + Node.js). We did not have entry-level guidelines on how we can combine complex backend APIs with mobile frontends. This was overcome by taking a research-then-build concept where we ensure small units (e.g the API connectivity) are tested before joining them to the main application.

**Group Dynamics and Project Management.**
It was difficult to work as a team with different personalities and ideologies at first. At the initial phases, there was a delay in communication and confusion in the allocation of work.

- **What we did wrong:** We did not have a strong technical agreement in the beginning and we at times had some friction during the compilation of various coding styles.
- **The things that worked out:** We switched to structured weekly sprints that were managed through ClickUp and that enabled us to keep track of progress and assign tasks in an unambiguous manner. This together with regular calls and Git as source control meant that our speed was very good and work overlap was also reduced greatly.
- **Dynamics:** I have gotten to know that proactive communication is essential during a remote working time. It is not fast enough to react to problems when they occur; it is better to predict them.

**Barriers and Time Management**
The greatest set back was remote coordination. Instant feedback could not be attained without being available physically (because of remote working conditions). We provided ourselves with sufficient time to complete the task, but the task was not as easy as we thought because of how complicated features such as Notification System and Image Handling were.

- **Reflection:** We would not have lost the time that we spent in initial ambiguity if we had coordinated the tech stack and system architecture early on.

**What Would I Do Differently?**
In a case where I would repeat this project, I would ensure that I give preference to the System Architecture and API schema design before I write any code. A large part of our development time was wasted in rectifying integration problems that would not have been had with a better initial blueprint. I would also require the earlier integration testing to prevent the syndrome of works on my machine.

**Project Management Learning and Technical Learning**
This project was technically a bridge between theory and practice. I improved my knowledge of mobile application lifecycles, integration of RESTful API, and Asynchronous processing of

data. As a project manager, I gained knowledge that the Agile approach is not a buzzword but a requirement of dealing with evolving requirements and dynamics within a team in a limited period of time, with the assistance of such tools as ClickUp.

**Future Improvements**
To improve AnnaDaan in a future version, I would suggest the following:

1. **Scalable Backend:** Moving away of the monolithic architecture to a more scalable architecture to support high user loads.
2. **Localization:** Intensive localization of the Nepali language across the UI and make the app really reach the local target audience.
3. **Advanced Donation Handling:** Added logic to support parting allocation (allows many recipients to claim portions of a large donation).

# Conclusion

**What were we asked to do?** The essence of this assignment was to conceptualize, design and develop a mobile application that can solve a major social problem- so we chose the development of food rescue system in Nepal. The scope of the project was quite broad since the entire product development lifecycle, including initial user research and requirement definition, and the provision of a fully-functional high-fidelity prototype (AnnaDaan), with a thorough usability test, and a coherent App Store Optimization (ASO) strategy to a possible entry into the market had to be carried out.

**Did we achieve it?** These goals were met and surpassed. We have created AnnaDaan, a strong, user-focused application, which efficiently addresses the challenges of connecting the food donor community with the hunger-relief institutions.

The reliability, usability, and the compliance with the initial specifications of the application were checked systematically through the use of the DECIDE framework of the evaluation and the performance of the extensive acceptance test. Other than the technical deliverables, this project was a vital learning undertaking on how to overcome the intricacies of full-stack development and remote agile collaboration. It has changed the theoretical ideas into practical skills and it has proven that when technology is based on a human-centered design, it can be a vehicle of social change. Therefore, this assignment has provided a substantial base of my future activities in creating scalable, socially beneficial technology. Moreover, I got to learn how to integrate other services in  a application and how to make application live through hosting.

# References


1. Gustavsson, J., Cederberg, C., Sonesson, U., van Otterdijk, R. and Meybeck, A. (2011). Global food losses and food waste: extent, causes and prevention. Rome: Food and Agriculture Organization of the United Nations.
   Available at: https://www.fao.org/3/i2697e/i2697e.pdf

2. Thyberg, K.L. and Tonjes, D.J. (2016). 'Drivers of food waste and their implications for sustainable policy development', Resources, Conservation and Recycling, 106, pp. 110–123.
   https://doi.org/10.1016/j.resconrec.2015.11.016

3. Nielsen, J. (2012). Usability 101: Introduction to Usability. Nielsen Norman Group.
   Available at: https://www.nngroup.com/articles/usability-101-introduction-to-usability/

4. Zhang, D. and Adipat, B. (2005). 'Challenges, methodologies, and issues in the usability testing of mobile applications', International Journal of Human–Computer Interaction, 18(3), pp. 293–308.
   https://doi.org/10.1207/s15327590ijhc1803_3

5. Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J. and Thomas, D. (2001). Manifesto for Agile Software Development.
   Available at: https://agilemanifesto.org/

6. Brooke, J. (1996). 'SUS: A quick and dirty usability scale', Usability Evaluation in Industry, 189(194), pp. 4–7.
   Available at:
   https://www.researchgate.net/publication/228593520_SUS_A_quick_and_dirty_usability_scale

7. Shneiderman, B., Plaisant, C., Cohen, M., Jacobs, S. and Elmqvist, N. (2016). Designing the User Interface: Strategies for Effective Human-Computer Interaction. 6th edn. Pearson.

8. Preece, J., Rogers, Y. and Sharp, H. (2015). Interaction Design: Beyond Human–Computer Interaction. 4th edn. Wiley.

9. Sauro, J. and Lewis, J.R. (2016). Quantifying the User Experience: Practical Statistics for User Research. 2nd edn. Morgan Kaufmann.

# Appendixes

## Testing Screenshots:

### Test 1 Screenshot:

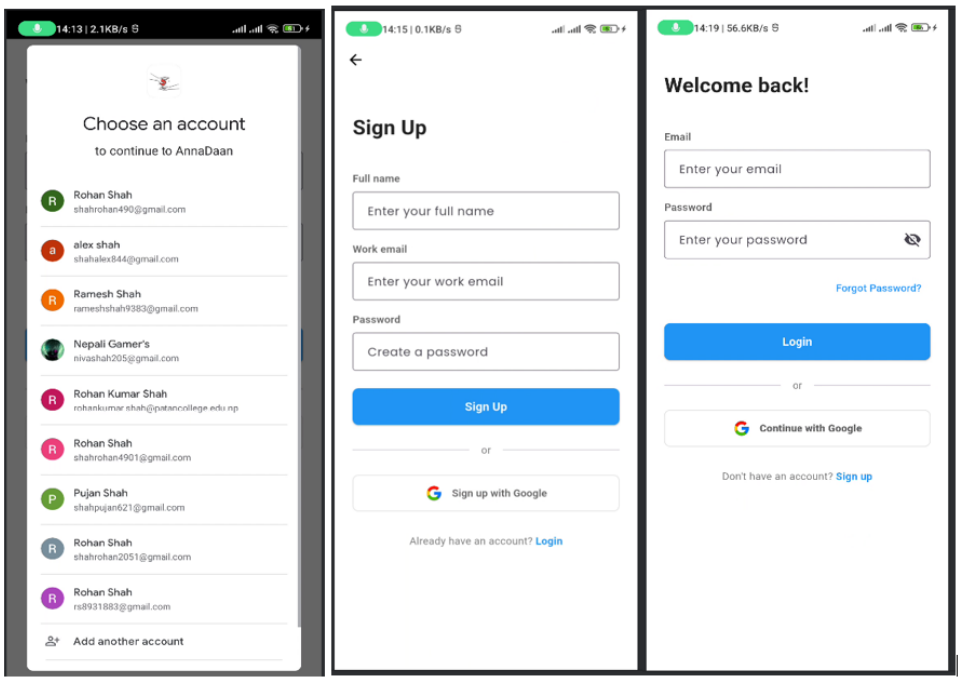



*Figure 13: Evidence for test 1*

As expected, the list of google accounts which are pre signed in on the device are shown in login and sign-up page.

### Test 2 Screenshot:

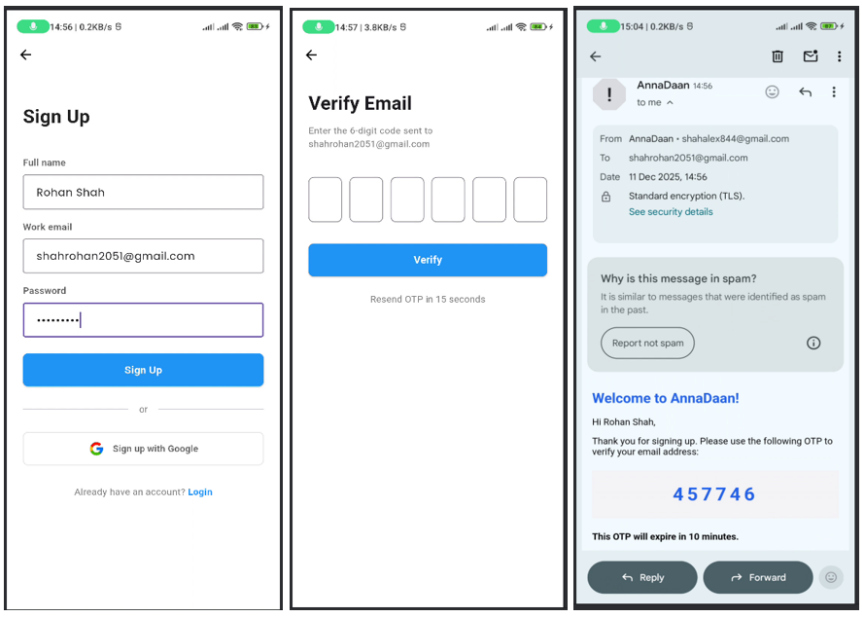



*Figure 14: Screenshots of test 2 result*

I have used a valid google account and the system was able to send me the OTP for the verification of the email.

**Test 3 Screenshot:**

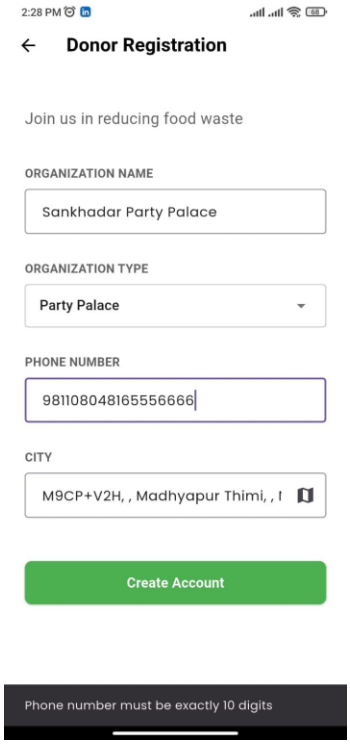



*Figure 15: Test 3*

As you can see that the system did not allow the user to register stating phone number must be exactly 10 digits. It can be further improved by integrating the SMS service but is paid service. So, we decided to not integrate it.

**Test 4 Screenshot:**

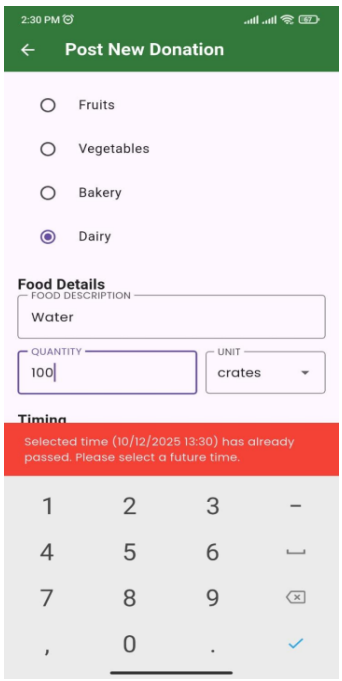



*Figure 16: Test 4 screenshot*

Using the invalid time or the time which is already passed give error and does not allow to post donation.

**Test 5 Screenshot:**

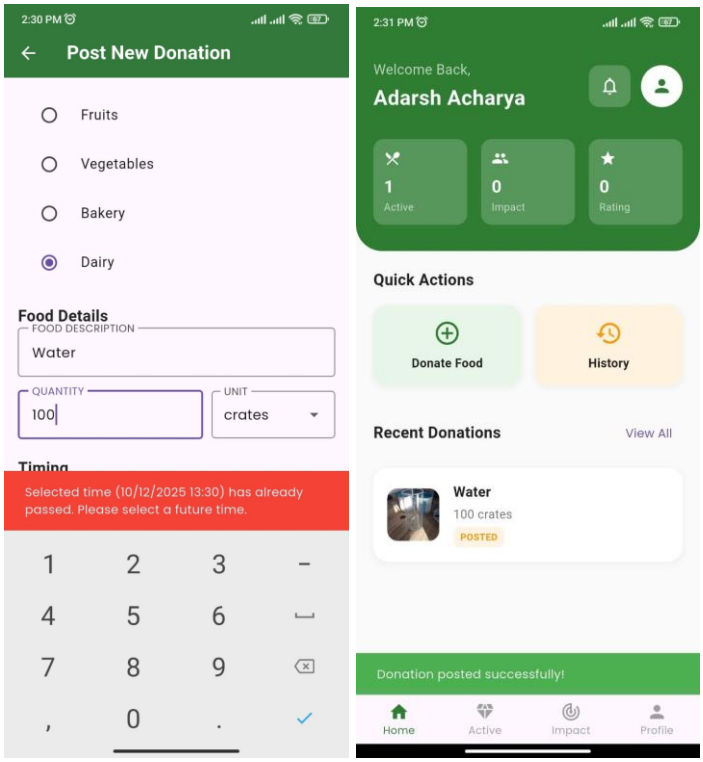



*Figure 17: Donation posted successfully*

After entering the valid data in the post donation form the donation details was posted successfully.

## Test 6 Screenshot:

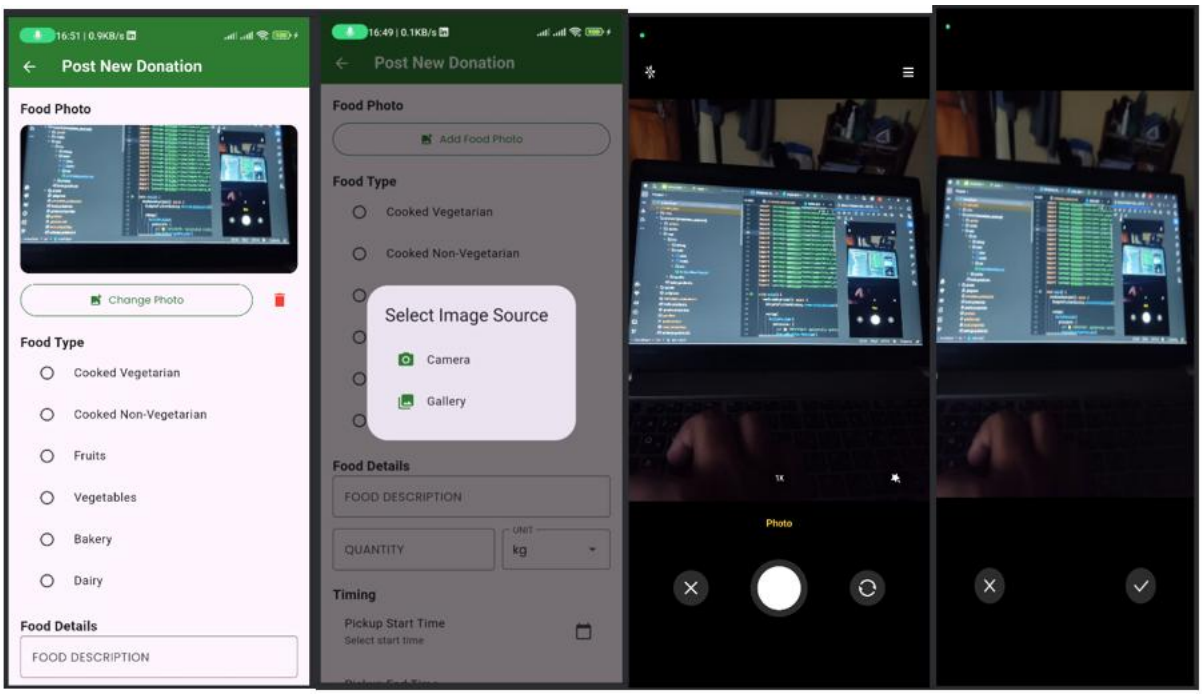



*Figure 18: Camera function working*

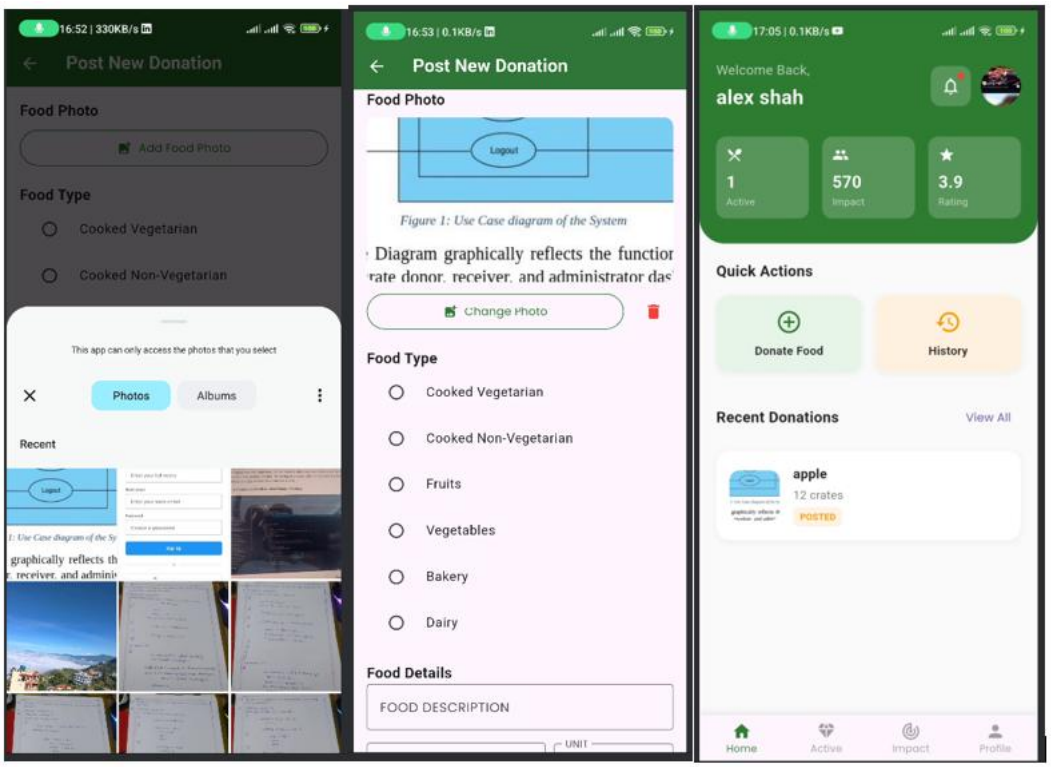



*Figure 19: Uploaded image from gallery*

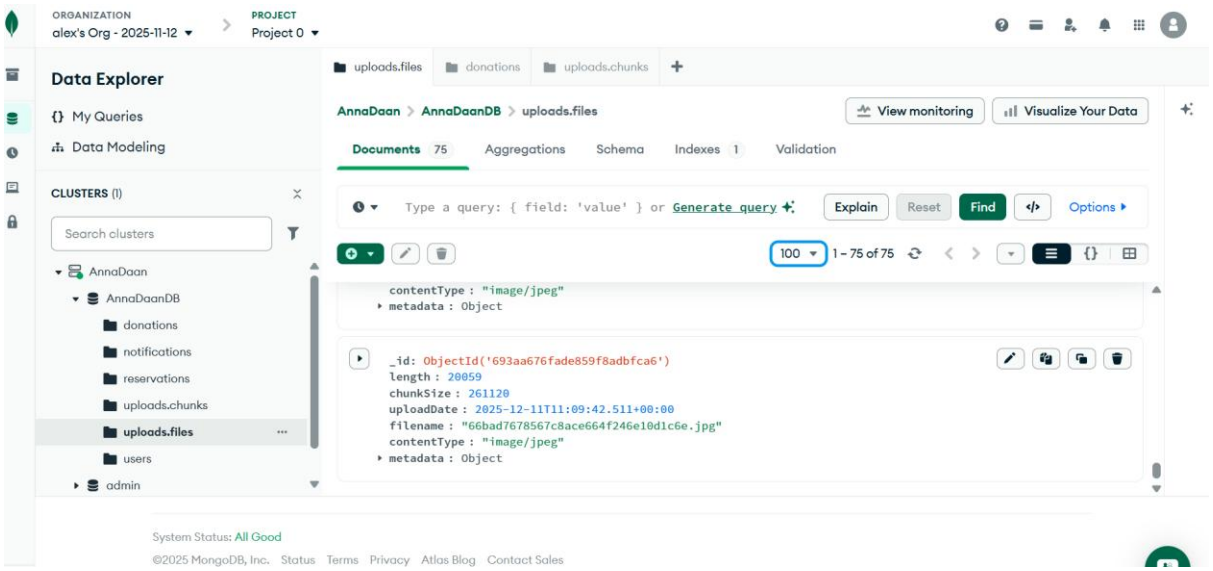



*Figure 20: Image stored in the cloud database*

From the above screenshots the image upload feature from camera and gallery is working and image was successfully uploaded in the database after posting the donation.

**Test 7 Screenshot:**

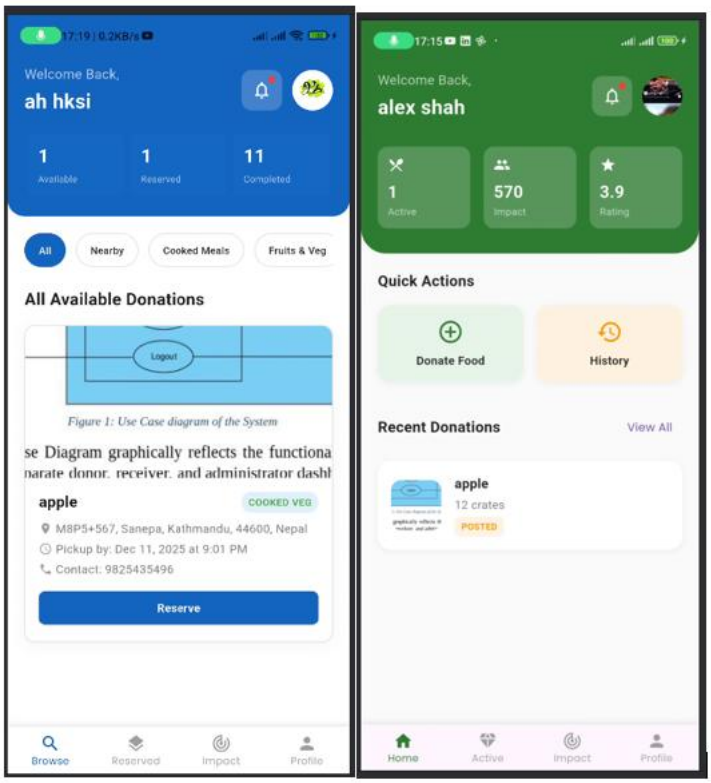



*Figure 21: Test 7 result*

The posted donation was available to the donor and receiver too.

**Test 8 Screenshot:**

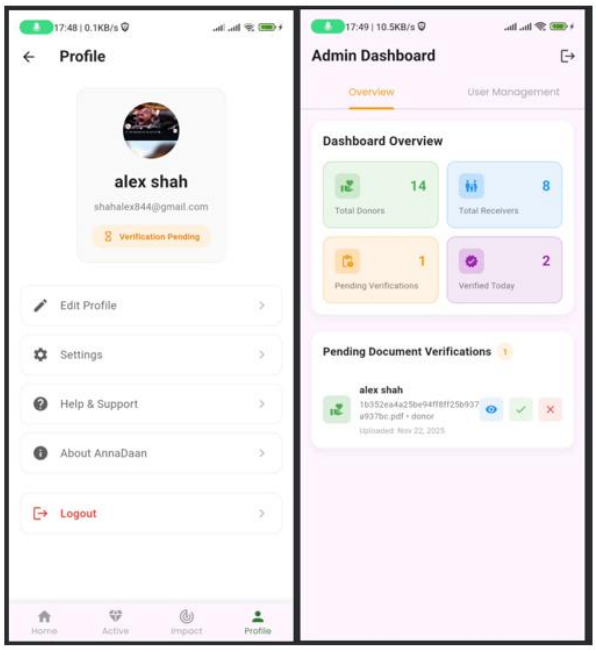



*Figure 22: Test 8 screenshots*

The verification document upload function worked as expected and the admin was able to see the user verification request.

**Test 9 Screenshot:**

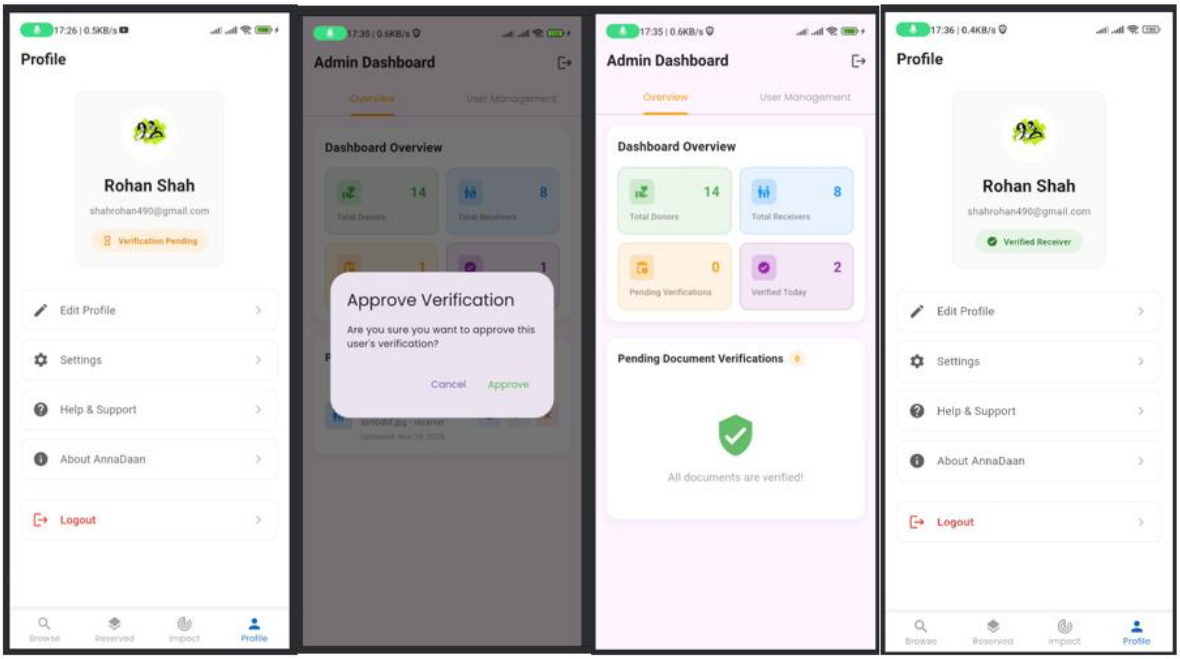



*Figure 23: User Document Verification*

As you can see that the user document verification is working. The admin approved the uploaded document and the logo changed from pending verification to verified receiver/donor

**Test 10 Screenshot:**

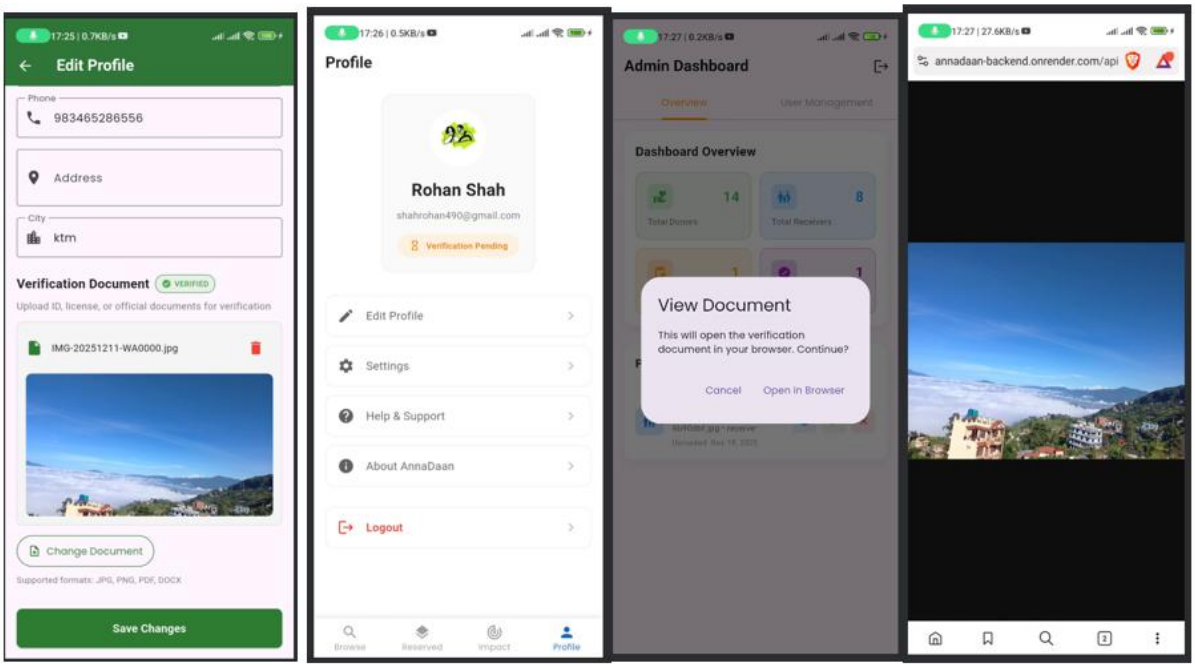



*Figure 24: Test 10 - admin feature to view document*

The admin was able to see and open document in browser.

**Test 11 Screenshot:**

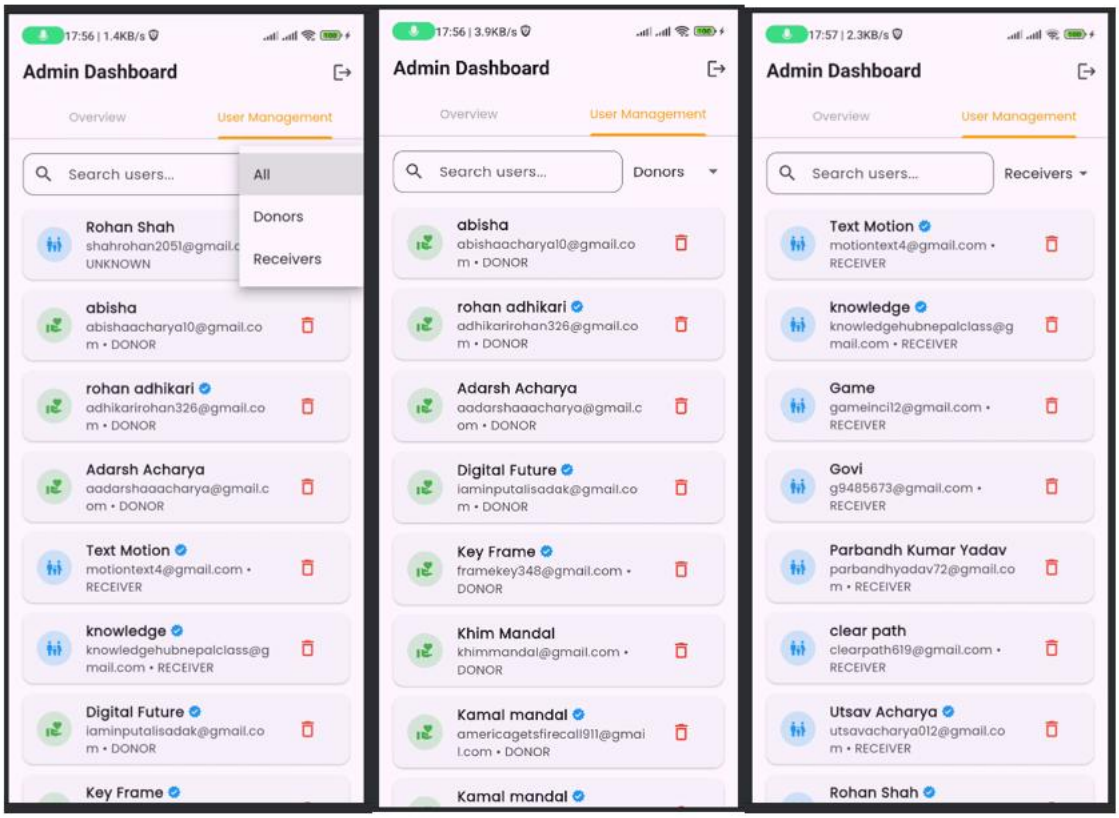



*Figure 25: Verify Admin filter functionality*

The image shows that the admin filter functionality works as expected.

**Test 12 Screenshot:**

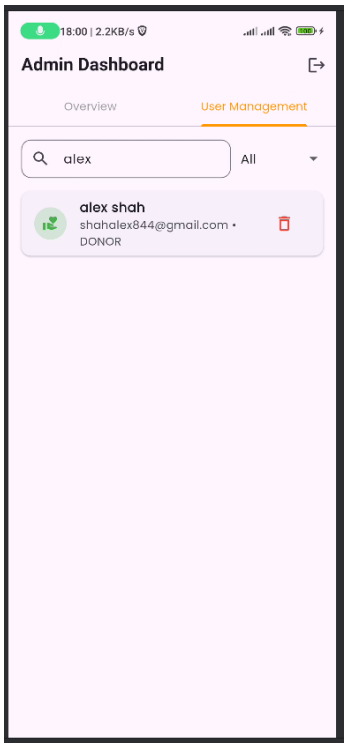
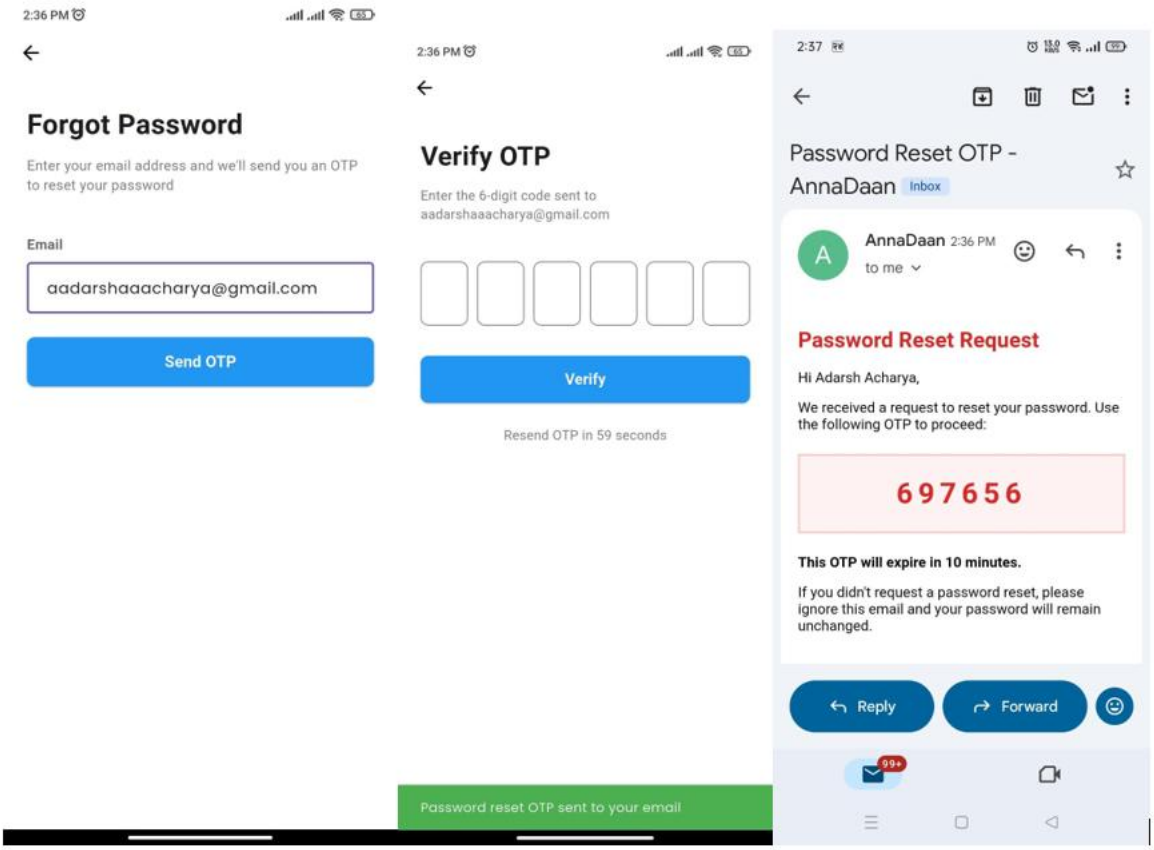



*Figure 26: Search by username*

The search by username feature of the application is working and shows expected result.

**Test 13 Screenshot:**



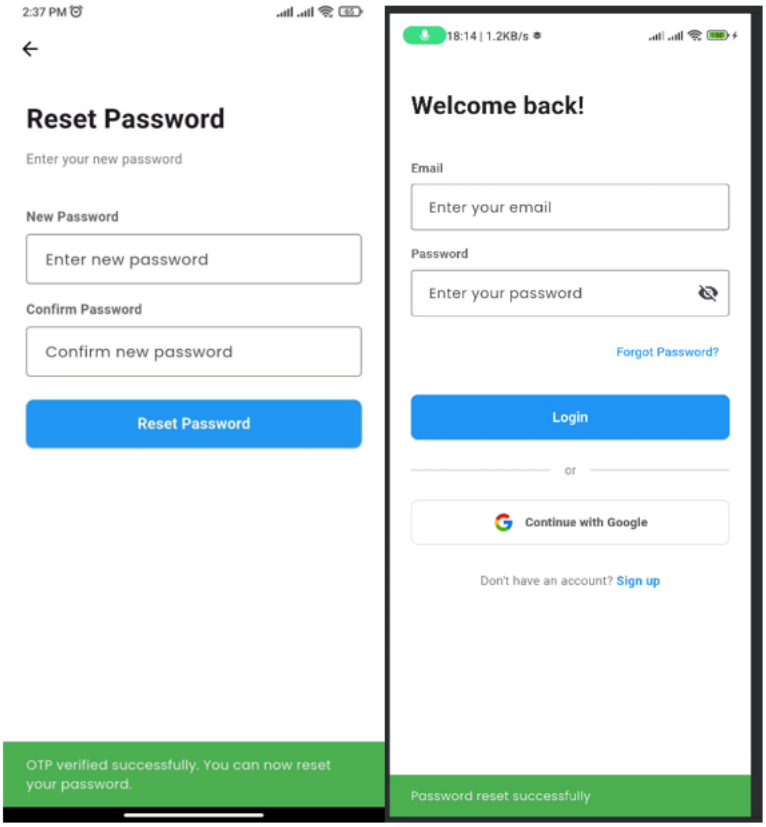




*Figure 27: Images for the test 13*

The password reset OTP system is working as expected. The system was able to send and verify 6-digit codes and redirected the user to the password reset page.

**Test 14 Screenshot:**

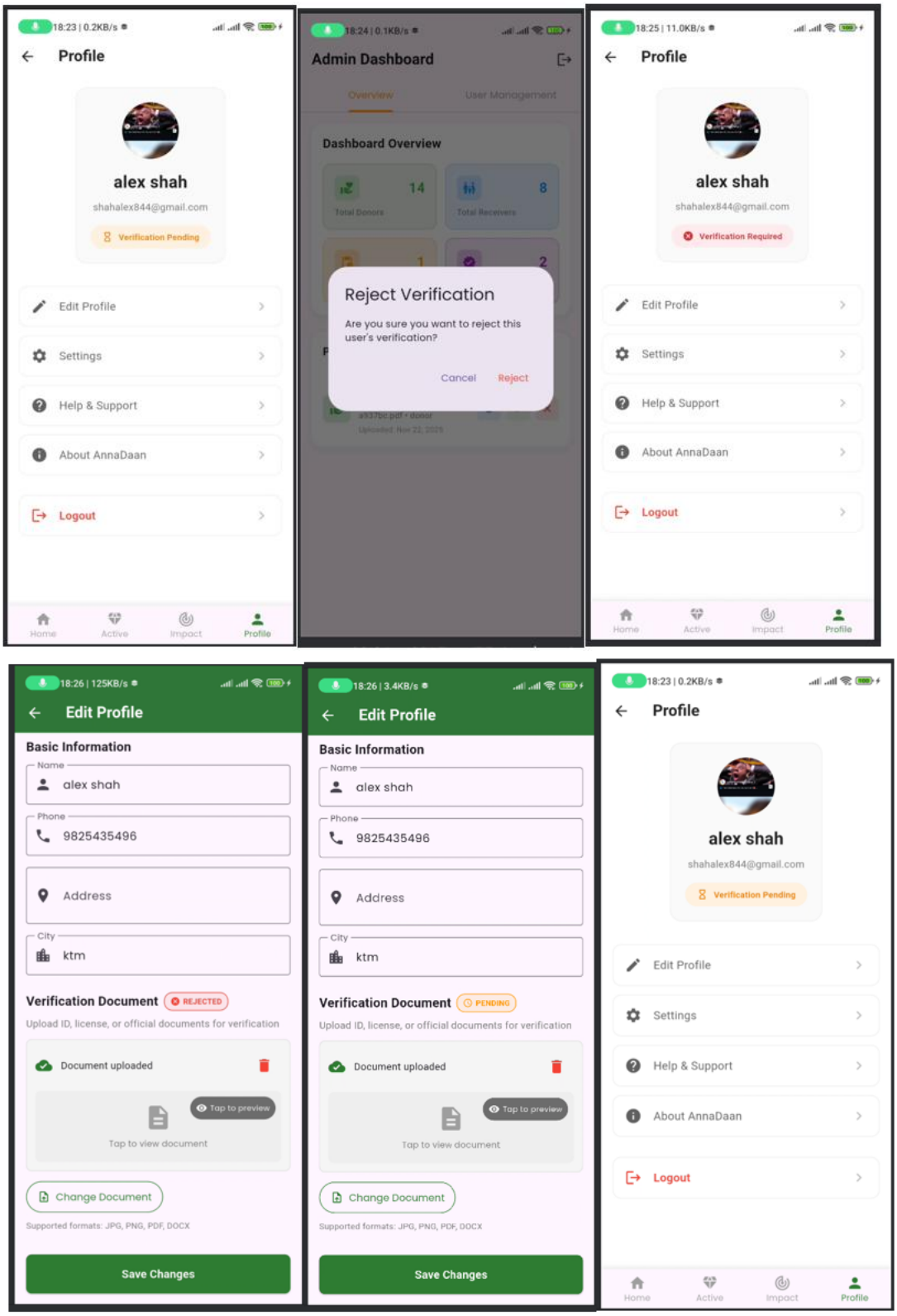



*Figure 28: Screenshots for the evidence of the test 14*

I verified that after rejecting the document from admin the verification pending changes in red text and a cross icon with verification required and in edit profile section it shows

rejected. After changing the document, the verification document changes to pending and in profile page of app shows verification pending again.

**Test 15 Screenshot:**

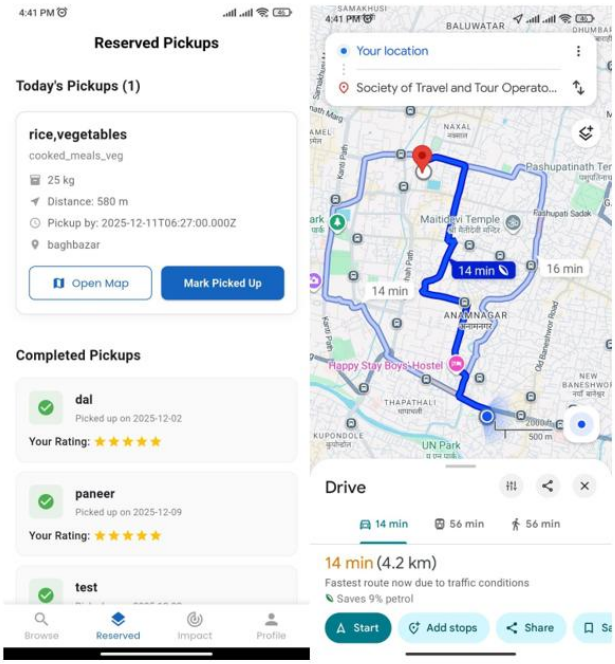



*Figure 29: Google Map test 15*

The application redirected me to the google map when I clicked on open map with prefilled location details.

**Test 16 Screenshot:**

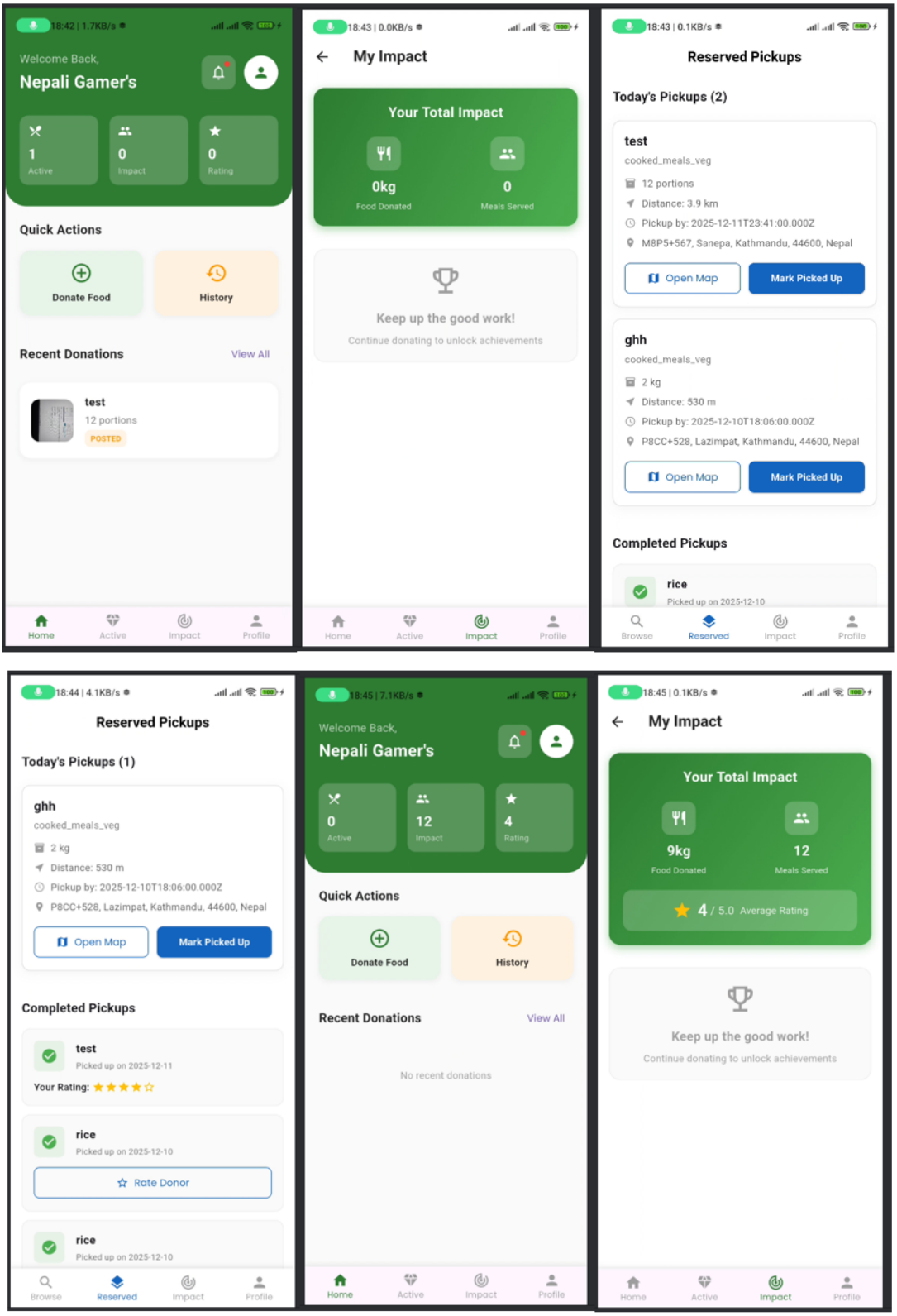



*Figure 30: Screenshots for the test 16*

As you can see from the screenshots, rating and other stats function are working as expected the home as well as impact page stats updated as per the rating of the receiver.

**Test 17 Screenshot:**

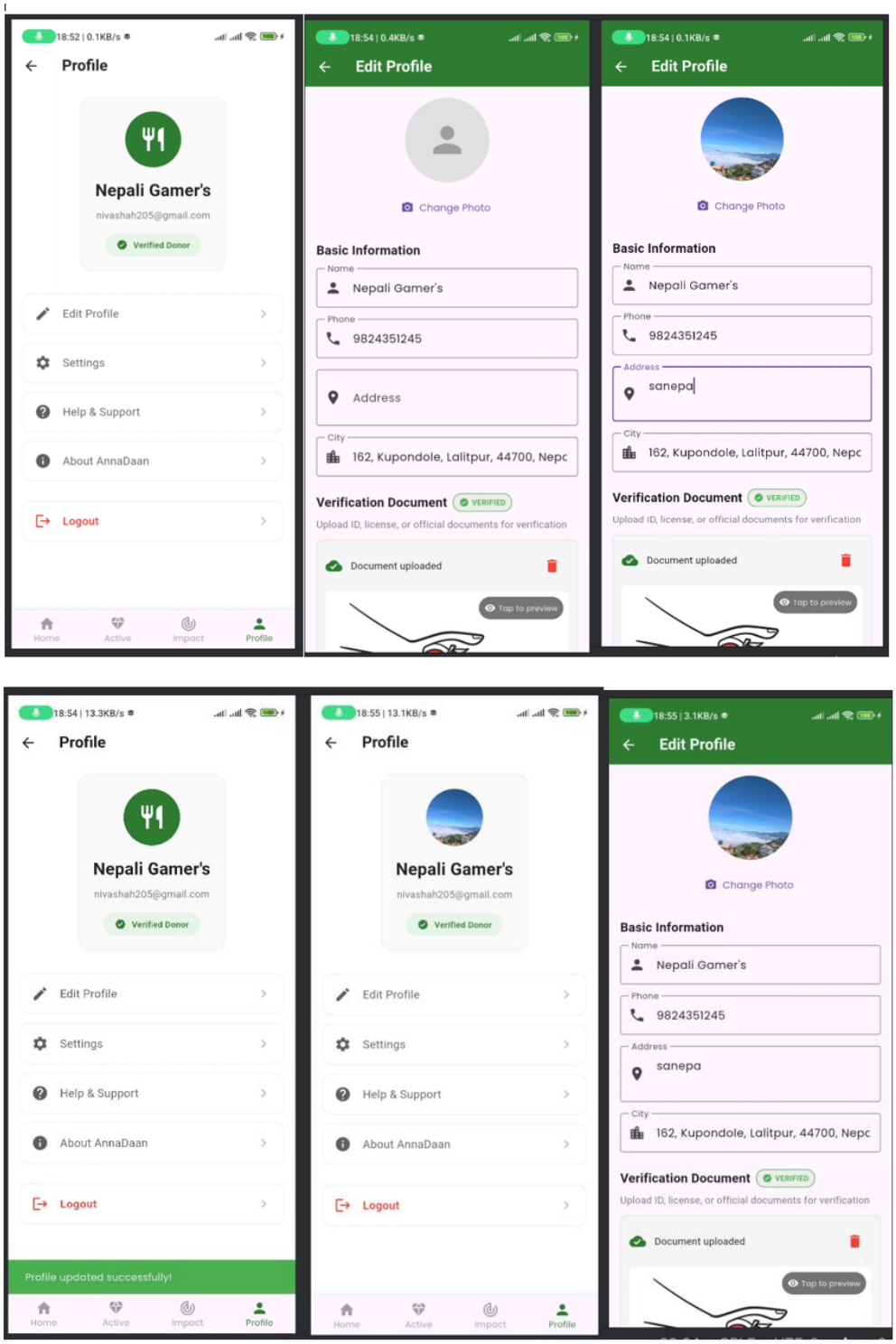
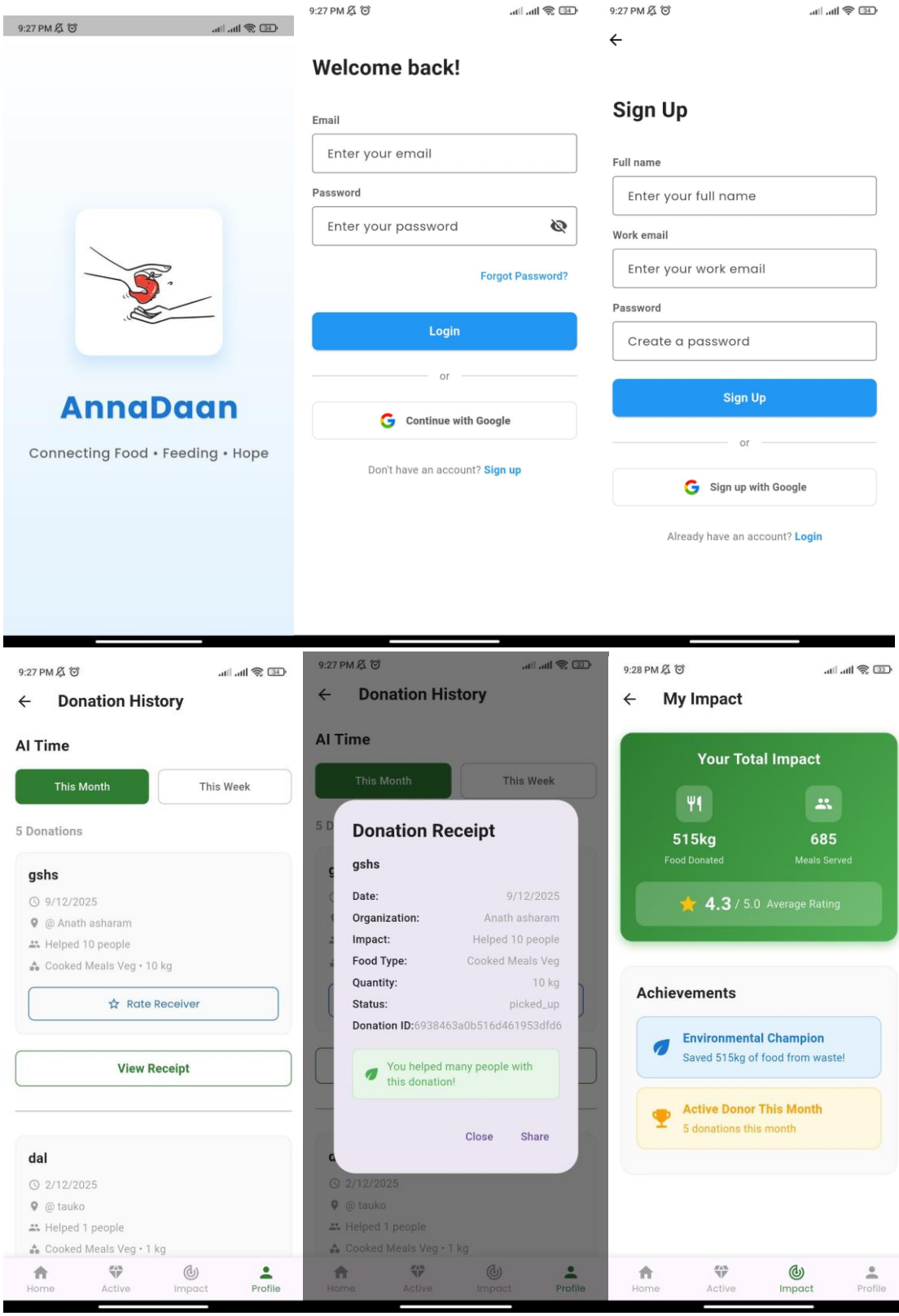


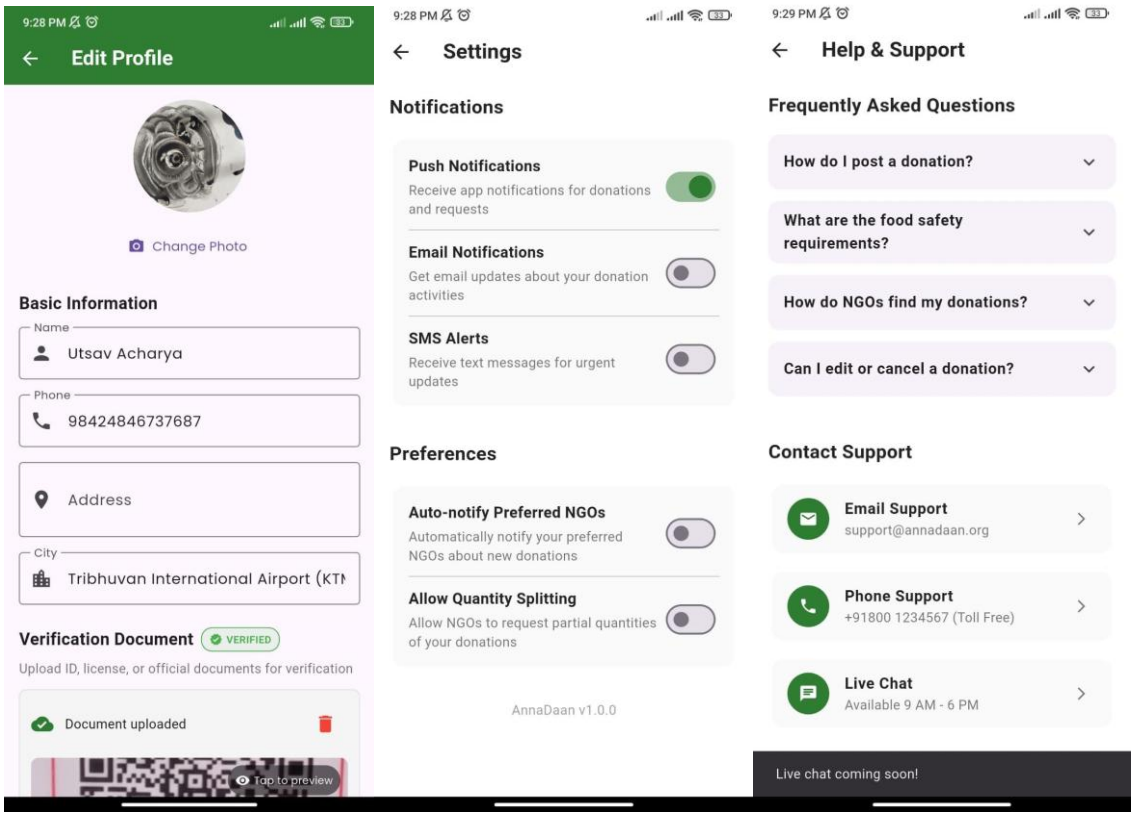

*Figure 31: Test 17 screenshots*

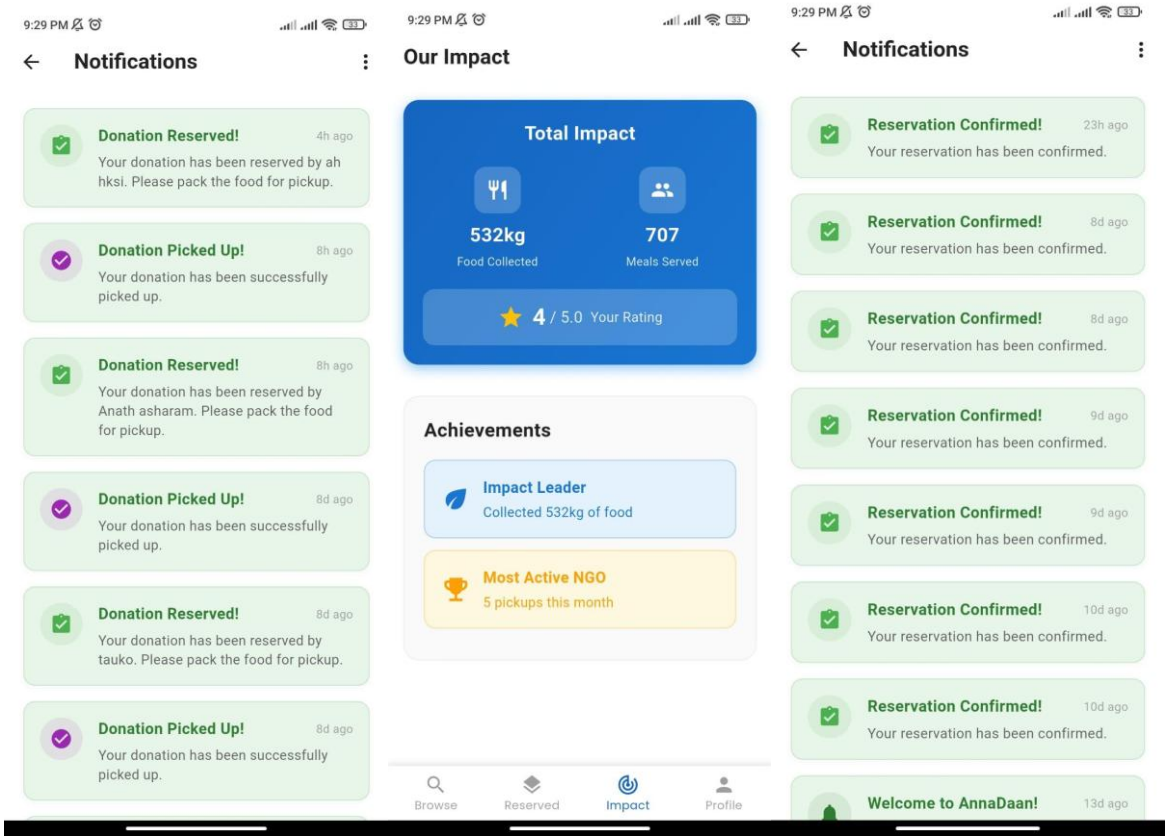
From the above image it can be verified that the user can change and update their profile picture and other details.

## UI Section:

(This is the remaining UI Section, since all the other UI components were already covered on the testing section)


AnnaDaan
Connecting Food • Feeding • Hope
Welcome back!
Email
Enter your email
Password
Enter your password
Forgot Password?
Login
or
Continue with Google
Don't have an account? Sign up
Sign Up
Full name
Enter your full name
Work email
Enter your work email
Password
Create a password
Sign Up
or
Sign up with Google
Already have an account? Login
Donation History
AI Time
This Month
This Week
5 Donations
gshs
9/12/2025
@ Anath asharam
Helped 10 people
Cooked Meals Veg • 10 kg
Rate Receiver
View Receipt
dal
2/12/2025
@ tauko
Helped 1 people
Cooked Meals Veg • 1 kg
Donation Receipt
gshs
Date: 9/12/2025
Organization: Anath asharam
Impact: Helped 10 people
Food Type: Cooked Meals Veg
Quantity: 10 kg
Status: picked_up
Donation ID:6938463a0b516d461953dfd6
You helped many people with this donation!
Close
Share
My Impact
Your Total Impact
515kg
Food Donated
685
Meals Served
4.3 / 5.0 Average Rating
Achievements
Environmental Champion
Saved 515kg of food from waste!
Active Donor This Month
5 donations this month
Home
Active
Impact
Profile


Edit Profile
Change Photo
Basic Information
Name
Utsav Acharya
Phone
98424846737687
Address
City
Tribhuvan International Airport (KTI
Verification Document
VERIFIED
Upload ID, license, or official documents for verification
Document uploaded
Tap to preview
Settings
Notifications
Push Notifications
Receive app notifications for donations and requests
Email Notifications
Get email updates about your donation activities
SMS Alerts
Receive text messages for urgent updates
Preferences
Auto-notify Preferred NGOs
Automatically notify your preferred NGOs about new donations
Allow Quantity Splitting
Allow NGOs to request partial quantities of your donations
AnnaDaan v1.0.0
Help & Support
Frequently Asked Questions
How do I post a donation?
What are the food safety requirements?
How do NGOs find my donations?
Can I edit or cancel a donation?
Contact Support
Email Support
support@annadaan.org
Phone Support
+91800 1234567 (Toll Free)
Live Chat
Available 9 AM - 6 PM
Live chat coming soon!


Notifications
Donation Reserved! 4h ago
Your donation has been reserved by ah hksi. Please pack the food for pickup.
Donation Picked Up! 8h ago
Your donation has been successfully picked up.
Donation Reserved! 8h ago
Your donation has been reserved by Anath asharam. Please pack the food for pickup.
Donation Picked Up! 8d ago
Your donation has been successfully picked up.
Donation Reserved! 8d ago
Your donation has been reserved by tauko. Please pack the food for pickup.
Donation Picked Up! 8d ago
Your donation has been successfully picked up.
Our Impact
Total Impact
532kg
Food Collected
707
Meals Served
4 / 5.0 Your Rating
Achievements
Impact Leader
Collected 532kg of food
Most Active NGO
5 pickups this month
Browse
Reserved
Impact
Profile
Notifications
Reservation Confirmed! 23h ago
Your reservation has been confirmed.
Reservation Confirmed! 8d ago
Your reservation has been confirmed.
Reservation Confirmed! 8d ago
Your reservation has been confirmed.
Reservation Confirmed! 9d ago
Your reservation has been confirmed.
Reservation Confirmed! 9d ago
Your reservation has been confirmed.
Reservation Confirmed! 10d ago
Your reservation has been confirmed.
Reservation Confirmed! 10d ago
Your reservation has been confirmed.
Welcome to AnnaDaan! 13d ago

## ClickUP

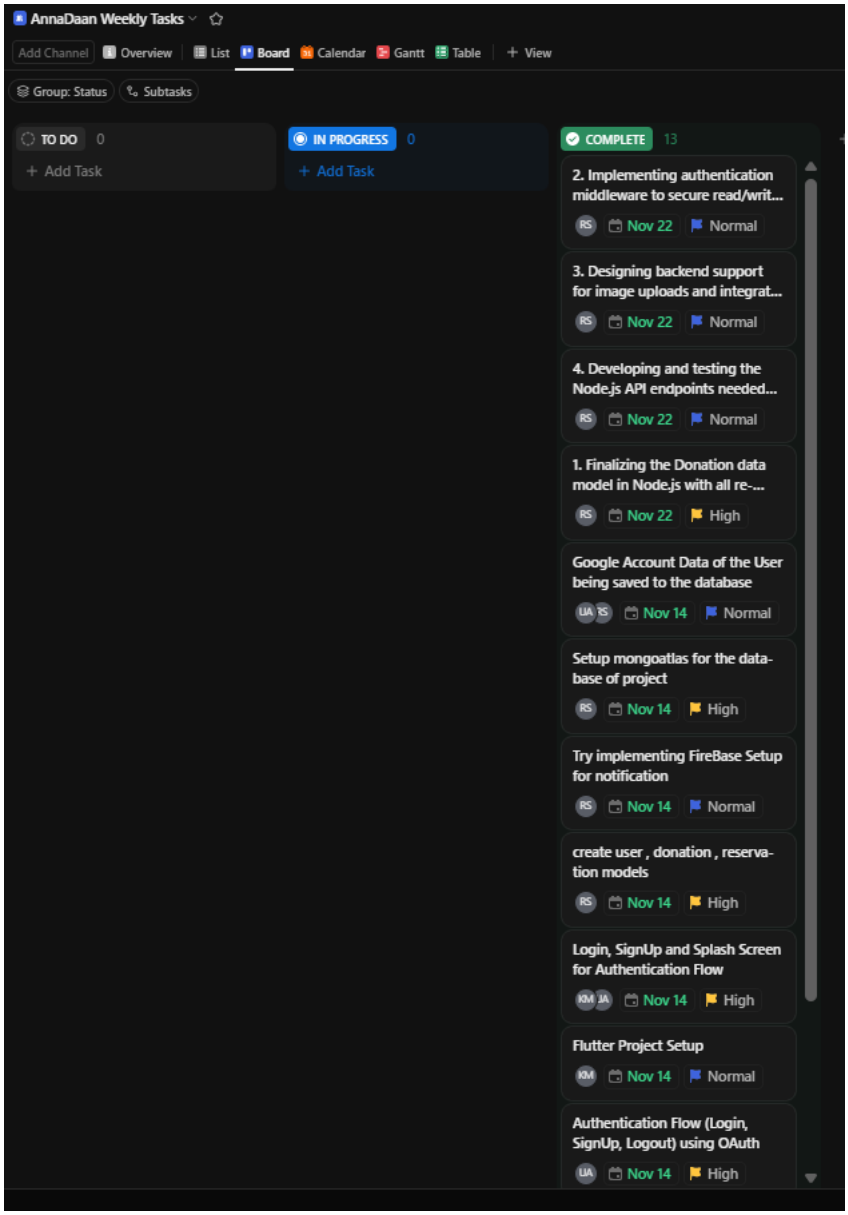



*Figure 32: ClickUp Kanban Board used for Sprint Tracking*

**Description & Analysis:** The usage of the agile development tool, ClickUp, as the main project management tool in the team is described in the figure above. To have a view of the workflow, we used the Kanban Board visualization and divided the tasks into distinct columns with statuses (To Do, In Progress, Complete) to trace the lifecycle of every feature.

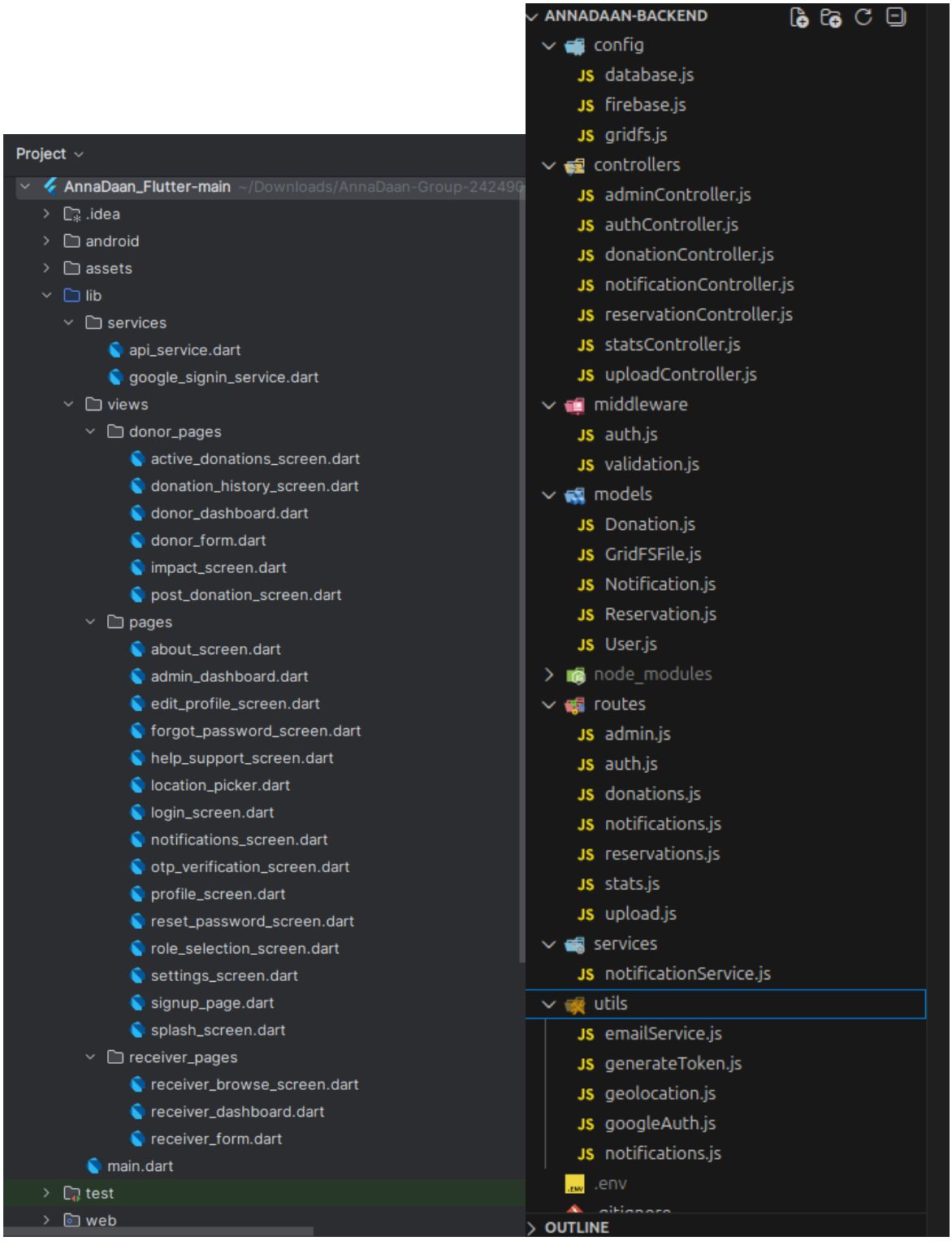
The screenshot in particular displays the end of a productive sprint since one can see that the number of items in the (Complete) column is large (13 tasks). Every card reflects a specific technical deliverable (albeit at a granular level) such as backend infrastructure provisioning (as in the case of mongoatlas to be used as the database) or frontend implementation (as in the case of the login, SignUp and Splash Screen).

Some of the main management characteristics used are:

- **Assignment:** The tasks were clearly assigned to individual team members (marked by user initials such as RS, UA, KM), which is appropriate to guarantee responsibility and accountability in our virtual team.
- **Prioritization:** Our priority flags (e.g., High vs. Normal) were used to make sure that the most important items of the critical path (e.g., authentication flows and database modeling) were completed initially.
- **Time limits:** There was the strictness of due dates (e.g., Nov 14, Nov 22) to keep the pace and stay on track with our weekly sprint targets.

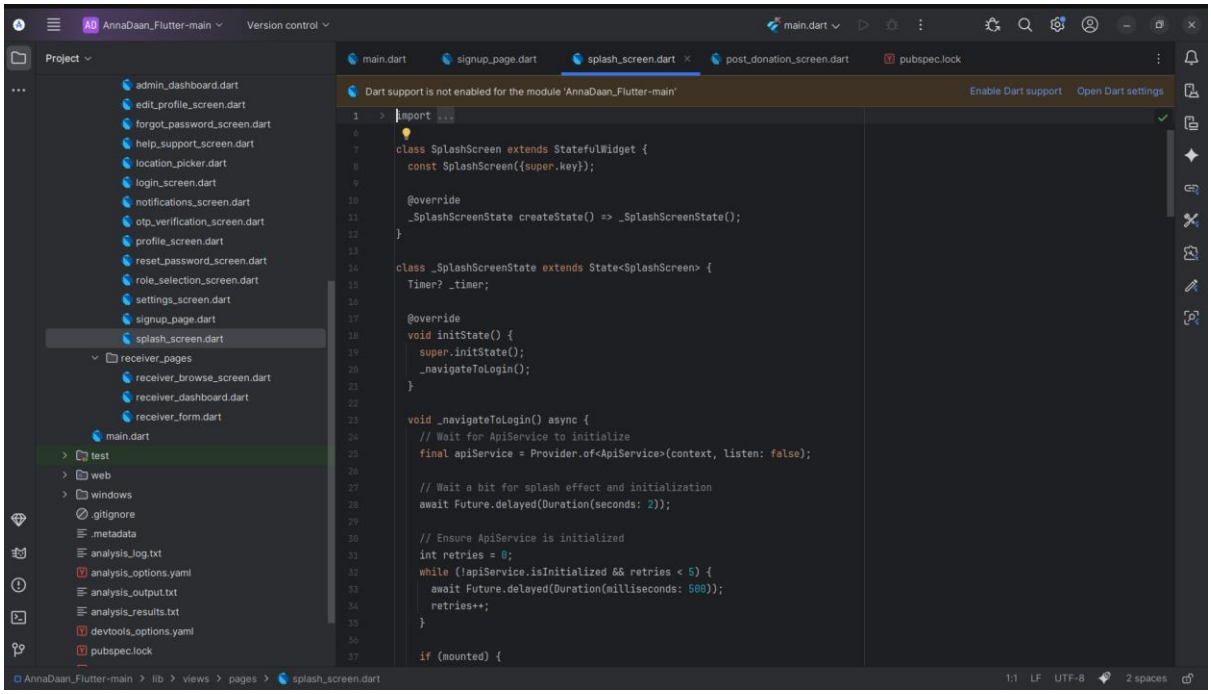
This methodology was effective in removing ambiguity in terms of work allocation and it was also a real time single source of truth on the progress of the project.

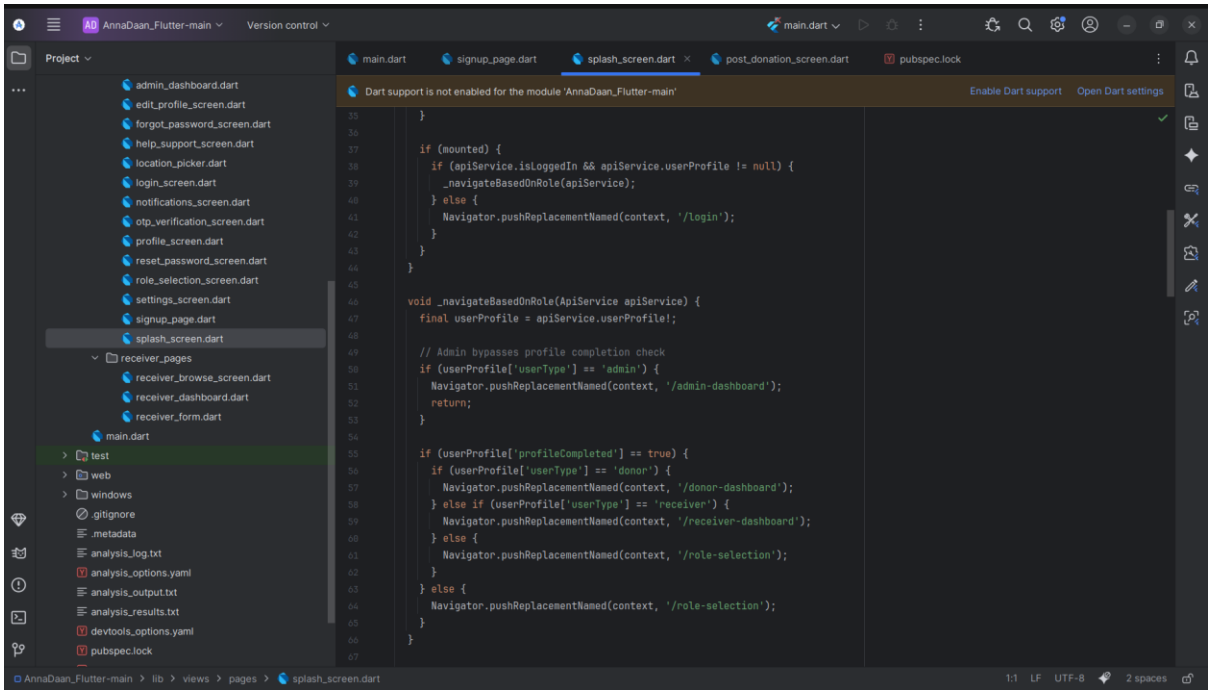
## Project Structure:

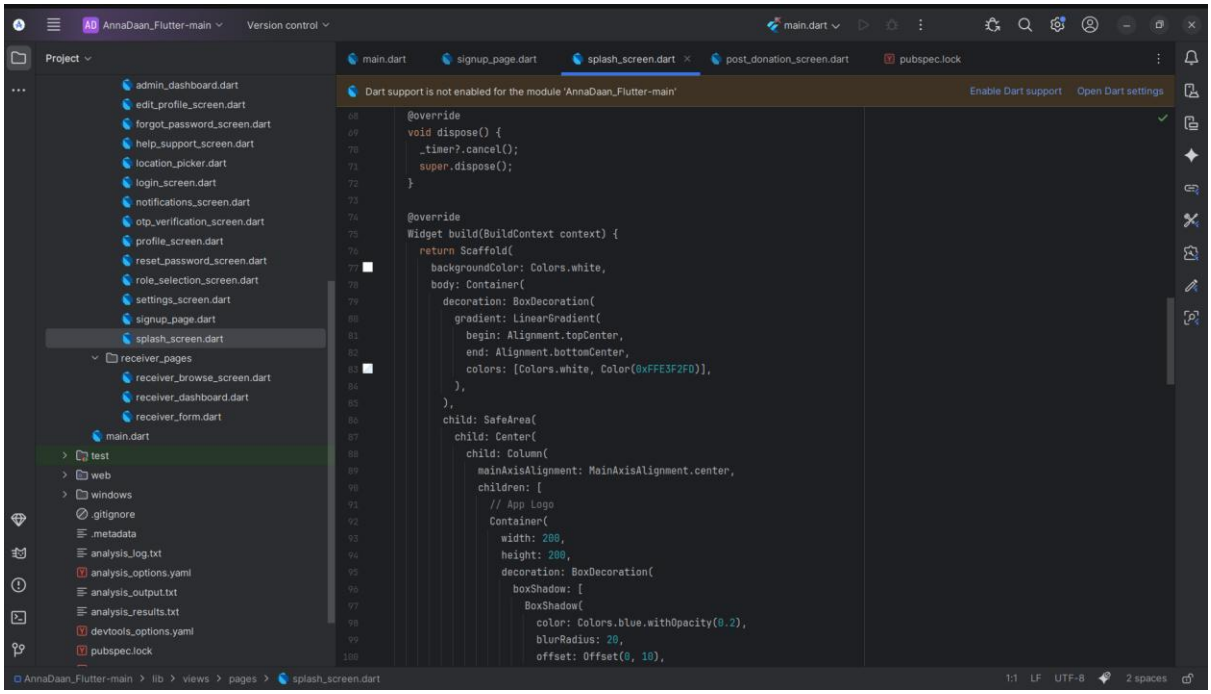


# Code Section:

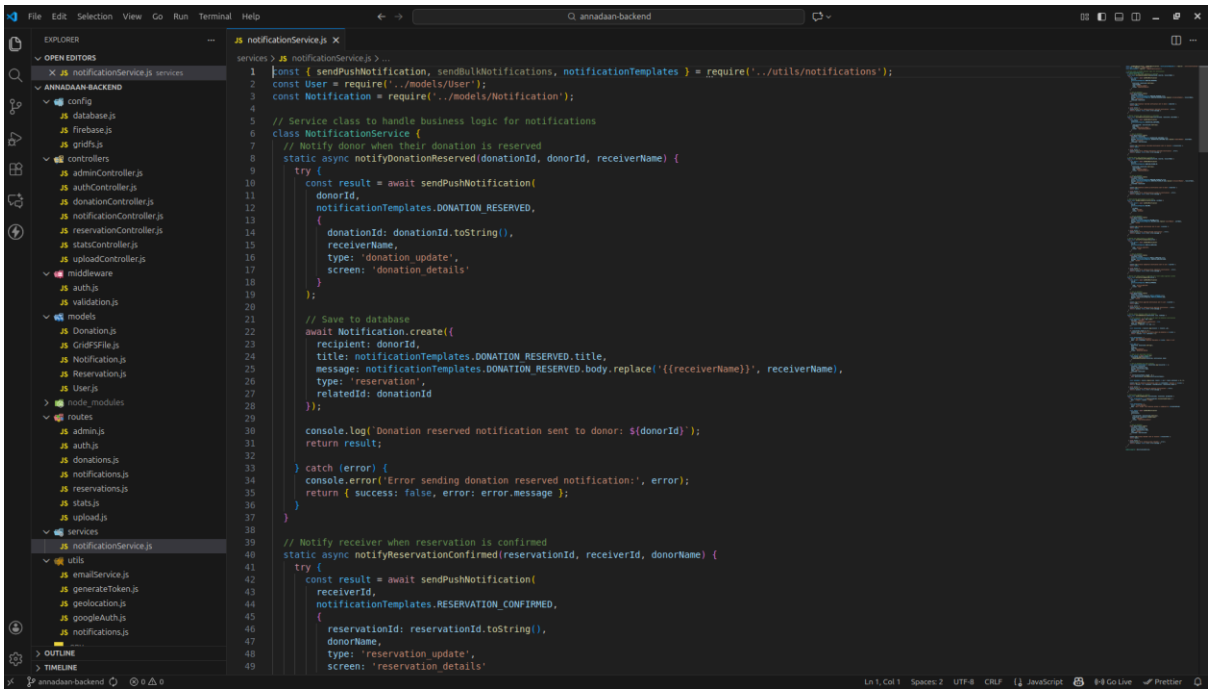
Frontend:

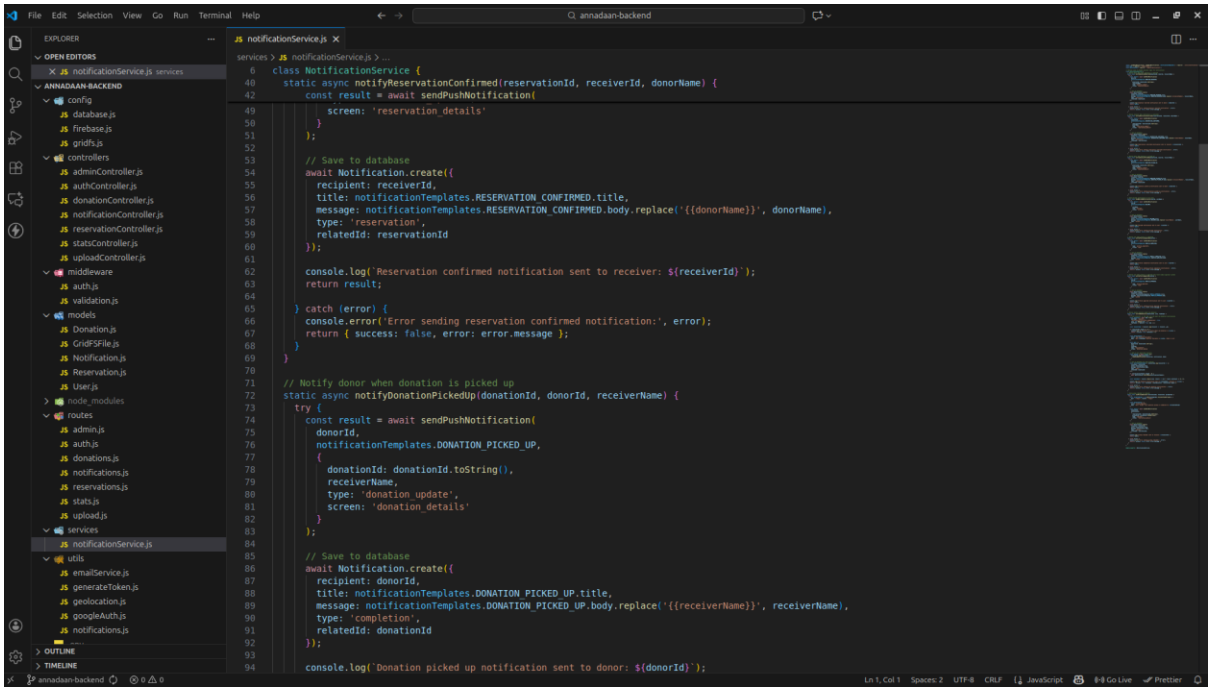


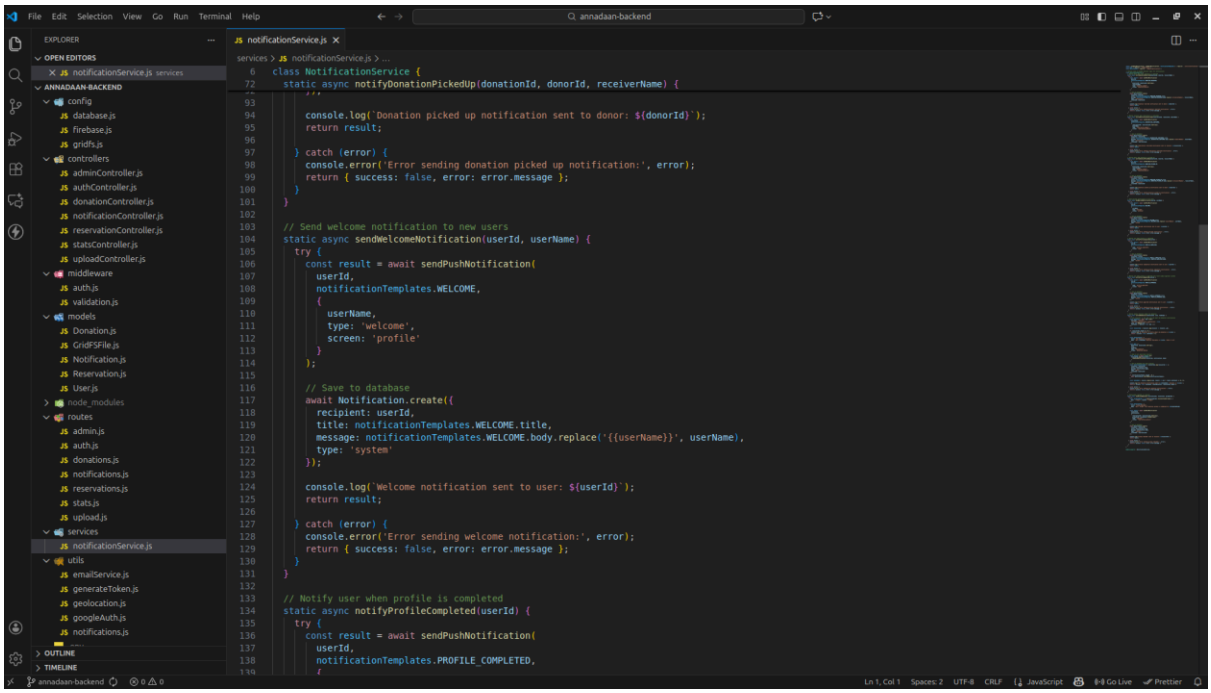

AnnaDaan_Flutter-main
Version control
main.dart
Project
main.dart
signup_page.dart
splash_screen.dart
post_donation_screen.dart
pubspec.lock
Dart support is not enabled for the module 'AnnaDaan_Flutter-main'
Enable Dart support
Open Dart settings
admin_dashboard.dart
edit_profile_screen.dart
forgot_password_screen.dart
help_support_screen.dart
location_picker.dart
login_screen.dart
notifications_screen.dart
otp_verification_screen.dart
profile_screen.dart
reset_password_screen.dart
role_selection_screen.dart
settings_screen.dart
signup_page.dart
splash_screen.dart
receiver_pages
receiver_browse_screen.dart
receiver_dashboard.dart
receiver_form.dart
main.dart
test
web
windows
.gitignore
.metadata
analysis_log.txt
analysis_options.yaml
analysis_output.txt
analysis_results.txt
devtools_options.yaml
pubspec.lock
@override
void dispose() {
_timer?.cancel();
super.dispose();
}
@override
Widget build(BuildContext context) {
return Scaffold(
backgroundColor: Colors.white,
body: Container(
decoration: BoxDecoration(
gradient: LinearGradient(
begin: Alignment.topCenter,
end: Alignment.bottomCenter,
colors: [Colors.white, Color(0xFFE3F2FD)],
),
),
child: SafeArea(
child: Center(
child: Column(
mainAxisAlignment: MainAxisAlignment.center,
children: [
// App Logo
Container(
width: 200,
height: 200,
decoration: BoxDecoration(
boxShadow: [
BoxShadow(
color: Colors.blue.withOpacity(0.2),
blurRadius: 20,
offset: Offset(0, 10),
AnnaDaan_Flutter-main
lib
views
pages
splash_screen.dart
1:1
LF
UTF-8
2 spaces

Backend:

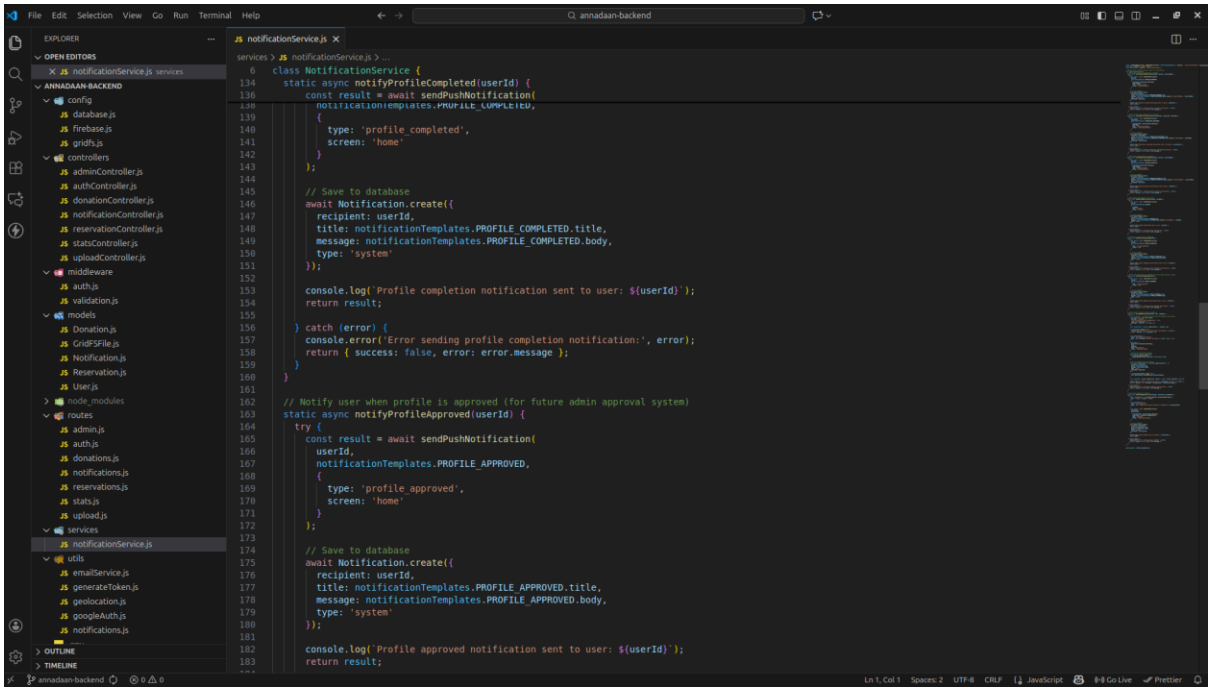

File Edit Selection View Go Run Terminal Help
annadaan-backend
EXPLORER
notificationService.js
OPEN EDITORS
notificationService.js services
ANNADAAN-BACKEND
config
database.js
firebase.js
gridfs.js
controllers
adminController.js
authController.js
donationController.js
notificationController.js
reservationController.js
statsController.js
uploadController.js
middleware
auth.js
validation.js
models
Donation.js
GridFSFile.js
Notification.js
Reservation.js
User.js
node_modules
routes
admin.js
auth.js
donations.js
notifications.js
reservations.js
stats.js
upload.js
services
notificationService.js
utils
emailService.js
generateToken.js
geolocation.js
googleAuth.js
notifications.js
OUTLINE
TIMELINE
services > notificationService.js > ...
const { sendPushNotification, sendBulkNotifications, notificationTemplates } = require('../utils/notifications');
const User = require('../models/User');
const Notification = require('../models/Notification');
// Service class to handle business logic for notifications
class NotificationService {
// Notify donor when their donation is reserved
static async notifyDonationReserved(donationId, donorId, receiverName) {
try {
const result = await sendPushNotification(
donorId,
notificationTemplates.DONATION_RESERVED,
{
donationId: donationId.toString(),
receiverName,
type: 'donation_update',
screen: 'donation_details'
}
);
// Save to database
await Notification.create({
recipient: donorId,
title: notificationTemplates.DONATION_RESERVED.title,
message: notificationTemplates.DONATION_RESERVED.body.replace('{{receiverName}}', receiverName),
type: 'reservation',
relatedId: donationId
});
console.log(`Donation reserved notification sent to donor: ${donorId}`);
return result;
} catch (error) {
console.error('Error sending donation reserved notification:', error);
return { success: false, error: error.message };
}
}
// Notify receiver when reservation is confirmed
static async notifyReservationConfirmed(reservationId, receiverId, donorName) {
try {
const result = await sendPushNotification(
receiverId,
notificationTemplates.RESERVATION_CONFIRMED,
{
reservationId: reservationId.toString(),
donorName,
type: 'reservation_update',
screen: 'reservation_details'
annadaan-backend
Ln 1, Col 1
Spaces: 2
UTF-8
CRLF
JavaScript
Go Live
Prettier



```
class NotificationService {
  static async notifyReservationConfirmed(reservationId, receiverId, donorName) {
    const result = await sendPushNotification(
      screen: 'reservation_details'
    );

    // Save to database
    await Notification.create({
      recipient: receiverId,
      title: notificationTemplates.RESERVATION_CONFIRMED.title,
      message: notificationTemplates.RESERVATION_CONFIRMED.body.replace('{{donorName}}', donorName),
      type: 'reservation',
      relatedId: reservationId
    });

    console.log(`Reservation confirmed notification sent to receiver: ${receiverId}`);
    return result;

  } catch (error) {
    console.error('Error sending reservation confirmed notification:', error);
    return { success: false, error: error.message };
  }
}

// Notify donor when donation is picked up
static async notifyDonationPickedUp(donationId, donorId, receiverName) {
  try {
    const result = await sendPushNotification(
      donorId,
      notificationTemplates.DONATION_PICKED_UP,
      {
        donationId: donationId.toString(),
        receiverName,
        type: 'donation_update',
        screen: 'donation_details'
      }
    );

    // Save to database
    await Notification.create({
      recipient: donorId,
      title: notificationTemplates.DONATION_PICKED_UP.title,
      message: notificationTemplates.DONATION_PICKED_UP.body.replace('{{receiverName}}', receiverName),
      type: 'completion',
      relatedId: donationId
    });

    console.log(`Donation picked up notification sent to donor: ${donorId}`);
```



```
class NotificationService {
  static async notifyDonationPickedUp(donationId, donorId, receiverName) {
    ...
    console.log(`Donation picked up notification sent to donor: ${donorId}`);
    return result;

  } catch (error) {
    console.error('Error sending donation picked up notification:', error);
    return { success: false, error: error.message };
  }
}

// Send welcome notification to new users
static async sendWelcomeNotification(userId, userName) {
  try {
    const result = await sendPushNotification(
      userId,
      notificationTemplates.WELCOME,
      {
        userName,
        type: 'welcome',
        screen: 'profile'
      }
    );

    // Save to database
    await Notification.create({
      recipient: userId,
      title: notificationTemplates.WELCOME.title,
      message: notificationTemplates.WELCOME.body.replace('{{userName}}', userName),
      type: 'system'
    });

    console.log(`Welcome notification sent to user: ${userId}`);
    return result;

  } catch (error) {
    console.error('Error sending welcome notification:', error);
    return { success: false, error: error.message };
  }
}

// Notify user when profile is completed
static async notifyProfileCompleted(userId) {
  try {
    const result = await sendPushNotification(
      userId,
      notificationTemplates.PROFILE_COMPLETED,
      {
```



This project is composed of two primary directories (backend and frontend) which each contain numerous files (over 400-500 lines per file) and it would not be practical to include in their entirety within this report since they are very large and complex.

You will find the entire project, including comments, laid out properly and clearly in the Git repository linked below; from there, you can access, inspect, and download all code for both back-end and front-end files.

## GitHub Link:

Frontend Repo:
https://github.com/Rohan-shah1/AnnaDaan_Flutter

Backend Repo:
https://github.com/Rohan-shah1/AnnaDaan_Backend