Diagram, logo

Description automatically generated

MEHRAN UNIVERSITY

OF ENGINEERING & TECHNOLOGY

JAMSHORO, PAKISTAN

Department of Software Engineering

**MAD Complex Engineering Problem Report**

***Submitted By:***

* *Aoun Jafri (22SW072)*
* *Muhmmad Haram (22SW121)*

***Submitted To:***

*Ma’am Mariam Memon*

## Introduction:

This report presents the complete documentation of the DonateNear mobile application, a Flutter-based project developed as a semester project to provide an easy and responsive platform for connecting donors with those in need. The report details the real-world problem, proposed solution, user interface design, data storage strategy, APIs and plugins used, and challenges encountered during development.

## 1. Real World Problem Identification

In many urban and semi-urban communities, there exists a significant gap between individuals willing to donate items such as food, clothes, and household essentials, and those in need who lack an effective medium to reach these donors. While various NGOs and charitable organizations work toward community welfare, there is no unified digital platform that efficiently connects small-scale donors directly with local organizations or needy individuals. This leads to wastage, inefficient resource distribution, and missed opportunities for impactful giving.

Additionally, donors often face uncertainty regarding where their items will go, and recipients or NGOs struggle to find local donations. Traditional social media posts or word-of-mouth are inefficient and unreliable for managing such donations. Therefore, a digital platform is essential to bridge this communication gap in a transparent, simple, and accessible way.

The core problem identified is the lack of a simple, accessible, and location-based donation system that facilitates meaningful community contributions without the need for complex registration or logistics management.

## 2. Proposed Solution

To address this problem, we designed and developed a Flutter mobile application called 'DonateNear'. The app enables donors to post details about items they wish to donate, including type (food, clothes, or other essentials), description, location, and an optional image. The donations are then displayed in a feed that can be viewed by nearby users or NGOs. The app prioritizes simplicity, anonymity, and accessibility over heavy registration procedures.

Key features include:  
• A simple and clean user interface using Flutter’s Material Design components.  
• Two main tabs – Feed and Add Donation – to make navigation intuitive.  
• Optional donor name field for anonymous giving.  
• Local and cloud-based storage integration to ensure reliability.  
• Responsive design for smooth experience across mobile, tablet, and web.

The project adopts an iterative development approach — initially storing data locally within temporary lists, and later integrating Firebase as a cloud backend for persistent storage and scalability. This ensures both quick prototyping and a realistic end-to-end implementation for production scenarios.

## 3. Responsive User Interfaces

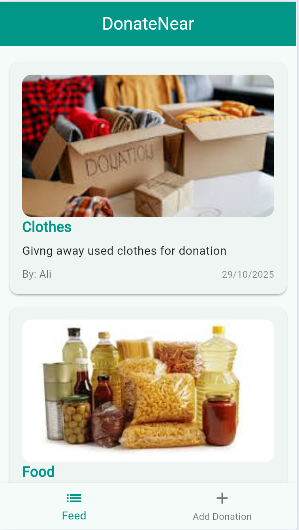
The DonateNear app was designed using Flutter’s responsive layout principles, ensuring optimal usability on mobile, tablet, and web platforms. Material Design widgets such as Scaffold, TabBar, and ListView were utilized to create a consistent and visually appealing interface across devices. Each screen adapts dynamically based on screen size and orientation.

• Splash Screen: Displays the DonateNear logo briefly before navigating to the home screen.  
• Home Screen: Contains two tabs – Feed (shows all donation posts) and Add Donation (allows users to submit donation details).  
• Donation Details Screen: Displays the complete details of a donation, including image, description, and donor info.

Screenshots of App in different screen sizes are provided below:

Mobile view:

|  |  |
| --- | --- |
|  | A screenshot of a donation box  AI-generated content may be incorrect. |



Tablet View:

A screen shot of a white box

AI-generated content may be incorrect.

A screenshot of a phone

AI-generated content may be incorrect.

Desktop/ Laptop View:

A screen shot of a donation

AI-generated content may be incorrect.

A screenshot of a donation box

AI-generated content may be incorrect.

## 4. Data Storage

Initially, the app utilized temporary local storage through in-memory lists to store donation entries. This approach facilitated rapid development and testing. However, for long-term scalability, Firebase Realtime Database was integrated to store donation data persistently and enable synchronization across devices.

Firebase was chosen due to the following advantages:  
• Real-time synchronization and offline persistence.  
• Seamless integration with Flutter using official plugins.  
• Scalable backend with minimal setup.  
• Secure authentication and data management options for future expansion.  
• Cost-effective and ideal for academic and small-scale deployments.

Each donation entry is stored as a structured object in Firebase with attributes such as donor name, type, description, location, date, and image data (either as a file path or base64 bytes for web compatibility).

## 5. APIs/Packages/Plugins Used

The project makes use of a few external dependencies to improve functionality and user experience. These include:  
• image\_picker: For selecting and uploading images from device gallery or camera.  
• firebase\_core and firebase\_database: For Firebase initialization and real-time database operations.  
• flutter/material.dart: For Material Design widgets and layouts.  
Each package was selected based on community reliability, stability, and compatibility with Flutter 3.x.

## 6. Issues and Bugs Encountered and Resolved

During the development of DonateNear, several issues were encountered and systematically resolved:

• Image Upload Issue on Web: Initially, the image picker plugin was not compatible with Chrome web builds, resulting in image loading failures. This was resolved by switching to a web-friendly implementation using `Uint8List` for storing image bytes.

• Navigation Delay from Splash Screen: The app initially did not transition properly from the splash screen to the home screen. This was fixed by introducing a timed delay using `Future.delayed()` and correct `Navigator.pushReplacement()` usage.

• State Management Conflicts: While adding new donations, the UI was not updating automatically. This issue was resolved by employing `setState()` within StatefulWidgets to refresh the feed dynamically.

• Firebase Configuration Errors: During Firebase setup, connection issues arose due to missing configuration files (google-services.json). This was resolved by correctly linking the Firebase project and ensuring necessary files were included.

These issues enhanced understanding of Flutter’s ecosystem and strengthened debugging and testing skills throughout the project.

## 7. Conclusion

The DonateNear project successfully demonstrates how a simple yet purposeful mobile application can address real-world problems by leveraging modern frameworks like Flutter and Firebase. It emphasizes ease of use, scalability, and community-driven impact. Through this project, the development process provided valuable insights into UI design, state management, and backend integration. Future improvements can include user authentication, location-based donation matching, and notification alerts for NGOs.