Ehsass App

Final Year Project

Session 2020-2024

A project submitted in partial fulfillment of the degree of

BS in Computer Sciences

Department of Computer Science

Faculty of Computer Science & Information Technology

The Superior University, Lahore

Spring 2024

Type (Nature of project)	[✓] **D**evelopment [ ] **R**esearch [ ] **R**&**D**
Area of specialization	Mobile base
FYP ID	**FYP-BCSM-F23-089**

Project Group Members

Sr.#	Reg. #	Student Name	Email ID	*Signature
(i)	BCSM-F20-033	Saif-ur-Rehman	bcsm-f20-033@superior.edu.pk	
(ii)	BCSM-F19-181	M. Nouman Ali	bcsm-f19-181@superior.edu.pk	
(iii)	BCSM-F20-022	Zeeshan Ahmad	bcsm-f20-022@superior.edu.pk	

*The candidates confirm that the work submitted is their own and appropriate credit has been given where reference has been made to work of others

Plagiarism Free Certificate

This is to certify that, I am **Saif Ur Rehman** S/D of **Muhammad Iftikhar**, group leader of FYP under registration no **FYP-BCSM-F23-089** at Computer science Department, The Superior University, Lahore. I declare that my FYP report is checked by my supervisor.

Date: 10/07/2024

Name of Group Leader: Saif Ur Rehman Signature: _____

Name of Supervisor: Ma'am Sabah Arif Co-Supervisor: Dr. Jawad Ahmad

Designation: Senior Lecturer Designation: Assistant Professor

Signature: _____ Signature: _____

HOD: Dr. Irfan-ud-din

Signature: _____

Ehsaas App

Change Record

Author(s)	Version	Date	Notes	Supervisor's Signature
	1.0		<Original Draft>	
			<Changes Based on Feedback from Supervisor>	
			<Changes Based on Feedback From Faculty>	
			<Added Project Plan>	
			<Changes Based on Feedback from Supervisor>	

APPROVAL

PROJECT SUPERVISOR

Comments: _____

_____

Name:_____

Date:_____ Signature:_____

PROJECT MANAGER

Comments: _____

_____

Date:_____ Signature:_____

HEAD OF THE DEPARTMENT

Comments: _____

_____

Date:_____ Signature:_____

Dedication

This work is dedicated to my university for providing us support and a platform for implementing our idea.

Acknowledgements

We are very thankful to our supervisor and our mentor who gave us this golden opportunity to build our future is the most supportive person we have ever met; he guides us more We wish to express our immense gratitude to GOD Almighty for his mercy, guidance, and protection towards us for seeing us through the rigors of this work. We are greatly indebted to our supervisor **Ma'am Saba Arif,** moreover all the staff our in department of Computer Science both academic and non-academic staff for their intellectual upbringing. Our special appreciation goes to our loving parents' brothers and sisters, whose moral and finical support cannot be overemphasized. Also, our sincere gratitude and special regards to us friends too many to mention whose encouragement led to the success of this work.

Executive Summary

In a world grappling with the challenges of food security and environmental sustainability, our Food Waste Management Project emerges as a beacon of responsible resource utilization and ecological stewardship. This initiative addresses the pressing issue of food waste through a comprehensive and innovative approach, aiming to create a positive impact on both communities and the environment.

The primary goal of our Food Waste Management Project is to significantly reduce the amount of food waste generated across various sectors of society. Through our online communication platform which is our application we can approach the donor and needy. Moreover forging partnerships with local governments, businesses, and NGOs to create a network for sharing best practices and optimizing resource utilization.
Collaborating with food retailers to implement waste reduction strategies and explore sustainable packaging alternatives.

Enhancing food security by redirecting surplus food to those in need through partnerships with food banks and community organizations. Fostering a sense of community responsibility and empowerment through active participation in waste reduction initiatives.

Our Food Waste Management Project stands at the intersection of environmental consciousness, technological innovation, and community engagement. By addressing the issue of food waste holistically, we aim to create a model for sustainable living that can be replicated and adapted globally, fostering a brighter, more resilient future for generations to come

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Chapter 1: Introduction

At its core, the Ehsas Food Waste Management Project aims to revolutionize our approach to food consumption and disposal. With a keen awareness of the interconnectedness of environmental health and societal well-being, our project endeavors to create a paradigm shift in the way we produce, consume, and discard food. We seek to establish a model by adopting a multifaceted strategy that incorporates technology, community engagement, and strategic partnerships.

1.1. Motivations and Challenges

Reducing food waste contributes directly to environmental sustainability by decreasing methane emissions from landfills and conserving resources used in food production. Food waste represents a significant misuse of resources such as water, energy, and land.

On the other hand the challenges are encouraging individuals, businesses, and communities to adopt new practices and mindsets regarding food consumption and waste disposal can be a significant challenge.

Developing efficient systems for collecting, transporting, and redistributing surplus food requires overcoming logistical challenges. Ensuring timely and safe delivery to those in need is crucial.

Ensuring user-friendly communication platforms for effective participation is essential, moreover limited financial resources may pose challenges in implementing comprehensive food waste management programs.

1.2. Goals and Objectives

Food waste is a significant global issue with economic, environmental, and social implications. The inefficient use of resources, energy, and water in food production, coupled with improper disposal methods, contributes to the depletion of natural resources and the generation of greenhouse gases. To address this problem, researchers and practitioners have explored

various technological solutions and management strategies for effective food waste reduction and management.

The complexity of food waste management arises from the entire supply chain, including production, distribution, retail, and consumption. Key challenges include overproduction, insufficient infrastructure, lack of awareness, and inadequate waste separation at source. These challenges necessitate a comprehensive approach that integrates technological innovations with behavioral changes and policy interventions.

- ***Smart Bins and IoT Sensors:***

  Smart bins equipped with sensors have been developed to monitor and manage food waste at various stages of the supply chain. These sensors can detect fill levels, expiration dates, and even identify types of discarded food. IoT technology enables real-time data collection, aiding in optimized waste collection routes and resource allocation.

- ***Food Tracking and Traceability Systems:***

  Block chain and other traceability systems have been explored to enhance transparency and traceability in the food supply chain. These systems enable stakeholders to track the journey of food products, facilitating better inventory management, timely recalls, and reducing the chances of waste due to spoilage.

- ***Data Analytics and Predictive Modeling:***

  Data analytics and machine learning algorithms have been employed to analyze historical data and predict future food waste patterns. These models assist in demand forecasting, inventory management, and production planning, minimizing overproduction and reducing waste throughout the supply chain.

1.3. Gap Analysis

The Food Waste Management System Project aims to develop and implement an efficient and sustainable system to address the global issue of food waste. A comprehensive gap analysis is crucial to identify the disparities between the current state of food waste management and the desired state envisioned by the project.

1. Technological Gaps:

a. Integration of IoT and Sensor Technologies:

Current systems may lack advanced sensors and IoT integration for real-time monitoring of food waste. The project should focus on bridging this gap by implementing smart bins and sensors throughout the supply chain to enable accurate data collection and analysis.

b. Data Analytics and Predictive Modeling:

Many existing systems may not fully leverage data analytics and predictive modeling to optimize supply chain processes. The project should emphasize the development of algorithms and analytics tools to predict food waste patterns, enabling proactive decision-making in production, distribution, and retail.

2. Infrastructure Gaps:

a. Waste Collection and Sorting Infrastructure:

Inadequate waste collection and sorting infrastructure at various stages of the supply chain can hinder effective food waste management. The project should assess and address gaps in infrastructure, ensuring that there is a robust system for efficient waste collection, separation, and disposal.

b. Technological Accessibility:

Accessibility to technology might be limited in certain regions or among smaller businesses. The project should consider developing scalable solutions that are accessible to a wide range of stakeholders, considering the diversity of technological infrastructure across different communities.

The existing regulatory framework may lack sufficient incentives for businesses to adopt food waste reduction practices or penalties for non-compliance. The project should explore gaps in current policies and recommend adjustments to encourage widespread adoption of sustainable practices. Inconsistent reporting standards and regulations related to food waste measurement and reporting can hinder accurate assessments. The project should advocate for standardized reporting frameworks to enable more transparent and comparable data across different entities. Lack of awareness among consumers regarding the impact of food waste and proper disposal methods is a significant gap. The project should prioritize educational initiatives to promote responsible consumption habits and reduce food waste at the household level.

Insufficient collaboration and engagement among stakeholders, including governments, businesses, and communities, may impede the effectiveness of food waste management efforts. The project should focus on fostering partnerships and creating a collaborative ecosystem to address food waste comprehensively.

1.4. Proposed Solution

Food waste presents a significant global challenge, impacting economies, the environment, and societal well-being. The ineffective utilization of resources, energy, and water during food production, combined with improper disposal practices, contributes to the depletion of natural resources and the emission of greenhouse gases. To tackle this issue, researchers and practitioners have explored diverse technological solutions and management strategies to efficiently reduce and handle food waste.

The intricacy of food waste management stems from the entire supply chain, encompassing production, distribution, retail, and consumption. Key obstacles include overproduction, inadequate infrastructure, lack of awareness, and suboptimal waste separation at the source. These challenges call for a holistic approach that integrates technological innovations with behavioral changes and policy interventions.

Advanced bins equipped with sensors have been devised to monitor and control food waste throughout different stages of the supply chain. These sensors can identify fill levels, expiration dates, and even categorize discarded food types. Utilizing IoT technology allows real-time data collection, facilitating optimized waste collection routes and resource allocation.

Blockchain and other traceability systems have been explored to enhance transparency and traceability in the food supply chain. These systems empower stakeholders to trace the journey of food products, improving inventory management, enabling timely recalls, and reducing the likelihood of waste due to spoilage.

Data analytics and machine learning algorithms have been deployed to analyze historical data and forecast future patterns of food waste. These models aid in demand forecasting, inventory management, and production planning, mitigating overproduction and minimizing waste across the supply chain.

1.5. Project Plan

The primary goal of the Food Waste Management Project is to reduce food waste in our community by implementing effective strategies for collection, processing, and disposal of food waste. This project aims to raise awareness about the environmental and social impacts of food waste while promoting sustainable practices.

Project Scope:

1. Research and Analysis;

Conduct a comprehensive study on the current state of food waste in the community. Identify key stakeholders, including local businesses, households, and waste management facilities, Analyze existing food waste management practices and regulations as well.

2. Community Engagement and Education;

Develop educational materials and campaigns to raise awareness about the importance of reducing food waste.

Organize workshops and seminars for local businesses, schools, and community groups.

3. Technology Integration;

Explore and implement technology solutions for tracking and monitoring food waste collection.

Develop a user-friendly mobile application to facilitate communication between stakeholders.

Integrate data analytics to assess the impact and effectiveness of the project.

Resource Allocation:

1. Human Resources:

- Project Manager
- Outreach and Education Coordinator
- Technology Specialist
- Regulatory Compliance Officer
- Monitoring and Evaluation Officer
- Communication and Media Liaison

2. **Financial Resources**;

- Budget for educational materials, workshops, and outreach campaigns
- ii) Funding for infrastructure development and technology integration
- Allocation for ongoing monitoring and evaluation activities

3. Partnerships;

- Collaborate with local businesses, schools, waste management facilities, and government agencies.

Risk Management:

- Identify potential risks such as community resistance, regulatory challenges, or technology integration issues.
- Develop mitigation strategies for each identified risk.
- Regularly review and update the risk management plan as the project progresses.

Communication Plan:

- Establish regular communication channels with stakeholders through newsletters, social media, and community meetings.
- Provide updates on project milestones, successes, and challenges.

Conclusion:

The Food Waste Management Project is a comprehensive initiative that addresses the multifaceted issue of food waste in our community. By engaging the community, implementing effective infrastructure, and leveraging technology, the project aims to make a significant impact on reducing food waste and promoting sustainable practices. Regular monitoring and evaluation will ensure the project's success and provide opportunities for continuous improvement.

1.6. Work Breakdown Structure:

The primary objective of our Food Waste Management Project is to significantly diminish the volume of food waste generated across diverse sectors of society. Leveraging our online communication platform, our application serves as a means to connect with donors and those in need. Additionally, we collaborate with local governments, businesses, and NGOs to establish a network that facilitates the sharing of best practices and optimizes resource utilization. This

involves working closely with food retailers to implement waste reduction strategies and explore sustainable packaging alternatives.

We contribute to enhancing food security by redirecting surplus food to individuals in need through strategic partnerships with food banks and community organizations. Furthermore, we foster a sense of community responsibility and empowerment by encouraging active participation in waste reduction initiatives.

Positioned at the intersection of environmental consciousness, technological innovation, and community engagement, our Food Waste Management Project strives to address the issue of food waste holistically. Our goal is to establish a model for sustainable living that can be replicated and adapted globally, thereby paving the way for a brighter and more resilient future for generations to come.

1.7. Roles & Responsibility Matrix:

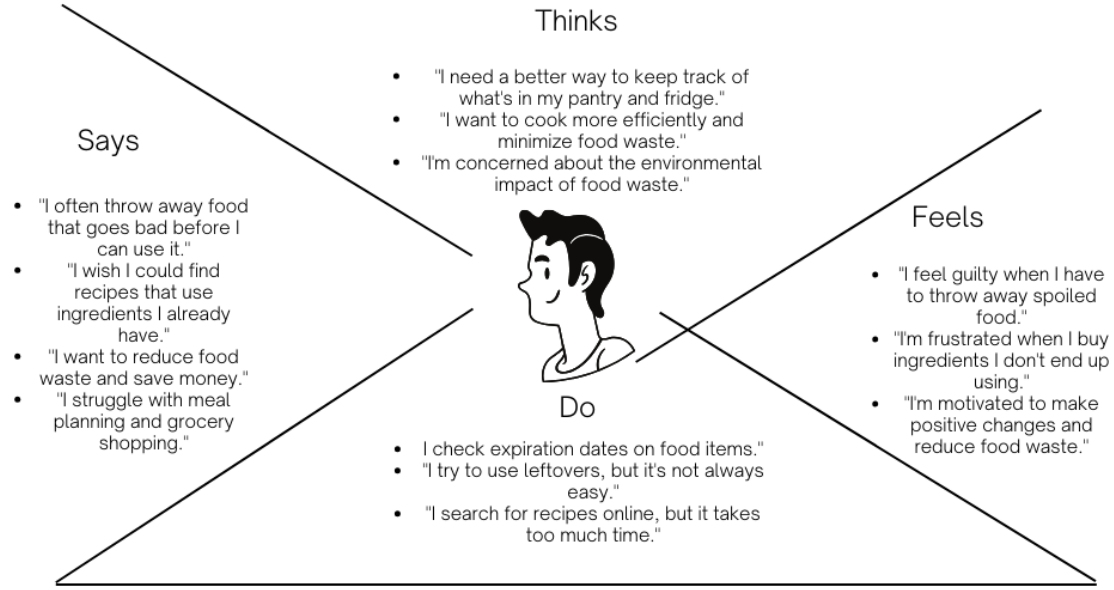
Role	Responsibilities
Project Manager	Develop and manage project plans, timelines, and budgets. - Coordinate communication among team members and stakeholders. - Ensure adherence to project milestones and goals. - Oversee risk management and mitigation strategies.
System Architect	Define the system architecture and technological stack. - Lead the design of the overall system structure. - Collaborate with the development team to ensure the system meets technical requirements.
Business Analyst	Conduct a feasibility study and gather project requirements. - Analyze and document business processes related to food waste management. - Ensure that system requirements align with organizational goals.
Database Administrator	Design and implement the database structure. - Ensure data integrity and security. - Optimize database performance for efficient data storage and retrieval.
User Interface	Designer - Design an intuitive and user-friendly interface for the online communication platform (application). - Ensure a positive user experience and accessibility.
Community Outreach Manager	Establish and maintain partnerships with local governments, businesses, and NGOs. - Develop and implement community engagement strategies. - Foster collaboration for sharing best practices in food waste reduction.
Food Redistribution	Collaborate with food banks and community organizations to redirect surplus food to those in need. - Develop and maintain relationships with

Coordinator	food donors and recipients. - Implement processes for efficient food redistribution.
Project Coordinator	Assist the project manager in coordinating day-to-day activities. - Facilitate communication between team members. - Monitor task progress and report to the project manager.

1.8. Gantt Chart

weeks / Activities	1	2	3	4	5
Research / Define Specification					
Project Planning					
Design / Development					
Test Plan					
Testing and Q A / Delivery					

1.9. Empathy Map



Pain

- Frustration when throwing away expired food.
- Guilt about wasting food and money.
- Overwhelmed by meal planning.

Gain

- Relief from reducing food waste.
- Satisfaction from contributing to environmental sustainability.
- Confidence in meal planning and saving money.

Chapter 2

Software Requirement Specifications

Chapter 2: Software Requirement Specifications

2.1. Introduction

The Software Requirements Specification (SRS) for the Ehsass App aims to provide a detailed roadmap for the development of a mobile application dedicated to managing food waste from various establishments such as hotels and marriage halls. The key purposes of this document include:

Ehsass App Management:

Efficiently manage and reduce food wastage from hotels, marriage halls, and similar places.

Donation Process:

Facilitate a seamless process for donors to contribute by donating excess food through location-pinpointed transactions.

Location-Based Food Delivery:

Enable receivers to collect donated food conveniently by locating and reaching the pin-designated areas.

User Record and Information:

Implement a user-friendly interface with a dedicated 'About' section, allowing users to showcase and share information.

Technology Stack:

Develop the application using Android Studio and Java language, leveraging the capabilities of Firebase for enhanced functionality and connectivity.

Promoting Social Responsibility:

Encourage a sense of social responsibility by creating a platform that addresses food wastage while facilitating contributions from the community.

Enhancing User Experience:

Focus on creating an intuitive and user-centric design to enhance the overall experience for both donors and receivers.

Scalability and Future Expansion:

Lay the groundwork for scalability and potential future expansions, ensuring the adaptability of the application to evolving needs.

Software Requirements

Operating System	Windows 10
Language	Java
Code Editor	Android Studio Electric Eel V2022.1.1
Database	Firebase

Hardware Requirements

Processor	2.6 GHz or Faster Processor
RAM	8 GB
Disk Space	20 GB of Available Hard Disk
Graphic	None
Display	1024X 768 or Higher Resolution

2.1.1. Document Conventions

Font Style:

Standard body text is presented in (Calibri 12).

Code snippets and technical terms are formatted in (Courier New, 12pt).

Priority Notation:

Requirements are prioritized using the following conventions:

[Priority Level 1]: Critical and non-negotiable requirements.

[Priority Level 2]: Important but not critical requirements.

[Priority Level 3]: Desirable and can be addressed in future releases.

Highlighting:

Important notes or critical information are highlighted using (Bold Text).

Inheritance of Priorities:

Higher-level requirements' priorities are assumed to be inherited by detailed requirements, unless explicitly stated otherwise.

2.1.2. Intended Audience and Reading Suggestions

The intended audience for the Ehsass App includes:

- Primary Audience
- Developers and Programmers:

Detailed technical specifications and code-related information will be valuable for developers working on the Ehsass App

Project Stakeholders:

Those involved in the planning, development, and execution of the Ehsass App, such as project managers and team leads.

Secondary Audience:

Testing Team:

Testers and quality assurance personnel who need to understand the functional and non-functional requirements of the Ehsass App for effective.

Future Developers:

Individuals who may take over the project in the future will benefit from a comprehensive understanding of the document for ongoing maintenance and updates.

Reading Suggestions:

Start with Overview:

Begin by reading the document's overview to grasp the general purpose, scope, and functionality of the Ehsass App.

Technical Details for Developers:

Developers should focus on the technical specifications, code conventions, and integration details, especially the sections related to Android Studio, Java, and Firebase integration.

Testing and Quality Assurance:

Testing teams should pay attention to the sections related to testing requirements, scenarios, and expected outcomes to ensure effective test planning and execution.

Future Development Considerations:

Individuals interested in future development or maintenance should review the documentation on the project's structure, coding conventions, and potential areas for improvement.

2.1.3. Product Scope

The product scope of the Ehsass App encompasses the following key features and functionalities:

Key Features:

I. **Ehsass App Management:**

The application facilitates the management of surplus food from various locations, including hotels, marriage halls, and other establishments.

II. **Location-Based Donation:**

Donors can contribute by pinning their location to indicate the availability of excess food for donation.

III. **Pin-Based Food Reception:**

Receivers can view pinned locations on the map and collect food donations from specified locations.

IV. **User Profile and Records:**

The app includes a user profile section where donors and receivers can maintain their records and track their contributions or receipts.

V. **About Section:**

Users have access to an "About" section providing information about the Ehsass App, its mission, and how it contributes to reducing food waste.

Development Environment:

The Ehsass App is developed using Android Studio and Java programming language, ensuring compatibility with Android devices. Firebase is integrated into the app to enhance functionality, such as real-time data updates, secure authentication, and efficient data storage[Exclusions:

The scope is limited to Android devices, and development for other platforms is not considered in this phase. Advanced analytical features beyond basic record-keeping are excluded from the initial scope.

2.2. Overall Description

2.2.1. Product Perspective

The product perspective for the Ehsass App is outlined as follows:

1. System Overview:

The Ehsass App operates as a mobile application designed for Android devices[1]. It addresses the issue of food waste management, providing a platform for donors to contribute excess food and for recipients to receive it.

2. Functionality:

Donation Process:

- Donors can use the app to initiate the food donation process by pinpointing their location on the map.
- Recipients can access donated food by specifying their location, connecting with donors through the app.

User Profile and Recording:

- The app includes a user profile section where users can record and showcase their contributions or activities.

3. Integration:

- The Ehsass App is developed using Android Studio and Java, ensuring compatibility with Android devices. Firebase is integrated to enhance functionalities such as real-time data synchronization and secure data storage.

4. External Systems:

- The app requires an internet connection for seamless functionality.

5. Relation to Existing Solutions:

- The Ehsass App addresses the global issue of food waste management, contributing to a more sustainable and community-driven approach.

2.2.2. User Classes and Characteristics

In the context of the Ehsass App, the user classes and their characteristics are defined as follows:

User Classes:

The Ehsass App for food waste management includes several user classes with distinct roles:

Individual Consumers: Food purchases, track consumption, and monitor waste. They receive alerts on expiring food and tips to reduce waste, and access reports on their waste patterns.

Households: Allow multiple users to coordinate food management, track inventory, and monitor collective waste, with household-level reports and alerts.

Restaurants and Food Businesses: Manage food inventory, track waste, and facilitate surplus donations. They get alerts on expiring stock and waste reduction tips, with detailed waste reports.

Food Banks and Charitable Organizations: Handle donations from individuals and businesses, coordinating collections and distributions. They receive alerts about donations and needs, with reports on their impact.

Municipal Waste Management Authorities: Monitor and coordinate waste collection and recycling. They provide community waste data and management updates.

Characteristics:

The Ehsass App for food waste management includes several user classes, each with distinct roles. Individual Consumers log food purchases, track consumption, and monitor waste, receiving alerts and tips. Households coordinate food management among multiple users, tracking inventory and waste collectively. Restaurants and Food Businesses manage inventory, track waste, and donate surplus food, with alerts and detailed reports. Food Banks and Charitable Organizations handle donations, coordinating collections and distributions. Municipal Waste Management Authorities monitor and coordinate waste collection and recycling, providing community data and updates - manage user accounts, ensure data security, and maintain app performance. Create educational content, organize campaigns, and engage users to promote waste reduction. Each user class contributes to comprehensive food waste management through the Ehsass App.

2.2.3. Operating Environment

The operating environment for the Ehsass App involves the platforms, technologies, and dependencies necessary for the app's proper functioning.

1. Mobile Platforms:

The Ehsass App is designed for Android devices, making it compatible with smartphones and tablets running the Android operating system.

2. Development Tools:

Integrated Development Environment (IDE):

The app is developed using Android Studio, a popular IDE for Android app development, ensuring compatibility and optimized code for the Android platform.

Programming Language:

Java is the primary programming language used in developing the Ehsass App. The use of Java aligns with Android development standards and provides a robust foundation for building the application.

3. Database and Backend:

Firebase Integration:

Firebase serves as the backend for the Ehsass App, providing features like real-time database functionality, authentication services, and cloud storage. This integration enhances the app's capabilities and ensures seamless data management.

4. Connectivity Requirements:

Internet Connection:

The app requires an internet connection, as mentioned in the documentation, indicating that certain features may rely on online functionality.

System Requirements:

Android Version:

The Ehsass App is designed to run on Android devices. The specific Android version compatibility should be outlined in the documentation, ensuring users have the required operating system version for optimal performance.

5. Device Capabilities:

Ensure that the app's features align with the capabilities of typical Android devices, considering factors like screen size, resolution, and processing power.\

2.2.4. Design and Implementation Constraints

1) Hardware Constraints:

Device Compatibility:

The Ehsass App is designed for Android devices. Ensure compatibility with a broad range of Android smartphones to maximize user reach.

2) Software Constraints:

Android Studio Version:

Development and testing are constrained to a specific version of Android Studio. Developers should use the recommended version for optimal performance[1].

Firebase Integration:

The app relies on Firebase for real-time data synchronization. Constraints may arise if there are limitations or changes in Firebase services.

3) Internet Connectivity:

The application requires a stable internet connection for functionalities like pinning locations and real-time donation tracking.

4) Security Constraints:

Data Privacy:

Implement measures to ensure user data privacy, especially considering sensitive location information. Adhere to data protection regulations.

5) Authentication Mechanism:

Firebase Authentication:

Ensure the secure implementation of Firebase authentication for user verification and access control.

2.2.5. Assumptions and Dependencies

Assumptions:

I) Internet Connectivity:

The assumption is made that users will have consistent internet connectivity for the proper functioning of the Ehsass App, considering its reliance on cloud services like Firebase.

II) Device Compatibility:

It is assumed that the users' devices will meet the necessary hardware and software requirements for running Android Studio and the Ehsass App effectively.

III) Location Accuracy:

The accuracy of the location services on users' devices is assumed to be reliable for the precise pinning of donation and receipt locations.

Dependencies:

Firebase Services:

The proper functioning of the Ehsass App is dependent on the seamless integration and availability of Firebase services for real-time data storage and retrieval

Android Studio and Java Development Environment:

The development and maintenance of the Ehsass App are dependent on the continuous support and updates of Android Studio and Java development tools.

User's Consent for Location Services:

The app's location-based features are dependent on users granting consent for location services on their devices.

2.3. External Interface Requirements

2.3.1. User Interfaces

The user interfaces for the Ehsass App are designed to provide an intuitive and seamless experience for both donors and receivers. The key interfaces include:

1. Donor Interface:

Donation Form:

A simple form where donors can input details about the type and quantity of food they wish to donate.

Location Pinning:

Integration of a map feature for donors to pin the location where the food is available for pickup.

2. Receiver Interface:

Food Request:

An interface for receivers to request food, including specifying dietary preferences or restrictions.

Location-based Search:

Functionality to search for available food donations based on pinned locations.

3. User Profile:

Donor Profile:

Displays a summary of a donor's donation history and preferences.

Receiver Profile:

Shows a summary of a receiver's food requests and preferences.

4. About Section Interface:

Details about the Ehsass App, its mission, and how it contributes to reducing food waste.

Team Information:

Section showcasing the development team and contributors.

5. Integration Interfaces:

Seamless integration with Firebase for real-time data updates, notifications, and user authentication.

2.3.2. Hardware Interfaces

The Ehsass App primarily operates as a mobile application, and its hardware interfaces are minimal as it relies on standard functionalities available in smartphones. The hardware interfaces include:

Smartphones:

Ehsass App is designed to run on Android smartphones, utilizing standard hardware components such as the touchscreen, camera, and GPS for functionalities like pinning locations during food donation.

2.3.3. Software Interfaces

The Ehsass App integrates with various software interfaces to ensure seamless functionality and connectivity:

Android Studio:

The primary development environment for the Ehsass App is Android Studio[2]. It provides a comprehensive set of tools for Android app development.

Java Language:

The Ehsass App is coded using Java, a versatile and widely used programming language, ensuring compatibility and efficient execution on Android devices.

Firebase Integration:

Firebase is integrated into the app, offering features like real-time database, authentication, and cloud messaging. This enhances data management and user interaction.

2.3.4. Communications Interfaces

Donation Interface:

Functionality: Allows donors to donate Ehsass App.

Features:

- Pin location for donation.
- Integration with Android Studio and Java for seamless functionality.
- Connection to Firebase for data management and security.
- Source: Ehsass App Donation Project | Firebase - YouTube

Receiver Interface:

Functionality: Enables receivers to collect food from a specified location.

Features:

- Pin location for convenient food collection.
- Utilizes Firebase for real-time tracking.
- Source: Ehsass App Donation Project | Firebase - YouTube

User Profile Interface:

Functionality: Displays user records and details.

Features:

- About section for additional user information.
- Developed using Android Studio and Java.
- Source: Ehsass App Donation Project | Firebase – YouTube

2.4. System Features

Food Donation Interface:

Donors can easily donate excess food by pinning their location on the app.

Pinpointing the location helps in efficient food pickup by receivers.

Utilizes the Android Studio and Java language for seamless interaction [4].

Receiver Food Collection:

Receivers can access donated food by navigating to the pinned location.

Geolocation integration ensures a precise pickup of donated items.

Enhances the process of redistributing surplus food from various sources.

User Profile and Records:

Users have a dedicated section to maintain and showcase their donation records.

Allows users to track their impact and contributions over time.

Integration with Firebase. Utilizes Firebase for real-time data synchronization and storage.

Enables secure and scalable handling of user information and donation records.

About Section:

An informative section providing details about the Ehsass App's mission and purpose.

Enhances user awareness and understanding of the app's objectives.

These system features collectively create an efficient, user-friendly, and impactful Ehsass App management system using mobile technology and Firebase integration.

2.4.1. Description and Priority

1. Project Description

Overview:

The Ehsass App is a mobile application designed to manage Ehsass App from various sources such as hotels, marriage halls, and other locations. The primary goal is to facilitate the donation process between food donors and receivers.

Features:

Location-Based Donation:

Donors can conveniently donate excess food by pinning their location within the app.

Receivers can access and collect donated food from the pinned location, ensuring a seamless process.

User Records and About Section:

The app includes a user records section, allowing users to keep track of their donation history.

An informative About section provides details about the application and its mission to reduce food wastage.

Development Platform:

The app is developed using Android Studio.

Java programming language is utilized for application logic.

Database Integration:

Firebase is connected to the app, enhancing data storage and retrieval capabilities.

2. Priority

High Priority Features:

Location-Based Functionality:

Given the nature of the app, ensuring the smooth operation of location-based donation and retrieval is of utmost importance.

Data Security and User Privacy:

Implement robust security measures to protect user data, especially considering the sensitive nature of food donation.

User-Friendly Interface:

Prioritize an intuitive and user-friendly design to enhance the overall user experience.

Medium Priority Features:

User Records Enhancement:

Implement additional features in the user records section, such as detailed donation statistics and achievements.

Community Engagement:

Explore options for community engagement features, encouraging more people to participate in food donation.

Low Priority Features:

Integration with External Platforms:

Consider integrating the app with external platforms to expand its reach and impact.

2.4.2. Stimulus/Response Sequences

1. Donor Registration

Stimulus: Donor opens the Ehsass App.

Response: Display a registration screen prompting the donor to enter details such as name, contact information, and address.

Source: food-donation-application.

2. Food Donation Process

Stimulus: Donor selects the option to donate food.

Response: Show a form allowing the donor to input the type and quantity of food, along with a map to drop a pin indicating the donation location.

Source: Ehsass App Donation Project | Firebase.

3. Receiver Location Confirmation

Stimulus: Receiver opens the Ehsass App.

Response: Provide an option to confirm the location on the map where they want to receive the donated food.

Source: food-donation-application

4. User Profile

Stimulus: User navigates to the profile section.

***Response*:** Display a section showcasing the user's donation history and impact, encouraging continued involvement.

***Source*:** food-donation.

5. Integration with Firebase

Stimulus: App performs data exchange or authentication.

Response: Utilize Firebase integration for real-time data updates, secure authentication, and efficient storage.

2.4.3. Description and Priority

Description

The Ehsass App is a mobile application designed to efficiently manage surplus food from hotels, marriage halls, and various locations. The app aims to connect donors and receivers to minimize food wastage. Key features include:

Ehsass App Donation: Donors can easily donate surplus food by specifying their location through pinning. This ensures a seamless and precise food pickup process for the receivers.

Pin-based Food Pickup: Receivers can locate and collect donated food based on pinned locations. This feature enhances the accessibility and convenience of food collection.

User Records and About Section: The app incorporates a user profile section where individuals can maintain their donation history. An 'About' section provides additional information about the app's mission and functionality.

Technology Stack: Developed using Android Studio and Java language, the app leverages Firebase for efficient data management and real-time communication.

Priority:

The priorities for the Ehsass App development are as follows:

Pin-Based Location System: Implement a robust system for accurate pinning of donation and pickup locations. This ensures the reliability and effectiveness of the food donation process.

User Interface and Experience: Prioritize a user-friendly interface to enhance the overall experience for both donors and receivers. Intuitive design elements will encourage widespread adoption and participation.

Firebase Integration: Ensure seamless integration with Firebase to facilitate real-time data synchronization, enhancing the app's performance and reliability.

Security Measures: Implement security protocols to safeguard user data and maintain the confidentiality of sensitive information.

2.4.4. Stimulus/Response Sequences

User Launches the App;

Stimulus: User opens the Ehsass App on their mobile device.

Response: Welcome screen with app branding and options.

Donor Initiates Food Donation;

Stimulus: Donor selects "Donate Food" from the main menu.

Response: Donation form prompts, allowing the donor to enter details like food type, quantity, and location.

Donor Pins Location;

Stimulus: Donor selects "Pin Location" to specify where the food is available.

Response: Map interface opens, allowing the donor to drop a pin at the exact location.

Donor Submits Donation;

Stimulus: Donor submits the donation form.

Response: Confirmation message and a notification sent to potential receivers.

Receiver Checks Available Donations;

Stimulus: Receiver opens the app and selects "Find Donations."

Response: List of available donations with details and pinned locations.

Receiver Selects a Donation;

Stimulus: Receiver chooses a specific donation.

Response: Details of the donation and an option to confirm acceptance.

User Accesses About Section;

Stimulus: User navigates to the "About" section.

Response: Information about the Ehsass App, its purpose, and how it operates.

User Records Personal Information;

Stimulus: User selects "Record" in the About section.

Response: User profile page allowing the recording of personal details and preferences.

Firebase Integration;

Stimulus: App interacts with Firebase for real-time data updates.

Response: Seamless data synchronization, ensuring efficient communication.

App Exit;

Stimulus: User closes the Ehsass App.

Response: App safely exits, ensuring data privacy and security.

2.4.5. Functional Requirements

User Authentication;

The app must have a secure user authentication system to ensure the credibility of both donors and receivers. Users should be able to create accounts, log in, and recover passwords.

Donation Management;

Donors should be able to input details about the donated food, including type, quantity, and expiry date. A feature for attaching images of the donated food can enhance transparency.

The app must allow donors to select the location by pinning it on a map.

Location-based Matching;

The app should employ a matching algorithm to connect donors with nearby receivers based on pinned locations. Notifications should be sent to potential receivers when a new donation is available in their vicinity.

Real-time Updates with Firebase;

Integration with Firebase for real-time data updates and synchronization.

This ensures instant notifications and up-to-date information for both donors and receivers.

User Profile and Record Keeping;

Users should have profiles where they can maintain a record of their donations or received food.

The "About" section should display user-contributed records and achievements.

App Navigation and User Interface;

Intuitive navigation for seamless user experience. Clear sections for donation, finding donations, user profiles, and the "About" section.

Privacy and Security Measures;

Implement robust security measures to protect user data and maintain privacy. Ensure secure transmission of sensitive information.

Offline Functionality;

Allow users to access essential features even in offline mode, with data synchronization once an internet connection is established.

Feedback and Ratings;

Incorporate a system for users to provide feedback on the donation process.

Allow users to rate and review their experiences.

Documentation and Help Section

2.5. Nonfunctional Requirements

2.5.1. Performance Requirements

Response Time;

- The app must provide quick responses to user actions, ensuring a seamless experience.
- Response time for critical functions like donation matching should be within seconds.

Load Time;

- The app should load swiftly, even with a substantial amount of data.
- Efficient loading is essential for a positive user experience.

Scalability;

- The system should handle a growing user base and an increasing number of donations without a significant decrease in performance.
- Regular performance testing should ensure scalability.
- Offline Performance

The app must maintain functionality in offline mode, allowing users to view previous donations and recorded data.

Synchronization with Firebase should occur efficiently once an internet connection is reestablished.

Firebase Real-time Updates;

Real-time updates through Firebase should be near-instantaneous, ensuring that users receive timely notifications.

Optimize the frequency of data synchronization to balance real-time functionality with resource efficiency.

Location Services;

The location-based features, including pinning and matching, should utilize device location services efficiently.

Minimize battery consumption while ensuring accurate location tracking.

Data Retrieval Efficiency;

Retrieval of user records, donation details, and other data should be optimized for speed.

Implement caching mechanisms to reduce the need for repeated data retrieval.

Security Performance;

The app's security features, including encryption and authentication, should not compromise overall performance.

Regular security audits should be conducted to identify and address potential vulnerabilities.

User Interface Responsiveness;

User interface elements should respond promptly to touch and input gestures, providing a smooth and interactive experience.

Optimize UI components for efficient rendering.

The help section and documentation should load quickly, allowing users to access assistance without delays.

Ensure that external links within the documentation are responsive.

2.5.2. Safety Requirements

Data Security and Privacy

Implement robust encryption methods to safeguard user data during transmission and storage.

Adhere to data protection regulations to ensure user privacy.

Conduct regular security audits to identify and address potential vulnerabilities.

User Authentication

Ensure a secure user authentication process to prevent unauthorized access.

Implement account lockout mechanisms to protect against brute force attacks.

Location Privacy

Prioritize user location privacy by allowing users to control location-sharing preferences.

Clearly communicate how location data is used and give users the option to opt out.

Emergency Situations

Implement emergency features, allowing users to quickly contact relevant authorities or services in case of accidents or unforeseen circumstances.

Include an emergency button for immediate assistance.

User Education and Guidelines

Provide clear guidelines within the app to educate users on safe and responsible usage.

Include information on food safety practices to ensure donated food is safe for consumption.

In-App Communication Safety

Implement secure in-app communication channels between donors and receivers.

Include reporting mechanisms for inappropriate content or behavior.

Real-time Notification Safety

Ensure that real-time notifications do not distract users while driving or engaging in other critical activities.

Implement a Do Not Disturb mode for users in specific situations.

Food Safety Compliance

Encourage donors to adhere to food safety regulations and guidelines.

Provide information on safe handling, storage, and transportation of donated food.

Emergency Response Information

Include emergency response information within the app, such as contact details for relevant health and safety organizations.

Regular App Maintenance

Conduct regular maintenance checks to identify and resolve safety-related issues promptly.

Keep the app updated with the latest security patches and enhancements.

2.5.3. Security Requirements

Employ secure authentication methods, such as multi-factor authentication, to verify user identities. Implement account lockout mechanisms after a specified number of unsuccessful login attempts.

Encrypt sensitive data, including user credentials, donation details, and location information.

Use industry-standard encryption algorithms to protect data both at rest and in transit.

Utilize HTTPS to ensure secure communication between the app and the server.

Implement secure sockets layer (SSL) for encrypted data transmission. Deploy a robust firewall to monitor and filter incoming and outgoing network traffic. Configure firewall rules to allow only essential connections and services. Enforce strict access controls, granting users and system components access only to the resources they need.

Regularly review and update access permissions. Implement input validation mechanisms to prevent common security threats, such as SQL injection and cross-site scripting (XSS).

Sanitize user inputs to ensure data integrity. Implement secure coding practices to protect against mobile device vulnerabilities.

Encourage users to keep their devices updated with the latest security patches.

Ensure secure integration with Firebase by following Firebase security best practices.

Regularly review Firebase security settings and configurations. Implement secure session management to prevent session hijacking. Use secure tokens and regularly update session keys.

Develop and document an incident response plan to address security breaches promptly.

Establish communication channels to notify users in case of a security incident.

Adhere to privacy regulations and standards to protect user privacy.

Clearly communicate the app's privacy policy to users

2.5.4. Usability Requirements

Intuitive User Interface

The app should have a user-friendly interface with clear navigation and easily understandable icons.

Ensure a straightforward and intuitive design to accommodate users of varying technological backgrounds.

Accessibility

Design the app to be accessible to users with disabilities, incorporating features such as voice commands and screen reader compatibility.

Provide adjustable font sizes and color contrast options for enhanced readability.

Responsive Design

Ensure the app's layout and features adapt seamlessly to various screen sizes and orientations.

Optimize the user experience for both smartphones and tablets.

Efficient Onboarding Process

Create a smooth onboarding process for new users, guiding them through the app's features and functionalities.

Include tooltips or tutorials to assist users in understanding how to use the app effectively.

Clear Donation Process

Streamline the donation process with a step-by-step guide for donors.

Provide real-time feedback and confirmation messages to keep users informed about the status of their donations.

Location Pinning

Simplify the location pinning process, allowing donors to easily mark the location of the food donation on a map.

Ensure that receivers can effortlessly access and navigate to the pinned locations.

Efficient Search and Matching

Implement a user-friendly search functionality for receivers to find available donations based on their location.

Facilitate quick matching between donors and receivers, minimizing wait times.

About Section Clarity

Design the "About" section to be concise and informative, providing users with a clear understanding of the app's purpose, mission, and how it operates.

Include contact information and support details for further assistance.

User Record Presentation

Display user records in a visually appealing and organized manner within the app.

Use charts or graphs to illustrate users' donation history and contributions.

Feedback Mechanism

Implement an easy-to-use feedback mechanism to gather user opinions on the app's usability.

Regularly analyze user feedback to make continuous improvements.

Multi-language Support

Include support for multiple languages to cater to a diverse user base.

Allow users to choose their preferred language within the app settings.

2.5.5. Reliability Requirements

High System Availability

Ensure the app is available for use 24/7, minimizing downtime for both donors and receivers.

Implement a reliable server infrastructure to support continuous operation.

Data Integrity

Implement measures to maintain the integrity of user data, donation records, and other critical information.

Regularly perform data integrity checks and backups.

Real-time Data Synchronization

Achieve reliable real-time data synchronization with Firebase to keep information up-to-date across all devices.

Address any potential synchronization conflicts promptly.

Fault Tolerance

Design the app to gracefully handle unexpected errors or faults, preventing a single point of failure.

Implement error logging and monitoring for proactive issue resolution.

Automated Testing

Conduct thorough automated testing for various scenarios, including high user loads and simultaneous transactions.

Use testing tools to identify and rectify reliability issues during the development phase.

Security Reliability

Ensure that security measures, such as encryption and authentication, are consistently reliable.

Regularly update security protocols to address emerging threats.

User Notification Reliability

Guarantee the reliability of user notifications, ensuring that donors and receivers receive timely updates about their activities within the app.

Implement fallback mechanisms for cases where push notifications may not be received.

Location-based Services Reliability

Ensure the reliability of location-based services, providing accurate information for both donors and receivers.

Regularly test and optimize location tracking functionality.

Backup and Recovery

Establish a robust backup and recovery system to quickly restore the app's functionality in the event of data loss or system failure.

Regularly perform and test data backups.

Load Balancing

Implement load balancing mechanisms to distribute user loads evenly across servers.

Ensure consistent performance under varying levels of user activity.

Scalability

Design the app to be scalable, accommodating a growing user base and increased data volume without compromising performance.

Regularly assess and optimize scalability parameters.

2.5.6. Maintainability/Supportability Requirements

Develop the app with a modular architecture, making it easier to update and maintain specific components independently.

Ensure a well-organized and documented codebase for efficient collaboration among developers.

Documentation

Create comprehensive documentation covering the app's architecture, code structure, and deployment processes.

Include guidelines for future developers to understand and extend the functionality of the app.

Version Control

Utilize a version control system (e.g., Git) to track changes in the codebase.

Maintain a clear version history with detailed commit messages for easier debugging and rollback if necessary.

Continuous Integration/Continuous Deployment (CI/CD)

Implement CI/CD pipelines to automate the testing, building, and deployment processes.

Facilitate quick and reliable releases of new features or bug fixes.

Error Logging and Monitoring

Integrate robust error logging and monitoring tools to track app performance and identify issues proactively.

Regularly review logs to address potential problems before they impact users.

Scalability Planning

Design the app with scalability in mind, allowing for easy expansion as the user base grows.

Regularly assess and update scalability plans to accommodate increasing demands.

Software Updates and Patching

Establish a process for deploying software updates and patches efficiently.

Ensure that users are notified of updates, and provide mechanisms for automatic or user-initiated updates.

User Support Channels

Set up user support channels, including a helpdesk, email support, or a community forum.

Ensure prompt and helpful responses to user queries and issues.

User Education Resources

Develop user education materials, including tutorials and FAQs, to assist users in navigating and utilizing the app effectively.

Provide in-app help features for immediate assistance.

Collaboration and Knowledge Transfer

Facilitate knowledge transfer among development team members.

Encourage collaboration through tools like documentation, team meetings, and shared repositories.

Security Audits and Updates

Conduct regular security audits to identify and address potential vulnerabilities.

Keep third-party libraries and dependencies updated with the latest security patches.

2.5.7. Portability Requirements

Cross-Platform Compatibility

Design the app to be compatible with multiple mobile platforms, including Android and potentially iOS in the future.

Ensure a consistent user experience across different devices and operating systems.

Device Compatibility

Optimize the app to function seamlessly on a variety of Android devices, considering different screen sizes, resolutions, and hardware specifications.

Conduct thorough testing on popular Android devices to ensure compatibility.

Minimum System Requirements

Clearly define and communicate the minimum system requirements for running the app.

Ensure that the app is accessible to users with a wide range of Android devices, including older models.

Network Flexibility

Design the app to work efficiently with varying network conditions, accommodating users with different internet speeds and connectivity.

Implement mechanisms for offline functionality and data synchronization.

Language Localization

Support multiple languages to cater to a diverse user base.

Allow users to choose their preferred language within the app settings.

Cross-Browser Compatibility

If applicable (e.g., for web-based components), ensure compatibility with major web browsers for additional accessibility.

Scalability

Design the app architecture to be scalable, enabling easy adaptation to future technological advancements and changes in platform requirements.

Stay informed about updates to Android Studio and other development tools to maintain compatibility.

Adaptability to Screen Orientations

Optimize the app for both portrait and landscape screen orientations to accommodate user preferences.

Ensure a seamless transition between different orientations without loss of functionality.

Ease of Installation

Simplify the installation process to make it user-friendly for individuals with varying levels of technical expertise.

Provide clear instructions for downloading and installing the app from app stores.

Third-Party Integration Compatibility

Ensure compatibility with third-party services and libraries, such as Firebase, for a smooth user experience.

Regularly check for updates and ensure continued compatibility with the latest versions of integrated services.

2.5.8. Efficiency Requirements

Response Time

The app should provide quick response times for essential functions, such as donation matching and location pinning.

Aim for response times of a few seconds to enhance user experience.

Data Retrieval Efficiency

Optimize the efficiency of data retrieval, ensuring that user records, donation details, and other information are fetched swiftly.

Implement caching mechanisms to reduce the need for repeated data retrieval.

Real-time Updates

Ensure real-time updates for user activities, donations, and notifications through Firebase.

Minimize latency to provide users with timely information.

Location Services

Optimize location-based features, including pinning and matching, for accuracy and efficiency.

Implement intelligent algorithms to reduce battery consumption during location tracking.

Offline Functionality

Design the app to maintain functionality in offline mode, allowing users to view previous donations and recorded data.

Ensure seamless synchronization with Firebase once an internet connection is reestablished.

Firebase Integration

Efficiently integrate Firebase services, utilizing features such as real-time database updates and cloud messaging.

Optimize the connection with Firebase to minimize latency.

User Interface Responsiveness

Ensure responsive user interface components that swiftly react to user interactions.

Optimize UI elements for efficient rendering on various screen sizes and resolutions.

Resource Utilization

Monitor and optimize resource utilization, including CPU, memory, and network usage, to enhance app performance.

Conduct regular performance testing to identify and address resource-intensive processes.

Search and Matching Algorithm

Optimize the search and matching algorithms to efficiently connect donors with suitable receivers.

Regularly refine algorithms based on usage patterns and feedback.

Security Performance

Ensure that security features, including encryption and authentication, do not compromise overall app performance.

Regularly conduct security audits to identify and address potential vulnerabilities without impacting efficiency.

Documentation Accessibility

Ensure that help documentation and guidelines are easily accessible within the app.

Provide an intuitive help section for users to quickly find information.

2.6. Domain Requirements

The app should facilitate a streamlined workflow for donors to easily donate surplus food by pinning their location.

Receivers should be able to view available donations, accept them, and receive the food from the specified pin location.

Location-Based Services

Implement accurate location-based services to enable donors and receivers to pin and access donation locations precisely.

Ensure compatibility with various GPS-enabled devices for location accuracy.

User Records and Profiles

Maintain comprehensive user records, including donation history and preferences.

Allow users to create and manage profiles, enhancing the personalized experience within the app.

About Section

Include an informative "About" section detailing the app's mission, objectives, and operational guidelines.

Provide information on how users can effectively use the app and contribute to its purpose.

Android Studio and Java Development

Develop the app using Android Studio and Java programming language, adhering to best coding practices.

Leverage Android Studio's features for efficient app development and debugging.

Firebase Integration

Connect the app to Firebase to leverage real-time database updates, cloud messaging, and other Firebase features.

Ensure secure and reliable communication between the app and Firebase services.

Ehsass App Management

Design the app to effectively manage Ehsass App from hotels, marriage halls, and other relevant places.

Incorporate features for categorizing and documenting the type and quantity of donated food.

Security and Privacy

Prioritize the security of user data, implementing encryption and authentication measures.

Adhere to privacy regulations to protect user information and maintain trust.

Communication Interfaces

Develop clear communication interfaces for donors and receivers to interact within the app.

Enable notifications and messaging functionalities to facilitate communication between users.

Efficiency and Performance

Optimize the app for efficient performance, including quick response times, real-time updates, and offline functionality.

Ensure that the app efficiently utilizes resources without compromising user experience.

Community Engagement

Encourage community engagement by providing avenues for users to share their experiences and success stories.

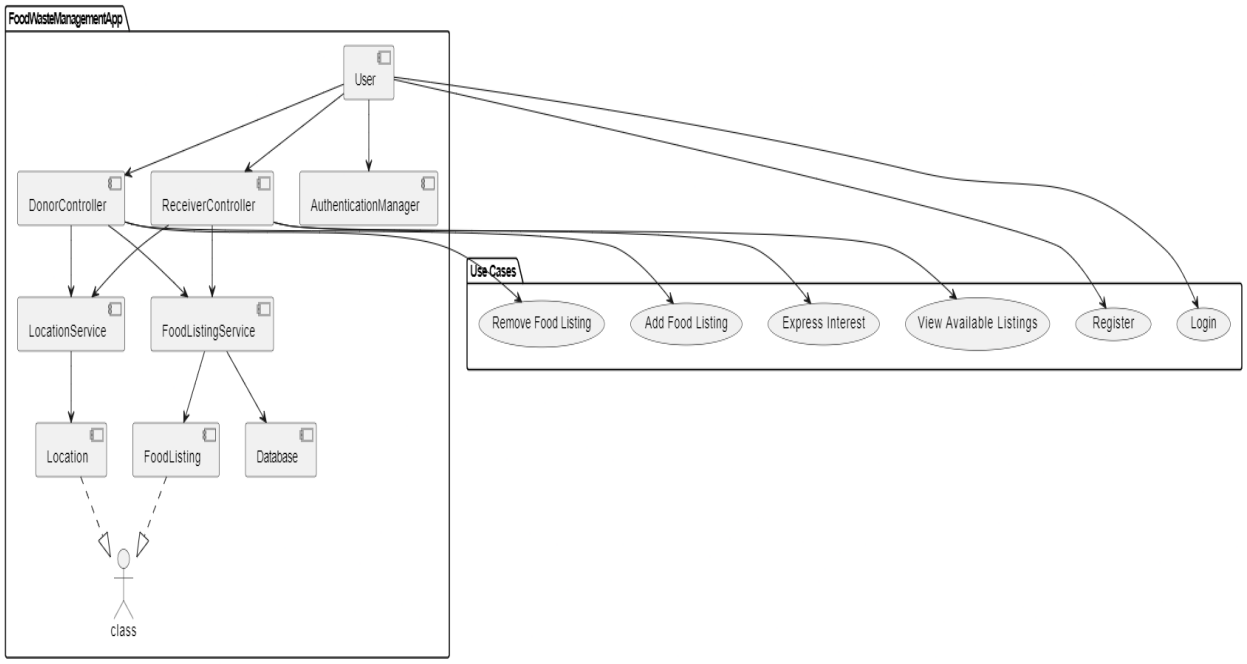
Implement feedback mechanisms to gather user opinions and continuously improve the app.

Chapter 3

Use Case Analysis

Use Case Analysis:

3.1. Use Case Model



3.2. Use Cases Description

Use Case 1: Donate Surplus Food
Actors:
Donor
Description:

- The Donor opens the Ehsass App on their Android device.
- The Donor logs in or creates a new account if they don't have one.
- The Donor selects the "Donate" option.
- The app prompts the Donor to pin the location where they want to donate surplus food.
- The Donor pins the location on the map and provides details about the type and quantity of food to be donated.
- After confirming the donation, the app records the information and updates the database in real-time using Firebase.

Use Case 2: Receive Donated Food

Actors:

Receiver

Description:

- The Receiver opens the Ehsass App on their Android device.
- The Receiver logs in or creates a new account if they don't have one.
- The Receiver selects the "Receive" option.
- The app displays available donations based on the Receiver's location.
- The Receiver reviews donation details and chooses a suitable donation.
- The app provides navigation to the Donor's pinned location for food pickup.
- After receiving the food, the Receiver updates the app, confirming the successful transaction.

Use Case 3: View User Records

Actors:

User

Description:

- The User opens the Ehsass App on their Android device.
- The User logs in.
- The User navigates to the "User Records" section.
- The app displays a summary of the User's donation and transaction history.
- The User can view details such as the types and quantities of donated food, pickup locations, and timestamps.

Use Case 4: About Section

Actors:

User

Description:

- The User opens the Ehsass App on their Android device.
- The User navigates to the "About" section.
- The app provides information about the app's mission, objectives, and guidelines.
- The User gains insights into how the app works and its impact on managing Ehsass App.

Use Case 5: App Maintenance and Updates

Actors:

System Administrator

Description:

- The System Administrator monitors app performance and user feedback.
- Based on feedback and analytics, the Administrator identifies areas for improvement.
- The Administrator releases updates, implementing bug fixes, new features, or optimizations through the app stores.
- Users receive notifications about the update and can choose to install it.

Chapter 4

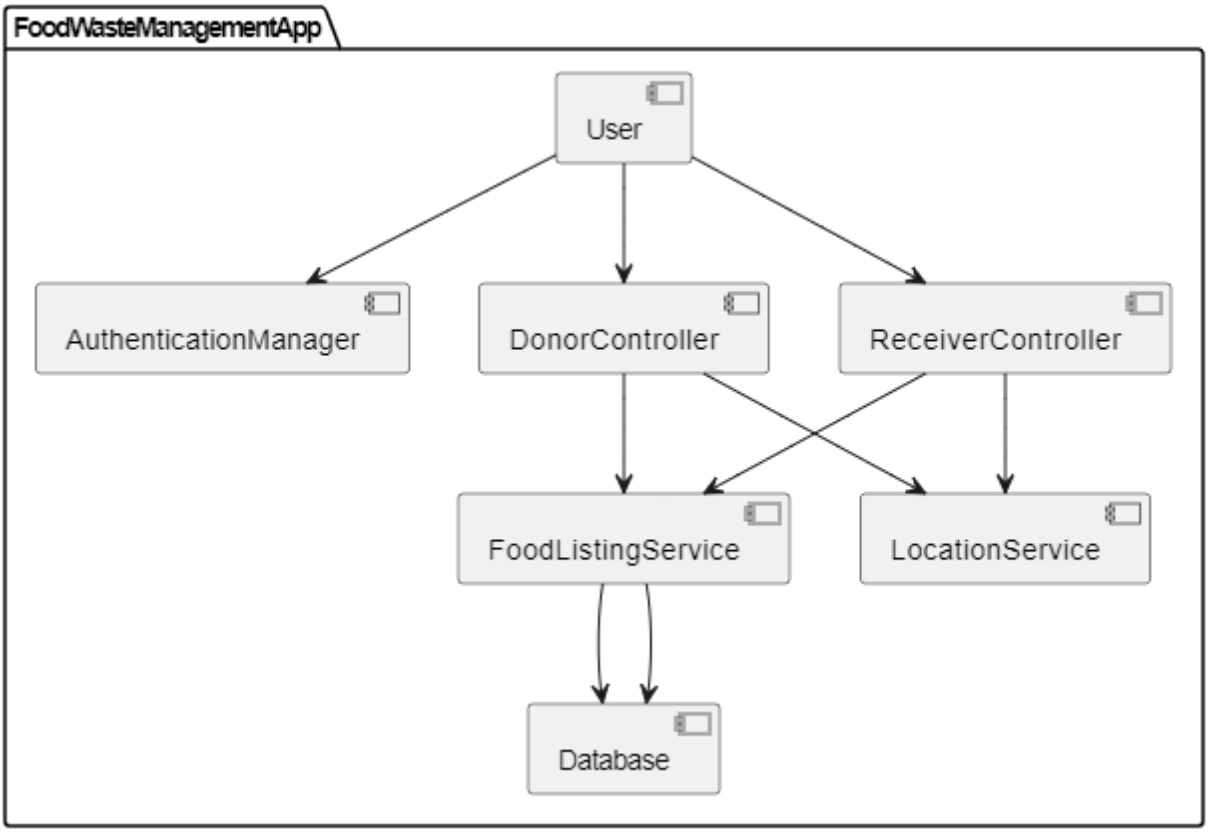
System Design

System Design:

As per the knowledge, the technology is going advanced and growing day by day. Over main motto is to help needy people. Many people who wish to donate things to needy organizations also can use the idea behind over project. We have tried to reduce food wastage by giving Ehsass App to people or organization who need it.

If someone is in need of food, that person can find a food donor directly through our app, and for that, only that person needs to install our app in his mobile, and open the food map in it and contact the nearby food donors. The biggest feature of our app is the food map in our app. In it, you can see both the users around donors who are going to donate food, and receivers who need food.

4.1. Architecture Diagram



4.2. Domain Model

The key concepts and relationships in the Ehsass App, a domain model can be created. The domain model outlines the major entities and their interactions within the system:

User:
Attributes: Use rid, Name, Email, Password
Subclasses: Donor, Receiver

Donor:
Attributes: DonationID, FoodType, Quantity, ExpiryDate, Location
Actions: SubmitDonation(), ViewHistory()

Receiver:
Attributes: RequestID, FoodType, Quantity, Location
Actions: PlaceRequest(), ViewReceivedRequests()

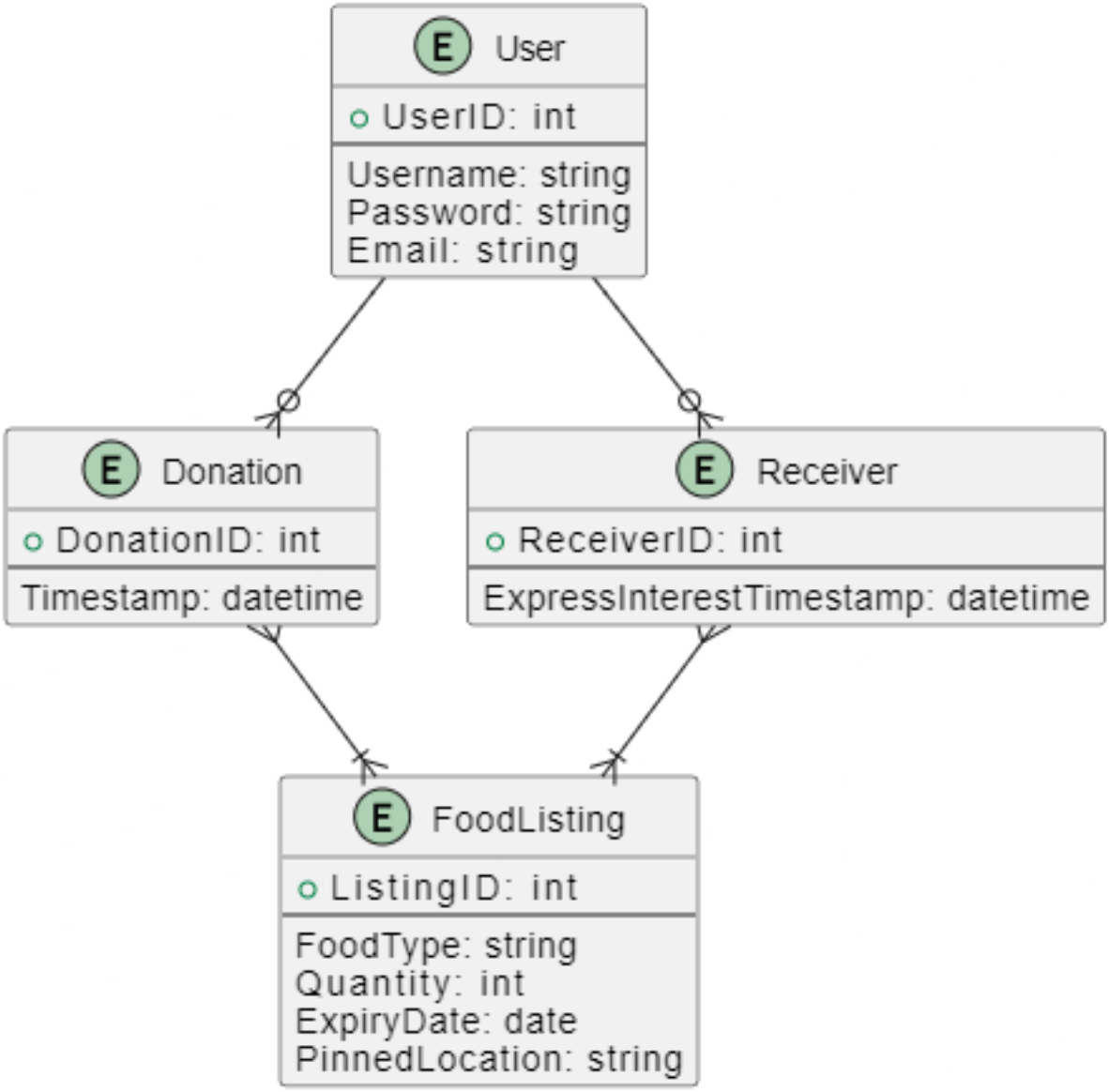
Location:
Attributes: LocationID, Latitude, Longitude
Actions: PinLocation()

App:
Actions: AuthenticateUser(), ProcessDonation(), MatchRequests()

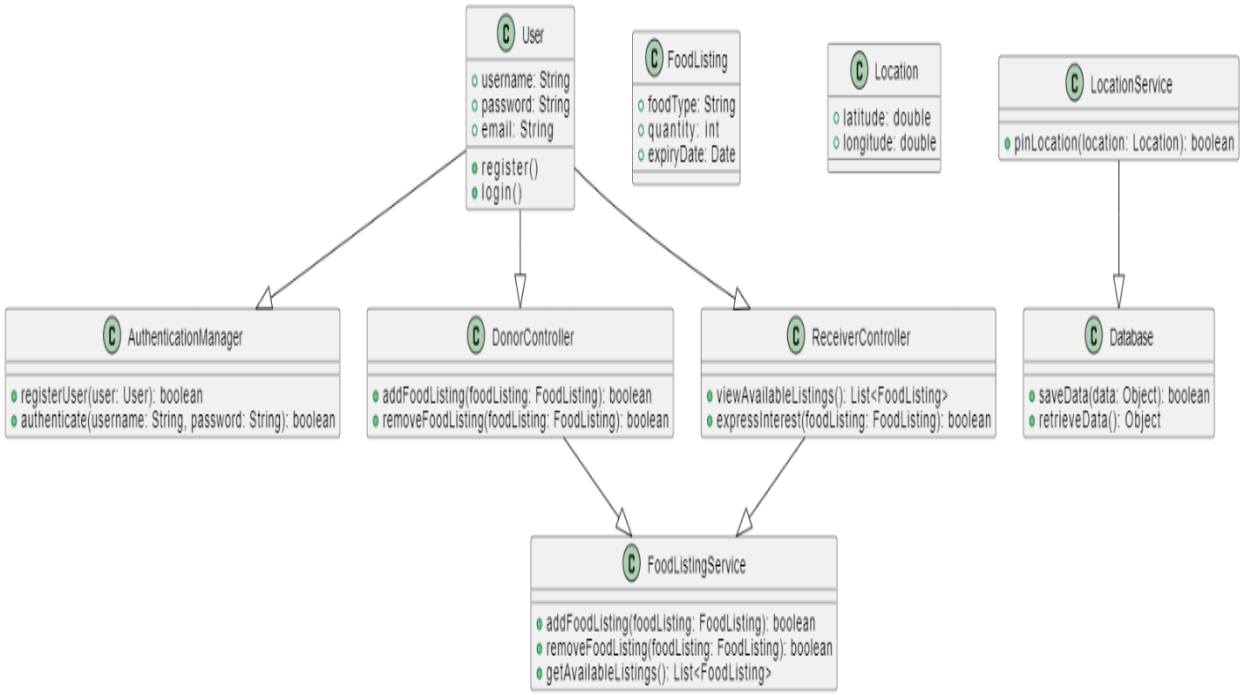
Database:
Actions: StoreUserData(), StoreDonationData(), StoreRequestData()

External Systems:
Attributes: FoodSafetyTemplate, RestaurantManagementSystem
Actions: VerifyFoodSafety(), OptimizeDeliveryRoute()

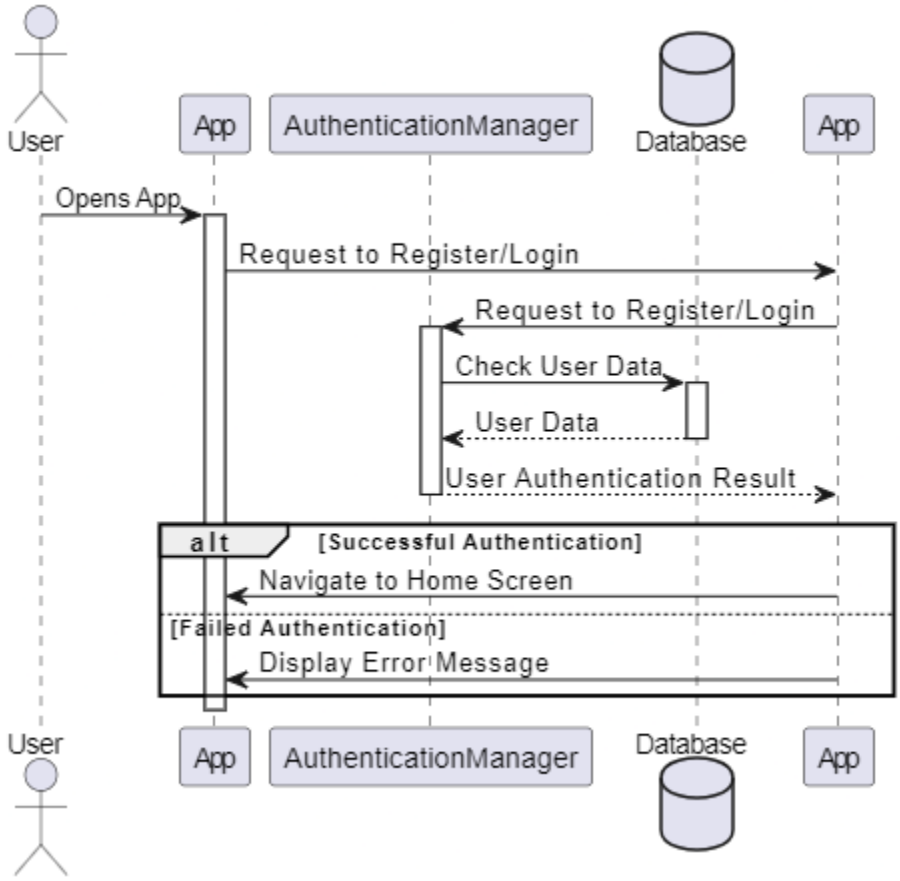
4.3. Entity Relationship Diagram with data dictionary



4.4. Class Diagram



4.5. Sequence / Collaboration Diagram



4.5. Operation contracts

Operation Contracts for Ehsass App - Food Waste Management Project

1. Donor Section Operations

1.1. Donation Submission:

- Donors can submit details of surplus food through the app.
- The app shall capture information such as food type, quantity, and expiry date.
- GPS coordinates will be recorded to locate the donation source.

Ehsass App - Food Donation

1.2. Verification Process:

The system will verify the donated food's quality and adherence to safety standards.

Donors will receive a confirmation once the food passes the verification process.

Food Safety Template Hub

1.3. Donor Feedback:

- Donors can provide feedback on the donation process and app usability.
- Continuous improvement based on donor feedback will be implemented.
- Solution for Food Waste Management

2. Receiver Section Operations

2.1. Food Request and Matching:

Receivers (organizations, individuals) can request specific types and quantities of food.

The app will match donation offerings with receiver requests for efficient distribution.

Food Waste Management Software

2.2. Delivery Logistics:

- The app will optimize the route for food delivery based on the receiver's location.
- Real-time tracking will be available for the delivery process.
- Food waste and its management in the foodservice sector

2. Pin Location and Data Recording

3.1. Pin Location:

- Both donors and receivers can view pinned locations of donation and delivery points.
- The app will use GPS to accurately display locations on a map.

Ehsass App - Pin Location

3.2. Data Recording:

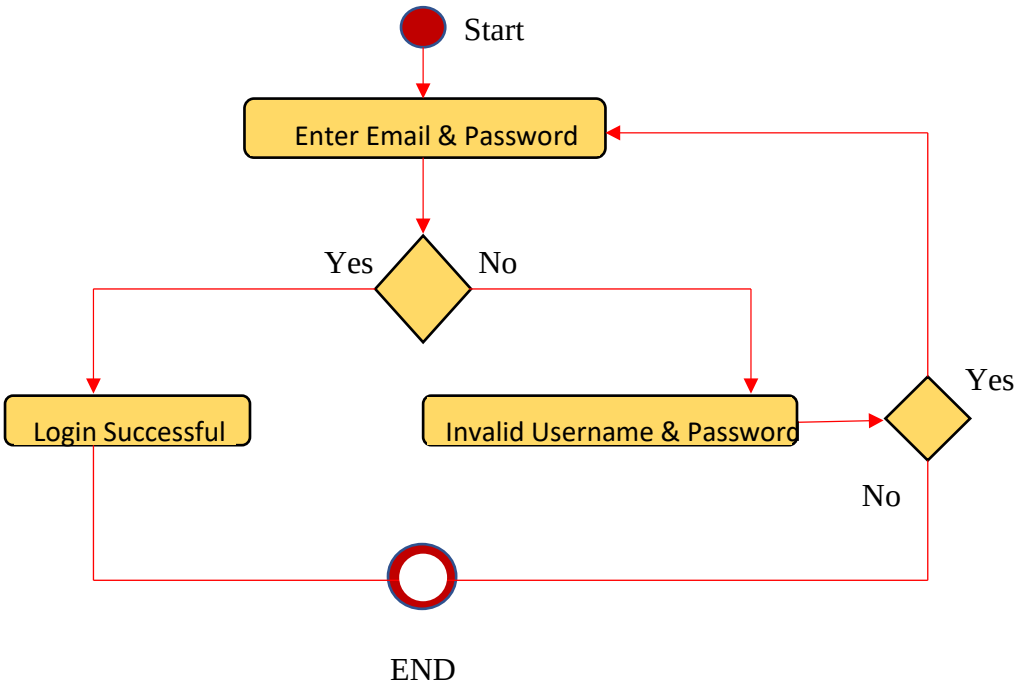
The app will maintain a secure database recording all transactions and interactions.

Data on donations, verifications, requests, and deliveries will be stored for analytics.

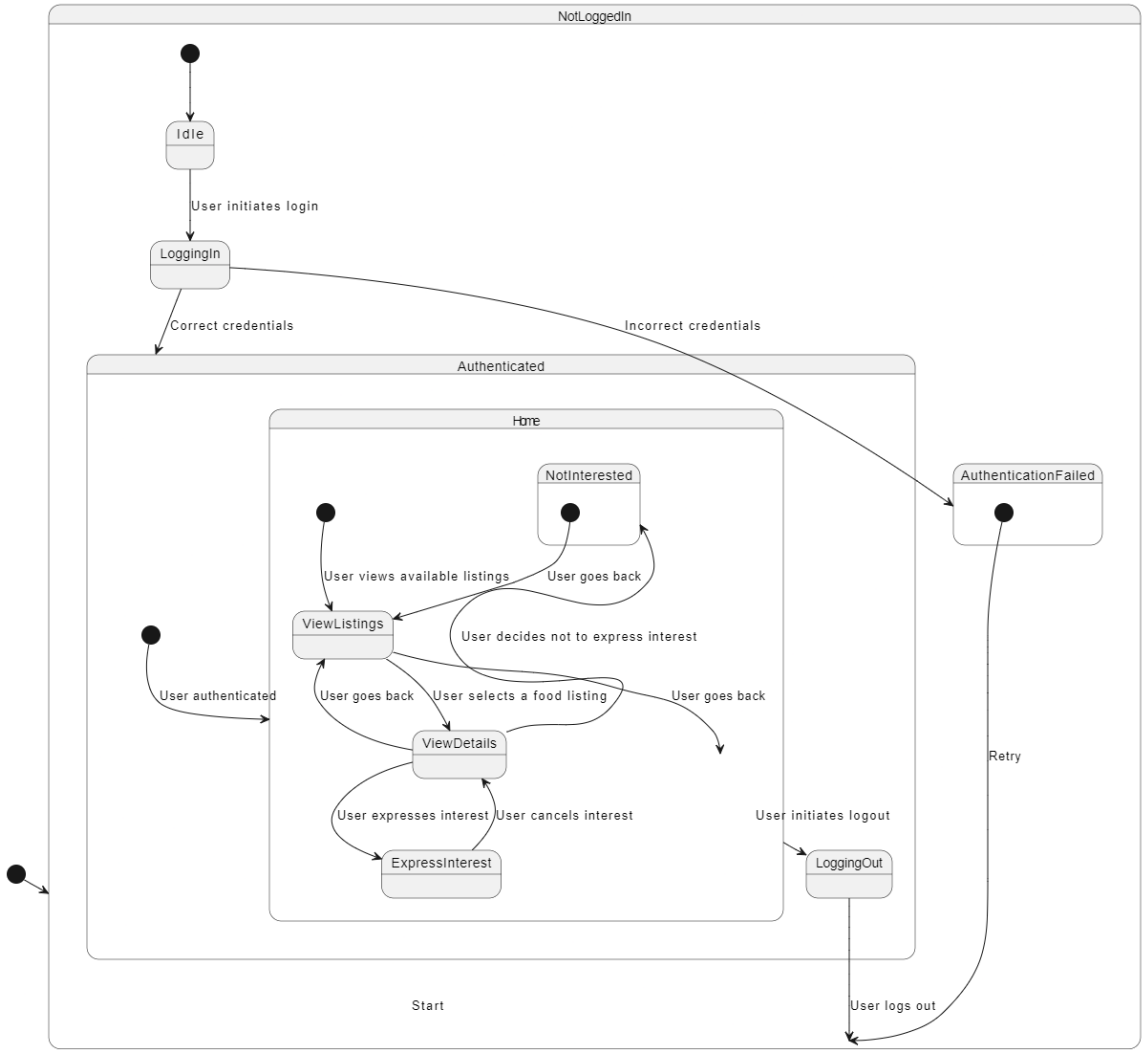
Service Agreement for Solid Waste Collection

4.6. Activity Diagram

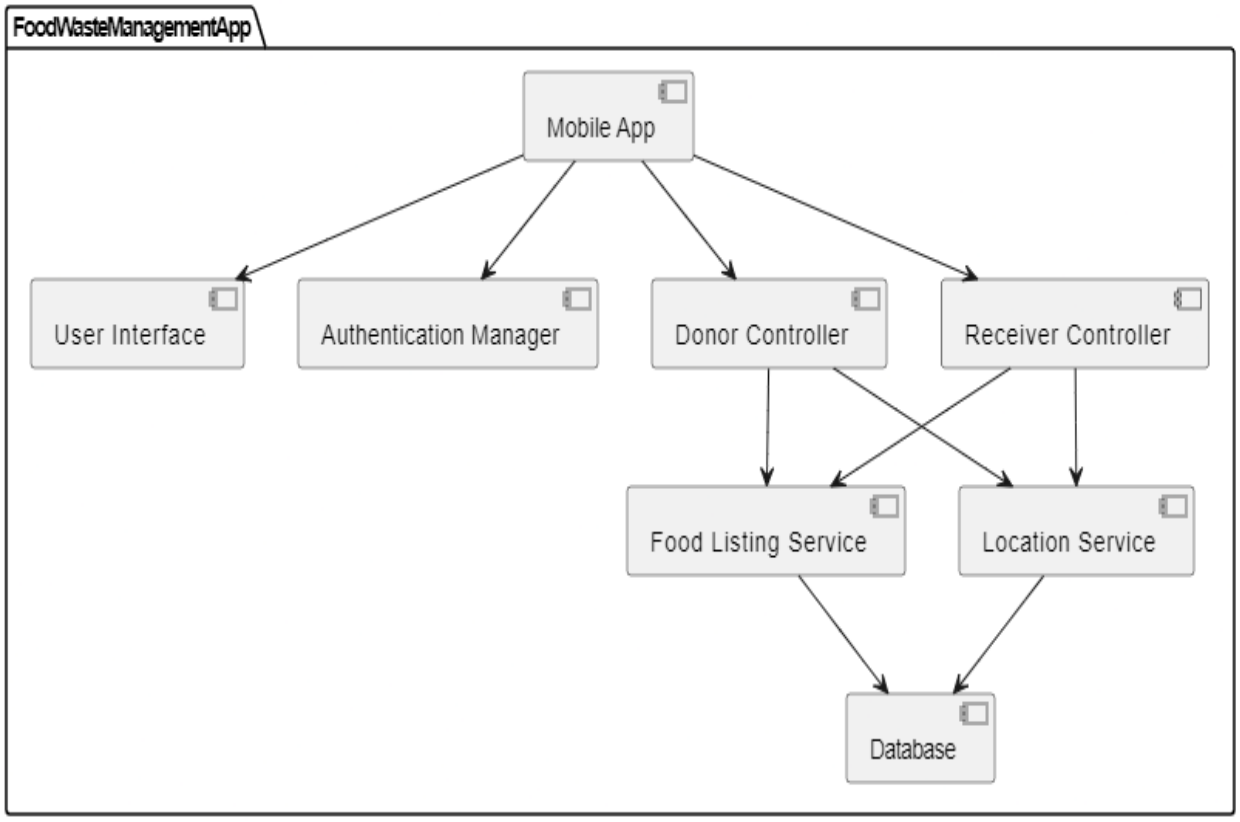
4.6.1 Activity Diagram for login :-



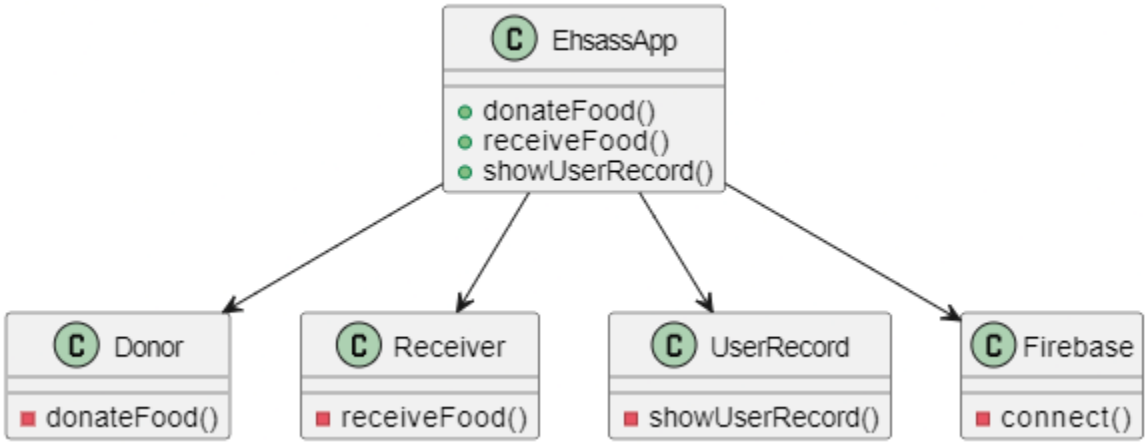
4.7. State Transition Diagram



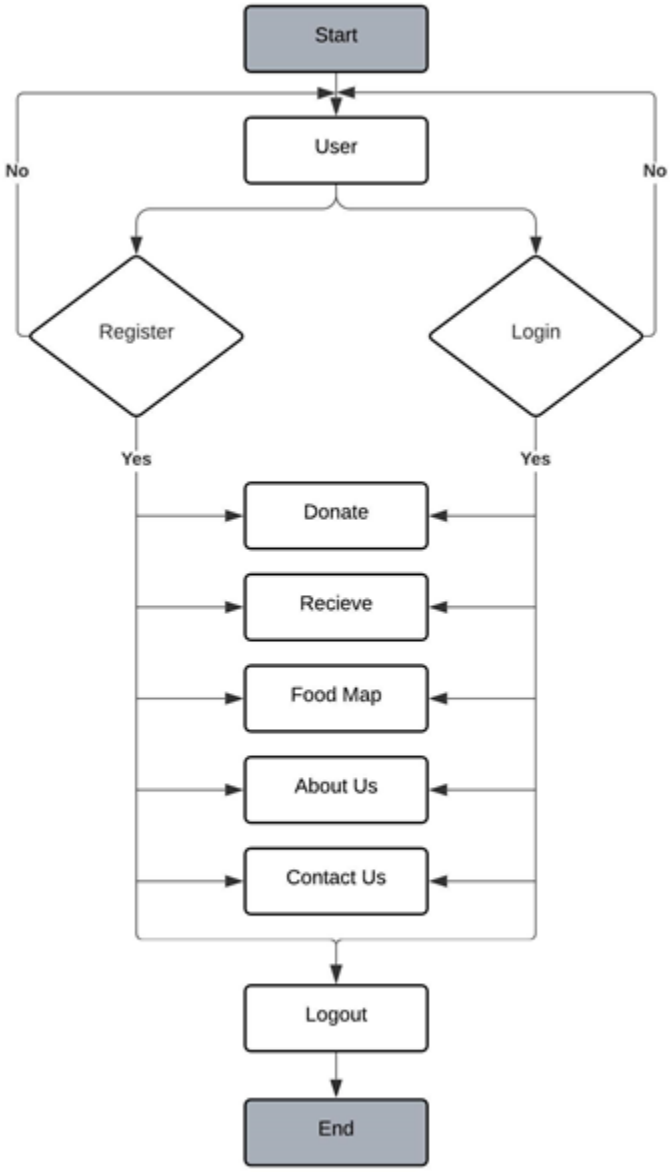
4.8. Component Diagram



4.9. Deployment Diagram



4.10. Data Flow diagram



Chapter 5

Implementation

Chapter 5: Implementation

5.1. Important Flow Control/Pseudo codes

```
class User:
   id
   name
   userType
   contactInfo

class IndividualConsumer(User):
   dietaryPreferences
   foodInventory

def logFoodPurchase(purchase):
      add to foodInventory

 def trackConsumption(consumption):
      update foodInventory

 def monitorWaste(waste):
      log waste

 def receiveAlerts():
      check foodInventory for expiring items
      send alerts

def generateReports():
      analyze waste patterns
      return report

class Household(User):
   members
   collectiveFoodInventory

 def addMember(member):
      add to members

 def logFoodPurchase(purchase):
      add to collectiveFoodInventory

def trackConsumption(consumption):
      update collectiveFoodInventory

 def monitorWaste(waste):
```

```
        log waste

def receiveAlerts():
        check collectiveFoodInventory for expiring items
        send alerts to all members

def generateReports():
        analyze household waste patterns
        return report

class RestaurantAndFoodBusiness(User):
    menuItems
    inventory

def manageInventory(purchase, usage):
        update inventory

def trackWaste(waste):
        log waste

def donateSurplusFood(foodItem):
        coordinate with FoodBank

def receiveAlerts():
        check inventory for expiring items
        send alerts

def generateReports():
        analyze waste and cost implications
        return report

class FoodBankAndCharitableOrganization(User):
    capacity
    donationsReceived

def receiveDonation(donation):
        update donationsReceived

def distributeFood(foodItem):
        update capacity

def generateReports():
        track donations and distributions
        return impact report
```

class MunicipalWasteManagementAuthority(User):
collectionSchedules
communityWasteData

def monitorWasteCollection():
track waste collection from users

def coordinateRecyclingPrograms():
implement recycling initiatives

def sendCommunityUpdates():
notify users about collection days and recycling tips

def generateReports():
analyze community waste data
return report

class AdministratorAndAppDeveloper(User):
systemSettings
appPerformanceData

def manageUserAccounts(userAccount):
update account settings

def ensureDataSecurity():
monitor and secure data

def maintainAppPerformance():
track app performance metrics
implement updates

def generateReports():
analyze usage and performance data
return report

class EducatorAndAdvocate(User):
educationalContent
campaigns

def createContent(content):
add to educationalContent

def organizeCampaigns(campaign):

schedule and promote campaign

def engageUsers():

interact with users to promote best practices

def generateReports():

measure campaign impact

return report

 # Main Program Logic

```
def main():
    # Example usage
    user = IndividualConsumer(id=1, name="John Doe")
    user.logFoodPurchase(purchase="Apple")
    user.trackConsumption(consumption="Apple")
    user.monitorWaste(waste="Apple Core")
    user.receiveAlerts()
    report = user.generateReports()
    print(report)

if __name__ == "__main__":
    main()
```

5.2. Components, Libraries, Web Services and stubs

To implement the Ehsass App efficiently, various components, libraries, web services, and stubs can be utilized:

Mobile App Components:

Donor Section UI:

Allows donors to submit food donations.

Captures food details, quantity, and expiration date.

Receiver Section UI:

Enables receivers to request specific types and quantities of food.

Displays a history of received donations.

Location Services:

Integrates GPS for pinning accurate donor and receiver locations.

Backend Components:

Authentication Module:

Manages user authentication and authorization.

Donation Processing Module:

Processes and verifies food donations.

Request Matching Module:

Matches donation offerings with receiver requests.

Libraries:

Geolocation Library:

Utilized for accurate mapping and location-based services.

Security Library:

Ensures secure data transmission and storage.

Database Access Library:

Facilitates interaction with the database for data storage and retrieval.

Web Services:

Food Safety Template Web Service:

Integrates with a food safety template system for verifying food donations.

External Restaurant Management Web Service:

Enhances handling of food requests by integrating with a restaurant management system.

Stubs;

Testing Stubs:

Simulate interactions with external systems during testing.

Location Stub:

Provides mock location data for testing location-based features.

Hosting Service

Hosts the app backend, database, and web services.

Domain Service

Manages the app's domain and ensures accessibility.

5.3. Deployment Environment

The deployment environment for the Ehsass App involves various components and services to ensure the effective operation of the system. Here's an outline of the deployment environment:

Client Devices:

Donor and receiver mobile devices with the Ehsass App installed.

Server Infrastructure:

Application Server:

- Hosts the backend logic and processes of the Ehsass App.
- Manages user authentication, donation processing, and request matching.
- Database Server:

- Stores and manages data related to users, donations, requests, and locations.
- Ensures data integrity and security.

External Services:

Geolocation Service:

- Provides accurate GPS data for pinning donor and receiver locations.
- Food Safety Template Service:
- Integrates with an external system to verify the safety of food donations.
- Restaurant Management System Service:
- Enhances food request handling through integration with an external restaurant management system.

5.4. Tools and Techniques

To develop the Ehsass App efficiently, leveraging Android Studio and Java, various tools and techniques can be employed:

Integrated Development Environment (IDE):

Android Studio:

Utilize Android Studio as the primary IDE for Android app development.

Leverage features such as code editing, debugging, and UI design.

Programming Language:

Java:

Use Java as the primary programming language for Android app development.

Leverage Java's object-oriented principles for modular and maintainable code.

UI/UX Design:

XML Layouts:

Design app layouts using XML for the Android user interface.

Implement responsive and user-friendly designs for the donor and receiver sections.

Backend Development:

Java Backend Logic:

- Develop backend logic in Java to handle donation processing, request matching, and user authentication.
- Ensure efficient and secure server-side operations.

Database Management:

SQLite Database:

- Utilize SQLite as the embedded database for storing app-related data.
- 0mplement database queries for efficient data retrieval and storage.

Geolocation Services:

Google Maps API:

- Integrate Google Maps API to access geolocation services.
- Pin accurate donor and receiver locations within the app.

Authentication and Security:

Firebase Authentication:

Implement Firebase Authentication for secure user authentication.

Ensure data security during communication between client devices and the backend.

Version Control:

Git:

- Use Git for version control to track changes and collaborate with team members.
- Enhance code management and collaboration efficiency.

Testing:

JUnit for Unit Testing:

- Implement JUnit for unit testing of individual components.
- Ensure code reliability and functionality.

Continuous Integration:

Jenkins:

- Implement Jenkins for continuous integration to automate building and testing processes.
- Streamline the development workflow and enhance code quality.

Documentation:

Javadoc:

- Use Javadoc for generating documentation for the Java code.
- Ensure comprehensive and well-documented codebase.

5.5. Best Practices / Coding Standards

The development of an "Ehsass Food Waste Management System" would require adherence to coding standards to ensure maintainability, extensibility, and consistency. Some of the key coding standards that should be considered include:

Naming conventions: Use consistent and descriptive naming conventions for variables, functions, and classes.

Code documentation: Document the code clearly and concisely to explain its purpose and usage.

Code modularity: Structure the code into modular and reusable components to promote maintainability and extensibility.

Error handling: Implement robust error handling mechanisms to ensure graceful handling of unexpected conditions.

Testing: Implement comprehensive testing strategies to ensure the system's functionality and reliability.

5.6. Version Control

Version control systems such as Git or Mercurial are essential for managing the development of the "Ehsass Food Waste Management System" project. These systems allow for tracking changes, maintaining different versions of the code, and collaborating with other developers. Version control practices should be followed diligently to ensure the project's integrity and traceability.

Chapter 6
Testing and Evaluation

6.1. Equivalence partitioning

Equivalence partitioning is a software testing technique that divides input data into partitions of equivalent data from which test cases can be derived. For a food waste management app like Ehsass, we can define equivalence classes based on different functionalities and user interactions.

Here are some equivalence partitions for various features of the food waste management app:

User Registration and Profile Management

1. Valid Registration Data:

Inputs: Valid email, strong password, correct user details (name, contact info).

Expected Output: Successful account creation and login.

2. Invalid Registration Data:

Inputs: Invalid email, weak password, incomplete user details.

Expected Output: Error messages and prompts to correct the input data.

3. Profile Updates:

Inputs: Valid updated user details.

Expected Output: Successful profile update.

4. Invalid Profile Updates:

Inputs: Invalid or incomplete user details.

Expected Output: Error messages and prompts to correct the input data.

Food Inventory Management

1. Logging Valid Food Purchase:

Inputs: Valid food item details (name, quantity, expiration date).

Expected Output: Food item added to inventory successfully.

2. Logging Invalid Food Purchase:

Inputs: Missing or invalid food item details.

Expected Output: Error message and prompt to provide correct details.

3. Tracking Consumption:

Inputs: Valid consumption data (food item, quantity).

Expected Output: Inventory updated to reflect consumption.

4. Invalid Consumption Data:

Inputs: Non-existent food item, invalid quantity.

Expected Output: Error message and prompt to correct the input data.

Waste Monitoring and Alerts

1. Valid Waste Logging:

- Inputs: Valid waste details (food item, quantity, reason).
- Expected Output: Waste logged successfully and inventory updated.

2. Invalid Waste Logging:

Inputs: Missing or invalid waste details.

Expected Output: Error message and prompt to provide correct details.

Donations Management

1. Valid Donation:

Inputs: Valid food item details, recipient organization details.

Expected Output: Donation logged and recipient notified.

2. Invalid Donation:

Inputs: Missing or invalid food item details, recipient organization details.

Expected Output: Error message and prompt to correct the input data.

Reports Generation

1. Valid Data for Report:

Inputs: Existing food inventory, consumption, and waste data.

Expected Output: Accurate report generated with insights.

2. No Data for Report:

Inputs: Empty food inventory, no consumption, and waste data.

Expected Output: Report indicating lack of data.

Notifications and Alerts

1. Valid Notification Setup:

Inputs: Valid user preferences for notifications (e.g., email, push notifications).

Expected Output: Notifications configured successfully and sent as per settings.

2. Invalid Notification Setup:

Inputs: Invalid user preferences or contact information.

Expected Output: Error message and prompt to correct the input data

By defining these equivalence partitions, you can ensure comprehensive testing coverage for the food waste management app, addressing various scenarios and user interactions systematically.

6.2. Boundary value analysis

Boundary Value Analysis (BVA) is a software testing technique crucial for ensuring the robustness and reliability of a food waste management app like Ehsass. For various functionalities such as user registration, profile management, food inventory management,

donations, reports generation, and notifications, BVA involves testing input parameters at their boundaries. For instance, in user registration, BVA entails testing the lower and upper boundaries of email and password lengths, as well as valid and invalid email formats. Similarly, in food inventory management, BVA involves testing the minimum and maximum allowed quantities for food purchases, consumption, and waste logging. This includes validating single unit transactions and those approaching the upper limits. For notifications and alerts, BVA examines enabling and disabling notification preferences, ensuring that the system behaves correctly at the boundary conditions. By systematically testing input parameters at their boundaries, BVA helps identify potential issues related to input validation, boundary conditions, and system integration, thereby enhancing the app's reliability and resilience.

6.3. Data flow testing

Data flow testing is a software testing technique that focuses on the flow of data within the application. For a food waste management app like Ehsass, data flow testing can be categorized into various scenarios based on how data is processed and manipulated within the system.

Here are some data flow testing scenarios for the food waste management app:

1. Input-Output Testing:

Scenario: Verify that data entered by the user is correctly processed and reflected in the output.

Example: Inputting a food item purchase and ensuring it appears in the user's inventory.

2. Control Flow Testing:

Scenario: Verify that the application navigates correctly through different screens or modules based on user input.

Example: Testing the flow from the registration/login screen to the main dashboard upon successful authentication.

3. Error Handling Testing:

Scenario: Verify that the application handles errors and exceptions appropriately.

Example: Testing the behavior when attempting to log consumption of a non-existent food item.

4. Data Transformation Testing:

Scenario: Verify that data is transformed accurately between different formats or representations.

Example: Testing the conversion of user-entered expiration date formats into a standardized format for internal processing.

5. Data Integration Testing:

Scenario: Verify that data from external sources or modules is integrated correctly into the application.

Example: Testing the integration of municipal waste collection schedules into the app's notification system.

6. Data Validation Testing:

Scenario: Verify that input data is validated correctly to ensure its integrity and accuracy.

Example: Testing the validation of user-entered email addresses during registration.

7. Data Persistence Testing:

Scenario: Verify that data is stored persistently and retrieved accurately when needed.

Example: Testing the storage and retrieval of user preferences for notification settings.

8. Data Privacy and Security Testing:

Scenario: Verify that sensitive data is handled securely and protected from unauthorized access.

Example: Testing the encryption of user passwords stored in the database.

9. Data Backup and Recovery Testing:

Scenario: Verify that data backup and recovery mechanisms work effectively to prevent data loss.

Example: Testing the restoration of user data from backup after a system failure.

10. Data Archiving and Purging Testing:

Scenario: Verify that data archiving and purging processes work correctly to manage data storage efficiently.

Example: Testing the archiving of expired food items and purging of old user activity logs.

By designing and executing tests based on these data flow testing scenarios, you can ensure that the food waste management app functions correctly and handles data effectively throughout its lifecycle.

6.4. Unit testing

Unit testing for a food waste management app project involves systematically testing individual units or components of the software to ensure they function correctly in isolation. In Python, the unit test framework provides a convenient way to write and execute unit tests. Within this framework, each test case typically starts with the `setup()` method, which initializes the necessary objects or resources before each test method runs. Subsequent test methods, such as `test_log_food_purchase_valid` and `test_track_consumption_invalid`, are written to target specific functionalities of the app's core classes, such as logging food purchases, tracking consumption, and monitoring waste. Within each test method, assertions are made using methods like `assertTrue()` or `assertFalse()` to verify whether the expected outcomes match the actual results. Through comprehensive unit testing, developers can identify and fix bugs early in the development process, ensuring the reliability and stability of the food waste management app.

6.5. Integration testing

Integration testing for a food waste management app involves testing the interaction between different modules or components to ensure they function together correctly. Here's how integration testing can be structured for the app:

1. User Registration and Authentication:

Test Case: Verify that user registration integrates with the authentication system.

Steps:

1. Register a new user with valid credentials.
2. Attempt to login using the registered credentials.

Expected Result: User is able to login successfully after registration.

2. Food Inventory Management Integration:

Test Case: Test integration between logging food purchases, tracking consumption, and monitoring waste.

Steps:

1. Log a food purchase.
2. Track consumption of the purchased food item.
3. Monitor waste for any leftover or expired items.

Expected Result: Food purchase reflects in inventory, consumption updates affect inventory levels, and waste monitoring accurately tracks disposal.

3. Notifications and Alerts Integration:

Test Case: Ensure integration between notifications/alerts and relevant app functionalities.

Steps:

1. Set up notification preferences.
2. Trigger alerts for expiring items or significant waste.

Expected Result: Users receive notifications/alerts as configured based on their preferences and relevant events in the app.

4. Donations Management Integration:

Test Case: Test integration between logging donations, managing surplus food, and coordinating with charitable organizations.

Steps:

1. Log a donation of surplus food items.
2. Verify that the donation is reflected in the inventory and properly recorded.
3. Ensure the donation is coordinated with the intended charitable organization.

Expected Result: Donations are accurately recorded, surplus food is properly managed, and donations are successfully coordinated with charitable organizations.

5. Reports Generation Integration:

Test Case: Verify integration between data collection, analysis, and report generation functionalities.

Steps:

1. Generate a report based on existing data (e.g., inventory, consumption, waste).
2. Verify that the report provides meaningful insights and analysis.

Expected Result: Reports are generated accurately based on collected data, providing valuable insights into food management and waste reduction efforts.

6. Municipal Waste Management Integration:

Test Case: Ensure integration with municipal waste management systems for waste collection coordination.

Steps:

1. Integrate municipal waste collection schedules and locations.
2. Verify that users are notified of collection schedules and locations.

Expected Result: Users receive accurate notifications about municipal waste collection schedules and locations, facilitating proper waste disposal.

7. Cross-Module Integration:

Test Case: Test integration between different modules to ensure seamless functionality across the app.

Steps:

1. Perform end-to-end scenarios involving multiple functionalities, such as purchasing food items, consuming them, and managing waste.

Expected Result: All modules interact smoothly, and the app functions as expected throughout various user workflows.

8. API Integration (if applicable):

Test Case: Verify integration with external APIs for functionalities like weather forecasts (for food storage recommendations), donation coordination, etc.

Steps:

1. Send requests to external APIs.
2. Verify that the app handles responses correctly and integrates the data into relevant functionalities.

Expected Result: App successfully integrates with external APIs, retrieves necessary data, and utilizes it within the app's functionalities.

Integration testing ensures that different parts of the food waste management app work together seamlessly, providing users with a cohesive and efficient experience while effectively managing food waste.

6.6. Performance testing

Performance testing for a food waste management app involves evaluating its responsiveness, scalability, and reliability under different scenarios. Here's an outline of performance testing scenarios for such an app:

1. Load Testing:

Scenario: Simulate typical usage patterns to determine how the app performs under normal load conditions.

Test Steps: Gradually increase the number of simultaneous users accessing the app and monitor response times, resource utilization, and throughput.

Metrics: Average response time, throughput, error rate, and server resource utilization.

2. Stress Testing:

Scenario: Determine the system's stability and robustness under extreme load conditions.

Test Steps: Subject the app to a load exceeding its capacity, either by increasing the number of concurrent users beyond normal levels or by generating excessive data traffic.

Metrics: System stability, maximum capacity, response time degradation, error rate under stress conditions.

3. Scalability Testing:

Scenario: Assess the app's ability to handle increased workload by adding more resources.

Test Steps: Increase the number of servers, database nodes, or other resources to observe how the app scales.

Metrics: Response time consistency, resource utilization as workload increases, ability to handle growing user base or data volume.

4. Endurance Testing:

Scenario: Evaluate the app's performance over an extended period to identify memory leaks, resource exhaustion, or degradation over time.

Test Steps: Maintain a consistent workload for an extended duration (e.g., 24 or 48 hours) and monitor system behavior.

Metrics: Memory and resource usage over time, response time stability, system crashes or slowdowns.

5. Concurrency Testing:

Scenario: Assess the app's ability to handle simultaneous user interactions and data updates.

Test Steps: Simulate multiple users accessing and updating data concurrently to identify any issues related to data integrity, locking, or contention.

Metrics: Response time under concurrent access, transaction throughput, database deadlock or contention.

6. Network Performance Testing:

Scenario: Evaluate how the app performs under different network conditions, such as low bandwidth or high latency.

Test Steps: Simulate various network conditions (e.g., 3G, 4G, Wi-Fi) and measure app response times and data transfer rates.

Metrics: Response time under different network conditions, data transfer rate, error rate due to network issues.

7. Database Performance Testing:

Scenario: Assess the app's performance related to database interactions, such as data retrieval, storage, and indexing.

Test Steps: Measure database read and write operations, query execution times, and index utilization under varying load levels.

Metrics: Database response time, query throughput, database locks and contention.

8. Third-Party Integration Testing:

Scenario: Evaluate the app's performance when interacting with external systems or APIs.
Test Steps: Monitor response times and reliability when making requests to third-party services for functionalities like geolocation, payment processing, or external data retrieval.
Metrics: Response time for third-party API calls, error rate, and reliability of integration.

By conducting performance testing with these scenarios, you can ensure that the food waste management app can handle the expected load, maintain responsiveness, and remain reliable under various conditions.

Chapter 7

Summary, Conclusion and Future Enhancements

7.1. Project Summary

Ehsaas App is a mobile application with a big heart, aiming to make a real difference in the fight against hunger. It's not just another app; it's a lifeline for those who have too much food and want to share it with those who don't have enough. In a world where food insecurity is a harsh reality for many, Ehsaas steps in to connect the dots, bringing together those with extra food and those in need.

Ehsaas App isn't just an app; it's a beacon of hope. In a world where many go to bed hungry, Ehsaas shines a light, showing that together, we can make a difference. With every meal shared, every connection made, Ehsaas brings us closer to a world where nobody has to go without.

Following are some highlighted aspects regarding the project;

- By rescuing surplus food, Ehsaas helps cut down on waste while filling empty stomachs.
- With Ehsaas, anyone can be a hero by sharing what they have with those who need it most, one meal at a time.
- Ehsaas isn't just about food; it's about building connections and showing kindness, making our communities stronger and more caring.
- With Ehsaas, giving and receiving food is easy, transparent, and gets the job done without any fuss.

7.2. Achievements and Improvements

So, here is the “Achievements and Improvements” of the project;

Achievements:

1. *Strengthened Food Donation Network*: The Ehsaas app effectively connected food donors with recipients, significantly reducing food wastage and addressing hunger through its user-friendly platform.

2. *Increased Donor Engagement*: By providing real-time donation updates, the app encouraged more frequent contributions, extending its reach in combating food insecurity.

3. *Improved Recipient Access*: Ehsaas enabled easy access to nearby food donations, ensuring timely meals for those in need and promoting inclusivity.

4. *Transparent Communication*: The app facilitated transparent communication between donors and recipients, fostering trust and accountability in the donation process.

5. *Community Impact*: Ehsaas sparked a culture of compassion and solidarity, addressing immediate food needs and inspiring broader initiatives to combat hunger.

Improvements:

1. *Enhanced User Experience*: Future updates could focus on simplifying navigation and enhancing design for a smoother donation process.

2. *Expanded Outreach*: Multilingual support and accessibility features would broaden the app's audience, along with targeted marketing campaigns and partnerships with local organizations.

3. *Feedback Integration*: Incorporating user feedback mechanisms would allow for continuous improvement and responsiveness to community needs.

4. *Data-Driven Optimization*: Analyzing donation trends and user behavior could inform more effective resource allocation and decision-making.

4. ***Sustainability Initiatives*:** Partnering with food rescue organizations and implementing eco-friendly practices would promote sustainability along the donation supply chain.

This revised version maintains the essence of the achievements and improvements while using alternative wording for clarity and variation.

6.1. Critical Review

The Ehsaas App emerges as a promising tool for connecting food donors with those in need, offering a platform for efficient communication and donation management. This critical assessment aims to evaluate the app's functionality, user experience, and effectiveness in facilitating food donations.

Functionality;

The app's functionality serves as its cornerstone, providing users with a seamless platform to donate surplus food and connect with recipients. Its user-friendly interface simplifies the process of uploading surplus food details and scheduling pickups. Recipients can easily browse available donations and request those that meet their needs, fostering direct and efficient exchanges.

User Experience;

The app's user-friendly interface stands out, making the donation process straightforward for both donors and recipients. The registration process is simple, and the layout is intuitive, enhancing overall usability. However, there may be potential accessibility challenges for users with limited technological proficiency or internet access.

Communication;

Effective communication is essential for facilitating food donations, and the Ehsaas App excels in this regard. Its built-in messaging system enables direct communication between donors and recipients, facilitating coordination and addressing concerns in real-time. Real-time notifications further enhance communication, ensuring timely responses and building trust among users.

Effectiveness;

The app shows significant potential in streamlining food donations and reducing waste. By connecting donors with recipients in real-time, it maximizes surplus food utilization while addressing immediate hunger needs. However, its success hinges on widespread adoption and active participation from both donors and recipients.

6.2. Lessons Learnt

Developing the Ehsaas app has taught us a profound lesson about the importance of human interaction in addressing food insecurity. In today's increasingly isolated world, our app has shown us the immense impact that direct communication between donors and recipients can have. Through this platform, we've come to understand that even in the digital age, the simple act of connecting with one another can build a sense of community and make a real difference for those in need. It's a powerful reminder that technology, when used with empathy and purpose, can be a vital tool in fostering connections and tackling social issues.

6.3. Future Enhancements/Recommendations

As we look ahead to potential enhancements for the Ehsaas app, there are several avenues we could explore to further improve its functionality and impact in facilitating food donations and fostering communication between donors and recipients i.e :

1. Enhanced Matching Algorithm: Introduce machine learning algorithms to enhance the matching process between donors and recipients. By analyzing user preferences, past donation patterns, and real-time demand, the app can suggest more relevant matches, ensuring efficient and timely food distribution.

2. Geolocation Integration: Incorporate geolocation services to provide donors and recipients with real-time information about nearby donation centers, food banks, and individuals in need. This feature can streamline the donation process and encourage local community engagement.

3. Feedback Mechanism: Implement a feedback mechanism to gather insights from both donors and recipients regarding their donation experiences. This feedback loop can help identify areas for improvement, address concerns, and enhance overall user satisfaction.

4. Volunteer Management System: Introduce a volunteer management system within the app to recruit, organize, and coordinate volunteers for various tasks such as food collection, distribution, and outreach efforts. This feature can amplify the app's impact by mobilizing community support and resources.

5. Multi-platform Accessibility: Extend the app's accessibility by developing versions compatible with various platforms, including web browsers, tablets, and wearable devices. This expansion can broaden the app's user base and reach individuals who may not have access to smartphones.

6. Social Media Integration: Integrate social media functionalities to enable users to share their donation activities, milestones, and success stories with their networks. By leveraging social media platforms, the app can raise awareness, inspire others to join the cause, and create a sense of community around food donation efforts.

7. Language Localization: Offer support for multiple languages to cater to a diverse user base. By providing localized content and interfaces, the app can ensure inclusivity and accessibility for users from different linguistic backgrounds.

By incorporating these future enhancements and recommendations, the Ehsaas app can continue to evolve as a transformative tool for facilitating food donations, fostering meaningful connections between donors and recipients, and making a tangible difference in the fight against hunger and food wastage.

Appendices

Appendix (A): Information / Promotional Material

A.1. Broacher

A.2. Flyer


FOOD
DONATION
Help to donate to needy
Your excessive food
Could be a life of someone
FOOD
DRIVE

A.3. Standee


SUPERIOR UNIVERSITY
Ehsass App
FOOD DONATION APP
Problem Statement:
• Inefficiencies in the food supply chain.
• Reducing food waste from production to consumption.
• Implementing recycling or repurposing strategies for food waste.
Solution:
Holistic Approach
▪ Education
▪ Composting
▪ Recycling
▪ Donation
• Stakeholder Collaboration
• Smart Technology
▪ Waste tracking
▪ Mobile Application
Supervised By
Mam Sabha Arif
FYP ID : FYP-BCSM-F20-089
Team
M. Nouman Ali
Saif-u-Rehman
Zeeshan Ahmad
Our Services
Food Donation Matching
Real-Time Inventory Management.
Community Engagement.
Co-relation with NGOs.
Dashboard
Donate Food
Receive
Food Map
My Pins
History

A.4. Banner

A.5. First Level heading
‘’Food Waste Management System’’

A.5.1. Second level heading
Food Waste Management System Project

- User Classes and Characteristics
- Functional Requirements
- Non-Functional Requirements
- System Architecture
- Database Design
- User Interface Design
- Performance Testing
- Security Considerations
- Integration with External Systems
- Data Analysis and Reporting
- Maintenance and Support
- Deployment Plan
- Future Enhancements

A.1.1.1 Third level heading

3. Food Waste Management System Project

- 3.1 User Classes and Characteristics
- 3.2 Equivalence Partitioning for Testing
- 3.3 Performance Testing Scenarios
- 3.4 Functional Requirements
- 3.5 Non-Functional Requirements
- 3.6 System Architecture
- 3.7 Database Design
- 3.8 User Interface Design
- 3.9 Implementation Plan
- 3.10 Testing Strategy
- 3.11 Deployment Plan
- 3.12 Maintenance and Support
- 3.13 Future Enhancements

Appendix: Food Waste Management System

Appendix: Appendix A: User Class Details

1. Individual Consumers

- ✓ Characteristics: Personal profiles, dietary preferences, food inventory logging.
- ✓ Key Functions: Log purchases, track consumption, monitor waste, receive alerts, generate reports.

2. Households

- ✓ Characteristics: Multiple users, collective inventory, coordinated food management.
- ✓ Key Functions: Log purchases, track consumption, monitor collective waste, receive household alerts, generate reports.

3. Restaurants and Food Businesses

- ✓ Characteristics: Menu items, inventory management, surplus food donations.
- ✓ Key Functions: Manage inventory, track waste, donate surplus, receive alerts, generate detailed reports.

4. Food Banks and Charitable Organizations

- ✓ Characteristics: Capacity for donations, coordination of collections and distributions.
- ✓ Key Functions: Receive and distribute food donations, generate impact reports.

5. Municipal Waste Management Authorities

- ✓ Characteristics: Waste collection schedules, community waste data.

- ✓ Key Functions: Monitor waste collection, coordinate recycling, send community updates, generate reports.
- ✓

6. Administrators and App Developers

- ✓ Characteristics: System settings, user account management, app performance monitoring.
- ✓ Key Functions: Manage accounts, ensure data security, maintain app performance, generate usage reports.

7. Educators and Advocates

- ✓ Characteristics: Educational content, campaigns, workshops.
- ✓ Key Functions: Create content, organize campaigns, engage users, generate impact reports.

Appendix B: Equivalence Partitioning Examples

1. User Registration and Profile Management

- ✓ Valid and invalid registration data
- ✓ Profile updates

2. Food Inventory Management

- ✓ Logging valid and invalid food purchases
- ✓ Tracking consumption
- ✓ Monitoring waste

3. Waste Monitoring and Alerts

- ✓ Logging valid and invalid waste
- ✓ Generating expiry alerts

4. Donations Management

- ✓ Valid and invalid donations

5. Reports Generation

- ✓ Valid data and no data scenarios

6. Notifications and Alerts

- ✓ Valid and invalid notification setups

7. Integration with Municipal Waste Management

- ✓ Valid and invalid collection data

Appendix C: Performance Testing Scenarios

1. Load Testing

- ✓ Typical usage patterns, response times, resource utilization

2. Stress Testing

- ✓ Extreme load conditions, system stability, maximum capacity

3. Scalability Testing

- ✓ Increasing resources, response time consistency

4. Endurance Testing

- ✓ Extended usage, memory and resource stability

5. Concurrency Testing

- ✓ Simultaneous user interactions, data integrity

6. Network Performance Testing

- ✓ Various network conditions, response times

7. Database Performance Testing

- ✓ Database interactions, query execution times

8. Third-Party Integration Testing

- ✓ External system interactions, API response times

Appendix D: Functional and Non-Functional Requirements

1. Functional Requirements

- ✓ User authentication
- ✓ Food inventory management
- ✓ Waste logging and monitoring
- ✓ Donation handling
- ✓ Report generation

2. Non-Functional Requirements

- ✓ Performance and scalability
- ✓ Security and data privacy
- ✓ Usability and user experience
- ✓ Integration with external systems

Appendix E: System Architecture and Design

1. System Architecture Diagram

- ✓ Overview of system components and interactions

2. Database Design

- ✓ Schema diagrams, table definitions

3. User Interface Design

- ✓ Wireframes, user flow diagrams

Appendix F: Implementation Plan

1. Development Phases

- ✓ Milestones, deliverables, timelines

2. Testing Strategy

- ✓ Unit testing, integration testing, user acceptance testing

Appendix G: Deployment and Maintenance

1. Deployment Plan

- ✓ Steps for deploying the app, rollback procedures

2. Maintenance and Support

- ✓ Maintenance schedules, support procedures

Appendix H: Future Enhancements

1. Planned Features

- ✓ Roadmap for additional functionalities
- ✓ User feedback integration

Reference and Bibliography

Reference and Bibliography

Websites:

1) https://www.javatpoint.com/android-tutorial
2) https://www.tutorialspoint.com/android/android_studio.htm
3) https://www.youtube.com

Index

INDEX

Chapter 1.docx

ORIGINALITY REPORT

8%	5%	1%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Higher Education Commission Pakistan Student Paper	1%
2	ijsrset.com Internet Source	1%
3	Submitted to University of Wales Institute, Cardiff Student Paper	1%
4	www.ijraset.com Internet Source	<1%
5	graffersid.com Internet Source	<1%
6	fastercapital.com Internet Source	<1%
7	lawire.com Internet Source	<1%
8	www.coursehero.com Internet Source	<1%
9	devtechnosys.com	