Software Requirement Specification Document:

**ShareBox**

**Prepared by: Nini Ola**

# **TABLE OF CONTENTS**

1. **Introduction........................................................................................................................1**

1.0 Overview……………………………………………………………………………....1

1.1 Purpose.......................................................................................................................... 1

1.2 Scope ............................................................................................................................ 1

1.3 Definitions, acronyms, and abbreviations .....................................................................1

1.4 References .................................................................................................................... 2

1.6 Overview………………………………………………………………………………3

1. **General Factors ............................................................................................................... 4**

2.1 Product perspective ...................................................................................................... 4

2.2 Product functions ......................................................................................................... 4

2.3 User characteristics ...................................................................................................... 5

2.4 General Constraints ...................................................................................................... 5

2.5 Assumptions and dependencies ................................................................................... 5

1. **Requirements Section....................................................................................................... 7**

3.1 INTERFACE REQUIREMENTS…….........................................................................5

3.1.1 User Interfaces ..............................................................................................5

3.1.2 Hardware Interfaces.................................................................................. ...25

3.1.3 External Software Interfaces ........................................................................25

3.1.4 Communications Interfaces..........................................................................25

3.2 FUNCTIONAL REQUIREMENTS ...........................................................................26

3.3 NON-FUNCTIONAL REQUIREMENTS .................................................................35

1. **Logical Database Requirements………………………………………………………..**
2. **Data Flow Diagram……………………………………………………………………..**
3. **Introduction**

This section gives a scope description and overview of everything included in this SRS document. Also,

The purpose for this document is described and a list of abbreviations and definitions is provided.

**1.1 Purpose**

The purpose of this Software Requirements Specification (SRS) document is to communicate the details of the features of the ShareBox. This document will explain Features and provide a visual of the User Interface (UI). The document will also detail the necessary constraints for developing the software. Additionally, it will cover hardware, software and communication interfaces as researched by the student

**1.2 Scope**

The ShareBox system is composed of an Electronic Lock Mechanism (ELM), Quality Assurance Machine Learning Algorithm (QAMLA), Firebase Database (FD) and finally the User Interface Web Extension (UIWE). The UIWE software is designed to be an addition to Government issued Community websites, in order to foster sharing and promote sustainability. The UIWE, when launched, will give users the option to be a Donor User (DU) or a Receiver User (RU). all RUs’ and DUs’ activities while using the UIWE platform will be tracked and stored in the FD. A DU’s donation undergoes a check process using the QAMLA. The QAMLA is an Object classification and Label classification algorithm responsible for assessing quality. Once an item is Accepted by the QAMLA, the UIWE provides the DU with a Location, Box No# and Box Code (LBBs), which the DU uses to donate the item in the ShareBox. Alternatively, RUs have the option to claim these items on the UIWE when qualified. RUs and DUs use the ELM to take/place items in the box, the ELM ensures safety and fairness, it must accept the correct LBBs info from UIWE to open the share box. Lastly, RUs and DUs have the option to view their activities to potentially check qualifications to receive more or track donations for tax benefits.

**1.3 Definitions, acronyms, and abbreviations**

| Word | Abbreviations |
| --- | --- |
| Electronic lock mechanism: Lock Box | ELM |
| Quality Assurance Machine Learning Algorithm | QAML |
| User Interface Web Extension | UIWE |
| Donor Users | DUs |
| Performance dashboard | PD |

**1.5 Overview**

The Software Requirements Specification will begin by providing a description of general factors, including the project perspective, product functions, user characteristics, and general constraints. In addition, it will communicate the different interfaces, software requirements, and constraints. The document will then move forward to the specifics of functional and non-functional requirements. Finally, it will present data flow diagrams to explain the internal flows throughout the system.

1. **Overall Description**

This section provides an overview of the ShareBox system, outlining its role and interactions with other systems. It will introduce the system’s core functionalities and describe the different users and explain the features available to each user type. Lastly, this section will present the key constraints and assumptions that shape the system’s design and operation.

**2.1 Product perspective**

The ShareBox system allows Two types of User interactions: Donation and Collection. Upon launching the app by clicking the UIWE icon, a User is required to create an account as either a DU or RU. The DUs use the app for making Donations, the UIWE will allow the DUs to make a donation by: uploading a picture or video of the item, filling out the item checklist, launching the QAMLA and lastly using their correct LBBs to drop off items. Alternatively the RUs use the app to collect donations and track their collections. Both RU and DU information will be stored in the FD, all Users will have access to their tracked activity and a dashboard; for RUs to track qualifications for next receive, and for DUs to track Accepted donations and potential tax deductions. ShareBox app will make use of Wifi for connectivity and communication, as well as a GPS locator to place Users in the appropriate radius using their address, the GPS locator will also help in locating the share box. Both RUs and DUs will make use of the ELM to open the sharebox, before a DU or RU can open a box they must enter the correct LBBs. The ELM requires a firebase realtime database (FRD), ESP32 microcontroller (ESPM) and Wifi to function correctly.

**2.2 Product functions**

With the UIWE, Users can create an account and share resources. DUs can find the nearest ShareBox hubs, this will be done based on their user inputs. DUs can then make donations by using the donation form, the donation form collects appropriate information on the item, origin/prep and user. The form uses a picture selector, list view and QAMLA to review a DU’s donation. The QAMLA will function as the main product reviewer, it will decide whether or not to accept the donation, request more info or reject the donation. The UIWE will then display the Accepted items to the Appropriate location homepage which RUs can then claim. The RUs will use their location input to find the appropriate ShareBox hub and can claim any box while available using the “first come first serve” policy. RUs can track their claims to ensure fairness, RUs can also highlight any allergies or donation preferences. Both Users have a performance dashboard to track activities, a profile and a help center to report any issues or suggestions. Finally to open the box both users make use of the ELM, which is responsible for box-User interaction maintenance.

**2.3 User characteristics**

There are three types of users that interact with the system, these users utilize the system differently and have different features and requirements.

* DUs: the DUs make use of the location feature to find closest boxes, submit form features for donations, launch the QAMLA for quality, obtain the LBBs to access the box and use a dashboard to track expenses and tax benefits.
* RUs: the RUs make use of location to find the available boxes in a 50 mile radius, personalized experience using the allergy and preference form, claim available donations,obtain the LBBs to open the correct box and a dashboard to track user qualifications for donations.
* The administrators: this user has access to all aspects of the database

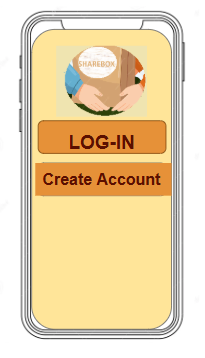
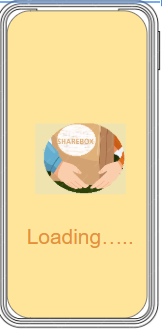
**2.4 Constraints**

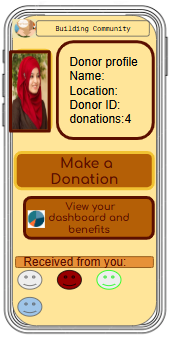
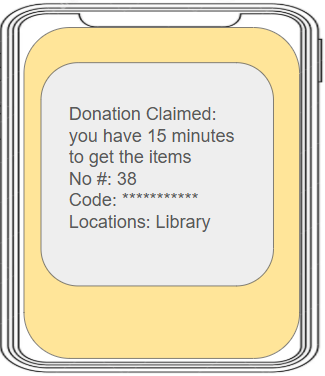
Internet connection is required for the DUs and RUs to receive real time updates of the App. This is due to the application retrieving data from the database over the Internet. The size of the Database matters Both as the application will be constrained by the size database and cost of a CRUD operation.

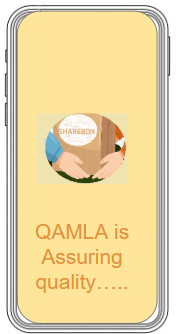
1. **Requirements Section**

**3.1.1 User interfaces**

The user first begins by waiting for the UIWE to load on the welcome screen (figure 1), they are then prompted to the next screen where they can either create an account or legion. If a user needs to create an account they are navigated to the create account screen then to the homepage , while the users who login are navigated to the homepage directly. The homepage differs for RUs and DUs, on the DU homepage they can donate and track benefits. Alternatively the RUs can claim and track claims.

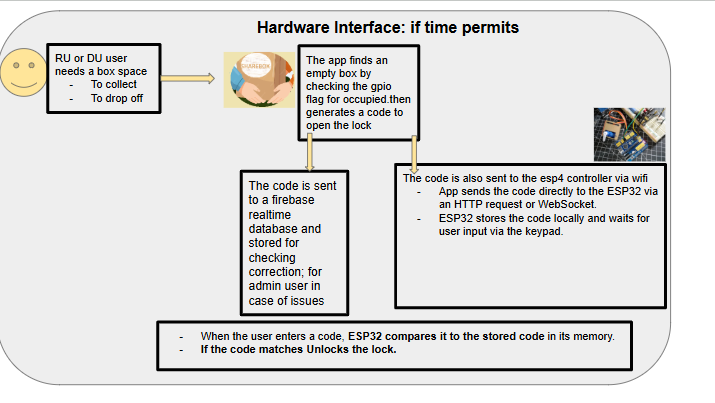




**3.1.2 Hardware interfaces**

| hardware | Hardware Interface | functions |
| --- | --- | --- |
| Power supply | Volts | Powers the lock |
| ESP32 Microcontroller | Wi-Fi, GPIO, I2C, UART | Manages logic, communicates with Firebase, and controls the lock |
| Keypad | GPIO Input | Unlocks when the correct code is entered |
| Solenoid Lock | GPIO Output | Provides visual feedback for lock status |
| Led indicator | GPIO Output | Sounds alert on failed attempts or successful unlocks |



**3.1.3 Software interfaces**

| software | interface | function |
| --- | --- | --- |
| Software Extension | Communicates with firebase and the esp4 controller program waiting for a pin. Uses gps to determine appropriate locations | Users can use the app |
| Technology used | React native, firebase, |  |
| Esp32 firmware | Retrieves code from firebase or ram to check correctness | Logs unlock attempts in Firebase |
| firebase | Store user data | Holds the unlock code and logs access attempts (success/failure)  Stores success and failure attempts with timestamps for security tracking |
| Communication interface | WIFI is required to use the app, http for communicating with the controller |  |

**3.2 Functional requirements**

| ID: FR1 | Description: Show a brief welcoming message to inform users that the system is loading (e.g., "Welcome! Loading..."). |
| --- | --- |
| Title: Welcome Page | Processing: Display a loading animation and transition to the main screen when the system is ready. |
| Input: None (automatic on app launch). | output:A visual welcome message and loading animation. |
| Dependency: none | Error handling: If loading fails, display a retry button or error message. |
| Priority high |  |

| ID: FR 2 | Description:A login screen that allows users to authenticate before accessing the system. It includes input fields for username/email and password, along with a login button. |
| --- | --- |
| Title: Login | Rationale: Ensures **secure access** to the system by preventing unauthorized users from proceeding. It acts as the **first layer of security** before interacting with protected features. |
| Processing : A user opens the app and is presented with a login screen. They enter their credentials, and upon successful authentication, they are granted access to the system. If the login fails, an error message is displayed. | Priority high  Error handling: Show appropriate error messages for incorrect credentials or network failure. |
| Dependency: - User authentication system (e.g., Firebase Auth, custom backend API)  - Internet connection (if authentication is remote)  - UI components (text fields, buttons) | Input: Email/Phone Number, Password.  Output: User is logged in and redirected to their respective homepage (Donor or Receiver). |

| ID: FR 3 | Description:A registration screen where new users can create an account by entering required details such as name, email, password, and optionally, phone number. A "Create Account" button submits the form |
| --- | --- |
| Title: Create Account | Rationale: Allows new users to register and gain access to the system, enabling authentication and secure identification. Ensures only authorized users can interact with protected features. |
| Processing : A user opens the app and navigates to the "Create Account" screen. They enter their details and submit the form. If all inputs are valid, the account is created, and they are redirected to the login screen or logged in automatically. If there’s an error (e.g., email already in use, weak password), an error message is displayed. | Priority high  Error Handling Display error messages for invalid input (e.g., weak password, existing email). |
| Dependency:  - User authentication system (e.g., Firebase Auth, custom backend API)  - Database to store user details  - UI components (input fields, buttons)  - Internet connection (if using a remote backend) | Input Name, Email, Phone Number, Password, User Type (Donor/Receiver).  Output User account is created and stored in the database. User is redirected to the homepage. |

| ID: FR 4 | Description:Allows users to upload images for verification (e.g., ID verification, donation items) |
| --- | --- |
| Title: Photo/Media picker | Rationale: Ensures donors can provide visual proof of their items and users can verify identity. |
| Processing : A user selects an image from their gallery or takes a photo using their phone’s camera to upload a document or item picture. | Priority high  Error Handling: Show error message if upload fails or the file type is unsupported. |
| Dependency: - File upload system (Firebase Storage, AWS S3)  - Camera/gallery access (Mobile SDK) | Input Image or video file (selected from device or captured with camera).  Output Uploaded media is stored and linked to the donation entry in the database. |

| ID: FR 5 | Description:A verification system that requires users to submit a form of identity (e.g., driver's license, library card, phone number, or school ID) to ensure legitimacy before accessing donor/receiver functionalities. |
| --- | --- |
| title: User Identity authentication | Rationale: Prevents fraud, ensures accountability, and builds trust within the system. Protects donors from fake receivers and ensures resources go to legitimate users. |
| Processing : A user registers or logs in and is prompted to verify their identity. They upload a government-issued or institutional ID (e.g., driver's license, school card), enter their phone number, and possibly receive an OTP for additional verification. Once verified, they gain full access to the platform. If verification fails, they are notified and given options to retry. | Priority high  Error Handling: Notify users of failed verification and allow reattempts. |
| Dependency: - Secure file upload system (for document verification)  - Database to store user verification status  - OCR (Optical Character Recognition) or manual verification process  - SMS API (e.g., Twilio) for phone number verification  - User authentication system (ensuring only verified users access key features) | Input Image of ID, Personal Details.  Output Verified status is stored in the user profile. |

| ID: FR 6 | Description:Captures and displays user’s location for donation pickup or delivery. |
| --- | --- |
| title: Location/gps | Rationale: Helps match donors and receivers efficiently based on proximity |
| Processing : A donor sets their location when submitting an item, or a receiver sees available donations nearby. | Priority high  Error Handling If location access is denied, prompt the user to enable it manually. |
| Dependency: GPS/Location API (Google Maps, OpenStreetMap)  - Permissions for location access | Input GPS coordinates./address  Output Nearby ShareBox locations are displayed on a map. |

| ID: FR7 | Description Uses AI to detect and verify donated items for quality control. |
| --- | --- |
| Title: QAMLA | Processing User submits item photo → AI scans the image → System confirms item category and checks for quality standards. |
| Input: Image of the item. | output:Verified item classification and quality approval. |
| Dependency:Machine Learning Model, Firebase Storage. | Error handling: If detection fails, prompt the user to retake the photo. |
| Priority high |  |

| ID: FR8 | Description: Allows users to view and edit their profile details (name, email, donations, etc.). |
| --- | --- |
| Title: View Profile | Processing: User clicks on profile → Profile details are fetched from Firebase → Display profile information. |
| Input: User details (name, email, donations, etc.) | output:Displayed profile information (name, donations made, etc.). Option to edit. |
| Dependency: Firebase Database, Firebase Authentication. | Error handlingIf data fails to load, display an error message and retry button. |
| Priority high |  |

| ID: FR9 | Description:Displays the user’s donation and receiving statistics (e.g., number of items donated, items claimed, total donation value). |
| --- | --- |
| Title: Performance Dashboard | Processing: User accesses dashboard → The app fetches user’s donation/receiving stats from the database → Display stats and performance insights. |
| Input: User's donation and receiving activity data from Firebase. | output:A graphical representation (charts/graphs) of donations, claims, etc. |
| Dependency: Firebase Database, Charting/Graphing Library (e.g., Chart.js). | Error handling: If data fetching fails, display a loading error message or empty state with retry option. |
| Priority high |  |

| ID: FR10 | Description: Show a brief welcoming message to inform users that the system is loading (e.g., "Welcome! Loading..."). |
| --- | --- |
| Title: Help/Suggestion Box | Processing: User selects feedback option → User writes and submits feedback → Store feedback in Firebase or send to admin email. |
| Input: Feedback or issue description. | output:Acknowledgement message confirming feedback submission |
| Dependency: Firebase Database, Email Service (for admin notifications). | Error handling:Display error message if feedback submission fails. |
| Priority high |  |

| ID: FR 11 | Description:Allows a receiver to request a donation. |
| --- | --- |
| title: Claim item | Rationale: Prevents multiple claims for the same item and ensures fair distribution. |
| Processing User browses available donations → Selects an item to claim → Updates the donation status in Firebase. | Priority high  Error Handling: If claiming fails (e.g., already claimed), show error message. |
| Dependency: - Firebase Database, Item Listing. | Input: Item selection by the user.  Output: Item claim confirmation and update of status in the database. |

| ID: FR 12 | Description:A secure mechanism that controls access to the donation box, unlocking when the correct code is entered. |
| --- | --- |
| title: lock system | Rationale: Ensures only authorized users can open the donation box, preventing unauthorized access. |
| Processing : A user enters a code via keypad/Bluetooth, and the system verifies it. If correct, the solenoid lock disengages for a set duration before re-locking. | Priority high  Error Handling:If the code is incorrect, display an error message and allow re-entry. |
| Dependency: - Microcontroller (ESP32)  - Relay/MOSFET for solenoid control  - Keypad/Bluetooth for input  - Power supply (Battery or Adapter) | Input:6-digit access code.  Output:Box unlocks upon successful code verification. |

| ID: FR 13 | Description:Allows donors to list an item for donation. |
| --- | --- |
| title: Submit a item | Rationale: Enables structured donation submissions with clear item descriptions and locations. |
| Processing : A donor fills out a form, uploads an image, sets pickup details, and submits it. | Priority high  Error Handling:Notify users if any required field is missing or if submission fails. |
| Dependency: - Database to store item details  - File upload system for images  - GPS for location tagging | Input:Item Name, Description, Category, Image, Location.  Output :Item entry is created and stored. |

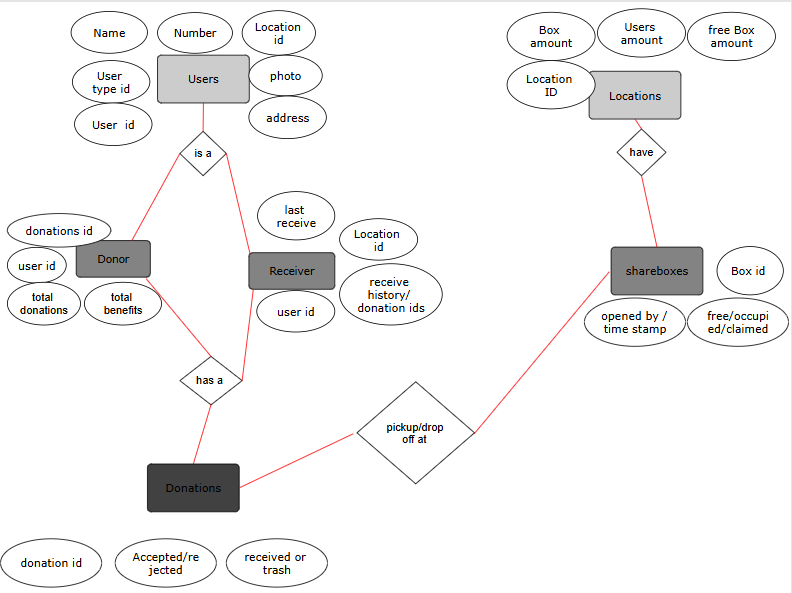
**3.3 Non-Functional requirements**

| ID: NFR1 | Description: The Donor's homepage allows users to browse submit donations, claim benefits, and track activity details |
| --- | --- |
| Title: Donor Home Page | Rationale: Provides a seamless experience for donors to manage and track their donations efficiently, ensuring transparency and engagement in the donation process. |
| Processing : A donor logs in and is directed to their homepage. They see an overview of their past donations, an option to add a new donation (e.g., food, clothes), and updates on donations that have been picked up or are pending. Notifications may alert them when a receiver claims their donation. | Priority high |
| Dependency: User authentication system (ensures only authorized donors access the page)  - Database for storing donation details  - UI components (cards, buttons, notifications)  - Map integration (optional, if showing pickup locations) |  |

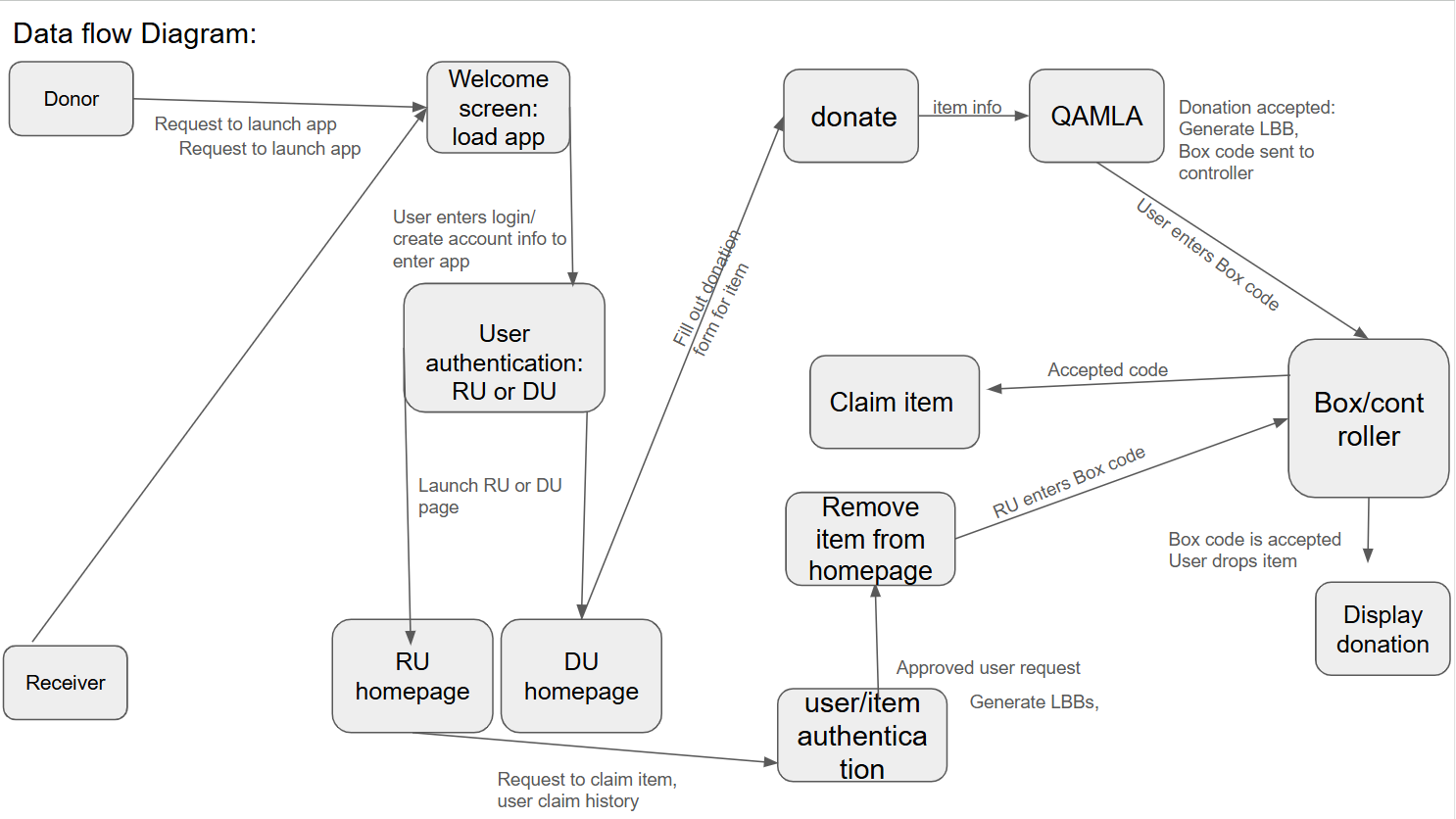
| ID: NFR 5 | Description: The receiver's homepage allows users in need to browse available donations, claim items, and track pickup details |
| --- | --- |
| Title: Receiver homepage | Rationale: Ensures a smooth and organized way for receivers to access and claim donations, making the process more efficient and reducing waste. |
| Processing : A receiver logs in and is directed to their homepage. They see a list of available donations, filter options based on category or location, and an option to claim an item. Once they claim an item, they receive pickup instructions and status updates | Priority high |
| Dependency: - User authentication system (ensures only authorized receivers access the page)  - Database for listing available donations  - UI components (item cards, filter/search bar, claim button)  - Notification system (alerts when a donation is successfully claimed) |  |

| ID: FR 9 | Description:Displays the terms and conditions to users and requires them to accept the agreement before proceeding with account creation or using the app. |
| --- | --- |
| Title: user agreement |  |
| Processing :User opens the app → User is presented with the terms and conditions → User must accept the agreement to proceed. | Priority high |

1. **Logical Database Requirements: Entity Relationship Diagram**

****

1. **Analysis Models:**Data Flow Diagrams (DFD): Represent the flow of data within processes or the entire system.

****