Software Requirements Specification for Food Donations and Redistribution System

Taha Shah , Aimal Amir , Jaleel Abbas

***Abstract*—This document outlines the requirements for the development of a food donations and redistribution system web application. It includes the problem statement, methodology, application architecture, tools & techniques, GUI design, functional and non-functional requirements, testing procedures, and any outsource libraries used.**

I. Introduction

A. Purpose

The purpose of this document is to provide a comprehensive overview of the software requirements for the food donations and redistribution system web application. It aims to define the features and functionalities of the system, as well as the technical specifications and constraints.

B. Scope

The scope of this project encompasses the development of a web-based platform that facilitates the donation, management, and redistribution of food items. The system will include features for users to donate food, administrators to manage donations and orders, and delivery personnel to track and fulfill orders.

C. Audience

The primary audience for this document includes software developers, project managers, and stakeholders involved in the development and deployment of the food donations and redistribution system. Additionally, it serves as a reference for quality assurance teams responsible for testing the application.

II. Problem Statement

The current methods for food donation and distribution lack efficiency and organization, leading to challenges such as food waste, logistical inefficiencies, and difficulty in tracking donations. There is a need for a centralized platform that streamlines the process of donating food, managing donations, and coordinating deliveries.

III. Methodology/Workflow

The development process will follow an Agile methodology, allowing for iterative development and continuous improvement based on user feedback. The workflow will consist of the following phases:

A. Requirement Analysis

Gather requirements from stakeholders and end-users through interviews, surveys, and workshops. Define user stories, use cases, and acceptance criteria to capture the functional and non-functional requirements of the system.

B. Design

Design the architecture of the system, including the frontend and backend components, database schema, and APIs. Create wireframes and mockups to visualize the user interface and user experience.

C. Implementation

Develop the frontend using HTML, CSS, and JavaScript, following responsive design principles for cross-device compatibility. Implement the backend logic using PHP and design the database schema using MySQL. Integrate third-party APIs for features such as geolocation and payment processing.

D. Testing

Conduct unit tests to verify the functionality of individual components and modules. Perform integration tests to ensure seamless communication between frontend and backend systems. Conduct acceptance tests with end-users to validate that the system meets their requirements and expectations.

E. Deployment

Deploy the application to a web server using a continuous integration and continuous deployment (CI/CD) pipeline. Monitor the performance and usage of the system in production and address any issues or bugs that arise.

IV. Application Architecture

The food donations and redistribution system will follow a client-server architecture, with the frontend and backend components interacting through RESTful APIs. The architecture will consist of the following layers:

A. Presentation Layer

The presentation layer will comprise the user interface components, including web pages, forms, and interactive elements. It will be developed using HTML, CSS, and JavaScript frameworks such as React.js or Vue.js.

B. Application Layer

The application layer will contain the business logic and processing functionality of the system. It will be implemented using PHP, which will handle user authentication, donation management, order processing, and other core features.

C. Data Access Layer

The data access layer will interface with the database to perform CRUD (Create, Read, Update, Delete) operations on data entities. It will use SQL queries to interact with the MySQL database and retrieve or manipulate data as required.

D. Database Layer

The database layer will store the persistent data of the system, including user profiles, donation records, order details, and administrative settings. It will utilize a MySQL database management system to ensure data integrity, consistency, and reliability.

V. Tools & Techniques

The development of the food donations and redistribution system will leverage a variety of tools and techniques to ensure efficiency, collaboration, and quality:

A. Version Control

Git will be used for version control, allowing multiple developers to collaborate on the same codebase, track changes, and manage code revisions effectively.

B. Integrated Development Environment (IDE)

Visual Studio Code will serve as the primary IDE for development, providing features such as syntax highlighting, code completion, and debugging capabilities.

C. Testing Framework

PHPUnit will be used as the testing framework for conducting unit tests, integration tests, and regression tests to ensure the reliability and stability of the application.

D. Continuous Integration/Continuous Deployment (CI/CD)

A CI/CD pipeline will be set up using tools such as Jenkins or GitLab CI/CD to automate the build, test, and deployment processes, enabling rapid and consistent delivery of updates to the production environment.

E. Wireframing & Mockup Tools

Tools such as Adobe XD or Sketch will be used to create wireframes and mockups of the user interface, allowing designers and developers to visualize the layout and interactions of the application before implementation.

VI. GUI Design

The graphical user interface (GUI) of the food donations and redistribution system will be designed with a focus on usability, accessibility, and aesthetics. The design principles will include:

A. User-Centric Design

The GUI will prioritize the needs and preferences of the end-users, providing intuitive navigation, clear instructions, and responsive feedback to enhance the user experience.

B. Consistent Branding

The GUI will adhere to a consistent visual identity, including colors, typography, and graphical elements, to reinforce the brand image and create a cohesive user interface across all pages.

C. Responsive Layout

The GUI will be designed to adapt seamlessly to different screen sizes and devices, including desktops, laptops, tablets, and smartphones, using responsive design techniques such as fluid grids and media queries.

D. Accessibility Compliance

The GUI will comply with accessibility standards such as WCAG (Web Content Accessibility Guidelines), ensuring that users with disabilities can access and interact with the application using assistive technologies.

E. Interactive Elements

The GUI will feature interactive elements such as buttons, forms, dropdown menus, and sliders to enable users to navigate, input data, and perform actions with ease and efficiency.

VII. GUI Screenshots

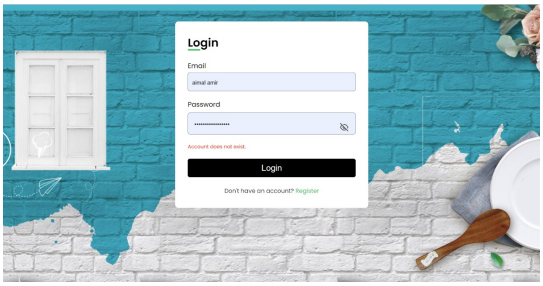

Fig. 1. GUI 1

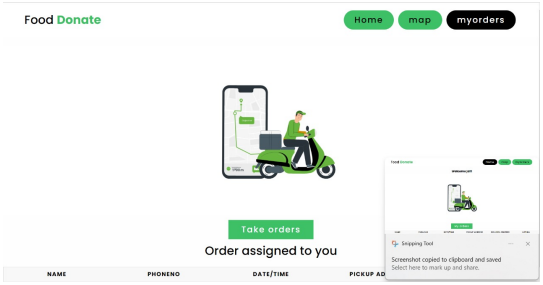

Fig. 2. GUI 2

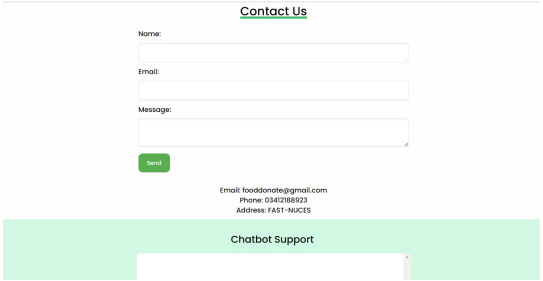

Fig. 3. Delivery Interface: Orders

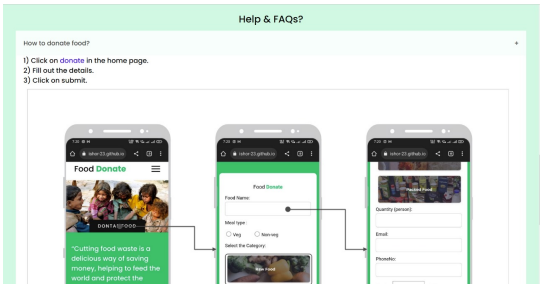

Fig. 4. GUI 4

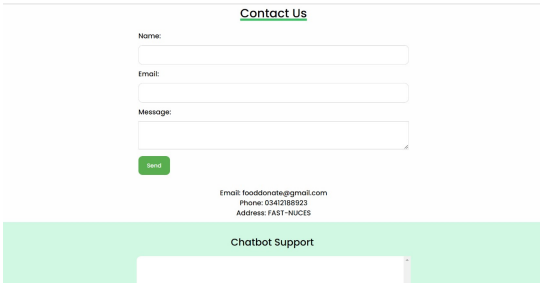

Fig. 5. GUI 5

VIII. Functional Requirements

The functional requirements of the food donations and redistribution system are categorized based on the user roles and system modules:

A. User Management

1) User Registration: Users should be able to create an account by providing basic information such as name, email address, and password. The registration process should include email verification to ensure the authenticity of user accounts.

2) User Login: Registered users should be able to log in to the system using their email address and password. The login process should authenticate user credentials and grant access to user-specific features and functionalities.

3) User Profile Management: Users should be able to update their profile information, including personal details, contact information, and preferences. They should also have the option to upload a profile picture and manage their notification settings.

B. Donation Management

1) Donation Submission: Users should be able to submit donation requests by providing details such as the type of food items, quantity, expiry date, and pickup location. They should also have the option to upload images or documents related to the donation.

2) Donation Approval: Administrators should be able to review and approve donation requests based on criteria such as availability, suitability, and compliance with guidelines. They should have the authority to accept or reject donations and communicate with donors if additional information is required.

3) Donation Tracking: Users should be able to track the status of their donation requests, including pending, approved, or rejected. They should receive notifications via email or SMS at each stage of the donation process, from submission to pickup.

C. Order Management

1) Order Placement: Users should be able to place orders for food items available for donation. They should be able to browse through available donations, select items of interest, and add them to their cart for checkout.

2) Order Fulfillment: Delivery personnel should be notified of new orders placed by users and assigned to fulfill them based on their availability and proximity to the delivery location. They should update the status of orders as they are dispatched, en route, and delivered.

3) Order History: Users should have access to their order history, including details such as order date, items ordered, delivery address, and status. They should be able to view and track the progress of past orders and provide feedback on their experience.

IX. Non-Functional Requirements

The non-functional requirements of the food donations and redistribution system define the quality attributes and constraints that govern its operation:

A. Performance

1) Response Time: The system should respond to user requests within acceptable time limits, with page load times optimized for minimal latency and delay.

2) Scalability: The system should be scalable to accommodate increasing user traffic and data volume, with provisions for horizontal scaling and load balancing.

B. Security

1) Data Encryption: Sensitive user data such as passwords and personal information should be encrypted during transmission and storage to prevent unauthorized access.

2) Access Control: Role-based access control (RBAC) should be implemented to restrict access to sensitive features and data based on user roles and permissions.

C. Usability

1) Intuitive Navigation: The user interface should be intuitive and easy to navigate, with clear labels, descriptive tooltips, and logical flow between pages.

2) Error Handling: The system should provide informative error messages and prompts to guide users in case of invalid inputs, unexpected errors, or system failures.

D. Reliability

1) Fault Tolerance: The system should be resilient to failures and errors, with mechanisms in place for error detection, recovery, and graceful degradation.

2) Data Backup: Regular backups of system data should be performed to prevent data loss in the event of hardware failures, software bugs, or security breaches.

E. Accessibility

1) Screen Reader Compatibility: The user interface should be compatible with screen readers and assistive technologies, ensuring accessibility for users with visual impairments.

2) Keyboard Navigation: Users should be able to navigate and interact with the application using keyboard shortcuts and commands, providing alternative input methods for users with motor disabilities.

X. Testing

The testing procedures for the food donations and redistribution system will encompass various levels and types of testing to ensure the reliability, functionality, and performance of the application:

A. Unit Testing

1) Component Testing:

- Test case 1: Verify that the user registration form accepts valid input data.
- Test case 2: Verify that the user registration form rejects invalid email addresses.
- Test case 3: Verify that the user login process authenticates valid credentials.

2) Mock Data Generation:

- Test case 1: Generate mock donation data with various item types and quantities.
- Test case 2: Generate mock order data with different delivery locations and preferences.

B. Integration Testing

1) API Integration:

- Test case 1: Validate that the frontend can successfully retrieve donation data from the backend API.
- Test case 2: Validate error handling when the backend API returns a 404 response.

2) Database Integration:

- Test case 1: Verify that donations are properly stored in the MySQL database upon submission.
- Test case 2: Verify that orders are correctly updated in the database upon delivery completion.

C. Acceptance Testing

1) User Acceptance Testing (UAT):

- Test case 1: End-user verifies that they can successfully register a new account.
- Test case 2: End-user verifies that they can place an order for donated food items.

2) Regression Testing:

- Test case 1: Verify that recent code changes do not affect the functionality of user registration.
- Test case 2: Verify that recent code changes do not introduce errors in the order fulfillment process.

D. Performance Testing

1) Load Testing:

- Test case 1: Simulate 100 simultaneous user registrations to assess system performance under load.
- Test case 2: Simulate 1000 concurrent order placements to evaluate system responsiveness.

2) Stress Testing:

- Test case 1: Simulate a sudden spike in user traffic to evaluate system stability and resource utilization.
- Test case 2: Simulate prolonged usage with maximum concurrent connections to identify potential bottlenecks.

XI. Outsource Libraries

The food donations and redistribution system may utilize third-party libraries or frameworks to enhance its functionality, improve development efficiency, or address specific requirements:

A. Bootstrap

Bootstrap may be used to streamline the development of responsive, mobile-first web interfaces, providing pre-designed components and layouts.

B. jQuery

jQuery may be used to simplify client-side scripting and DOM manipulation, facilitating interactive and dynamic user experiences.

C. Google Maps API

The Google Maps API may be integrated to provide geolocation services, including mapping, routing, and location-based search functionality.

D. PHPUnit

PHPUnit may be used as the testing framework for conducting unit tests, integration tests, and regression tests to ensure the reliability and stability of the application.

E. PHPMailer

PHPMailer may be utilized for sending transactional emails, notifications, and alerts to users, administrators, and delivery personnel.

XII. Conclusion

The software requirements specification for the food donations and redistribution system provides a comprehensive overview of the system's features, functionalities, architecture, and testing procedures. By following the outlined requirements and utilizing the recommended tools and techniques, the development team can build a robust and scalable solution that addresses the challenges of food donation and distribution effectively.