Phineas Kalambay

ST10295602

SOEN6222

SOFTWARE ENGINEER



Software Requirements Specification

1. Introduction

1.1 Purpose

The purpose of this document is to outline the software requirements for the Helping Hands Software Solution. This solution is designed to enhance the efficiency and effectiveness of Helping Hands, a non-profit organization dedicated to providing essential resources such as food, clothing, shelter, and education to those in need. The software will automate various operations, including resource distribution, program management, and volunteer coordination, thereby addressing the current challenges the organization faces with manual processes. (Krüger, 2023)

1.2 Agile Methodology and Scrum Framework

Agile methodology is a flexible, iterative approach to software development that focuses on delivering small, functional components of the software incrementally. It emphasizes collaboration, continuous feedback, and adaptability to change. Agile ensures that the development process can respond quickly to shifting requirements or changes in priorities. (simplilearn, 2024)

Why Agile for Helping Hands:

The dynamic nature of non-profit organizations like Helping Hands, which face ever-evolving resource needs, makes Agile the best methodology for this project. The ability to adjust requirements and deliver functional parts of the system, such as volunteer management or resource tracking, in stages ensures that the organization benefits early on while the system continues to evolve. (simplilearn, 2024)

Scrum Framework in Agile:

The Scrum framework, a specific implementation of Agile, organizes the project into short sprints (2-4 weeks). Each sprint focuses on developing and delivering specific system features. The Product Owner, Scrum Master, and development team collaborate closely to ensure each sprint's success, making this an ideal approach for managing the complexity and scale of the Helping Hands project. (simplilearn, 2024)

1.3 Scope

The Helping Hands Software Solution will be a web-based application that enables the organization to manage its resources, programs, volunteers, and beneficiaries more effectively. The software will include features such as user authentication, resource tracking, reporting, and communication tools. It will be designed to support scalability, security, and usability, ensuring that it can meet the growing needs of the organization.

3. Functional Requirements

Functional requirements define the specific operations, behaviours, or functions that the software must perform to meet the organization's needs. These requirements focus on the system’s capabilities and how it interacts with users. (altexsoft, 2023)

3.1 User Authentication

FR1: The system shall allow users to register using their email address and a secure password.

FR2: The system shall provide role-based access control, where different roles (e.g., Administrator, Volunteer, Beneficiary) have access to specific features.

3.2 Volunteer Management

FR3: The system shall allow administrators to create and manage volunteer profiles, including personal information and availability.

FR4: The system shall enable volunteers to view and update their assignment details.

3.3 Program Management

FR5: The system shall allow administrators to create, edit, and delete programs and initiatives.

FR6: The system shall track program outcomes and generate reports.

3.4 Resource Tracking

FR7: The system shall allow administrators to manage resource inventories, including adding, updating, and removing items.

FR8: The system shall track the distribution of resources to beneficiaries and generate inventory reports.

3.5 Beneficiary Management

FR9: The system shall allow administrators to create and manage beneficiary profiles, including personal information and needs assessment.

FR10: The system shall track the allocation of resources to beneficiaries.

3.6 Reporting

FR11: The system shall generate detailed reports on program outcomes, resource distribution, and volunteer activities.

FR12: The system shall allow administrators to export reports in PDF and Excel formats.

3.7 Notifications

FR13: The system shall send email notifications to volunteers about upcoming events or tasks.

FR14: The system shall notify beneficiaries about resource allocations or important updates.

3.8 Communication

FR15: The system shall include an internal messaging system for communication between volunteers and beneficiaries.

3.9 Data Backup

FR16: The system shall automatically back up data every 24 hours to a secure cloud storage.

3.10 Audit Trail

FR17: The system shall log all user activities for audit purposes, including login attempts, data modifications, and report generation.

4. External Interface Requirements

External Interface refers to the interaction points between the system and other external entities like third-party APIs, users, or other systems. It defines how the system communicates with external services to extend its functionality. (Anastasia, 2023)

4.3 Software Interfaces

Payment Gateway API: The system shall integrate with a payment gateway (e.g., PayPal, Stripe) for processing donations. This allows the organization to collect funds securely and efficiently. (Payment application program interfaces (APIs): What they are, different types, and their benefits for businesses, 2023) (Detailing the External Interfaces in the Software Requirements, 2022)

Resource Management API: The system shall integrate with a third-party API to update and manage resource inventories. This integration will enable seamless real-time updates for the organization’s resource management. (Resource Management integrations and API, 2024) (Detailing the External Interfaces in the Software Requirements, 2022)

Why are external interfaces important?

External interfaces are crucial for ensuring that the system can extend its functionality beyond internal operations. They allow the system to interact with other services, providing essential capabilities like payment processing and real-time resource updates, without having to build these features from scratch. This enhances the efficiency and effectiveness of the solution. (Detailing the External Interfaces in the Software Requirements, 2022)

5. Non-functional Requirements

Non-functional requirements define the quality attributes of a system, focusing on how the system performs rather than the specific behaviours it should exhibit. These include aspects such as performance, security, usability, and scalability. (altexsoft, 2023)

5.1 Security

NFR1: The system shall use industry-standard encryption (e.g., TLS) to protect data in transit.

NFR2: The system shall enforce strong password policies and multi-factor authentication for all user accounts. (altexsoft, 2023)

5.2 Performance

NFR3: The system shall handle up to 1,000 concurrent users without performance degradation.

NFR4: The system shall load the homepage within 3 seconds under normal network conditions.

5.3 Usability

NFR5: The system shall have an intuitive and user-friendly interface that requires minimal training for new users.

NFR6: The system shall provide context-sensitive help and tooltips throughout the UI. (altexsoft, 2023)

5.4 Scalability

NFR7: The system shall be scalable to support future growth, including additional users, programs, and resources. (altexsoft, 2023)

5.5 Reliability

NFR9: The system shall have an uptime of 99.9%, ensuring it is available when needed by the organization.

NFR10: The system shall be capable of recovering from failures with minimal data loss. (altexsoft, 2023)

# References

altexsoft. (2023, November 30). *Functional and Nonfunctional Requirements: Specification and Types*. Retrieved from altexsoft: https://www.altexsoft.com/blog/functional-and-non-functional-requirements-specification-and-types/

Anastasia, D. (2023, December 20). *External Interface Requirements*. Retrieved from cadabra: https://cadabra.studio/what-is-software-requirements-specification-and-how-to-write-it-with-example/#:~:text=External%20interface%20requirements%20define%20the,by%20Richard%20Thayer%20in%202002.

*Detailing the External Interfaces in the Software Requirements*. (2022, April 24). Retrieved from qat Global: https://qat.com/external-interfaces-software-requirements/#:~:text=External%20interfaces%20are%20critical%20in,party%20services%2C%20or%20hardware%20devices.

Krüger, G. (2023, January 17). *How to Write an SRS Document (Software Requirements Specification Document)*. Retrieved from Perforce: https://www.techtarget.com/searchsoftwarequality/tip/A-guide-to-software-design-documentation-and-specifications#:~:text=Before%20development%20teams%20can%20get,are%20going%20to%20get%20there.

*Payment application program interfaces (APIs): What they are, different types, and their benefits for businesses*. (2023, October 2). Retrieved from stripe: https://stripe.com/resources/more/payment-application-program-interfaces-apis#:~:text=Payment%20APIs%20serve%20as%20a,role%20it's%20intended%20to%20play.

*Resource Management integrations and API*. (2024, August 19). Retrieved from help: https://help.smartsheet.com/learning-track/resource-management-reporting/integrations-and-api

simplilearn. (2024, August 30). *Agile vs Scrum: Understanding the Core Differences*. Retrieved from simplilearn: https://www.simplilearn.com/agile-vs-scrum-article#:~:text=In%20Agile%20methodology%2C%20software%20is,Scrum%20is%20an%20Agile%20approach.