SOEN 6222

REQUIREMENT
DOCUMENT

Sajana Bidesi
ST10249843

Table of Contents

INTRODUCTION 2

FUNCTIONAL REQUIREMENTS 3

EXTERNAL INTERFACE REQUIREMENTS 4

External System: Payment Gateway 4

1. System Overview 4

2. Integration Requirements 4

3. Testability 4

4. Feasibility 5

Non-Functional Requirements 6

Performance 6

Security 6

Usability 6

Availability 6

Scalability 6

REFERENCES 7

INTRODUCTION

"Helping Hands" is a charity that provides support to individuals and families by giving them basic needs like food, clothes, residence, and education. The present manual processes, through the use of spreadsheets and paper-based systems, are quite ineffective and prone to mistakes, thus complicating the task of tracking the distribution of resources, the supervision of program efficiency and the involvement of two parties: the volunteers and the beneficiaries. To tackle these problems, a web-based software solution will be designed.

This software will simply be used to automate the program management, to manage the distribution of resources, to coordinate volunteers, and to communicate. This will improve the operation by reducing the time of work, provide real-time tracking, and make transparency easier during the management of the organization's activities. The system will also give Helping Hands the chance to involve the community better through improved communication and collaboration.

The aim of this project is the development and efficient running of a website that helps control programs, volunteers, beneficiaries, donations, and resource distribution. The site will enable both an administrator and a user to the directives that they will deliver.

The Agile methodology is a flexible approach that develops software in small, iterative cycles. For Helping Hands, this means the web-based solution will be built in phases, focusing on resource tracking, volunteer management, and donation handling. The project is divided into short sprints with regular stakeholder feedback, allowing continuous adjustments to meet real needs. Agile enables Helping Hands to respond quickly to new demands, improving efficiency by automating manual processes. By involving staff and volunteers throughout development, the system ensures it solves real problems, making Helping Hands more effective in serving individuals and families.

Scrum, as an Agile methodology, is particularly well-suited for Helping Hands. Its iterative process delivers small features in short sprints, focusing on specific functions like resource tracking, volunteer coordination, and donation management. With regular feedback and active involvement from stakeholders, Scrum ensures the system remains aligned with the charity’s evolving needs. Daily standups and sprint reviews promote ongoing improvement, while each sprint delivers usable features that improve efficiency and transparency, allowing Helping Hands to better manage resources and respond quickly to changing demands. (Veritis, n.d.)

FUNCTIONAL REQUIREMENTS

Functional Requirement	**Requirement Name**	**Testable**	**Feasible**
User Registration and Authentication	The system will allow users (volunteers, staff, donors) to create accounts and log in securely.	Test registration form, password strength, error handling, and login success.	Implement with OAuth or JWT protocols. (Mukhadin Beschokov, 2024)
Volunteer Management System	The system will track volunteer availability, skills, and tasks, and allow volunteers to update their profiles.	Test profile updates, task assignment, and schedule notifications.	Implement using relational databases and scheduling modules.
Donation Management System	The system will track donations, generate receipts, and store donor information.	Test donation amounts, payment processing, and receipt generation.	Integrate with payment gateways (e.g: PayFast, Zapper, PayU, Yoco). (Lacroix, 2022)
Resource Distribution Tracking	The system will log and track the distribution of resources (food, clothing, etc.) to beneficiaries.	Test resource logging, stock level updates, and distribution reports.	Implement using inventory management systems and databases. (Volha Belakurska, 2024)
Program Outcome Reporting	The system will generate reports on program outcomes (e.g., families helped, volunteer hours logged).	Test report generation accuracy based on program types and date filters.	Use SQL Server reporting tools or Business Intelligence (BI) tools for data visualization. (Atlassian, 2019)
Volunteer Scheduling and Task Assignment	The system will assign tasks and schedules based on volunteer availability and display them on a dashboard.	Test schedule updates, conflict management, and notification systems.	Use scheduling libraries and integrated calendars for task management.
Beneficiary Registration and Tracking	The system will allow beneficiaries to register for programs and track their application statuses.	Test application process, status updates, and notifications.	Implement using user profiles and automated workflows for program registration.
Automated Notifications for Volunteers and Donors	The system will send email notifications to volunteers (tasks) and donors (confirmation, thank you).	Test email notification triggers for different scenarios (e.g., donation, task assignment).	Integrate for communication services.
Search and Filter Functionality for Programs and Resources	The system will allow users to search and filter programs or resources by categories like location or type of support.	Test search queries for accuracy and filters for correct refinement.	Implement using database queries and search tools for fast performance.
User Roles and Permissions Management	The system will control user access based on roles (admin, staff, volunteers, donors) to limit functionalities accordingly.	Test role-based access to restricted areas and permissions for each user type.	Implement using Role-Based Access Control (RBAC) via middleware or authentication libraries. (Microsoft, 2022)

EXTERNAL INTERFACE REQUIREMENTS

External System: Payment Gateway

1. System Overview

The payment gateway plays a crucial role in managing online transactions for the Helping Hands website, enabling secure processing of donations through a variety of methods including credit and debit cards, mobile payments, and bank transfers. By integrating with payment gateways such as PayFast, Zapper, PayU, and Yoco, the website can offer flexible and reliable payment options to users, ensuring that all donations are handled securely and efficiently. (Stripe, n.d.)

2. Integration Requirements

API Integration

To manage payment transactions effectively, integration with the payment gateway's API is vital. This involves sending payment requests, handling the processing of transactions, and receiving responses from the gateway. The system must guarantee secure transmission of payment information and provide real-time updates on transaction statuses, as well as handle payment confirmations efficiently to ensure smooth and accurate processing. (Dantzer, 2024)

Data Security

Ensuring data security is essential to protect sensitive payment information. Payment data should be encrypted during transmission using SSL/TLS (Digicert, 2023)to safeguard communication between the website and the payment gateway. Additionally, implementing tokenization is important for securely managing sensitive payment data, thus minimizing the risk of exposing credit card information and enhancing overall security.

Transaction Management

Effective transaction management requires the payment gateway to support real-time processing, which offers immediate feedback on donation statuses and keeps records current. The system must also handle payment errors appropriately, providing users with clear guidance and information if any transaction issues arise. This ensures users are well-informed and supported throughout their payment experience. (Dantzer, 2024)

Reporting and Reconciliation

The website should record successful transactions in its database, capturing details such as donor information, donation amount, and transaction ID. The system must also support reconciliation by comparing transaction records with payment gateway reports to ensure accuracy and address any discrepancies. This helps maintain precise financial records and resolves any issues efficiently.

3. Testability

- Payment Methods: Test various payment methods supported by the gateway (e.g., credit/debit cards, mobile payments) to confirm transactions are processed correctly and efficiently.
- Error Scenarios: Simulate different error conditions (e.g., declined transactions, network issues) to ensure the system handles errors appropriately and provides useful feedback.
- Security: Perform security testing to verify that encryption (Digicert, 2023)and tokenization are effectively implemented, and that payment data is well-protected.

4. Feasibility

The feasibility of integrating a payment gateway into the Helping Hands website is supported by the availability of comprehensive APIs from various payment providers, such as PayFast, Zapper, PayU, and Yoco. These payment gateways offer well-documented APIs, which simplify the integration process by providing clear guidelines and technical details necessary for implementation. The thorough documentation ensures that developers have the resources needed to understand and utilize the APIs effectively, reducing the complexity of integration (Dantzer, 2024). Additionally, these payment gateways provide robust support to assist developers with any issues that may arise during the integration process, further enhancing the feasibility of incorporating these solutions into the website. This support helps to ensure that the integration is both efficient and effective, allowing the Helping Hands website to manage donations seamlessly and securely.

Non-Functional Requirements

Performance

The system must be designed to support up to 1,000 concurrent users while ensuring optimal performance. It is crucial that page load times consistently stay under 3 seconds to provide a responsive and efficient user experience, especially during peak traffic times. (Rome, 2020)

Security

To protect sensitive data, the system must use strong encryption protocols like SSL/TLS for data transmission. Additionally, user information storage must adhere to data protection laws, including the Protection of Personal Information Act (POPIA) in South Africa (POPIA, 2021). Following POPIA is vital for safeguarding personal data and ensuring compliance with local data protection regulations. (Rome, 2020)

Usability

The system should have a user-friendly interface that makes navigation and task completion easy for volunteers, donors, and beneficiaries. Good usability is essential so that users with different technical skills can interact with the system effectively and with minimal hassle. (Rome, 2020)

Availability

The system must maintain a minimum availability rate of 99.9%, ensuring it is almost always accessible. Scheduled maintenance should be communicated to users ahead of time to reduce disruptions and maintain user trust and satisfaction. (Rome, 2020)

Scalability

The system's design should allow for scalability, making it easy to add users, resources, and programs as the organization grows. This ability to scale ensures that the system can meet increasing demands and growth without sacrificing performance. (Rome, 2020)

REFERENCES

Atlassian (2019). *Top 10 BI Tools for Data Visualization & Analysis*. [online] Atlassian. Available at: https://www.atlassian.com/data/business-intelligence/10-data-visualization-tools#:~:text=Google%20Data%20Studio [Accessed 13 Sep. 2024].

Dantzer, J. (2024). *Payment API explained: Connecting a payment gateway*. [online] Checkout.com. Available at: https://www.checkout.com/blog/online-payment-api-explained#:~:text=a%20payment%20API%3F- [Accessed 13 Sep. 2024].

Digicert (2023). *What is SSL, TLS and HTTPS? | DigiCert*. [online] www.digicert.com. Available at: https://www.digicert.com/what-is-ssl-tls-and-https.

Lacroix, B. (2022). *The Best Payment Gateways in South Africa*. [online] Portmoni. Available at: https://portmoni.com/best-payment-gateway-south-africa/.

Microsoft (2022). *What is Azure role-based access control (Azure RBAC)?* [online] learn.microsoft.com. Available at: https://learn.microsoft.com/en-us/azure/role-based-access-control/overview.

Mukhadin Beschokov (2024). *OAuth vs JWT*. [online] Wallarm.com. Available at: https://www.wallarm.com/what/oauth-vs-jwt-detailed-comparison#:~:text=JWT%20token%20vs%20oauth%20token.

POPIA (2021). *Protection of Personal Information Act (POPI Act)*. [online] POPIA. Available at: https://popia.co.za/.

Rome, P. (2020). *What are Non Functional Requirements — With Examples*. [online] Perforce Software. Available at: https://www.perforce.com/blog/alm/what-are-non-functional-requirements-examples.

Stripe (n.d.). *What is a payment gateway? | Stripe*. [online] stripe.com. Available at: https://stripe.com/resources/more/payment-gateways-101#:~:text=A%20payment%20gateway%20is%20a.

Veritis (n.d.). 7 Popular Types of Agile Methodologies. [online] Available at: https://www.veritis.com/blog/7-important-types-of-agile-methodologies/.

Volha Belakurska (2024). *Inventory Management Database: Understanding Inventory Database and Inventory Management System*. [online] Resources. Available at: https://synder.com/blog/inventory-management-database/#:~:text=An%20inventory%20management%20database%20is [Accessed 13 Sep. 2024].