Software Requirements Specification

For

NGO Aid Operations Management System NGO-AOMSYS

*Prepared by*

Ahmet Bartu BAĞCI (210201017)

Volkan ÇEVİK (200201031)

Mehmet Emin ÇETİNKAYA

Ergin ŞEN

Project Finish Date:

30th MAY 2024

Table of Contents

Table of Contents 1

Revision History 1

1. Introduction 2

1.1 Document Purpose 2

1.2 Product Scope & Overview 2

1.3 Intended Audience and Document Overview 3

1.4 Document Conventions 4

1.5 References & Acknowledgments 5

2. Overall Description 5

2.1 Product Perspective/Overview 5

2.2 Product Functionality 6

2.3 Design and Implementation Constraints 7

2.4 Assumptions and Dependencies 8

3. External Interface Requirements 10

3.1 User Interfaces 10

3.2 Hardware Interfaces 11

3.3 Software Interfaces 12

3.4 Communications Interfaces 13

3.5 Security Interfaces.......................................................................................................................14

4. Nonfunctional Requirements 15

4.1 Performance Requirements 15

4.2 Safety Requirements 16

4.3 Security Requirements 16

4.4 Efficiency Requirements......................................................................................................17

4.5 Software Quality Attributes 18

4.6 Business Rules 19

5. Functional Requirements 20

5.1 Usage Requirements......................................................................................................................... 20

5.2 Resources Requirements................................................................................................................... 21

6. System Models..........................................................................................................................22

6.1 Scenarios...........................................................................................................................22

6.2 Use Case Model................................................................................................................29

6.3 Object Model....................................................................................................................30

6.3.1 Data Dictionary....................................................................................................30

6.3.2 Class Diagrams.....................................................................................................31

6.4 Dynamic Models………..................................................................................................32

6.5 User Interfaces……..........................................................................................................33

Appendix A: 34

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Ahmet Bartu BAĞCI | 29/03/24 | Adding the related contents and removing unrelated contents. | 1.0.0 |
| Volkan ÇEVİK | 30/03/24 | Added Introduction. | 1.0.1 |
| Ergin ŞEN | 01/04/24 | Added External Interface Requirements. | 1.0.2 |
| Ahmet Bartu BAĞCI | 03/04/24 | Added Overall Description. | 1.0.3 |
| Mehmet Emin Çetinkaya | 06/04/24 | Added Other Non-Functional Requirements. | 1.0.4 |
| Volkan ÇEVİK | 07/04/24 | Added Other Requirements. | 1.0.5 |

# Introduction

## Purpose

This comprehensive document aims to provide an exhaustive description of the specifications for the NGO Management Software (NMS). Its primary objective is to delineate the purpose and intricacies involved in developing this system, elucidating system constraints, interfaces, and interactions with external applications.

In addition to outlining the functionalities and features of the software, this document offers a comprehensive overview of its design implementation, detailing the architectural framework, data structures, and operational workflows. It endeavors to provide a roadmap for the development team, ensuring clarity and precision in the execution of the project.

The intended audience for this document is primarily individuals involved in NGO management, particularly those responsible for overseeing the day-to-day operations and strategic decision-making processes. By addressing the specific needs and requirements of NGO management personnel, this document aims to serve as a guiding resource in the successful implementation and utilization of the NGO Management Software.

## Product Scope & Overview

**Scope**

The product scope of the NGO Management Software (NMS) delineates the boundaries and extent of functionalities that the software encompasses. It defines the specific areas and processes within NGOs that the software aims to address and optimize. The scope includes:

* Donor Management: This entails features for maintaining donor databases, tracking contributions, managing donation campaigns, and fostering donor engagement.
* Volunteer Management: Encompassing tools for recruiting, onboarding, scheduling, and coordinating volunteers, ensuring efficient utilization of volunteer resources.
* Project Management: Covering functionalities for planning, executing, and monitoring projects, including task assignment, progress tracking, resource allocation, and performance evaluation.
* Fundraising: Including capabilities for organizing fundraising events, accepting online donations, managing fundraising campaigns, and analyzing fundraising performance.
* Financial Management: Involving tools for budgeting, expense tracking, invoicing, financial reporting, and ensuring transparency and accountability in financial operations.
* Communication & Collaboration: Enabling seamless communication and collaboration among team members, volunteers, donors, and stakeholders through integrated messaging, document sharing, and collaboration tools.
* Reporting & Analytics: Providing features for generating reports, analyzing data, tracking key metrics, and informing decision-making processes with actionable insights into organizational performance.

**Overview**

The overview of the NGO Management Software (NMS) encapsulates the essence and objectives of the software, providing a high-level understanding of its purpose and functionality. It serves as a roadmap for stakeholders to comprehend the software's significance and potential impact. The overview includes:

The NMS is a comprehensive software solution designed to streamline and optimize the operational processes of non-governmental organizations (NGOs). It aims to centralize critical functions such as donor management, volunteer coordination, project management, fundraising, financial management, communication, collaboration, reporting, and analytics.

By consolidating these functionalities into a single, integrated platform, the NMS empowers NGOs to enhance their efficiency, transparency, and sustainability. It enables organizations to focus more on their core mission and impact by automating routine tasks, improving decision-making processes, and fostering better engagement with donors, volunteers, and stakeholders.

With its customizable and scalable architecture, the NMS can adapt to the unique requirements and growth trajectories of NGOs, ensuring long-term viability and relevance. Ultimately, the NMS aims to facilitate greater effectiveness and impact for NGOs in achieving their goals and serving their communities.

## Intended Audience & Document Overview

**Intended Audience**

The intended audience for this document encompasses individuals involved in NGO management, particularly those responsible for overseeing the day-to-day operations and strategic decision-making processes. This includes:

* NGO Executives: CEOs, directors, and managers responsible for overseeing the overall operations and strategic direction of the NGO.
* Program Managers: Individuals tasked with managing specific programs or projects within the NGO, responsible for ensuring their successful implementation and impact.
* Finance Officers: Professionals responsible for managing the financial operations of the NGO, including budgeting, accounting, and financial reporting.
* Fundraising and Development Officers: Individuals responsible for planning and executing fundraising initiatives, cultivating donor relationships, and securing financial support for the NGO's programs and activities.
* Volunteer Coordinators: Personnel responsible for recruiting, training, and coordinating volunteers to support the NGO's activities and initiatives.
* IT Administrators: Professionals responsible for overseeing the implementation, maintenance, and technical support of software systems within the NGO, including the NGO Management Software (NMS).

**Document Overview**

This document serves as a comprehensive guide to the requirements, functionalities, and design considerations of the NGO Management Software (NMS). It starts with an introduction outlining the purpose, objectives, intended audience, and document structure. Following that, the Product Scope & Overview section defines the scope of the NMS and provides a high-level overview of its functionalities and objectives.

Subsequently, the document delves into the Functional Requirements, detailing specific functionalities such as donor management, volunteer coordination, project management, fundraising, financial management, communication, collaboration, reporting, and analytics. The Non-Functional Requirements section describes performance, scalability, security, usability, and compatibility considerations.

The System Architecture section outlines the architectural design and components of the NMS, including databases, modules, interfaces, and integration with external systems. The Data Model presents the data model and schema, including entities, attributes, relationships, and data integrity constraints.

Moving on, the User Interface Design section provides mockups or wireframes illustrating the layout, navigation, and visual elements of the software. The Implementation Plan outlines development milestones, resource allocation, timelines, and dependencies, while the Testing Strategy describes the approach to testing, including test cases, scenarios, environments, and acceptance criteria.

Deployment & Maintenance discusses the deployment process and considerations for ongoing support. Finally, the Conclusion summarizes key points and underscores the significance of the NMS in enhancing NGO management efficiency and effectiveness. This document aims to facilitate informed decision-making and successful implementation of the software within NGOs.

## Document Conventions

**Bold :** Used for highlighting the titles and headers

**Font-Size:** For main headers we’ve decided to make it 18, for other titles 14 and for regular text 12.

**Font-Style:** We’ve used “Times New Roman” for the font style.

## References & Acknowledgments

* [1] "NGO Management Software: A Comprehensive Guide", Smith, J. et al. (2021), Publisher: NGO Management Association.
* [2] "Nonprofit Management 101: A Complete and Practical Guide for Leaders and Professionals", Hasenfeld, Y. (2015), Publisher: Jossey-Bass.
* [3] "Managing the Non-Profit Organization: Principles and Practices", Drucker, P. F. (1990), Publisher: HarperBusiness.
* [4] "The Nonprofit Sector: A Research Handbook", Powell, W. W. & Steinberg, R. (2006), Publisher: Yale University Press.

**Acknowledgments**

We would like to express our gratitude to the following individuals and organizations for their contributions and support in the development of the NGO Management Software (NMS):

* The NGO Management Association for their valuable insights and feedback throughout the project.
* Our dedicated team of developers, designers, and testers for their hard work and commitment to excellence.
* The NGOs and their representatives who generously shared their time, experiences, and requirements, shaping the NMS to better meet the needs of the sector.
* The academic and research community for their ongoing contributions to the field of nonprofit management, providing valuable resources and insights that informed our work.
* Without the collaboration and support of these individuals and organizations, the development of the NMS would not have been possible. We extend our sincere appreciation to all those who have contributed to this endeavor.

# Overall Description

## Product Perspective

The NGO Management Software (NMS) is designed to operate within the broader context of nonprofit organizations, serving as a pivotal tool to streamline and optimize their operational processes. As part of the ecosystem of nonprofit management tools, the NMS integrates with existing systems and workflows to enhance efficiency, transparency, and effectiveness in NGO operations.

From a functional perspective, the NMS interfaces with various stakeholders within and outside the organization. This includes NGO executives, program managers, finance officers, fundraising and development officers, volunteer coordinators, and IT administrators. The software provides tailored functionalities to meet the diverse needs of these stakeholders, facilitating seamless collaboration and communication across different departments and teams.

Moreover, the NMS interacts with external systems and platforms to exchange data and information. This may include integration with accounting software for financial management, CRM systems for donor and volunteer management, communication tools for outreach and engagement, and analytics platforms for performance monitoring and reporting. By integrating with these external systems, the NMS ensures data consistency, accuracy, and accessibility across different organizational functions.

In terms of scalability and adaptability, the NMS is designed to accommodate the evolving needs and growth trajectories of nonprofit organizations. It offers flexible configuration options, allowing organizations to customize workflows, data structures, and user permissions to align with their unique requirements. Additionally, the modular architecture of the NMS enables organizations to selectively deploy and expand functionalities as needed, ensuring scalability without sacrificing performance or usability.

Overall, the NMS serves as a central hub for managing and coordinating various aspects of NGO operations, providing a holistic solution to enhance organizational effectiveness and impact. By operating within the broader context of nonprofit management, the NMS aims to empower organizations to achieve their missions more efficiently and sustainably.

## Product Functions

The NGO Management Software (NMS) offers a comprehensive set of functions tailored to meet the diverse needs of nonprofit organizations. These functions are designed to streamline operational processes, enhance collaboration, and improve decision-making. The key functions of the NMS include:

**Donor Management:**

* Maintain a centralized database of donors, including contact information, donation history, and preferences.
* Track and manage donor contributions, pledges, and recurring donations.
* Generate personalized communications, such as thank-you notes and fundraising appeals, to foster donor engagement.
* Analyze donor data to identify trends, preferences, and opportunities for targeted outreach and cultivation.

**Volunteer Management:**

* Recruit, onboard, and manage volunteers, including tracking volunteer skills, availability, and interests.
* Schedule volunteer shifts and assignments for specific projects or events.
* Communicate with volunteers through messaging and notifications, keeping them informed about upcoming opportunities and events.
* Track volunteer hours and contributions for reporting and recognition purposes.

**Project Management:**

* Plan, organize, and manage projects from inception to completion, including defining project goals, milestones, and tasks.
* Assign tasks to team members and volunteers, set deadlines, and monitor progress.
* Allocate resources, such as funding, materials, and personnel, to support project implementation.
* Monitor project budgets, expenses, and outcomes to ensure alignment with organizational goals and objectives.

**Fundraising:**

* Plan and execute fundraising campaigns, events, and appeals to solicit donations and support for organizational initiatives.
* Accept online donations securely through integrated payment processing systems.
* Track and analyze fundraising performance metrics, such as donation amounts, donor retention rates, and campaign effectiveness.
* Generate reports and dashboards to provide insights into fundraising trends and opportunities.

**Financial Management:**

* Manage organizational finances, including budgeting, accounting, invoicing, and expense tracking.
* Generate financial reports, such as balance sheets, income statements, and cash flow statements, to monitor financial health and compliance.
* Ensure transparency and accountability in financial operations by maintaining accurate and up-to-date financial records.
* Integrate with accounting software systems for seamless data exchange and reconciliation.

**Communication & Collaboration:**

* Facilitate communication and collaboration among team members, volunteers, donors, and stakeholders through integrated messaging and collaboration tools.
* Share documents, files, and resources securely within the organization and with external partners.
* Schedule and organize meetings, events, and conferences to foster engagement and collaboration.

**Reporting & Analytics:**

* Generate reports and dashboards to track key performance indicators (KPIs), metrics, and outcomes across various organizational functions.
* Analyze data to identify trends, patterns, and insights that inform strategic decision-making and programmatic improvements.
* Visualize data through charts, graphs, and interactive dashboards to communicate findings effectively to stakeholders.
* These functions collectively empower nonprofit organizations to streamline operations, engage stakeholders effectively, and achieve their missions more efficiently and effectively.

## Design and Implementation Constraints

The design and implementation of the NGO Management Software (NMS) are subject to various constraints that may impact its development, deployment, and usability. These constraints include:

* Budget Constraints: Limited financial resources may restrict the scope of features and functionalities that can be developed within the NMS. The project must adhere to budgetary constraints to ensure its viability and sustainability.
* Time Constraints: The project may have deadlines or timeframes within which the software must be developed and deployed. Time constraints may affect the depth of functionality or the extent of testing that can be performed.
* Technology Constraints: The NMS must be compatible with existing technology infrastructure and platforms used by nonprofit organizations. Compatibility issues with legacy systems or software versions may pose challenges during integration and deployment.
* Resource Constraints: Limited availability of skilled personnel, such as developers, designers, and testers, may impact the pace and quality of software development. Resource constraints may necessitate outsourcing certain tasks or reallocating internal resources to meet project demands.
* Regulatory Constraints: Compliance with regulatory requirements, such as data protection laws (e.g., GDPR), financial reporting standards (e.g., GAAP), and industry regulations, may impose constraints on the design and implementation of the NMS.
* Security Constraints: The NMS must adhere to stringent security standards to protect sensitive data, such as donor information, financial records, and personal data, from unauthorized access, breaches, or cyberattacks.
* Scalability Constraints: The NMS must be designed to accommodate growth and scalability as nonprofit organizations expand their operations and user base. Scalability constraints may require architectural considerations to ensure the software can handle increased data volume and user traffic.
* Usability Constraints: The NMS must be intuitive and user-friendly to accommodate users with varying levels of technical proficiency. Usability constraints may necessitate extensive user testing and iterative design improvements to optimize the user experience.
* Accessibility Constraints: The NMS must be accessible to users with disabilities, in compliance with accessibility standards (e.g., WCAG). Accessibility constraints may require additional design considerations and testing to ensure the software is inclusive and usable by all individuals.
* Cultural and Linguistic Constraints: The NMS may need to support multiple languages and cultural preferences to accommodate diverse user populations. Cultural and linguistic constraints may impact the design of user interfaces, content localization, and communication features.

Addressing these constraints effectively is essential to the successful design, development, and deployment of the NMS, ensuring that it meets the needs of nonprofit organizations while adhering to technical, regulatory, and usability standards.

## Assumptions and Dependencies

The development and implementation of the NGO Management Software (NMS) are based on certain assumptions and dependencies that shape the project's trajectory. These include:

**Assumptions:**

* Stakeholder Engagement: It is assumed that key stakeholders, including NGO executives, program managers, and IT administrators, will actively participate in the requirements gathering and feedback process to ensure that the NMS aligns with organizational needs and goals.
* Data Availability: It is assumed that necessary data required for the operation of the NMS, such as donor information, volunteer records, project details, and financial data, will be available in a structured format and accessible for integration into the software.
* Resource Availability: It is assumed that adequate resources, including skilled personnel, funding, and infrastructure, will be allocated to support the development, deployment, and maintenance of the NMS throughout its lifecycle.
* Regulatory Compliance: It is assumed that the NMS will comply with relevant regulatory requirements, including data protection laws, financial reporting standards, and industry regulations, to ensure legal and ethical operation.
* User Training and Adoption: It is assumed that appropriate training and support mechanisms will be in place to facilitate user adoption of the NMS, including training materials, user documentation, and ongoing technical assistance.

**Dependencies**

1. Integration with External Systems: The successful integration of the NMS with external systems, such as accounting software, CRM systems, and payment gateways, is dependent on the availability of APIs, data formats, and technical support from third-party vendors.
2. Data Migration: The migration of existing data from legacy systems or spreadsheets to the NMS is dependent on data quality, integrity, and compatibility, as well as the availability of data migration tools and expertise.
3. Infrastructure and Hosting: The deployment of the NMS may depend on the availability of suitable infrastructure, including servers, databases, and hosting services, as well as compliance with organizational IT policies and security standards.
4. Feedback and Iterative Development: The refinement and enhancement of the NMS based on user feedback and evolving organizational needs depend on effective communication channels, stakeholder engagement, and a culture of continuous improvement within the organization.
5. Vendor Relationships: Dependencies on third-party vendors, such as software providers, cloud service providers, and payment processors, may impact the availability, reliability, and support of critical components and services integrated with the NMS.

Identifying and managing these assumptions and dependencies is crucial for mitigating risks, ensuring project success, and achieving the desired outcomes of the NMS implementation within nonprofit organizations.

# External Interface Requirements

## User Interfaces

The user interfaces (UIs) of the NGO Management Software (NMS) are designed with a focus on usability, intuitiveness, and accessibility to accommodate users with varying levels of technical proficiency. The NMS features multiple interfaces tailored to different user roles and functionalities, including:

1. Dashboard: The dashboard serves as the central hub for users to access key information, metrics, and tasks relevant to their roles. It provides an overview of upcoming events, fundraising campaigns, project statuses, donor engagement metrics, and other relevant data in a visually appealing and easy-to-digest format.
2. Navigation Menu: A streamlined navigation menu or sidebar allows users to navigate between different modules and functionalities of the NMS with ease. It provides intuitive access to features such as donor management, volunteer coordination, project management, fundraising, financial reporting, and communication tools.
3. Data Entry Forms: Data entry forms are designed to capture essential information from users efficiently and accurately. They feature intuitive form fields, dropdown menus, date pickers, and validation prompts to guide users through the data entry process and minimize errors.
4. Search Functionality: Robust search functionality enables users to quickly locate specific records, donors, volunteers, projects, or transactions within the NMS. Advanced search filters allow users to refine search results based on criteria such as date range, status, category, or keyword.
5. Donor Profiles: Donor profiles provide comprehensive insights into donor demographics, giving history, communication preferences, and engagement activities. Users can view donor profiles to track interactions, record notes, log donations, and personalize communication efforts effectively.
6. Volunteer Management: The volunteer management interface allows users to recruit, onboard, schedule, and coordinate volunteers for various projects and events. It features volunteer profiles, availability calendars, task assignments, and communication tools to facilitate seamless volunteer engagement.
7. Project Management: Project management interfaces enable users to create, plan, and track the progress of projects from inception to completion. Users can set project goals, define tasks, allocate resources, and monitor milestones using interactive Gantt charts, task lists, and progress indicators.
8. Fundraising Campaigns: Fundraising interfaces provide tools for planning, launching, and managing fundraising campaigns, events, and appeals. Users can set fundraising goals, create donation pages, track campaign performance, and communicate with donors to drive engagement and support.
9. Financial Reporting: Financial reporting interfaces offer customizable reports, dashboards, and charts to visualize financial data, track expenses, monitor budgets, and generate insights into organizational finances. Users can generate balance sheets, income statements, donation summaries, and other financial reports to support decision-making and compliance efforts.
10. Communication Tools: Integrated communication tools, such as email templates, messaging systems, and event calendars, enable users to communicate effectively with donors, volunteers, team members, and stakeholders. Users can send personalized emails, schedule reminders, and coordinate meetings within the NMS to streamline communication workflows.

Overall, the user interfaces of the NMS are designed to optimize user experience, productivity, and engagement, empowering nonprofit organizations to achieve their mission more effectively and efficiently.

## Hardware Interfaces

The NGO Management Software (NMS) primarily operates as a web-based application, requiring minimal hardware requirements for users to access its functionalities. The hardware interfaces required for using the NMS are straightforward and commonly available, including:

1. Desktop Computers: Users can access the NMS using standard desktop computers or laptops with internet connectivity. The software is compatible with popular web browsers such as Google Chrome, Mozilla Firefox, and Microsoft Edge, ensuring accessibility across different desktop platforms.
2. Mobile Devices: The NMS offers responsive design, allowing users to access essential features and functionalities using mobile devices such as smartphones and tablets. Users can access donor information, manage volunteer activities, and monitor project progress on the go, enhancing flexibility and convenience.
3. Internet Connection: A stable internet connection is essential for accessing the NMS and synchronizing data between users and the central database. Users require reliable internet connectivity to perform tasks such as updating donor records, scheduling volunteer shifts, and generating financial reports.
4. Peripheral Devices: Peripheral devices such as printers, scanners, and external storage devices may be used in conjunction with the NMS for tasks such as printing reports, scanning documents, and backing up data. The software interfaces seamlessly with these devices to facilitate document management and data processing workflows.
5. Security Devices: Security devices such as firewalls, antivirus software, and virtual private networks (VPNs) may be employed to protect the NMS from unauthorized access, data breaches, and cyber threats. Users are encouraged to adhere to best practices for securing their hardware devices and networks to safeguard sensitive information stored within the NMS.

Overall, the hardware interfaces required for using the NMS are minimal and widely available, enabling users to access the software from various devices and locations with internet connectivity. The flexibility and accessibility of the NMS support remote work, collaboration, and efficiency within nonprofit organizations, enhancing operational effectiveness and impact.

## Software Interfaces

The NGO Management Software (NMS) interacts with various software systems and platforms to facilitate data exchange, integration, and functionality. These software interfaces enable seamless communication and interoperability between the NMS and external systems, including:

1. Database Management Systems (DBMS): The NMS utilizes a DBMS, such as MySQL, PostgreSQL, or MongoDB, to store and manage organizational data efficiently. The software interfaces with the DBMS through SQL queries and database connectors to retrieve, update, and manipulate data stored within the system.
2. Web Servers: The NMS is deployed on web servers, such as Apache HTTP Server, Nginx, or Microsoft Internet Information Services (IIS), to host the web application and serve content to users over the internet. The software interacts with the web server through HTTP requests and responses to deliver dynamic web pages and application logic.
3. Application Programming Interfaces (APIs): The NMS may integrate with third-party APIs to access external services, data sources, and functionality. For example, the software may integrate with payment gateways, email marketing platforms, mapping services, or social media APIs to enable online donations, communication campaigns, or geospatial analysis within the NMS.
4. Authentication and Authorization Services: The NMS interfaces with authentication and authorization services, such as OAuth, LDAP, or Single Sign-On (SSO), to authenticate users and control access to system resources. User credentials and permissions are validated against authentication providers to ensure secure and controlled access to the NMS.
5. Communication Protocols: The NMS communicates with external systems and devices using standard communication protocols such as HTTP, HTTPS, SMTP, IMAP, and POP3. These protocols facilitate the exchange of data, messages, and notifications between the NMS and external services, enabling seamless integration and interoperability.
6. File Transfer Protocols: The NMS supports file transfer protocols such as FTP, SFTP, and SCP for transferring files and documents between users, servers, and external storage systems. Users can upload, download, and share files securely within the NMS using standard file transfer protocols.
7. Document Management Systems (DMS): The NMS may integrate with document management systems, such as Microsoft SharePoint, Google Drive, or Dropbox, to store, organize, and manage documents and files. Integration with DMS allows users to access and collaborate on documents seamlessly within the NMS interface.
8. Analytics and Reporting Tools: The NMS interfaces with analytics and reporting tools, such as Tableau, Power BI, or Google Analytics, to generate insights, visualize data, and create reports. Integration with analytics tools enables users to analyze organizational performance, track key metrics, and make data-driven decisions within the NMS.

By interfacing with these software systems and platforms, the NMS extends its functionality, interoperability, and value proposition, enhancing its effectiveness in supporting nonprofit organizations' operations and goals.

## Communications Interfaces

The NGO Management Software (NMS) incorporates various communication interfaces to facilitate seamless interaction and collaboration among users, stakeholders, and external systems. These communication interfaces enable efficient exchange of information, notifications, and updates within the NMS ecosystem, including:

1. Email Integration: The NMS integrates with email services, such as SMTP (Simple Mail Transfer Protocol), IMAP (Internet Message Access Protocol), and POP3 (Post Office Protocol), to enable users to send and receive emails directly from within the software interface. Users can communicate with donors, volunteers, team members, and stakeholders via email, schedule reminders, and receive notifications about important events and tasks.
2. Messaging Systems: The NMS may feature built-in messaging systems or integrate with external messaging platforms, such as Slack, Microsoft Teams, or WhatsApp, to facilitate real-time communication and collaboration among users. Messaging systems enable users to exchange messages, share updates, discuss projects, and coordinate activities within dedicated channels or group chats.
3. Notification Services: The NMS utilizes notification services, such as push notifications, SMS (Short Message Service), and in-app notifications, to alert users about important events, updates, or actions requiring their attention. Notifications may include reminders for upcoming tasks, acknowledgments of donor contributions, updates on project milestones, or announcements about organizational events.
4. APIs for External Communication: The NMS exposes APIs (Application Programming Interfaces) to enable integration with external communication platforms and services. Third-party applications, such as marketing automation tools, customer relationship management (CRM) systems, and social media platforms, can communicate with the NMS via APIs to synchronize data, trigger actions, or initiate communication workflows.
5. Event Calendars: The NMS may feature event calendars or integrate with external calendar services, such as Google Calendar or Microsoft Outlook Calendar, to schedule and organize meetings, appointments, and events. Users can view, create, and RSVP to events directly within the NMS interface, facilitating coordination and attendance tracking.
6. User Notifications Preferences: The NMS allows users to customize their notification preferences, including frequency, delivery channels, and types of notifications received. Users can opt-in or opt-out of specific notifications, set preferences for email digests, or configure notification settings based on their roles and responsibilities within the organization.
7. Multi-channel Communication: The NMS supports multi-channel communication, allowing users to interact via various communication channels, including email, messaging, notifications, and event calendars. Multi-channel communication ensures flexibility and accessibility, enabling users to choose the most convenient and effective communication channels for their needs.

By incorporating these communication interfaces, the NMS facilitates efficient, timely, and personalized communication among users, stakeholders, and external systems, fostering collaboration, engagement, and productivity within nonprofit organizations.

**3.5 Security Interfaces**

Security is paramount in the design and implementation of the NGO Management Software (NMS). The NMS incorporates various security interfaces and measures to safeguard sensitive data, protect against unauthorized access, and mitigate potential security threats. These security interfaces include:

1. Authentication Mechanisms: The NMS implements robust authentication mechanisms to verify the identity of users accessing the system. This may include username/password authentication, multi-factor authentication (MFA), biometric authentication, or integration with single sign-on (SSO) providers. Users must authenticate themselves before accessing sensitive data or performing privileged actions within the NMS.
2. Authorization Controls: The NMS employs granular authorization controls to enforce access restrictions based on users' roles, permissions, and privileges. Role-based access control (RBAC) or attribute-based access control (ABAC) mechanisms are used to ensure that users can only access the functionalities and data relevant to their responsibilities and requirements. Unauthorized access attempts are prevented through strict authorization policies and access restrictions.
3. Encryption Protocols: The NMS utilizes encryption protocols, such as SSL/TLS (Secure Sockets Layer/Transport Layer Security), to encrypt data transmitted over the network and protect it from eavesdropping or interception. Encryption ensures that sensitive information, including user credentials, financial transactions, and personal data, remains secure during transit between users' devices and the NMS servers.
4. Data Masking and Anonymization: The NMS employs data masking and anonymization techniques to protect sensitive data stored within the system. Personally identifiable information (PII), such as donor contact details or volunteer records, may be masked or anonymized to prevent unauthorized access or disclosure. Data masking ensures that only authorized users with appropriate permissions can access sensitive information while maintaining data privacy and confidentiality.
5. Security Auditing and Logging: The NMS logs security-relevant events, activities, and access attempts to facilitate security auditing and monitoring. Audit logs capture details such as user login/logout events, access to sensitive data, changes to user permissions, and security-related incidents. Security administrators can review audit logs to identify suspicious activities, detect potential security breaches, and enforce compliance with security policies and regulations.
6. Vulnerability Management: The NMS implements vulnerability management processes to identify, assess, and mitigate security vulnerabilities within the software and underlying infrastructure. Regular security assessments, code reviews, penetration testing, and vulnerability scanning are conducted to proactively identify and address potential security weaknesses. Vulnerabilities are promptly patched and remediated to maintain the integrity and security of the NMS.
7. Incident Response Procedures: The NMS establishes incident response procedures and protocols to handle security incidents, breaches, or data breaches effectively. Security incidents are reported, investigated, and remediated following predefined escalation procedures and response plans. Incident response teams are trained and equipped to respond promptly to security incidents, minimize impact, and restore normal operations.

By incorporating these security interfaces and measures, the NMS ensures the confidentiality, integrity, and availability of sensitive data, mitigates security risks, and maintains trust and confidence among users, stakeholders, and regulatory authorities.

# Nonfunctional Requirements

**4.1 Performance Requirements**

This section outlines various nonfunctional requirements that impact the system's performance, security, and usability, including specific standards and objectives.Performance Requirements.

**Reg1:** Response Time Requirement:

The NMS should respond to user interactions, such as data queries and form submissions, within 2 seconds to ensure a seamless user experience.

**Req2:** Throughput Requirement:

It must support a minimum of 100 simultaneous users during peak usage periods without experiencing performance degradation or bottlenecks.

**Req3:** Scalability Requirement:

The system should be capable of scaling horizontally and vertically to accommodate a growing number of users, volunteers, donors, and projects over time without significant performance degradation.

**Req4:** Availability Requirement:

High availability is essential, with the system uptime exceeding 99% to minimize downtime and ensure uninterrupted access to critical functionalities and data.

**Req5:** Reliability Requirement:

The NMS should demonstrate high reliability and stability in its operation, minimizing the occurrence of errors, crashes, or data corruption.

**Req6:** Resource Utilization Requirement:

It should optimize resource utilization, including CPU, memory, disk space, and network bandwidth, to maximize system performance and efficiency.

**Req7:** Load Testing Requirement:

Rigorous load testing should be conducted to validate the NMS's performance under expected and peak load conditions, ensuring it can handle user traffic effectively without compromising performance.

## 4.2 Safety Requirements

**Req1:** Data Security:

The NMS should implement robust data security measures to protect sensitive information, such as donor details, financial records, and personal data, from unauthorized access, breaches, or theft.

**Req2:** Privacy Protection:

The software must adhere to privacy regulations and best practices to safeguard users' privacy rights and prevent unauthorized disclosure or misuse of personal information.

**Req3:** Backup and Recovery:

Regular data backups and disaster recovery procedures should be in place to minimize data loss and ensure business continuity in the event of system failures, natural disasters, or cyberattacks.

**Req4:** Access Control:

Access to sensitive functionalities and data within the NMS should be restricted based on users' roles, permissions, and privileges to prevent unauthorized actions or data manipulation.

**Req5**: Audit Trails:

Comprehensive audit trails and logging mechanisms should be implemented to track user activities, system events, and data modifications for accountability, compliance, and forensic analysis purposes.

**Req6**: Training and Awareness:

Users should receive adequate training and awareness programs on security best practices, data protection policies, and incident response procedures to mitigate risks and maintain a secure computing environment.

**Req7**: Regulatory Compliance:

The NMS should comply with relevant regulatory requirements, industry standards, and contractual obligations related to data security, privacy, and confidentiality to avoid legal liabilities and penalties.

**Req8:** Incident Response:

The software should have predefined incident response procedures and protocols to handle security incidents, breaches, or data breaches effectively, minimizing the impact on users and stakeholders.

## 4.3 Security Requirements

**Req1:** Authentication Mechanisms:

The NMS must employ robust authentication mechanisms, such as username/password, multi-factor authentication (MFA), or biometric authentication, to verify the identity of users accessing the system.

**Req2:** Authorization Controls:

Access to sensitive functionalities and data within the NMS should be controlled through granular authorization mechanisms, such as role-based access control (RBAC) or attribute-based access control (ABAC), ensuring that users only have access to the resources necessary for their roles.

**Req3:** Data Encryption:

Sensitive data stored within the NMS, including donor information, financial records, and personal data, must be encrypted both at rest and in transit using strong encryption algorithms and protocols to prevent unauthorized access or disclosure.

**Req4:** Secure Communication:

The NMS should use secure communication protocols, such as SSL/TLS, for transmitting data over the network to protect against eavesdropping, tampering, or interception of sensitive information.

**Req5:** Security Patching:

Regular security patches and updates must be applied to the NMS software, underlying operating systems, libraries, and dependencies to address known vulnerabilities and mitigate the risk of exploitation by malicious actors.

**Req6:** Intrusion Detection and Prevention:

The NMS should be equipped with intrusion detection and prevention systems (IDPS) to monitor network traffic, detect suspicious activities or anomalies, and prevent unauthorized access or attacks in real-time.

**Req7:** Data Loss Prevention:

Measures should be implemented to prevent data loss or leakage, including access controls, data masking, encryption, and monitoring of data access and movement within the NMS..

**Req8:** Secure Coding Practices:

Developers should follow secure coding practices, such as input validation, output encoding, and parameterized queries, to prevent common security vulnerabilities such as injection attacks, cross-site scripting (XSS), and cross-site request forgery (CSRF).

**Req9:** Security Auditing and Logging:

The NMS should maintain comprehensive audit logs and logging mechanisms to record security-relevant events, user activities, and system changes for forensic analysis, compliance, and incident response purposes.

**Req10:** Incident Response Plan:

A documented incident response plan should be in place to outline procedures for detecting, responding to, and recovering from security incidents, breaches, or data breaches in a timely and effective manner.

**4.4 Efficiency Requirements**

ER1 - Optimized Resource Utilization:

The NMS should efficiently utilize computing resources, including CPU, memory, disk space, and network bandwidth, to minimize resource contention and maximize system performance.

ER2 - Fast Response Time:

The system must respond to user interactions, such as data queries and form submissions, swiftly to provide a responsive and seamless user experience.

ER3 - Minimal Latency:

The NMS should minimize latency in data processing, transaction execution, and communication to ensure real-time or near-real-time performance for critical operations.

ER4 - Scalability:

The software should be scalable to accommodate increasing user loads, data volumes, and concurrent transactions without significant degradation in performance or responsiveness.

ER5 - Transaction Throughput:

The system should support high transaction throughput to handle a large number of concurrent users and data transactions efficiently during peak usage periods.

ER6 - Batch Processing Efficiency:

Batch processing tasks, such as data imports, exports, and batch updates, should be executed efficiently within predefined timeframes to minimize processing delays and optimize system performance.

ER7 - Caching Mechanisms:

Caching mechanisms should be implemented to cache frequently accessed data, queries, or computations, reducing the need for repetitive processing and improving response times for subsequent requests.

ER8 - Database Optimization:

Database queries and operations should be optimized for efficiency, including indexing, query optimization, and data partitioning, to minimize query execution times and enhance overall system performance.

ER9 - Asynchronous Processing:

Long-running or resource-intensive tasks should be processed asynchronously to avoid blocking the user interface and ensure smooth user interactions while background tasks are executed.

ER10 - Load Balancing:

Load balancing mechanisms should be implemented to distribute user requests and workload evenly across multiple servers or instances, preventing overload on individual servers and ensuring optimal resource utilization.

## 4.5 Software Quality Attributes

* Reliability: The NMS should operate reliably under normal and exceptional conditions, ensuring consistent performance and minimal downtime to maintain user trust and satisfaction.
* Maintainability: The software should be designed with clean, modular, and well-documented code that facilitates ease of maintenance, updates, and enhancements over time, reducing maintenance costs and effort.
* Usability: The NMS should be user-friendly, intuitive, and easy to navigate, with clear and concise interfaces that minimize user errors and facilitate efficient task completion.
* Portability: The software should be portable across different hardware platforms, operating systems, and environments, enabling seamless deployment and interoperability in diverse computing environments.
* Scalability: The NMS should scale effectively to accommodate increasing user loads, data volumes, and system complexity without compromising performance, ensuring optimal responsiveness and resource utilization.
* Security: The software should adhere to industry-standard security practices and protocols to protect against unauthorized access, data breaches, and cyber threats, safeguarding sensitive information and maintaining regulatory compliance.
* Performance: The NMS should meet performance requirements, including response times, throughput, and resource utilization, to deliver a responsive, efficient, and scalable user experience under varying usage scenarios.
* Flexibility: The software should be flexible and adaptable to accommodate evolving user needs, organizational requirements, and technological advancements, supporting customization, configuration, and extensibility.
* Interoperability: The NMS should seamlessly integrate and interoperate with external systems, applications, and data sources through standard interfaces and protocols, facilitating data exchange and interoperability across the ecosystem.
* Testability: The software should be designed with testability in mind, with well-defined test cases, automated testing frameworks, and diagnostic tools that enable thorough testing and validation of system functionality, reliability, and performance.

By prioritizing these software quality attributes, the NMS can ensure high-quality, robust, and sustainable software that meets user expectations, enhances organizational effectiveness, and drives positive outcomes for nonprofit organizations.

## 4.6 Business Rules

* Donor Eligibility: Only individuals or organizations meeting predefined eligibility criteria, such as legal age, financial standing, or alignment with the NGO's mission, are eligible to make donations through the NMS.
* Volunteer Qualifications: Volunteers must meet specified qualifications, such as background checks, skills assessments, or training requirements, before they can participate in volunteer activities or projects coordinated through the NMS.
* Project Approval: Projects proposed by NGO staff or volunteers must undergo a review and approval process by authorized personnel before they are published or promoted through the NMS platform.
* Funding Allocation: Funds received through donations or grants are allocated to specific projects or initiatives based on predefined criteria, such as project budgets, funding priorities, or donor preferences, as managed within the NMS.
* Event Registration: Individuals or organizations interested in attending NGO events or activities must register through the NMS platform, providing necessary information and completing any required registrations or RSVPs.
* Communication Preferences: Donors, volunteers, and other stakeholders can specify their communication preferences within the NMS, indicating their preferred channels, frequency, and content types for receiving updates, newsletters, and notifications from the NGO.
* Data Privacy: The NMS adheres to data privacy regulations and organizational policies, ensuring that sensitive information collected from donors, volunteers, and other users is stored, processed, and protected in accordance with applicable laws and best practices.
* Financial Reporting: The NMS generates financial reports and statements, including donation summaries, expense reports, and balance sheets, to provide transparency and accountability regarding the NGO's financial activities and performance.
* Feedback and Evaluation: Users can provide feedback and evaluations for projects, events, and volunteer opportunities facilitated through the NMS, enabling continuous improvement and accountability within the organization.
* Conflict of Interest: NGO staff, volunteers, and board members are required to disclose any conflicts of interest related to their involvement in projects, fundraising activities, or decision-making processes facilitated through the NMS, as governed by organizational policies and regulations.

By adhering to these business rules, the NMS ensures transparency, accountability, and efficiency in managing donations, volunteers, projects, and events within the NGO ecosystem.

# 5.Functional Requirements

**5.1** **Usage Requirements**

**UR1** - User Registration: The system shall allow users to register accounts with unique usernames and passwords.

**UR2** - Login: Users shall be able to log in to their accounts using their credentials.

**UR3** - User Roles: The system shall support multiple user roles such as administrators, donors, volunteers, and staff, each with different access permissions and functionalities.

**UR4** - Profile Management: Users shall be able to update their profiles, including personal information, contact details, and communication preferences.

**UR5** - Donation Management: Donors shall be able to make one-time or recurring donations through the system, with options to specify donation amounts, payment methods, and allocation preferences.

**UR6** - Volunteer Registration: Prospective volunteers shall be able to register through the system by providing relevant information such as skills, availability, and areas of interest.

**UR7** - Project Creation: Authorized staff members shall be able to create new projects within the system, including project descriptions, goals, timelines, and resource requirements.

**UR8** - Volunteer Assignment: Staff members shall be able to assign volunteers to projects based on their skills, availability, and preferences.

**UR9** - Event Management: The system shall support the creation, promotion, and management of events, including registration, attendance tracking, and communication with attendees.

**UR10** - Communication Tools: The system shall provide communication tools such as email templates, messaging systems, and event calendars to facilitate communication among users, volunteers, donors, and staff.

**UR11** - Reporting: Administrators and staff members shall be able to generate reports on various aspects of NGO operations, including donor contributions, volunteer activities, project progress, and financial performance.

**UR12** - Analytics: The system shall provide analytics and data visualization tools to analyze trends, track key metrics, and gain insights into organizational performance and impact.

**UR13** - Accessibility: The system shall be accessible to users with disabilities, complying with accessibility standards such as WCAG (Web Content Accessibility Guidelines) to ensure equal access for all users.

**UR14** - Localization: The system shall support multiple languages and localization options to accommodate users from diverse linguistic and cultural backgrounds.

**UR15** - Documentation: The system shall provide comprehensive documentation, tutorials, and help resources to guide users in navigating the platform, performing tasks, and troubleshooting issues.

**5.2 Resource Requirements**

**RR1** - Hardware:

The system shall require hardware components including servers, storage devices, and network infrastructure to host and operate the NMS software. The hardware specifications should be capable of supporting the anticipated user load, data volume, and performance requirements of the system.

**RR2** - Software:

The system shall require software components such as operating systems, web servers, database management systems, and application frameworks to run the NMS software. The software versions and configurations should be compatible with the NMS requirements and deployment environment.

**RR3** - Database:

The system shall utilize a database management system (DBMS) to store and manage organizational data, including donor information, volunteer records, project details, and financial transactions. The database should be scalable, reliable, and secure to accommodate data growth and ensure data integrity.

**RR4** - Networking:

The system shall require networking infrastructure, including routers, switches, firewalls, and internet connectivity, to facilitate communication between users, servers, and external systems. The network should be reliable, secure, and capable of supporting the required data transfer rates and bandwidth demands of the NMS.

**RR5** - Human Resources:

The system shall require human resources including IT personnel, administrators, developers, and support staff to manage, maintain, and operate the NMS software. The personnel should possess the necessary skills, expertise, and training to effectively deploy, configure, and support the system.

**RR6** - Training and Documentation:

The system shall require training materials, documentation, and user guides to educate users, administrators, and staff on how to use the NMS software effectively. Training sessions and workshops may be conducted to familiarize users with the system functionalities and best practices.

**RR7** - Financial Resources:

The system shall require financial resources for procurement, licensing, maintenance, and support of hardware, software, and other resources necessary for the operation of the NMS. Budget allocations should be made for ongoing expenses such as hosting fees, software subscriptions, and personnel salaries.

**RR8** - Security Measures:

The system shall require security measures including firewalls, antivirus software, intrusion detection systems, encryption tools, and access controls to protect against cyber threats, data breaches, and unauthorized access. Investments in security infrastructure and services may be necessary to ensure the integrity and confidentiality of organizational data.

# System Models

**6.1 Scenarios**

**Use Case ID: 1**

Primary System Actor: Donor

Participating Actor: -

Definition:

This use case enables individuals interested in supporting the NGO's mission to register as donors within the system, thereby gaining access to donation functionalities and contributing to charitable causes.

Precondition:

The system must be operational and accessible to users.

Trigger:

When a potential donor decides to register within the system to contribute to NGO initiatives.

Basic Path:

* The donor navigates to the NGO's website or designated platform.
* The donor selects the "Sign Up" option, typically found within the "Sign In" or "Register" section.
* The donor completes the registration form, providing required personal information such as name, surname, email, and password.
* Upon completion, the donor submits the registration form by clicking the "Register" button.
* The system validates the provided information, ensuring accuracy and compliance with security standards.
* Once successfully verified, the donor receives confirmation of registration and gains access to the system.
* The registered donor can now log in to the system using their credentials.

Alternative Path:

Step 5: If the system detects any discrepancies or missing information in the registration form, an error message is displayed, prompting the donor to rectify the issues before resubmitting.

Exceptional Case:

Step 3: If the donor fails to provide all required information or inputs invalid data, the system rejects the registration attempt and prompts the donor to correct the errors.

Postcondition:

The donor is successfully registered within the system, granting access to donation features and functionalities.

Rules:

* Usernames must be unique within the system to avoid duplication and ensure user identification.
* Email addresses provided during registration must be accurate and valid for communication purposes.
* Passwords must adhere to security protocols, including minimum length and inclusion of uppercase letters, lowercase letters, numbers, and special characters, to enhance account security.

Explanation:

This use case facilitates the seamless registration of donors within the NGO's system, enabling individuals to contribute to charitable causes and support the organization's mission effectively.

**Use Case Name: Aid Request Submission**

**Use Case ID: 2**

Primary System Actor: Individual in Need

Participating Actor: -

Definition:

This use case enables individuals facing economic hardship to submit requests for assistance through the NGO's system, seeking support for their essential needs.

Precondition:

The system must be operational and accessible to users seeking aid.

Trigger:

When an individual experiencing financial difficulties decides to seek assistance through the NGO's system.

Basic Path:

* The individual in need accesses the NGO's website or designated platform.
* The individual selects the "Request Help" or similar option available on the platform.
* The user completes a form providing personal and economic information, including details of their financial situation and the assistance required.
* Upon completing the form, the user submits the aid request by clicking the "Send" or equivalent button.
* The system validates the submitted information, ensuring accuracy and completeness.
* Once verified, the system records the aid request, saving it for further processing and examination.
* The individual in need has the option to update or follow up on their aid request as needed.

Alternative Path:

Step 5: If the system identifies any inconsistencies or deficiencies in the submitted information, it displays an appropriate error message, prompting the user to review and correct the information before resubmitting the request.

Exceptional Case:

Step 3: If the user fails to provide all required information or inputs invalid data, the system rejects the aid request and notifies the user to rectify the errors.

Postcondition:

The aid request submitted by the individual in need is recorded within the system, awaiting examination and processing by relevant units within the NGO.

Rules:

* All information requested in the aid request form must be provided accurately and completely to facilitate the assessment process.
* The reason for seeking assistance and the amount of aid required must be clearly stated to enable efficient evaluation of the request.
* Each user should be allowed to submit only one aid request to prevent duplication and ensure fair distribution of resources.

Explanation:

* This use case streamlines the process for individuals experiencing financial hardship to seek assistance through the NGO's system, ensuring that their requests are accurately recorded and processed for potential support.

Use Case Name: Volunteer Application Review

**Use Case ID: 3**

**Primary System Actor: Administrator**

Participating Actor: Volunteer

Definition:

This use case facilitates the review and approval process of volunteer applications within the system, ensuring that all volunteer profiles are thoroughly assessed and approved by the administrator.

Precondition:

The system must be operational and accessible. Volunteer applications must be submitted for review.

Trigger:

When a volunteer registers in the system or updates their existing volunteer profile, prompting the need for administrator review.

Basic Path:

* The administrator accesses the system's administration panel using appropriate credentials.
* Within the administration panel, the administrator navigates to the section labeled "Volunteer Applications" or similar.
* The system displays a list of pending volunteer applications awaiting review.
* The administrator selects an application from the list for detailed review.
* The administrator thoroughly examines the volunteer application, ensuring all required information is provided and meets predefined criteria.
* If necessary, the administrator contacts the volunteer for additional information or clarification.
* Based on the assessment, the administrator either approves or rejects the volunteer application.
* The system records the administrator's decision and updates the volunteer's status accordingly.
* Upon approval, the volunteer's profile becomes active within the system, granting access to volunteer functionalities.

Alternative Path:

Step 5: During the application review process, the administrator may request additional information from the volunteer or reject the application if criteria are not met.

Exceptional Case:

Step 3: If there are no pending volunteer applications at the time of review, the system notifies the administrator accordingly.

Postcondition:

The volunteer application undergoes thorough review and is either approved or rejected by the administrator, with the volunteer's status updated accordingly within the system.

Rules:

* Volunteer applications must be evaluated based on objective criteria outlined by the organization to ensure fair and consistent assessment.
* All volunteer applications must contain complete and accurate information to facilitate the review process.
* Volunteers must be promptly notified of the decision regarding their application status.

Explanation:

This use case ensures that all volunteer applications submitted to the system are diligently reviewed and approved by the administrator, maintaining the integrity and effectiveness of the volunteer management process within the organization.

**Use Case Name: Donation Process**

**Use Case ID: 4**

Primary System Actor: Donor

Participating Actor: -

Definition:

This use case enables registered donors to contribute funds or resources to designated projects or organizations through the system.

Precondition:

The system must be operational and accessible. Donors must be registered users within the system.

Trigger:

When a registered donor decides to make a donation through the system.

Basic Path:

* The donor accesses the system's website and logs in using their credentials.
* Upon successful login, the donor navigates to the donation section and selects the "Donate" or similar option.
* The system presents the donation screen to the donor, providing options for selecting the donation area (e.g., education, healthcare, disaster relief).
* The donor chooses the specific project or organization they wish to support with their donation.
* The donor enters the donation amount, specifying whether it is a monetary contribution, materials, or time.
* If applicable, the donor indicates their shipping preference for physical items.
* The donor confirms the donation by clicking the "Donate" or equivalent button to finalize the transaction.
* The system records the donor's information, donation amount, and selected project or organization, completing the donation process.
* The donor can access their donation history and view the status of their contributions.

Alternative Path:

Step 4: If the donor fails to select a donation area or cancels the transaction, the donation process terminates.

Step 8: If the donor opts out of donating physical items, this step can be skipped.

Exceptional Case:

Step 6: If the donor enters an invalid donation amount or leaves required fields blank, the system rejects the donation and displays an error message.

Postcondition:

The donor's contribution is recorded within the system and directed to the designated project or organization. The donor has access to their donation history and can monitor the status of their contributions.

Rules:

* The donation amount must be a positive numerical value.
* Donors must select the area and project to which they wish to donate.
* Completion of the transaction requires filling in all required fields.

Explanation:

This use case facilitates the seamless process of donors logging into the system and making contributions, ensuring their donations are accurately recorded and directed to the intended beneficiaries.

**Use Case Name: Operation Coordination**

**Use Case ID: 5**

Primary System Actor: Coordinator

Participating Actor: -

Definition:

This use case empowers coordinators to effectively coordinate operations by assessing assistance requests within the system.

Precondition:

The system must be operational and accessible, and there must be active requests for assistance.

Trigger:

When a coordinator needs to strategize and organize operations or review incoming assistance requests.

Basic Path:

* The coordinator accesses the system's website and logs in using their credentials.
* Upon successful login, the coordinator navigates to the "Operations Planning" section or its equivalent.
* The system displays a list of current assistance requests to the coordinator.
* The coordinator selects a specific assistance request they wish to review or plan for.
* Reviewing the request details, the coordinator assesses the necessary resources required for the operation.
* The coordinator assembles or assigns teams to address the assistance request effectively.
* Creating the operation plan, the coordinator allocates the requisite resources for the task at hand.
* Determining the timeline for the operation, the coordinator initiates the implementation of the plan.

Alternative Path:

Step 3: If no current assistance requests are present or if the coordinator seeks to plan another operation, they can access a blank operation planning form.

Exceptional Case:

Step 5: If the coordinator encounters incomplete or unclear request details, they may request additional information or reject the request.

Step 7: In case of insufficient resources or impracticability of the plan, the coordinator may halt or revise the planning process.

Postcondition:

Upon completion of the operation planning, the system registers the operation plan and ensures the allocation of necessary resources. The coordinator sets the timeline for the operation and commences its implementation.

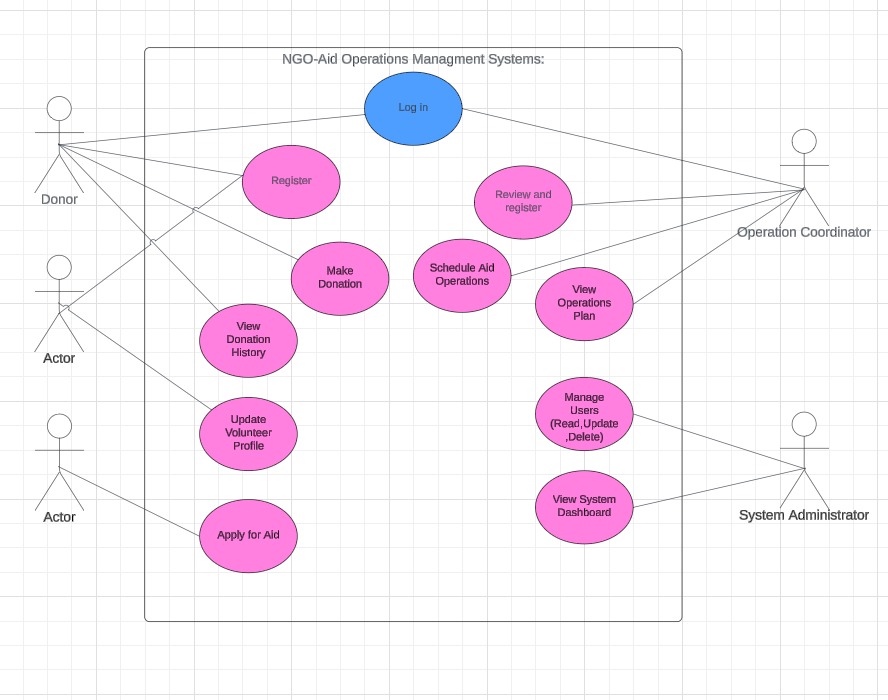
Rules:

* The operation plan must align with the specific details and resource availability associated with the assistance request.
* A clear timeline for the planned operation must be established and communicated to the relevant teams.
* Resources and team assignments during the operation must adhere to the defined operation plan.

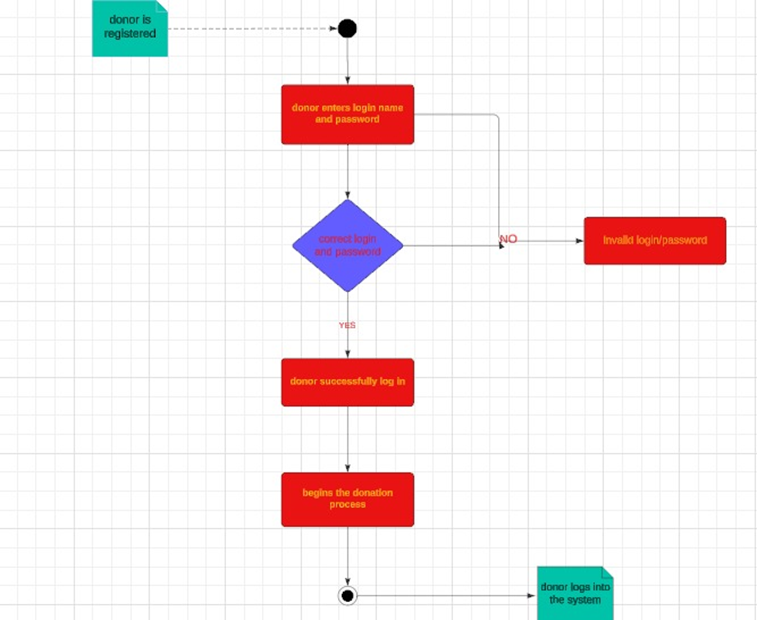
Explanation:

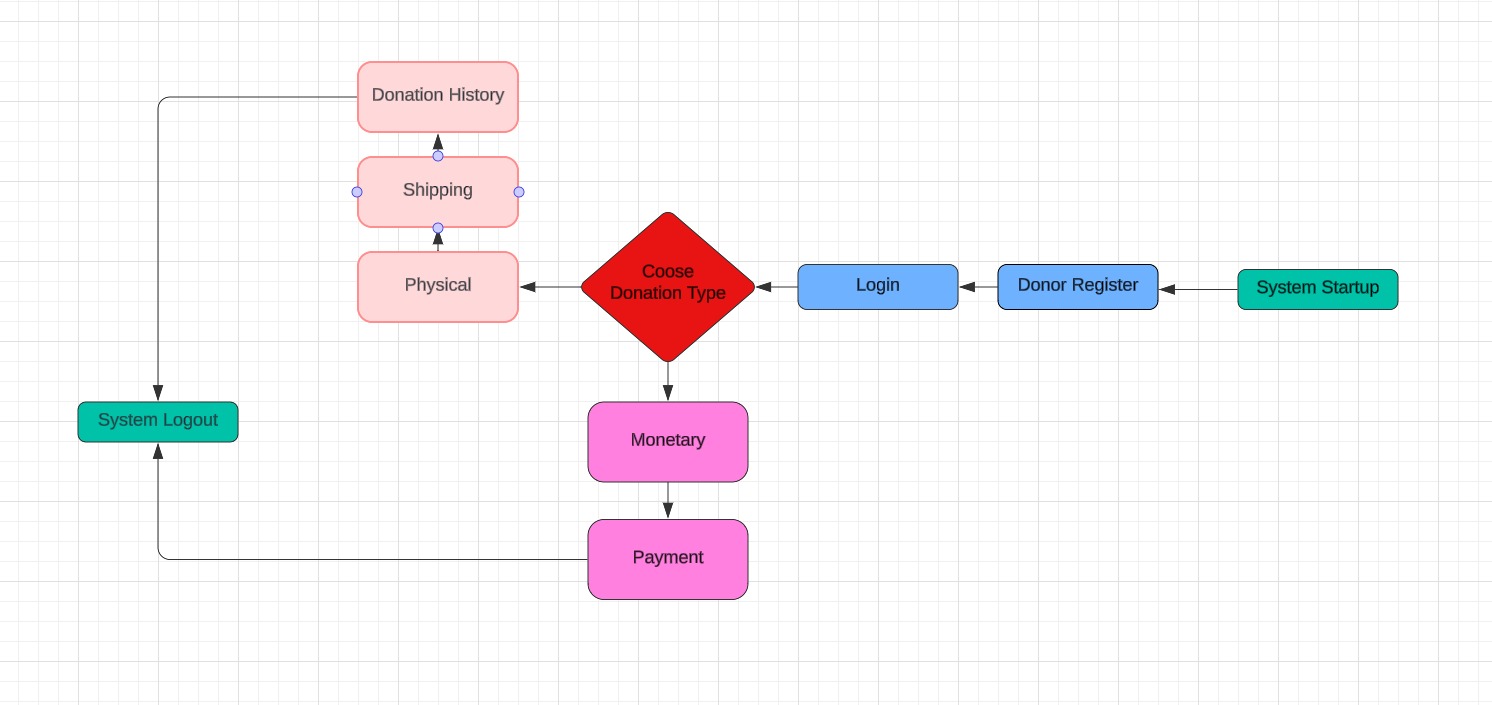
This use case facilitates the efficient coordination of operations by allowing coordinators to review and strategize based on incoming assistance requests within the system, ensuring timely and effective responses to the identified needs.

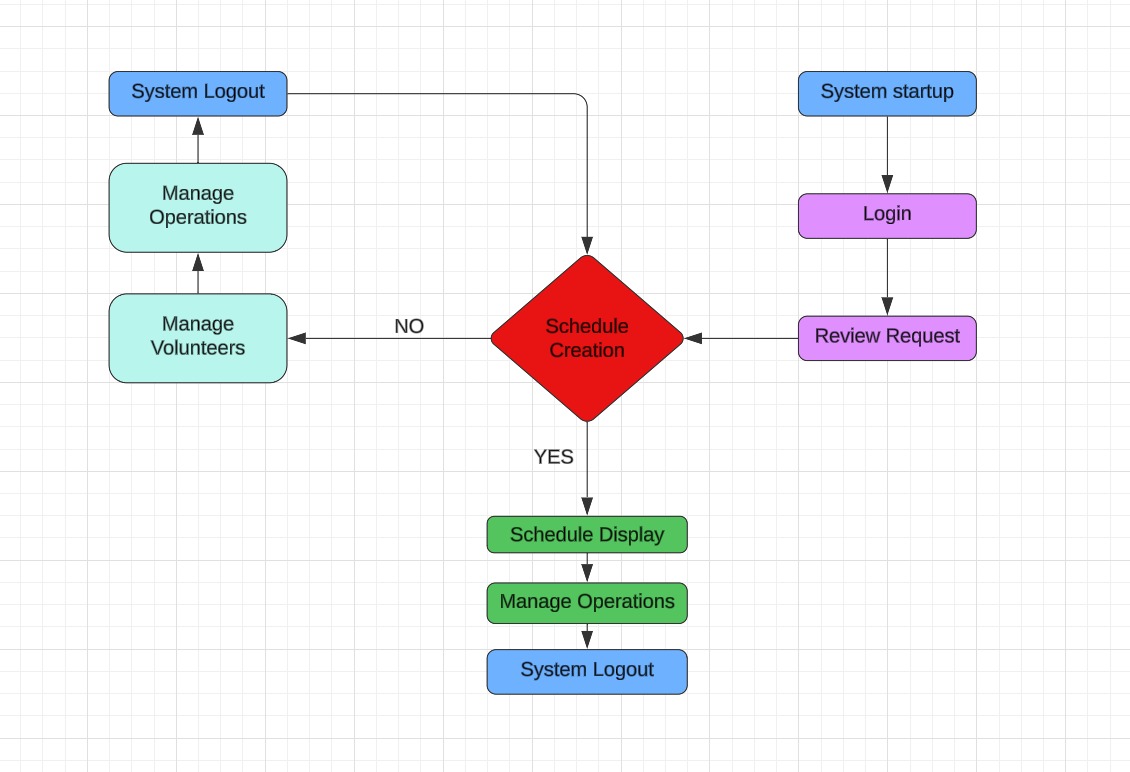
**6.2 Use Case Model**

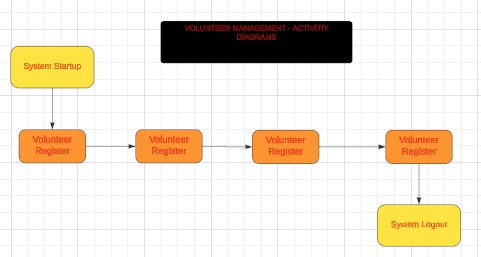


**6.3 Object Model**

****

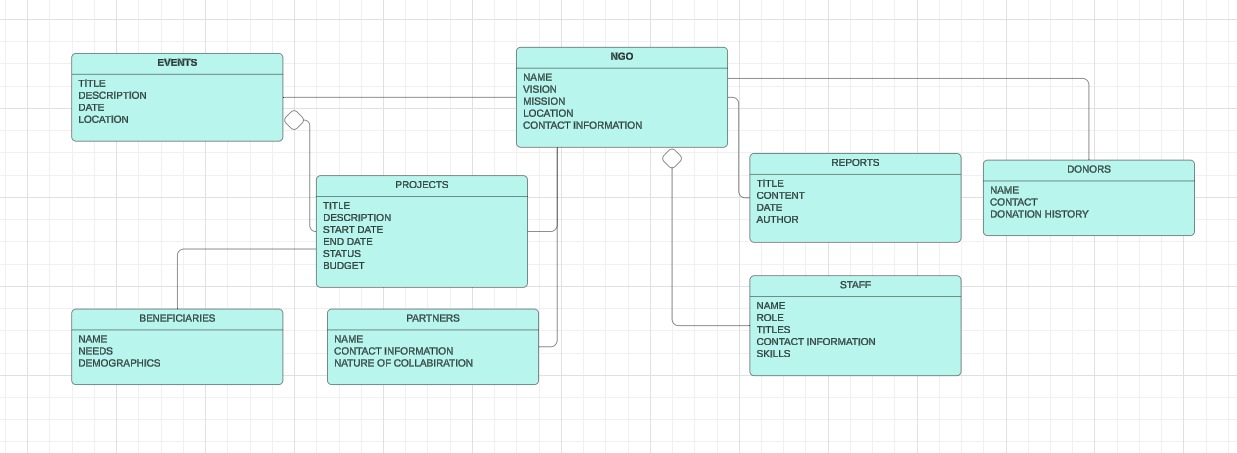




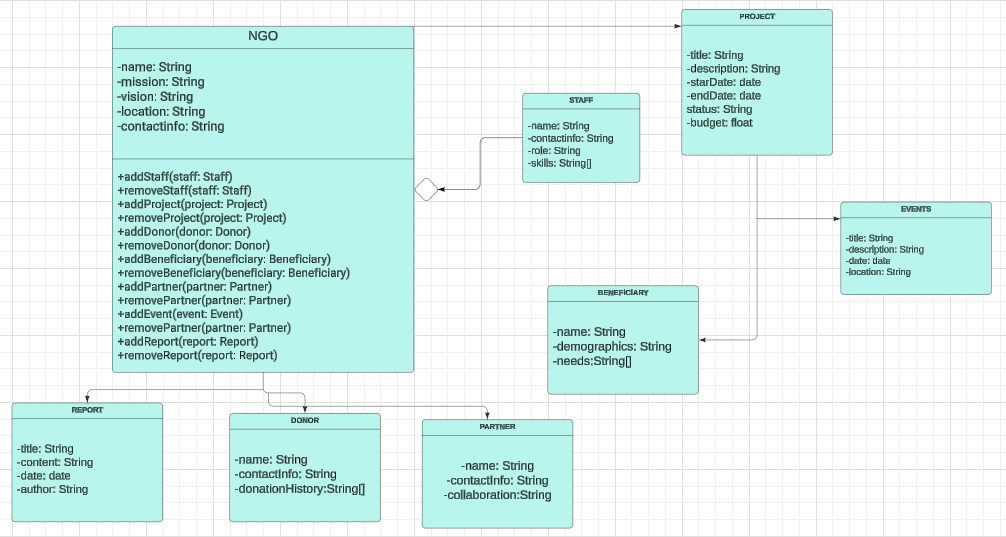


* 1. **Dynamic Models**

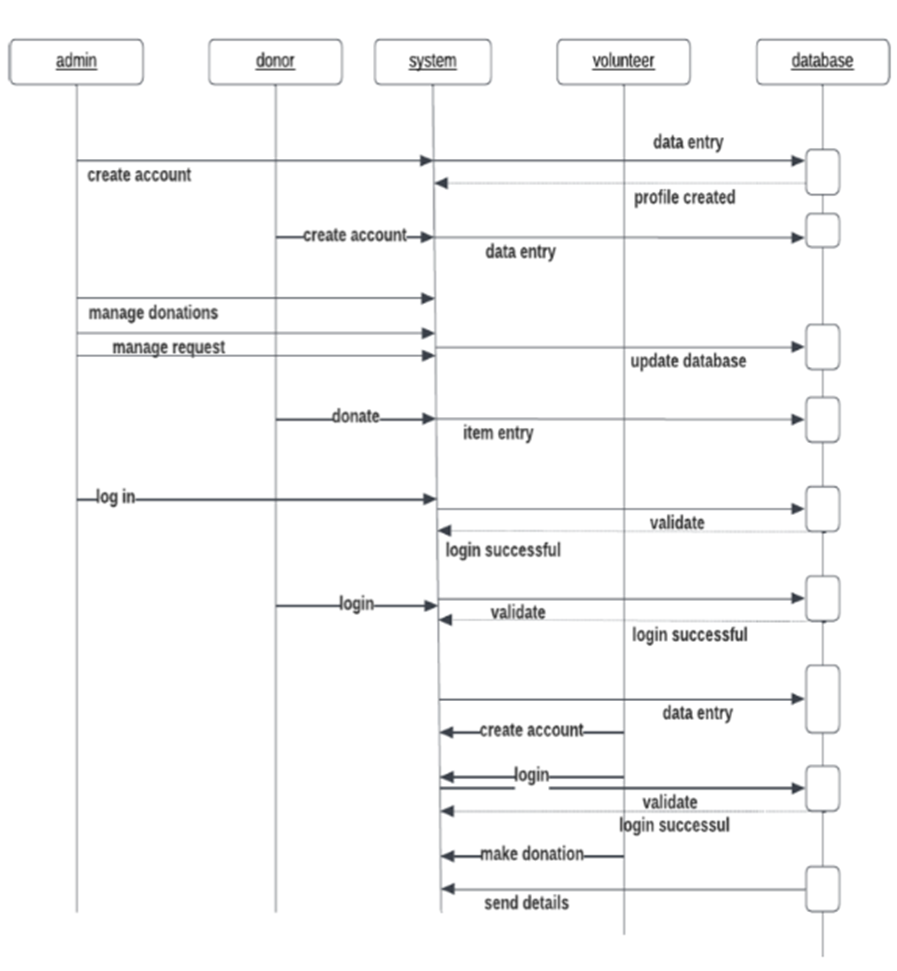
**Object Diagram**



**CLASS DIAGRAM**



* 1. **User Interfaces**

****

**<< Not Applicable>>**

Appendix A: Glossary

SRS: System Requirement Specification

UR: Usage Requirement

ER: Efficiency Requirement

RR: Resources Requirement