Analysis: FoodBridge

By: Simeon Markov

Institution: Fontys UAS

Course/Class: ICT/EN08

Date: 2025-10-13

Introduction

The goal of this document is to present the user requirements, mentioning possible user stories, going through functional, non-functional and technical requirements. It also contains the MoSCoW requirements prioritization method, used to ensure that core requirements are met, while less important ones (secondary) could be later realized (postponed).

Targeted users

Primary users

- _Staff managing inventory:_ Manages food supplies, donors.
- _Volunteer coordinator:_ Seeks scheduling shifts.

Secondary users

- _Volunteers:_ Checking their own schedule.
- _Donors:_ Viewing donations receipts.

Functional requirements

Admin dashboard

- *CRUD Operations (Create, Read, Update, Delete) for managing volunteers, inventory, donations and schedules.*
- *Real-time updates for changes in the state of the dashboard (e.g. new record added/modified).*
- *Data visualization for displaying charts, tables (analytics and insights).*
- *Administrative control for system management.*

Operational management

- *Staff can schedule volunteers in the dashboard.*
- *All data is stored and managed within the system.*
- *Communication and coordination happen through the platform.*

Analytics and insights

- *Charts and metrics update according to the changes in the operational data.*
- *Visualization of volunteers, staff schedules,* donations and trends.

Wants and needs

Ref.	**User Role**	**User Story**	**Business/Data Value**
US-1	**Staff Managing Inventory**	As a staff member, I want to add, delete and update food inventory so that stock levels remain accurate and usable.	Ensures efficient resource management and waste reduction
US-2		As a staff member, I want to link donations with donors so we can acknowledge them and maintain relations.	Improves donor engagement and reporting accuracy
US-3	**Administrator**	As an administrator, I want to add, edit users roles (assigning roles like coordinator, volunteer…) as well as adding, editing and deleting resources from the dashboard.	Having an extra level of authentication, administrative control.
US-4	**Volunteer Coordinator**	As a coordinator, I want to schedule volunteer shifts (create, update, retrieve, delete) so that all required activities are properly staffed.	Improves volunteer allocation and event coverage
US-5		As a coordinator, I want to communicate shift changes to volunteers to keep everyone informed.	Enhances volunteer participation and reduces no-shows
US-6	**Volunteers**	As a volunteer, I want to check my assigned shifts so I know when to report for work.	Increases volunteer reliability and satisfaction
US-7		As a volunteer, I want to update my contact info so coordinators can contact me easily.	Ensures up-to-date communication channels
US-8	**Donors**	As a donor, I want to view donation receipts so I can keep records for tax purposes.	Enhances donor trust and recordkeeping
US-9		As a donor, I want to see impact reports (meals provided, people helped) so I feel valued and motivated.	Strengthens donor relationship and encourages ongoing support

US-10		As a donor, I want to update my contact details to receive accurate communication and acknowledgments.	Maintains effective communication for fundraising

Table 1: User stories

Non-functional requirements

Performance

- During peak and off-peak hours, the system has to process the data without falling.

Scalability

- The architecture has to accommodate for future growth.

Usability

- The interface should be intuitive, ensuring admin and managers could easily navigate throughout the system.

Security

- User data must be encrypted. Protection against common vulnerabilities, access through security protocols.

Technical requirements

Frontend (Designing & Tools)

<u>*Web:*</u>

- React 18: Modern UI framework with hooks support.
- JavaScript.
- Tailwind CSS - Utility-first CSS framework for rapid UI development.
- Recharts - Declarative charting library built specifically for React, perfect for data visualization.

Backend (Software & Infrastructure)

- <u>*Languages:*</u> C# & JavaScript.

- *Web:* ASP.NET Core 8 (latest maintainable, supported version) - Modern, high-performance framework for building APIs.
- *Database:* SQL Server - Enterprise-grade relational database, integrates seamlessly with C#.

Prioritization

For the requirements prioritization, the MoSCoW technique was used.

MoSCoW technique

Must have	US-1, US-2, US-3, US-4
Should have	US-5, US-6, US-7
Could have	US-9, US-10
Will/Wish have	US-8

Table 3: MoSCoW requirements prioritization

References

- Abhay Talreja (2025). What is a User Story in Agile? Definition, Examples & Template. https://teachingagile.com/agile/user-story/what-is-user-story.
- Functional and Nonfunctional Requirements of Inventory Management Systems. https://www.nexstrideconsulting.com/insights-and-trends/functional-nonfunctional-inventory-requirements.
- Refine. https://refine.dev/blog/recharts/#introduction.
- Jaydeep Patil. Product Management Application using .NET Core and React JS with CRUD Operations. https://www.c-sharpcorner.com/article/product-management-application-using-net-core-and-react-js-with-crud-operations/
- Hitesh Jethva. C# SQL Server Connection: 3 Easy Methods. https://hevodata.com/learn/c-sql-server/#m1
- AI transperancy: PerplexityAI for extensive in-depth research, summarizing articles, papers.