Validation document: FoodBridge

By: Simeon Markov

Institution Name: Fontys UAS

Course/Class: ICT/EN08

Date: 2025-12-06

Introduction

This document presents the last stage of development of the project. It includes the sections test plan (doing validation from the perspective of the user on different features mentioned in the analysis document under ‘functional requirements’).

Validation

This section presents the test plan, where is evaluated if certain features have passed the validation.

Feature	Brief Description	Passed
Managing GRUD operation on products, donations, waste resources by authorized user	Admin could create, edit, delete a resource.	Passed
Link donations to donors with their products.	Authorized users could manage the relationships between donors and donation items.	Currently only the administrator could do this action.
History of changes	Database admin could see the performed operations like when, who did what at what time (Audit trace).	Passed
Dashboard overview	The dynamic dashboard fetches the data from the database and display it in different forms of charts.	Passed

Table 1: Features table validation

Validation of non-functional requirements

1. Performance
Requirement

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

Current State Analysis
PARTIALLY MET - Foundation Set, Testing Required

Already Implemented:

☑ Async/Await Pattern: Backend services utilize asynchronous programming to handle concurrent requests efficiently without blocking threads.

☑ Vite Build Tool: Frontend utilizes Vite for optimized bundling and faster initial load times compared to traditional CRA.

☑ Loading States: UI implements loading indicators to manage user perception during data processing.

☑ Clean Architecture: Separation of concerns prevents logic bottlenecks.

2. Scalability
Requirement

The architecture has to accommodate for future growth.

Current State Analysis
MET (Functionally) - Architecture supports growth

Already Implemented:

☑ Interface-Based Services: Use of IAuthenticationService allows for easy swapping of implementations or mocking for testing.

☑ EF Core + SQL Server: Enterprise-grade database foundation capable of handling significant data growth.

☑ Stateless Authentication: JWT implementation is stateless, making it easier to scale the backend horizontally (add more servers) without sticky sessions.

☑ Separation of Concerns: Backend logic is decoupled from controllers, making refactoring into microservices easier in the future.

3. Usability
Requirement

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

Current State Analysis
MET - Modern UX Best Practices Applied

Already Implemented:

☑ Feedback Systems: Validations, error messages, and loading states provide immediate system feedback to users.

☑ Visual Hierarchy: "Support for black, white theme design" and "Quick actions grid" guide user attention to key tasks.

☑ Responsive Layout: Dashboard and sidebar adapt to screen sizes, ensuring usability across devices.

☑ Routing: react-router-dom ensures logical navigation flows (Login -> Dashboard).

4. Security
Requirement

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

Current State Analysis
MET - Strong Standards Implemented

Already Implemented:

☑ Encryption At Rest: Passwords are hashed using ASP.NET Core Identity.

☑ Encryption In Transit: HTTPS enforcement ensures data is encrypted over the network.

☑ Access Control: JWT Token authentication (HMAC SHA256) with role-based authorization readiness.

☑ Vulnerability Protection: CORS policy configured, account lockout protection enabled, and secure password requirements enforced.

Summary Table

Requirement	**Status**	**Priority**	**Effort to be improved**
Performance(Process data w/o falling)	Partial	High	2-4 weeks (Caching/Testing)
Scalability(Architecture for growth)	Met (Functionally)	High	2-4 weeks (Containerization)
Usability(Intuitive interface)	Met	Medium	2-4 weeks (UAT & Tweaks)
Security(Encryption & Protocols)	Met	Critical	2-4 weeks (Rate limiting)

Table 2: Summary table of met non-functional requirements

References

- Perplexity AI (Research AI): Used for Suggesting possible ways of improving, the non-technical requirements concerning the application's overall performance.