TQS: Product specification report

Nuno Afonso Fontes Vieira [107283]
v2026-01-28

1 Introdução

1.1 Visão geral do projeto

Este projeto foi devolvido no âmbito da unidade curricular **Tecnologias e Qualidade de Software (TQS)**, tendo como principal objetivo aplicar conceitos e boas práticas de engenharia de software, nomeadamente especificação de requisitos, desenho arquitetural, testes automatizados, integração contínua e garantia de qualidade.

O trabalho consiste no desenvolvimento de uma aplicação web completa, com backend e frontend, suportada por uma API REST, permitindo explorar práticas de desenvolvimento orientadas a testes, controlo de qualidade e avaliação contínua do software.

1.2 Limitações e problemas conhecidos

Apesar do projeto cumprir os principais objetivos definidos para o desenvolvimento do produto e da estratégia de garantia de qualidade, existem algumas limitações e aspetos que não foram totalmente implementados. Estas limitações resultam principalmente de falta de tempo e do facto de se tratar de um projeto individual e não em grupo.

Algumas das principais limitações são a ausência de mecanismos de recomendação com base em competências e disponibilidade, de dashboards públicos para métricas, de um deployment em Docker e a ausência de testes de aceitação e de carga mais compreensivos.

1.3 Referências e recursos

O desenvolvimento do projeto recorreu a um conjunto de ferramentas, bibliotecas e recursos que foram fundamentais para a implementação da aplicação e da estratégia de garantia de qualidade. A nível das tecnologias usadas, destacam-se o Spring Boot como framework principal para o backend, JUnit 5, Mockito e AssertJ para testes unitários e Selenium WebDriver para testes de aceitação automatizados.

No contexto de qualidade e integração contínua, foram utilizadas ferramentas de CI/CD integradas com o repositório Git, bem como ferramentas de análise estática de código e cobertura de testes (SonarQube), que permitiram acompanhar métricas de qualidade ao longo do desenvolvimento. A documentação oficial destas ferramentas, bem como materiais disponibilizados na unidade curricular de TQS, constituíram as principais referências técnicas.

2 Conceito do produto e requisitos

2.1 Visão

A plataforma tem como objetivo promover o envolvimento cívico, reforçar os laços comunitários e valorizar o contributo dos membros da comunidade académica através de um sistema estruturado de reconhecimento e recompensas. O sistema é concebido como um marketplace digital de voluntariado, facilitando a ligação entre entidades promotoras e indivíduos interessados em participar em atividades de impacto social.

Neste contexto, os promotores são responsáveis pela criação e divulgação de oportunidades de voluntariado, que podem ser propostas por serviços, departamentos ou núcleos de estudantes, desde que estejam enquadradas na missão e atuação da Universidade de Aveiro. Estas oportunidades incluem, entre outras, apoio à organização de eventos académicos e conferências, participação em sessões de mentoria, acolhimento de visitas institucionais, atividades de responsabilidade social e projetos de extensão comunitária, nomeadamente em colaboração com entidades externas como a Câmara Municipal.

Os voluntários são maioritariamente estudantes da Universidade de Aveiro, podendo igualmente incluir docentes, investigadores e outros colaboradores que pretendam contribuir ativamente para a comunidade académica e local. A plataforma procura, assim, criar um

ecossistema inclusivo que incentive a participação e reconheça o esforço individual, promovendo uma cultura de solidariedade e envolvimento cívico sustentável.

2.2 Personas e cenários

Persona 1 – João Silva (Voluntário)

- Idade: 21 anos
- Curso: Engenharia Informática
- Objetivo: Participar em atividades extracurriculares e obter reconhecimento académico
- Frustrações: Falta de visibilidade das oportunidades existentes

Persona 2 – Marta Costa (Promotora)

- Idade: 38 anos
- Função: Coordenadora de projeto académico
- Objetivo: Recrutar voluntários de forma simples e organizada
- Frustrações: Gestão manual de candidaturas e comunicação dispersa

Cenário – Voluntário
João consulta a plataforma Opportunity4UA, encontra uma oportunidade de apoio a um evento académico, candidata-se e acompanha o estado da sua candidatura. Após concluir a atividade, recebe pontos que pode trocar por recompensas institucionais.

Cenário – Promotora
Marta cria uma nova oportunidade de voluntariado, analisa as candidaturas recebidas, aceita voluntários e, após a conclusão do evento, confirma a participação, atribuindo automaticamente os pontos correspondentes.

2.3 Épicos e prioridades

O desenvolvimento do **Opportunity4UA** foi organizado em torno de um conjunto de **epics funcionais**, permitindo uma implementação incremental e estruturada da solução ao longo de várias iterações. Cada epic agrega funcionalidades relacionadas e foi priorizado de acordo com o valor entregue ao utilizador e com as dependências técnicas entre funcionalidades.

2.3.1 Épico 1 – User Management

Este epic estabelece a base do sistema, sendo essencial para todos os restantes.
Abrange a gestão de utilizadores e autenticação, distinguindo os diferentes papéis existentes na plataforma.
Principais funcionalidades:

- Registo e autenticação de utilizadores
- Identificação de papéis (Voluntário e Promotor)

- Gestão de perfis de utilizador
- Atualização de informações específicas por papel (ex.: competências do voluntário)

Sem este epic, não é possível utilizar qualquer funcionalidade da plataforma.

2.3.2 Épico 2 – Opportunity Management

Este epic é responsável pela criação e gestão das oportunidades de voluntariado por parte dos promotores.
Principais funcionalidades:

- Criação de oportunidades de voluntariado
- Definição de datas, número de vagas, pontos e requisitos
- Listagem de oportunidades criadas por um promotor
- Encerramento de oportunidades para novas candidaturas

É um epic central para permitir a existência de oportunidades no sistema.

2.3.3 Épico 3 – Opportunity Discovery

Este epic foca-se na experiência do voluntário ao procurar oportunidades disponíveis, garantindo facilidade de acesso e visibilidade.
Principais funcionalidades:

- Listagem de oportunidades abertas
- Visualização detalhada de uma oportunidade
- Filtros básicos (estado, disponibilidade, competências requeridas)
- Navegação intuitiva entre oportunidades

Depende da existência de oportunidades, mas é essencial para maximizar a participação dos voluntários.

2.3.4 Épico 4 – Opportunity Participation

Este epic cobre todo o ciclo de vida da participação de um voluntário numa oportunidade, desde a candidatura até à conclusão.
Principais funcionalidades:

- Submissão de candidaturas a oportunidades
- Aceitação e rejeição de candidaturas por promotores
- Acompanhamento do estado das candidaturas
- Confirmação da conclusão da participação
- Atualização automática do estado da oportunidade

Representa o núcleo funcional da plataforma, ligando voluntários e promotores de forma efetiva.

2.3.5 Épico 5 – Rewards

Este epic introduz mecanismos de motivação e reconhecimento, recompensando a participação ativa dos voluntários.
Principais funcionalidades:

- Atribuição de pontos após a conclusão de oportunidades
- Visualização do saldo de pontos do voluntário
- Catálogo de recompensas disponíveis
- Resgate de pontos por itens institucionais
- Histórico de recompensas resgatadas

Embora não seja essencial para o funcionamento básico do sistema, aumenta significativamente o valor e o envolvimento dos utilizadores.

3 Modelo de domínio

O sistema **Opportunity4UA** gere um conjunto de entidades centrais que suportam todo o ciclo de vida das oportunidades de voluntariado, a participação dos utilizadores e a gestão de recompensas.

A entidade central é o **User**, que representa todos os atores autenticados do sistema. Um utilizador pode assumir diferentes papéis (Volunteer ou Promoter). Os voluntários possuem informação de perfil como competências, disponibilidade e saldo de pontos, enquanto os promotores estão associados à criação e gestão de oportunidades. Cada utilizador pode ainda ter múltiplos **Tokens**, utilizados para suportar autenticação baseada em sessões.

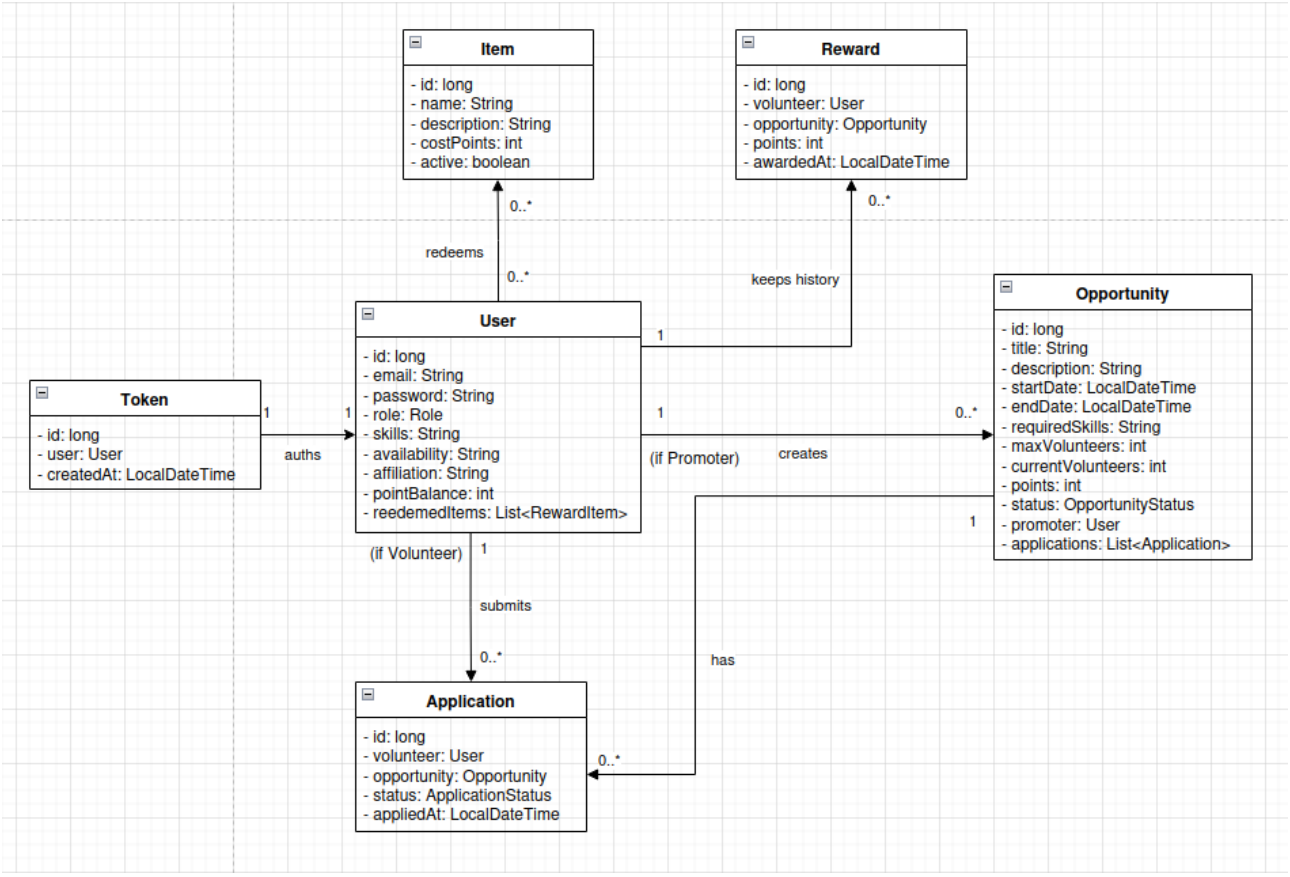
Uma **Opportunity** representa uma atividade de voluntariado criada por um promotor. Inclui informação descritiva (título, descrição, competências requeridas), dados temporais, restrições de capacidade e um estado que indica se a oportunidade está aberta, fechada ou concluída. Cada oportunidade está associada a exatamente um promotor e pode receber múltiplas **Applications**.

Uma **Application** modela o pedido de participação submetido por um voluntário a uma oportunidade. Estabelece a ligação entre um voluntário e uma oportunidade e acompanha o estado da candidatura (pendente, aceite, rejeitada ou concluída), bem como a data de submissão. Uma oportunidade pode ter várias candidaturas, mas cada candidatura refere-se a uma única oportunidade e a um único voluntário.

Quando um voluntário conclui com sucesso uma candidatura aceite, é gerada uma **Reward**. A recompensa regista o número de pontos atribuídos, a oportunidade associada, o voluntário e a data de atribuição, permitindo ao sistema manter um histórico completo de recompensas por utilizador.

O sistema suporta ainda a troca de pontos através das entidades **Item**. Os itens de recompensa representam benefícios resgatáveis (por exemplo, serviços, descontos ou merchandising institucional), cada um com um custo em pontos e um estado de disponibilidade. Um utilizador pode resgatar vários itens ao longo do tempo, resultando numa relação muitos-para-muitos entre utilizadores e itens de recompensa, que é mantida como parte do histórico de resgates do utilizador.

O diagrama de classes UML abaixo define as entidades e as respetivas relações, constituindo assim a base para os fluxos principais da plataforma.



4 Notebook de arquitetura

4.1 Requisitos chave e restrições

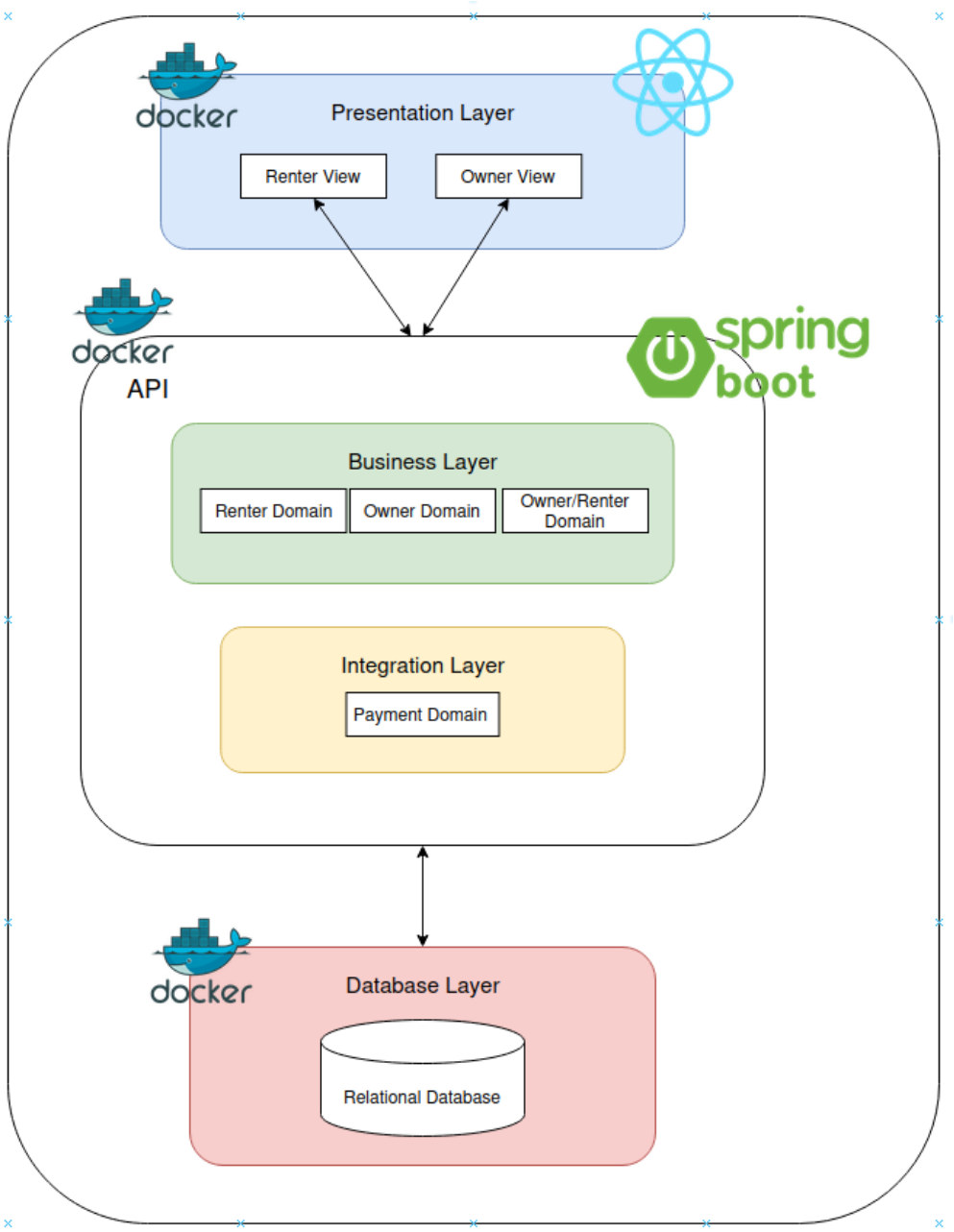
A definição da arquitetura do sistema foi orientada por um conjunto de requisitos chave e restrições que influenciaram diretamente as decisões de desenho e implementação.

Um dos requisitos fundamentais foi a **separação clara de responsabilidades entre componentes**, de forma a facilitar a manutenção, a testabilidade e a evolução futura do sistema. Esta necessidade conduziu à adoção de uma arquitetura em camadas, distinguindo claramente a camada de apresentação, a camada de controlo, a lógica de negócio e a camada de persistência. Esta separação permite isolar alterações numa camada sem impactar diretamente as restantes, contribuindo para a robustez da solução.

Do ponto de vista dos utilizadores, a arquitetura teve de acomodar diferentes perfis e papéis, como voluntários e promotores, garantindo controlo de acessos e comportamentos distintos consoante o tipo de utilizador. Esta característica é central para o domínio da aplicação e influencia tanto a lógica de negócio como os mecanismos de autenticação e autorização.

Em termos de extensibilidade, a arquitetura foi concebida de forma a permitir a adição futura de novas funcionalidades, como novos tipos de recompensas, métricas de participação ou algoritmos de recomendação, sem necessidade de alterações profundas à estrutura existente. Esta preocupação com a extensibilidade justifica a modularização do código e a utilização de abstrações claras entre componentes.

4.2 Arquitetura



5 API para developers

A aplicação disponibiliza uma **API REST** que permite a interação entre o frontend e o backend. A API foi concebida de forma a expor as principais entidades do domínio como recursos, utilizando os métodos HTTP padrão (GET, POST, PUT, DELETE) para a sua manipulação.

A API encontra-se organizada sob o prefixo comum /api e utiliza **autenticação baseada em tokens**, enviados em cada pedido através de um cabeçalho HTTP dedicado. Este mecanismo garante que apenas utilizadores autenticados possam aceder a recursos protegidos, com validação adicional baseada no papel do utilizador (voluntário ou promotor).

Os principais recursos expostos pela API incluem:

- **Users** (/users) — gestão de utilizadores, autenticação e atualização de perfis;
- **Opportunities** (/opportunity) — criação, consulta e gestão de oportunidades de voluntariado;
- **Applications** (/application) — submissão e gestão de candidaturas às oportunidades;
- **Rewards** (/rewards) — atribuição de recompensas associadas à conclusão de oportunidades;

A documentação detalhada da API, incluindo a descrição completa dos endpoints, parâmetros, exemplos de pedidos e respostas, encontra-se disponível através de uma interface **Swagger** gerada automaticamente a partir do código da aplicação. Esta documentação é utilizada como referência principal para o desenvolvimento e integração com a API.