TQS: Product specification report

Roberto Rolão de Castro [107133], Mariana Figueiredo Perna [108067], Tiago Caridade Gomes [108307], Rafaela Espírito Santo Dias [108782]

v2024-05-07

[This report should be written as the main source of technical documentation on the project, clarifying the functional scope and architectural choices. Provide concise, but informative content, allowing other software engineers to understand the product and quickly access the related resources.
Tips on the expected content, along the document, are meant to be removed.
You may use English or Portuguese; do not mix.]

1 Introduction

1.1 Overview of the project

<contextualize the objectives of this project assignment in the scope of the TQS course>
<introduce your application/product: brief overview of the solution. What is it good for? Introduce the name of the product if it has one>

The objective of this project within the Test and Software Quality (TQS) course is to synthesize quality assurance practices with agile software development to create a functional software solution. We are tasked with developing an application that not only serves a practical purpose but also embodies the principles of robust software architecture, clean coding practices, and rigorous testing methodologies. This aligns with the TQS course's aim to instill a mindset where quality is not an afterthought but a pervasive aspect of the software development lifecycle.

Our application, CliniConnect, is designed to revolutionize the patient experience in scheduling and managing healthcare appointments. At its core, CliniConnect is a digital platform that serves as a nexus between patients and healthcare providers, offering features such as appointment scheduling, patient registration, and real-time queue management. Its purpose is to enhance service delivery within healthcare facilities by optimizing appointment booking processes, reducing wait times, and facilitating efficient patient flow and billing post-consultation. CliniConnect stands out by providing a comprehensive, intuitive, and accessible tool for healthcare interaction, laying the groundwork for future expansions like mobile app integration and self-service check-in functionalities.

1.2 Limitations

<explain the known limitations/unimplemented (but planned) features>

2 Product concept and requirements

2.1 Vision statement

<functional (black-box) description of the application: what will you system be used for? Which is the high-level/business problem being solved by your system?>
<if needed, clarify what was planned/expected to be included but was changed to a different approach/concept >
<optional: how is your system different or similar to other well-known products?>
<optional: you may include a UML Use Case diagram to support the explanation>
<optional: additional details on the process for the requirements gathering and selection (how did we developed the concept? Who helped us with the requirements? etc)>

Our system is designed to streamline the management of patient appointments and admissions across a network of hospitals in Portugal. It will be used by patients who want to make, see, or cancel doctors' appointments, by healthcare professionals to schedule and manage consultations, and by administrative staff to oversee and facilitate the overall patient flow from check-in to billing.

2.2 Personas and scenarios

<Uma Persona é uma personagem utilizada para contar histórias representativas da futura utilização do sistema. Uma Persona é um Actor instanciado, à qual se dá um conjunto de caraterísticas para a humanizar e definir o contexto em que usará o sistema e as suas motivações.
"Personas are fictional people. They have names, likenesses, clothes, occupations, families, friends, pets, possessions, and so forth. They have age, gender, ethnicity, educational achievement, and socioeconomic status. They have life stories, goals and tasks. Scenarios can be constructed around personas, but the personas come first. They are not 'agents' or 'actors' in a script, they are people. Photographs of the personas and their workplaces are created and displayed. […] It is to obtain a more powerful level of identification and engagement that enable design, development, and testing to move forward more effectively". Adapted from Grudin, J. and Pruitt, J., 2002, June. Personas, participatory design and product development: An infrastructure for engagement. In Proc. PDC (Vol. 2).

Exemplo: ver secção 4.1, neste artigo (open access)] >

<You don't need to include all possible details. Pick the main scenarios, related to the core value of the system.>
<The scenarios tell the story of the Personas in their lives, doing their daily/professional activities that are relevant to find the points of contact with the system under specification.
Scenarios are somewhat similar to use cases (they have a goal and tell a story), but, unlike use cases, they capture a larger process, with activities that may not use the software. Scenarios don't required a "template", like the usual use cases description.>
Exemplo: ver secção 4.2 neste artigo (open access)] >

Persona 1: Mariana Perna

Mariana Perna, a 20-year-old university student studying environmental science, has been dealing with significant myopia since childhood, necessitating annual ophthalmology check-ups. She is a tech-savvy individual who relies heavily on her smartphone for managing her busy academic and social life. Mariana values the convenience of digital solutions that allow her to maintain her health without disrupting her schedule.

Scenario: Routine Ophthalmology Appointment Booking Mariana receives a reminder notification from the CliniConnect app about scheduling her annual ophthalmology appointment. Within minutes, using her smartphone, she accesses the app to view Dr. Dionisio Cortesão's availability at Hospital da Luz in Aveiro. She selects a slot that fits between her lectures and finalizes the booking. Mariana appreciates the seamless experience and the reassurance that her eye health is being monitored regularly without any hassle.

Persona 2: Adelaide Bicho

At 53, Adelaide Bicho is an experienced receptionist at Trofa Saúde Hospital in Porto. She prides herself on her efficiency and her ability to provide a warm and welcoming environment for patients. Adelaide is the first point of contact for many visitors, and she takes her role seriously, understanding that she sets the tone for their healthcare experience.

Scenario: Managing Priority Queue As the day begins, Adelaide logs into the CliniConnect system at her reception desk, checking the day's schedule and patient appointments. She calls the next priority queue number, D43, which belongs to an elderly patient requiring assistance. With CliniConnect's intuitive interface, Adelaide quickly accesses the patient's details, verifies their priority status due to mobility issues, and smoothly directs them to the appropriate department, ensuring they receive prompt care.

Persona 3: Bernardo Oliveira

Bernardo Oliveira is a vibrant 5-year-old starting kindergarten. Accompanied by his mother, Sofia Oliveira, a 35-year-old architect, they both value timely and effective healthcare services. Sofia, as a working professional and a mother, looks for healthcare solutions that minimize waiting time and provide a stress-free experience for Bernardo's healthcare needs.

Scenario: Pediatric Appointment Check-In Bernardo, holding his mother's hand, enters the pediatric wing of their local clinic for a routine check-up. Sofia uses the self-service kiosk to check in, selecting the normal queue option on the CliniConnect system. They are immediately issued ticket number N26. Bernardo is fascinated by the digital signage that will soon display his number, and Sofia appreciates that the system respects their time, allowing for a smooth and orderly visit.

2.3 Project epics and priorities

[Apresentar um plano indicativo para a implementação incremental da solução ao longo de várias iterações/releases, explicando as funcionalidades a atingir por *epics*]

The implementation of CliniConnect will be carried out incrementally across several iterations, with each epic representing a broad category of functionality crucial to the application's operation. Below are the defined epics, each composed of several user stories that contribute to a cohesive set of functionalities to be developed and refined in successive releases.

Epic 1: Appointment Management System

User Story 1.1: As Mariana Perna, I want to receive timely notifications to schedule my annual ophthalmology check-up, so that I can maintain my eye health with minimal disruption to my schedule.

User Story 1.2: As a patient, I want to view and select available time slots for appointments, so that I can book appointments at my convenience.

User Story 1.3: As a patient, I want to confirm and cancel appointments through the app, ensuring flexibility in managing my schedule.

User Story 1.4: As a receptionist, I want to view the daily appointment schedule, so that I can effectively manage patient flow.

Epic 2: Queue Management and Prioritization

User Story 2.1: As Adelaide Bicho, I want to efficiently manage priority queues, ensuring that patients with urgent needs receive timely care.

User Story 2.2: As a receptionist, I want to quickly access patient details and verify their priority status, ensuring that I can direct them to the appropriate care promptly.

User Story 2.3: As a patient or caregiver, I want to take a queue number through a self-service kiosk, minimizing wait time and streamlining check-in procedures.

Epic 3: Patient-Doctor Matching

User Story 3.1: As a patient, I want to search for doctors by specialty, location, and availability, so that I can find the right healthcare provider for my needs.

User Story 3.2: As a doctor, I want to manage my availability on the platform, ensuring that patients can book appointments with me based on my real-time schedule.

Epic 4: Notification and Alerts System

User Story 4.1: As a patient, I want to receive reminders for upcoming appointments, ensuring that I don't miss them.

User Story 4.2: As a clinic staff, I want to be alerted about no-shows or cancellations, so I can adjust the schedule accordingly and optimize the use of resources.

Epic 5: User Experience and Accessibility

User Story 5.1: As a young patient, I want an engaging and understandable display system that notifies me when it's my turn, making the waiting experience less daunting.

User Story 5.2: As a working parent, I want to easily check in my child for appointments using a system that respects our time and needs for a stress-free experience.

User Story 5.3: As a patient with accessibility needs, I want an application interface that is easy to navigate and use, regardless of my technical abilities.

Implementation Plan

The project will be structured in a series of sprints, each designed to deliver a working increment of the system:

Sprint 1: Focus on setting up the basic appointment booking and notification functionality, targeting user stories from Epics 1 and 4.

Sprint 2: Develop the queue management features and integrate priority handling as specified in Epic 2.

Sprint 3: Enhance the patient-doctor matching features, with emphasis on search functionality and real-time availability updates from Epic 3.

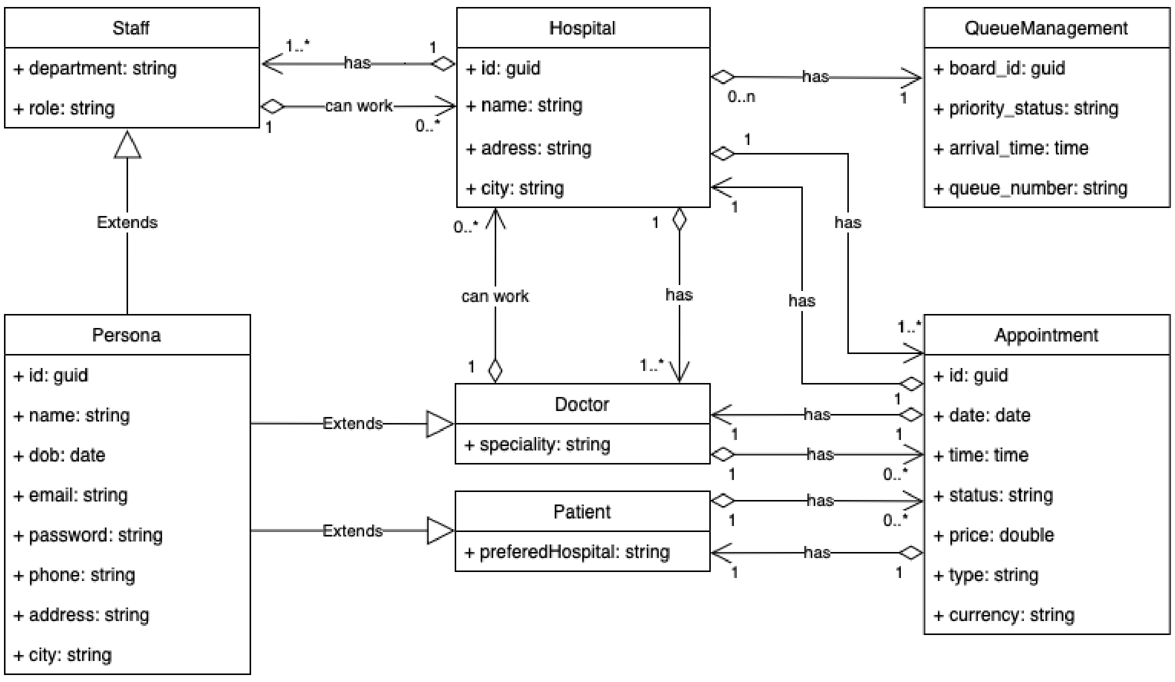
Sprint 4 and beyond: Continuous refinement of user experience, accessibility features from Epic 5, and additional functionalities based on feedback and user testing.

3 Domain model

<which information concepts will be managed in this domain? How are they related?>
<use a logical model (UML classes) to explain the concepts of the domain and their attributes>

The class diagram represents a hospital management system with seven entities:

- ***Persona*** is the base class for individuals, containing personal information such as ID, name, date of birth, and contact details.
- ***Staff*** extends ***Persona*** and includes additional attributes for department and function, indicating the function taken in the hospital. Staff can work in multiple hospitals.
- ***Doctor*** also extends ***Persona***, with a specific attribute for specialty. Doctors can work at multiple hospitals and have a one-to-many relationship with ***Patient***.
- ***Patient*** extends ***Persona*** with a preference for a particular hospital.
- ***Appointment*** is associated with ***Patient*** and ***Doctors*** and contains details about the appointment such as ID, date, time, status, as well as financial attributes like price and currency.
- ***Hospital*** is a separate entity with its own attributes like ID, name, address, and city. It has associations with *Staff*, ***Doctor***, and ***QueueManagement***.
- QueueManagement is linked to Hospital and manages patient queues with attributes such as board ID, priority status, arrival time, and queue number.



4 Architecture notebook

4.1 Key requirements and constrains

<**Identify issues that will drive the choices for the architecture** such as: Will the system be driven by complex deployment concerns, adapting to legacy systems, or performance issues? Does it need to be robust for long-term maintenance?
Identify critical issues that must be addressed by the architecture, such as: Are there hardware dependencies that should be isolated from the rest of the system? Does the system need to function efficiently under unusual conditions? Are there integrations with external systems? Is the system to be offered in different user-interfacing platforms (web, mobile devices, big screens,…)?
E.g.: (the references cited in [XX] would be hypothetical links to previous specification documents/deliverables)

There are some key requirements and system constraints that have a significant bearing on the architecture. They are:

- The existing legacy Course Catalog System at Wylie College must be accessed to retrieve all course information for the current semester. The C-Registration System must support the data formats and DBMS of the legacy Course Catalog System [E2].
- The existing legacy Billing System at Wylie College must be interfaced with to support billing of students. This interface is defined in the Course Billing Interface Specification [E1].
- All student, professor, and Registrar functionality must be available from both local campus PCs and remote PCs with internet dial up connections.
- The C-Registration System must ensure complete protection of data from unauthorized access. All remote accesses are subject to user identification and password control.
- The C-Registration System will be implemented as a client-server system. The client portion resides on PCs and the server portion must operate on the Wylie College UNIX Server. [E2]

- All performance and loading requirements, as stipulated in the Vision Document [E2] and the Supplementary Specification [15], must be taken into consideration as the architecture is being developed.>

4.2 Architecture view

→ Discuss architecture planned for the software solution.

→ include a diagram (a package or block diagram)

→ explain how the identified modules will interact. Use sequence diagrams to clarify the interactions along time, when needed

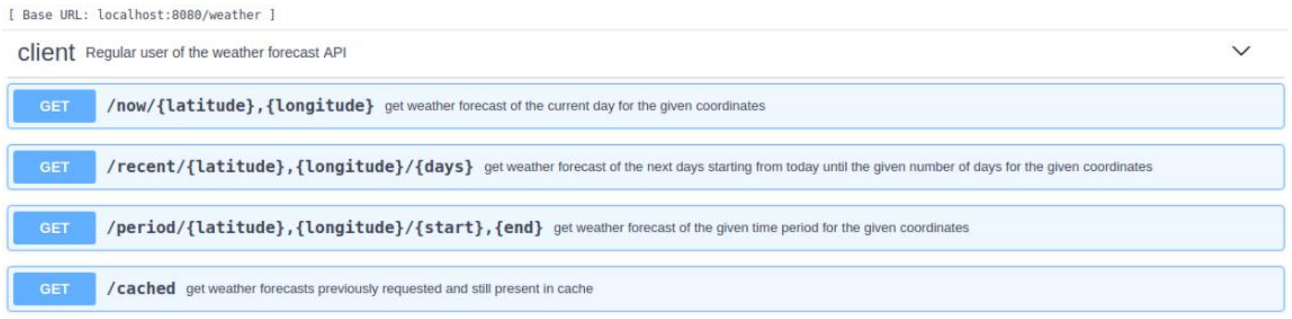
→ discuss more advanced app design issues: integration with Internet-based external services, data synchronization strategy, distributed workflows, push notifications mechanism, distribution of updates to distributed devices, etc.>

4.3 Deployment architecture

[Explicar a organização prevista da solução em termos configuração de produção (*deployment*). Anotar, no diagrama, as tecnologias de implementação, e.g.: colo aro simbolo do PostgreSQL na Base de dados,…]

5 API for developers

[Explicar a organização da API. Os detalhes detalhes/documentação dos métodos devem ficar numa solução *hosted* de documentação de APIs, como o Swagger, Postman documentation, ou incluída no próprio desenvolvimento (e.g.: maven site)
<what services/resources can a developer obtain from your REST-API?>
<document the support endpoints>



[Base URL: localhost:8080/weather]

client Regular user of the weather forecast API

GET `/now/{latitude},{longitude}` get weather forecast of the current day for the given coordinates

GET `/recent/{latitude},{longitude}/{days}` get weather forecast of the next days starting from today until the given number of days for the given coordinates

GET `/period/{latitude},{longitude}/{start},{end}` get weather forecast of the given time period for the given coordinates

GET `/cached` get weather forecasts previously requested and still present in cache

6 References and resources

<document the key components (e.g.: libraries, web services) or key references (e.g.: blog post) used that were really helpful and certainly would help other students pursuing a similar work>