TQS: Product specification report

***Pedro Ramos [107348], João Luís [107403]*, Daniel *Madureira [107603], Rodrigo Aguiar [108969]***

v2024-04-19

[**1**](#_heading=h.gjdgxs) **Introduction 1**

[1.1](#_heading=h.30j0zll) Overview of the project 1

[1.2](#_heading=h.1fob9te) Limitations 1

[**2**](#_heading=h.3znysh7) **Product concept 2**

[2.1](#_heading=h.2et92p0) Vision statement 2

[2.2](#_heading=h.tyjcwt) Personas 2

[2.3](#_heading=h.35nkun2) Main scenarios 2

[2.4](#_heading=h.3dy6vkm) Project epics and priorities 2

[**3**](#_heading=h.1t3h5sf) **Domain model 2**

[**4**](#_heading=h.4d34og8) **Architecture notebook 3**

[4.1](#_heading=h.2s8eyo1) Key requirements and constraints 3

[4.2](#_heading=h.17dp8vu) Architetural view 3

[4.3](#_heading=h.3rdcrjn) Deployment architeture 4

[**5**](#_heading=h.26in1rg) **API for developers 4**

[**6**](#_heading=h.lnxbz9) **References and resources 4**

# Introduction

## 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 is to propose, conceptualize and implement a multi-layer application, using a software architecture based on *enterprise* *frameworks*. Also apply a *Software Quality Assurance* (SQA) strategy, and integrate CI/CD on our project.

Therefore, **PhiHub** is going to be developed to enhance patient management at their upcoming hospital facilities. The system will be able to schedule new appointments for patients, and also check their “agenda” and access summaries of past consultations. Patients will also be able to follow the process of calling for appointments via screens.

The staff will be able to check-in and handle registrations from patients and handle administrative tasks.

## Limitations

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

# Product concept and requirements

## 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?>

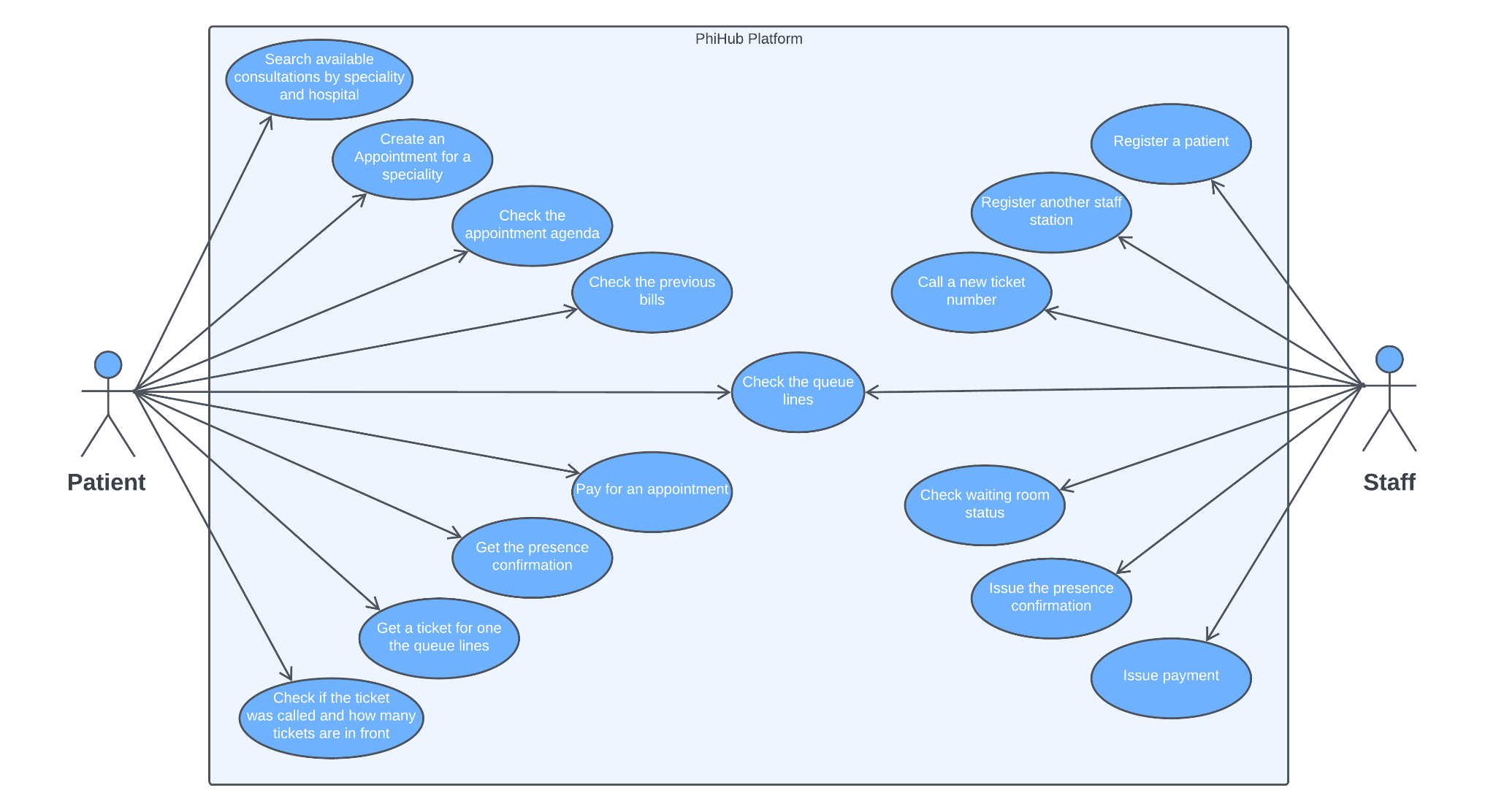
<if needed, clarify what was planned/expected to be included but was changed to a different approach/concept >

<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 aim with PhiHub is to develop a system where patients, technical staff and doctors can associate appointments, bills, tickets and queues, as to simulate the entire workflow between registering an appointment, waiting for the appointment inside the hospital and paying for said appointment;
* This project tries to resolve the complex problem of organizing and maintaining a good flow of patients as to minimize wait time and mistakes from the staff;
* Our system will be similar to several others already in use by hospitals around Portugal, but while these systems usually only focus on one of the three main functionalities (setting up appointments, managing line tickets and schedule all the appointments for the staff), our system will provide all of these functionalities;
* In summary, we want to include the complete workflows of searching and setting up an appointment, waiting for your ticket to be called, paying for the appointment and issue the bills and presence confirmation.



## 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](http://www.mdpi.com/1424-8220/18/4/1285) (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](http://www.mdpi.com/1424-8220/18/4/1285) (open access)] >

**Personas:**

1. **Emily Smith**

Is 32 years old and suffers from a chronic condition. Therefore, Emily has to go to the hospital frequently and is looking for a way to schedule/manage their appointments via digital platforms and to check her appointment history.

Emily lives in a rural region so is often forced to drive long distances to go to the hospital, many times having to face long waiting queues,often reaching more than 12h, sometimes not even getting an appointment.

**Motivation:**

Emily’s bad experiences with hospital check-ins and appointments are making her look for an easier and more in-hand way of scheduling her appointments while at home, through her smartphone or computer.

Emily is also looking for a way to check her registered appointments so she doesn’t forget about them.

1. **Tom Desk**

Tom is 43 years old and works at Hospital reception. He feels constantly frustrated about his work because sometimes the reception area is full of patients and he can’t do the check-in process as quickly as he'd like. He would appreciate a system that could resolve this problem and also guarantee that patients are being called by order.

Tom is also constantly getting phone calls from customers trying to make appointments, often leading to being stressed during the work days.

**Motivation:**

Tom is looking for a digital system that can facilitate his job, making sure that patients know when it's their turn to do the check-in process and automating the process of calling patients to appointments.

Tom would like a digital system where patients can register their appointments, without having to contact the reception directly, that is easy to manage.

1. **Lisa Moose**

Lisa is 51 years old and works as an administrator of a Hospital. She finds it hard to know if the current working protocols are efficient, both for patients and the hospital workers, often having to read through various files from the hospital areas.

**Motivation:**

Lisa would like a centralized system where she could review the hospital statistics, such as consultations or appointments done by each medic, the number of total consultations done in a day or how many patients visited the hospital that day.

**Main Scenarios:**

**Scenario 1:**

Emily wakes up feeling an unusual discomfort which she suspects could be related to her chronic condition. She uses the PhiHub to search for an available consultation with a specialist in her area. After finding a suitable time slot, she books the appointment and receives confirmation. On the day of her appointment, she uses the digital signage portal to self-check-in upon arrival at the hospital.

**Scenario 2:**

Tom starts his shift at the reception and logs into the PhiHub. He reviews the list of patients scheduled for the day and prepares the necessary paperwork. As patients arrive, he checks them in using the system, updating their status so that the screen at the reception area displays current waiting times and queue numbers. Tom uses the system to manage priority cases, ensuring they are attended to promptly. He also handles payment transactions and issues presence confirmations after consultations.

**Scenario 3:**

During the flu season, it is very common for Tom to encounter a mix of regular appointments and walk-ins. This way, Tom uses PhiHub to manage the flux of patients, prioritizing appointments, but also fitting walk-ins by viewing no-shows and last minute cancellations.

**Scenario 4:**

Lisa reviews the various logs from the hospital areas on PhiHub and uses it to train the hospital staff and further implement new work protocols so the hospital is able to serve more patients while ensuring the staff isn’t overworked.

## 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*](https://www.atlassian.com/agile/project-management/epics-stories-themes) ]

**Epic 1: Schedule/manage appointments**

User Story #1: As Emily, I want to be able to schedule, cancel and check appointments remotely without the need for actually going to the hospital;

**Epic 2: Manage patients check-ins**

User Story #5: As Tom, I want to be able to register new patients quickly as to optimize the time used in checking in first time patients;

User Story #6: As Tom, I want regular patients to be able to check-in by themselves so that I can help new patients.

**Epic 3: Manage administrative personnel**

User Story #10: As Lisa, I want to be able to easily check the day-to-day statistics of patients that pass through the hospital, as well as the average amount of time that a patient spent waiting to be attended to.

User Story #11: As Lisa, I want to be able to quickly check how many patients per hour are being attended to, as to optimize the working hours of the staff.

**Epic 4: Real-Time monitoring of waiting lines and prioritization of patients**

User Story #7: As Tom, I want to check which lines are the longest and how long the estimated queue time is so that I can prioritize certain queues for a better patient experience and time management.

User Story #8: As Tom, I want the system to automatically call in a patient from the queue line with the most priority or the most wait time, as to optimize patient satisfaction;

**Epic 5: Self-check-in feature via mobile when arriving at the hospital**

User Story #2: As Emily, I want to be able to check-in for my consults after arriving in the hospital from my mobile phone, to circumvent having to wait in a queue for the check-in;

**Epic 6: Check agenda, processed bills and presence confirmations**

User Story #3: As Emily, I want to be able to check all my previous bills so I can know exactly how much money I am spending with the visits;

User Story #4: As Emily, I want to be able to check my current agenda so I will not miss my next appointments;

User Story #12: As Lisa, I want to be able to check how many bills are passed and how much money is generated by the hospital.

**Epic 7: Call and manage tickets**

User Story #9: As Tom, I want the system to quickly show the patients which ticket is being called, as to waste as little time as possible in the calling process;

# 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>

# Architecture notebook

## Key requirements and constraints

<**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.>

## 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.>

## Deployment architecture

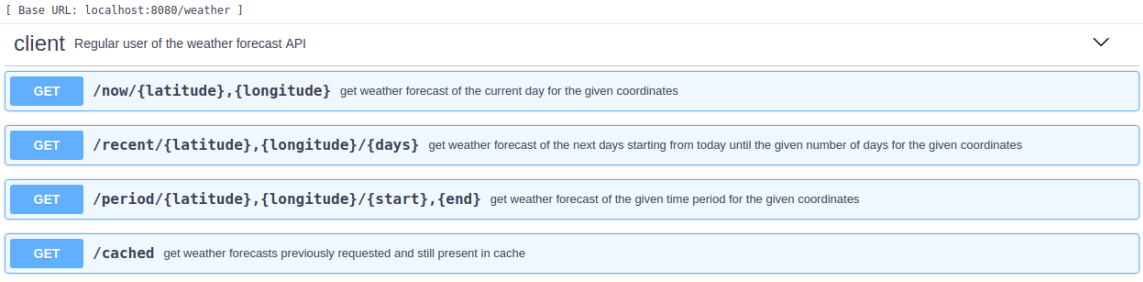
[Explicar a organização prevista da solução em termos configuração de produção (*deployment*). Anotar, no diagrama, as tecnologias de implementação, e.g.: colo aro simbolo do PostgreSQL na Base de dados,…]

# 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](https://swagger.io/), 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>



# 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>