universidade de aveiro deti departamento de eletrónica, telecomunicações e informática

CoworkClub

5 Jun 2022

*Final project for the subject of **System Analysis**, class **P6**.*

Team

Daniel Capitão	75943
David Araújo	93444
Filipe Costa	77548
Samuel Teixeira	103325

1. Introduction

1.1 Executive Summary

In this report we focus on the results of the *Elaboration* and *Development* stages. This is divided into two sections: Architecture validation and Functionality implementation.

When validating the architecture we focused on picking the most important components for a user to accomplish a satisfactory simple interaction like booking a space. Since the system will be constructed with Vue JS, component reusability is also very efficient which leads to easy recognizance by the user. It also contributes to a very high level of compatibility across devices. This is important as **CoworkClub** is expected to be accessible globally and is not dependent on end-user device specificities.

Implementing further functionalities will consist in implementing capabilities and tools that increase the simplicity and speed of processes, on the user side; and increase robustness and reliability on the back office side of the processes, this includes the creation of shortcuts, more control over data by the user and increased quality in the visual aspect in order to improve appeal and UX.

1.2 Version Control

Date	Editor	Changes
7 Jun 2022	David Araújo	Introduction
7 Jun 2022	Samuel Teixeira	System Architecture
9 Jun 2022	Daniel Capitão	System Architecture and Use Cases
10 Jun 2022	David Araújo	Software architecture diagram Physiscal architecture diagram Stratagie amd Estate of implementation

1.3 References & Supplementary Resources

- Refine the Architecture

2. System Architecture

2.1 General Objectives

Our architecture will be done by prioritizing the user's characteristics. **CoworkClub** will have the following specifications:

- The System shall give quick notifications if something happens to a Booking. For example, if the Coworking space gets damaged to the point it's simply not usable, **CoworkClub** should warn the user about this change and provide help for repayment.
- Since **CoworkClub** is going to be a PWA, it will work in very similar ways in Desktop or Mobile, which makes the usage of the service very convenient.
- The system must be capable of making payments with various methods of online payment, such as MBWay, Paypal, etc.
- The System must not accept more than one booking for the same time frame. If that is the case, every Booking after the first one in these circumstances will not be allowed.
- Reviews of a certain Coworking space will only be allowed if the user has made at least one Booking of that specific space

2.2 Requirements with impact on the architecture

Requirements	**Description**
RD-1	User-friendly interface with the integrated payment platform.
RD-2	The system will be able to support up to 250 users and will have cross-platform support.
RS-1	Secure storage and payment platform.
RU-1	Results based on user location integrable with calendar applications.
RS-1	Login operations should rely on Firebase services and social login
RU-2	Re-use of components should be preferred in order for the user to recollect the various components' functions.
RU-3	The user can search without any input or with incomplete input, the system, in this case, must assume default values. This is done so a user that does not know what is his/her ideal space, can still browse through the available ones
RD-3	A minimalistic design, and resorting to industry-standard keywords and

Requirements	Description
	icons must be used to accomplish a more user-friendly environment.

2.3 Decisions and Justifications

- TeleportHQ is our choice for front-end websites and UI elements developments. It comes with a visual builder and allows us to create our own custom functionalities with integrated front-end development tools.
- VueJS (JavaScript framework) for building user interfaces and single-page applications is ideal for modularization and fast development.
- ExpressJS for back-end web application framework, free and open-source software with built-in libraries for HTTP and advanced routing mechanisms.
- Jira for project management purposes allows us to track user stories' stage of development.

2.4 Software Architecture

Our implementation will be subdivided into three main domains, these being: User Interfaces, Logical Domains, and Infrastructure. A package diagram can be found in the Annexe section.

In the UI section, we highlight the two main packages: website and *progressive web application*. Our solutions intend to implement both in order to increase compatibility across devices and encourage "persistence" with the user by enabling it to keep an application of our system locally.

In the logical domain level, we focus on what is the core domains of our platform. User are comprised of two types, workers and providers, which can overlap. Also we have spaces, and spaces can have their own attributes, but most importantly, a collection of office, that can be of different types. Its this spaces that can be subject to booking.

2.5 Physical Architecture

For our physical infrastructure, we envision a simple implementation. Since the application is mostly rendered on the side of the user, there is no need for a highly capable server system.

For it to give location based results, we would need access to the devices gps capabilities, or if in a non mobile system, access to the WiFi information in order to estimate a location. On the other end, the server will only need be run an API and a database, probably a JSON base one. A physical diagram can be found in the Annexe section.

3. First Increment

3.1 Use Cases on the First Increment

Story/Use Case Slice	Acceptance Criteria
Joaquim Searches for a Coworking Space Joaquim needs a Coworking Space for his online meetings, so he decides to use CoworkClub to search for it	**1º Scenario: Searches for an existing name** Joaquim opens the app and searches for a Coworking Space he knows. When Joaquim presses the search button, the System finds out that there is a Coworking Space with that name and gives that result to him **2º Scenario: Searches for a non-existing name** Joaquim opens the app and, by mistake, searches “askfdskz”. As he presses the search button of mere curiosity, the System doesn’t find any Coworking Spaces/Geographical regions, and gives no results to him **3º Scenario: Searches with filtering** Joaquim doesn’t know a specific Coworking Space, but he knows what he wants in one. So, he uses the System’s filters to filter by region and private room with free Wi-Fi and Coffee. After pressing the search button, the System finds out 2 options for Joaquim and gives to him as a result.
Ana wants to make a booking for a Coworking Space After finding a Coworking Space that fits her, Ana wants to	Financial institution. Its goal at **CoworkClub** is to help the process of payment of Coworking Spaces by the Coworkers. The Bank can either accept or decline the transaction
José is a digital nomad and loves to travel, he	**CoworkClub** offers numerous Coworking Spaces based on location. José is in a new city he doesn't know about, so he trusts

Story/Use Case Slice	Acceptance Criteria
needs a professional working space close to his current location José searched Coworking Space on **CoworkClub** based on his current location and availability	our platform to help him find a solution. He uses the search filter to find a professional working space based on his preferences, books for a week, and searches for coworking spaces in the next city he will stay.
Pedro puts his Coworking Space available at CoworkClub Pedro owns a potential Coworking Space, thus, he wants to get more clients and finds out a solution for that at **CoworkClub**	The System asks for the space's features, and, after that, as he presses the add button, the System puts Pedro's space available for searching/making bookings at **CoworkClub**.
Rita's Coworking space had a problem Rita has a coworking space at **CoworkClub** and that space had an unpredicted incident that severely affect the space, making it impossible for Bookings to occur. She wants to inform the Coworkers.	After reporting the situation to **CoworkClub**, the System will sent a notification to every Coworker affected by this event. The Coworking space will be temporarily unavailable until the problems are solved. Also, **CoworkClub** will need to help the Coworkers by providing repayment to them.
Office/Coworking Space	The space owned by the Space provider that was paid by the Coworker with the primary goal of using that space to work for a determined period of time
Calendar	An agenda with all the Bookings of an Office/Coworking Space of a Space Provider for all time frames

3.2 Selected Stories of Use

When a Booking is made, it needs two approvals:

1. The Space Provider's approval:

  - The Space Provider will authorize or decline the Coworker (since the Space Provider owns the space that is being booked
2. The Bank's payment approval
  - The Bank must make sure that the payment has been made or if the Coworker is capable of paying for the service

If any of those approvals fail, the Booking ends. If both are conceded, the Space Provider must make sure that the Office/Coworking Space has all conditions to host the Coworker at the time of the Booking. If it does have, the Coworker will then enjoy the Booking. If something goes wrong, the Coworker must be warned about the impossibility of having the Booking.

3.3 Strategy and Implementation Stage

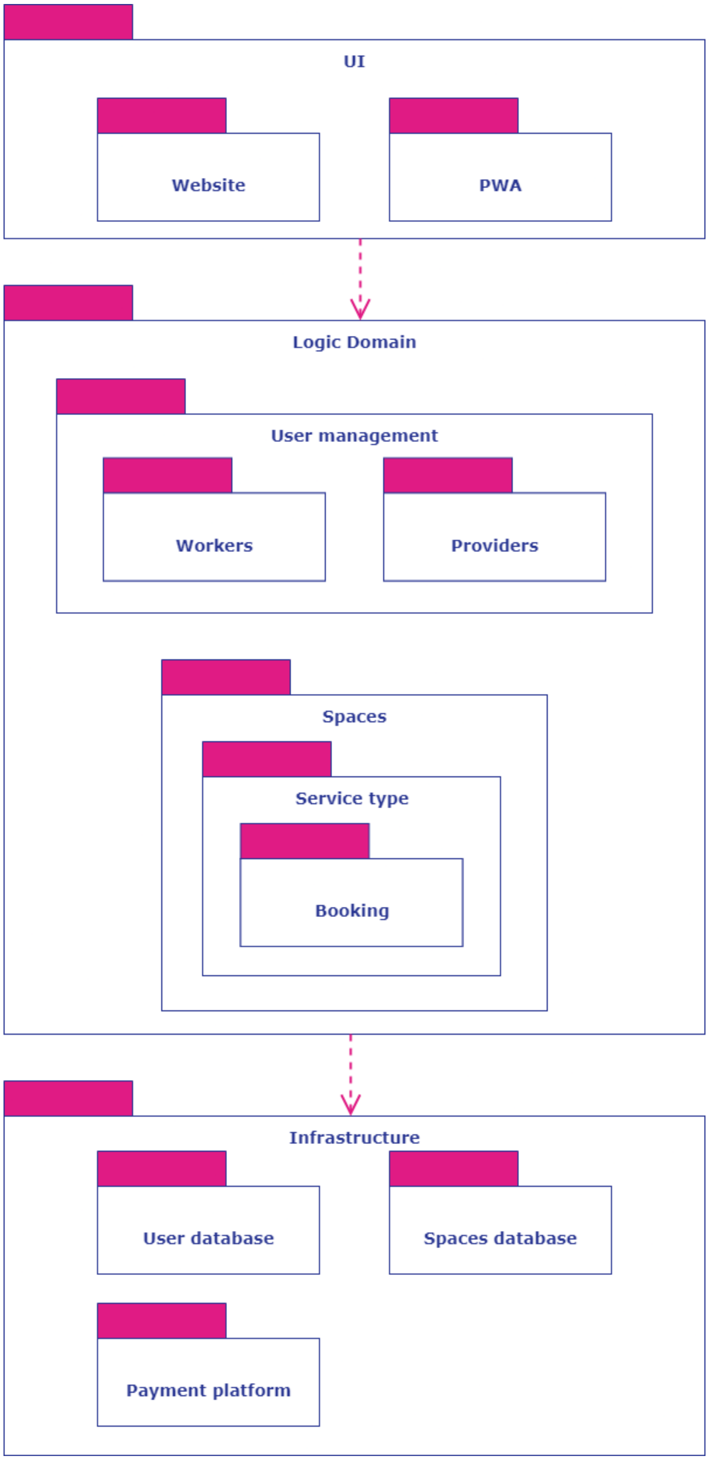
As of this stage the implementation is still on the design of the user interface, using a Teleport HQ, the team is designing a interface so later it can be exported in Vue Js code. Home and search pages are already complete, and now follows the spaces's pages , profile and booking pages too.

The API in Express JS is already prepared and designed, requests and responses are already constructed, and are being implemented into the code.

What is lacking as of this stage is, besides the missing pages, the connection between front-end and API.

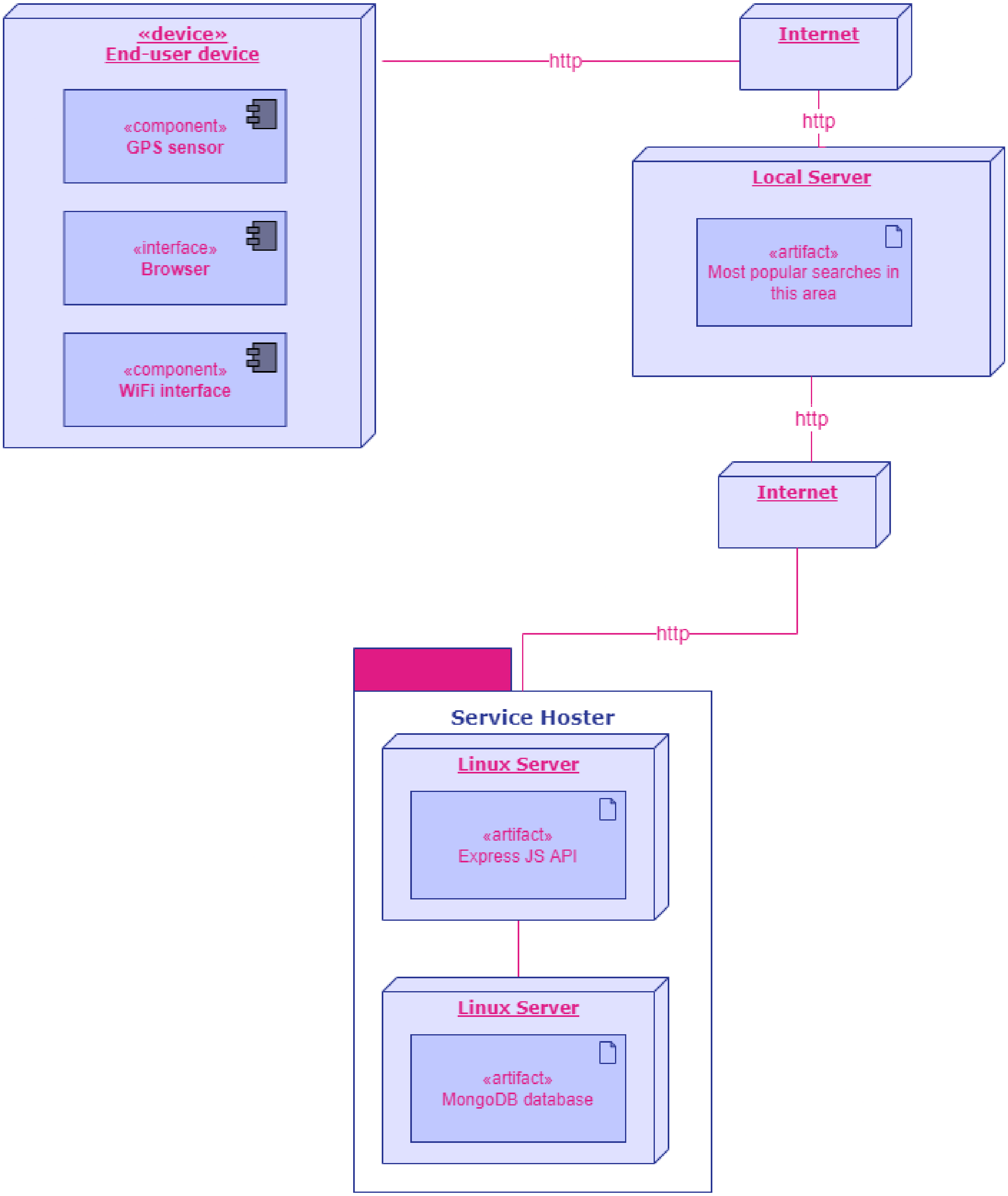
Annex

Software Architecture



Picture 1 - Package Diagram of the domains of the architecture.

Software Architecture



Picture 2 - Physical Diagram.