

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

A Gamified Application for Service Plan Management

Tiago Augusto Pacheco Ludovico Pinto de Barros



Mestrado em Engenharia Informática e Computação

Supervisor: Dr. António Fernando Vasconcelos Cunha Castro Coelho

February 3rd, 2022

A Gamified Application for Service Plan Management

Tiago Augusto Pacheco Ludovico Pinto de Barros

Mestrado em Engenharia Informática e Computação

February 3rd, 2022

Resumo

No contexto da indústria das pequenas empresas, um profissional tem de lidar com aspetos logísticos complexos para contratar os vários serviços para se manter operacional, nomeadamente: serviço de electricista, canalização, contabilidade, fornecimento de materiais, entre outros. Esta necessidade de gerir e contratar cada serviço individualmente pode tornar-se constrangedora e trabalhosa. De modo a responder ao problema anteriormente mencionado, propomos o desenvolvimento de uma aplicação móvel para a gestão deste tipo de serviços, na qual um utilizador terá um certo número de serviços disponíveis para subscrever, podendo escolher os que necessita. Após a subscrição, o utilizador terá acesso rápido a cada um deles, num único local. A aplicação também integra o processo administrativo onde os administradores conseguem gerir aspetos do negócio com gamificação integrada para impulsionar desempenho e compromisso. A aplicação em questão trará benefícios para o sector das pequenas empresas, nomeadamente qualidade de vida, tendo em conta a simplicidade de solicitar um serviço e a facilidade de utilização da aplicação. Além do sector das pequenas empresas, a aplicação também trará contribuições para o sector das TI, visto que tem uma arquitetura flexível que lhe permite servir qualquer tipo de negócio, bem como utilizar uma abordagem inovadora à gamificação onde os administradores recebem pontos por fazerem tarefas básicas, assim como Badges/Achievements. Desta forma, pode ter aplicações abrangentes.

Abstract

In the context of the small business industry, a professional has to deal with complex logistical aspects for contracting the various services to remain operational, namely: electrician service, plumbing, accounting, supply of materials, among others. This need to manage and contract each service individually can become constraining and laborious. In order to respond to the problem previously mentioned, we propose the development of a mobile application for the management of this type of services, in which a user will have a certain number of services available to subscribe, being able to choose the ones he needs. After subscribing, the user will have quick access to each of them in one place. The application also features an administration side where admins are able to manage aspects of the business with integrated gamification to drive up performance and engagement. The application in question will bring benefits to the small business' sector, namely quality of life, taking into account the simplicity to request a service and the application's ease of use. Apart from the small business's sector, the application will also bring contributions to the IT sector since it has a flexible architecture allowing it to serve any type of business as well as using an innovative approach to gamification in which admins receive points for doing basic task as well as achievements. In this way, it can have comprehensive applications.

Acknowledgments

In the first place, I would like to thank my advisor and supervisor Doctor António Coelho for the opportunity and for advising me in this dissertation. I would like to thank FEUP and all its staff for teaching me and for all the valuable experiences that brought me here. I also would like to thank my family and especially my father Manuel de Barros for supporting me and helping me through highs and lows during all these years of College. A big thanks to all my friends, those I already had and those I made along the way for all the good times we had as well. Wouldn't have done it without you all.

Tiago Augusto Pacheco Ludovico Pinto de Barros

“Three Rules of Work: Out of clutter find simplicity; From discord find harmony; In the middle of difficulty lies opportunity.”

Albert Einstein

Contents

1	Introduction	1
1.1	Context and Motivation	1
1.2	Objectives	1
1.3	Research Questions and Methodologies	2
1.3.1	Research Questions	2
1.3.2	Methodologies	2
1.4	Document Structure	2
2	State of the Art	5
2.1	Business and Technology	5
2.2	CRM (Customer Relationship Management)	5
2.2.1	Operational CRM	6
2.2.2	Collaborative CRM	6
2.2.3	Analytical CRM	6
2.3	Gamification	6
2.3.1	Gamification Elements	7
2.3.2	Examples of gamification	7
2.4	Related Work	10
2.4.1	Design of a mobile service for the farming sector	10
2.4.2	Glovo and Uber Eats	10
2.4.3	Gamified mobile application for creative introduction to programming	11
2.4.4	Zoho CRM	12
2.5	Technologies for App Development	13
2.5.1	Android and iOS independent Apps	13
2.5.2	Progressive Web App	15
2.5.3	Back-end technologies	17
2.6	Conclusion	18
3	Specification and Early Prototypes	19
3.1	High-level Overview	19
3.2	Requirements Analysis	21
3.2.1	Functional Requirements	21
3.2.2	Non-functional Requirements	24
3.3	Technical Architecture	24
3.4	Navigation	25

4	Implementation of the solution	27
4.1	Project Organization	27
4.2	Database	28
4.3	Register/Login	31
4.4	User Environment	33
4.5	Admin Control Sections	34
4.5.1	Service Requests	35
4.5.2	Registration Requests	36
4.5.3	Manage Services	37
4.5.4	Add Admin	38
4.6	Admin Gamification	39
4.6.1	Admin Leaderboard	40
4.6.2	Achievements	41
4.7	Controllers and API	42
4.7.1	Authentication	43
4.8	User evaluation	43
4.8.1	Client component evaluation	44
4.8.2	Admin component evaluation	45
5	Conclusions and Future Work	47
5.1	Satisfaction of the Objectives	47
5.2	Research Questions	49
5.3	Future Work	49
5.3.1	General Improvements:	49
5.3.2	Client Component Improvements:	49
5.3.3	Admin Component Improvements:	50
	References	51

List of Figures

2.1	screenshot of Duolingo	8
2.2	screenshot of Habitica App	9
2.3	Functionality example of MySugr	9
2.4	Mockups for the App: mobile service for the farming sector	10
2.5	Front Pages of Uber Eats and Glovo	11
2.6	Example of a mini-game	12
2.7	Example of a Zoho Dashboard	13
2.8	Android Studio development environment	14
2.9	Comparison between Native Apps, PWAs and Web Apps. source: web.dev/what-are-pwas	16
2.10	Example of a complex React component composed of simpler components	17
3.1	Application screens example: Mockups for user Registration	20
3.2	User and Guest use case diagram for functional requirements	22
3.3	Admin and Super Admin use case diagram for functional requirements	23
3.4	Non-Functional Requirements for the Application	24
3.5	Architecture of the Application	25
3.6	Navigation flow of the application	26
4.1	Project's relevant Folder Structure	28
4.2	UML diagram for the database	30
4.3	Registration screenshots	32
4.4	Login screenshot	33
4.5	User services page	34
4.6	Navigational dropdown for Admins	35
4.7	Handling of a service request	36
4.8	Handling of a registration request	37
4.9	1: "Gerir Serviços" page, 2: delete service alert, 3: create new service form.	38
4.10	Form to create an Admin	39
4.11	Admin Leaderboard	41
4.12	Admin Achievements Page	42

List of Tables

4.1	File/Folder organization of the Project's src folder.	27
4.2	Description of the various collections within the database	31
4.3	Ladder of Priorities for Point attribution	40
4.4	Description of the various controllers within the App	43
4.5	SUS scores for the client component	44
4.6	SUS scores for the Admin component	45

Abbreviations and Symbols

PWA	Progressive Web Application
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
UI	User Interface
UX	User Experience
PC	Personal Computer
API	Application Programming Interface
AI	Artificial Intelligence
CRM	Customer relationship management

Chapter 1

Introduction

1.1 Context and Motivation

Managing a small business and keeping it afloat may be an arduous and constant process, wherein the owner needs to balance different aspects of its business daily, such as: accounting, cleaning, laundry, buying supplies etc... but also other aspects related to maintenance, such as contracting: an electrician, plumbing services, showcase professionals, etc...

“I didn’t realize how hard it was to run a small business.” *Andrew Mason (founder and former CEO of Groupon)* (1) (2)

This need to search and look up what kind of service needed and to hire it, can be time-consuming, as much searching may be required, either to find the best service or the best price for it. The approach discussed in this dissertation consists on making the professional’s life easier by removing the process of having to search/hire a service when needed and instead letting them subscribe to all the services that may be needed and when these are needed they can be requested easily. This approach also includes a gamified Admin component where administrators will be able to manage clients’ service requests, registration requests as well as earn points and achievements for actions done within the system.

1.2 Objectives

The objectives of this dissertation pass through studying, designing, developing and validating the gamified application for service plan management, more concretely the specific objectives are:

- Explore and study an approach and technologies to design a mobile App for business professionals;
- Develop an App that facilitates the hiring of different services;
- Develop an intuitive and easy to use App;
- Provide the ability for the App to be customized and suit different businesses;

- Validate the App and identify aspects to improve with the help of input from a focus group;
- Provide a gamified back office for administrative use;

1.3 Research Questions and Methodologies

In order to research and develop an Application to help small business owners, some questions are worth posing, as well as some methodologies on how to proceed.

1.3.1 Research Questions

In order to develop the App, research needs to be conducted in order to better understand what needs to be done on a deeper level and how, as such, some questions need to be asked and answered in order to get a deeper understanding of the related areas and context, some of those questions are:

- What is the best architecture to provide these services?
- What are the most fitting technologies to develop our App?
- How can we gamify the App in an engaging way?

1.3.2 Methodologies

This project was developed iteratively and in an exploratory way, some steps were required in order to reach a functional prototype.

First, we started by doing research for the state of the art, by finding several articles, thesis works, related concepts and Apps in order to perceive the surrounding ecosystem and our position in it. Second, we started finding ways to develop the App, making extensive research on what kind of technologies would better fit the development of our App. Third, we started prototyping the App by making the requirements Analysis, defining the technical architecture and prototyping page navigation. Fourth, we started developing our App by developing the database alongside the front-end, with meetings to analyze the progress made, until the App prototype was fully developed. Fifth, we arranged a meeting with different people in order to evaluate the App by use of the System Usability Scale.

1.4 Document Structure

The present document is composed of 5 chapters being them:

- Chapter 1, the introduction, where we describe the context and motivation, objectives, research questions, methodologies and document structure of this dissertation;

- Chapter 2, the state of the art, where it is discussed relevant concepts and themes, related work as well as different possible technologies for the development of the application;
- Chapter 3, the specification and early prototypes, giving insight on how we conceptualized the solution alongside its architecture and technologies used;
- Chapter 4, the implementation of the solution described in detail and the user evaluations;
- Chapter 5, the conclusions and future work;

Chapter 2

State of the Art

In this chapter we talk about relevant concepts and themes to the dissertation, some possible technologies that were considered for the development of the application and related work.

2.1 Business and Technology

Business is the activity of making money by producing or buying and selling goods, or providing services (3) as such it has always been a part of society since time immemorial. Ever since society existed there was a need to trade materials, resources, goods, which in turn gave rise to business. Nowadays, business has evolved alongside technology and current trends/ideologies to suit different and bigger markets, as such, there is a need for enterprises to adapt and adopt technology more than ever (4). There exist several ways in which technology has improved business, be it with tools that give insight on the current state of the business, monitoring different aspects of it such as revenue, number of clients, inventory tracking, or with more applied tools that facilitate the business process, by cutting the middleman, such as labor management platforms for restaurants that let owners calculate employees' salary automatically and funnel clocked hours straight into the payroll (5) making easier what would have been a manual and cumbersome task. This way, with a greater need to keep up with demand and growing markets, it is beneficial and needed to find ways in which the business process may be facilitated and/or streamlined.

2.2 CRM (Customer Relationship Management)

Customer relationship management, or CRM for short, is a methodology/process/software that helps companies to understand and deal with customers. In a more software centric sense, it can be a tool to manage different aspects of the business, be it as the name implies dealing with the relationships with customers, by allowing to manage complaints, market to customers, among other things, or be it to condense different business metrics into easy to read and analyze graphics as well as allowing the management of different business facets. Three different kinds of CRM exist (6):

2.2.1 Operational CRM

Operational CRMS focus on streamlining and improving business processes, by means of automation of processes or providing an easier way to manage them. An operational CRM can help business owners/companies to generate leads, convert them into contacts and retain them. Operational CRMS achieve this goals by automatizing:

- Marketing;
- Sales;
- Services;

2.2.2 Collaborative CRM

A collaborative CRM focuses on establishing communications between different departments of a company so that synchronicity within the business is achieved, providing each department with a better understanding of their clients' interests, wants and needs. This kind of CRM achieves its goals by 2 means, interaction management which focuses on keeping track of every costumer interaction and logging them, and channel management which is the next step after interaction management that focuses on using the information gathered from it to identify and pursue the communication channels that better suit the costumer's preferences.

2.2.3 Analytical CRM

An analytical CRM uses data gathering and analysis to serve their customers. The analytical CRM is a kind of CRM that facilitates business processes by providing different analytics and functions that let business owners more closely monitor their business and understand different facets of it on a deeper level. This can include a selection of graphics, such as clients per month, revenue per month, costs per month, etc.

2.3 Gamification

With the purpose of making the work environment less tedious and more engaging, gamification comes to mind as a way of engaging workers with the intention of leading to better performance. Gamification, is the idea of using game design elements in non-game contexts in order to drive up engagement (7), it has been rapidly gaining traction and has spawned numerous applications - ranging from productivity, finance, health, education, sustainability as well as, news and entertainment (8). Gamification is the craft of extracting the fun elements of games and applying them in non-gaming contexts, for example the work environment, it is what is referred to as "Human-Focused Design" instead of "Function-Focused Design" according to Yu-kai Chou (9), meaning, it is a design process focused on human beings, on what drives them and trying to drive up that engagement instead of being designed for maximum efficiency.

2.3.1 Gamification Elements

There exist many ways in which an App or system can be gamified, as stated above gamification is the process of extracting game elements into non-game environments, this can include rewarding workers among a variety of things ((10), (11)), such as:

- **Badges or Achievements:** these are a visual representation of a user's achievements within the App. For example in a fitness App a user might have a badge of an x amount of steps they took one time;
- **Levels:** These can correspond to parts of the game world, with each level the complexity of the challenges within the App increases;
- **Performance Charts:** These charts can show how a player has improved by comparing previous data with new data, for example in an exercise App if a user has spent more calories or walked more in a day, it can compare showcasing the increase in calories lost or miles/kilometers walked vs a previous day;
- **Points:** Points serve as an intangible reward and can be a measure of performance, depending on how many points a user has. Points can be given by accomplishing different tasks within a system;
- **Leaderboards or Scoreboards:** these serve to showcase and compare the performances of different users within a system, ranking them and showcasing the ones that perform best;
- **In-game currency:** This is a means of "economy" for game-related benefits. The user can get currency by accomplishing tasks within the system or as a daily bonus, and use this currency to get items in the game;

2.3.2 Examples of gamification

There exist a variety of ways an App or system can be gamified, ranging from rewarding workers/users with badges/achievements, implementing levels, performance charts, points, among many other techniques. Below, we talk about some Apps renowned for their gamification design.

2.3.2.1 Duolingo

Duolingo is one of the most popular platforms for learning new languages in the world, in order for Duolingo to motivate its users it uses a number of gamification elements in an attempt to make learning a new language fun. It uses techniques such as awarding users badges for performance, intangible rewards, leaderboards and keeping track of their engagement with the platform on a daily basis, rewarding them for using the platform for consecutive days (12)(figure 2.1).

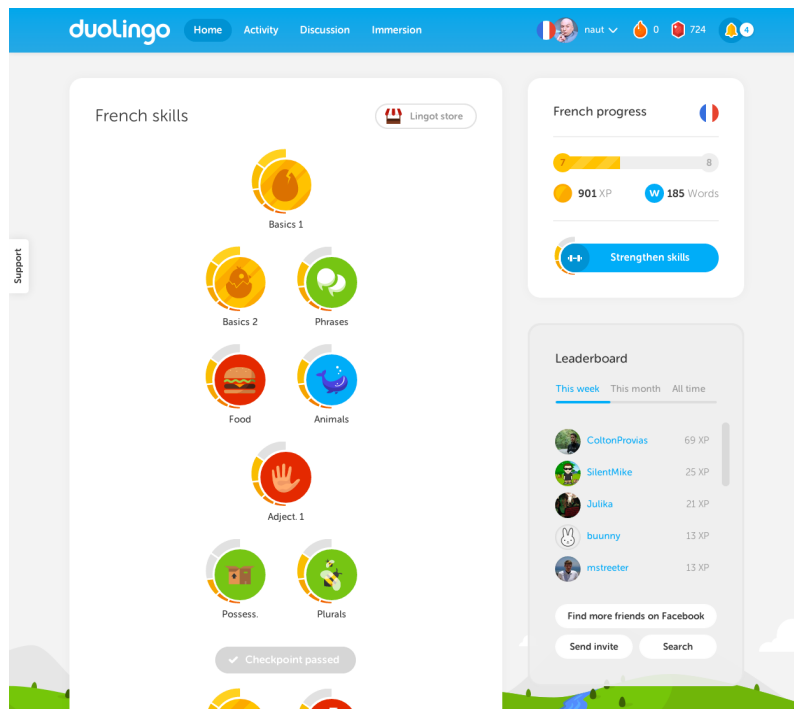


Figure 2.1: screenshot of Duolingo

2.3.2.2 Habitica

Habitica is a free habit building and productivity App that aims to help the user do their day-to-day tasks and achieve their goals by gamifying these mundane aspects of life. It uses in-game awards and punishments in order to motivate its users (13). It allows its users to:

- Improve their habits and achieve their goals by tracking and managing user habits, daily goals and to-do lists;
- Earn rewards for their goals by checking off tasks to level up and unlock in-game features, such as, different armors for their characters, skills and magic;
- Battle monsters with friends and with the rewards buy in-game things or custom rewards like watching an episode of a TV show;

In figure 2.2 we can see a screenshot of the Habitica App.

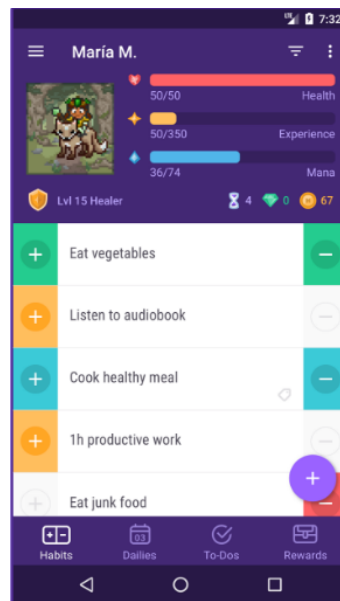


Figure 2.2: screenshot of Habitica App

2.3.2.3 MySugr

MySugr is a healthcare gamified App designed especially for people who have diabetes. This gamified healthcare App depicts the disease as a monster that has to be tamed. Diabetes patients need to look out for how much sugar they consume, which can be something boring to keep track off. This App attempts to make the routine of dealing with diabetes less dull by implementing all the necessary gamification aspects, such as progress tracking and rewards (figure 2.3).

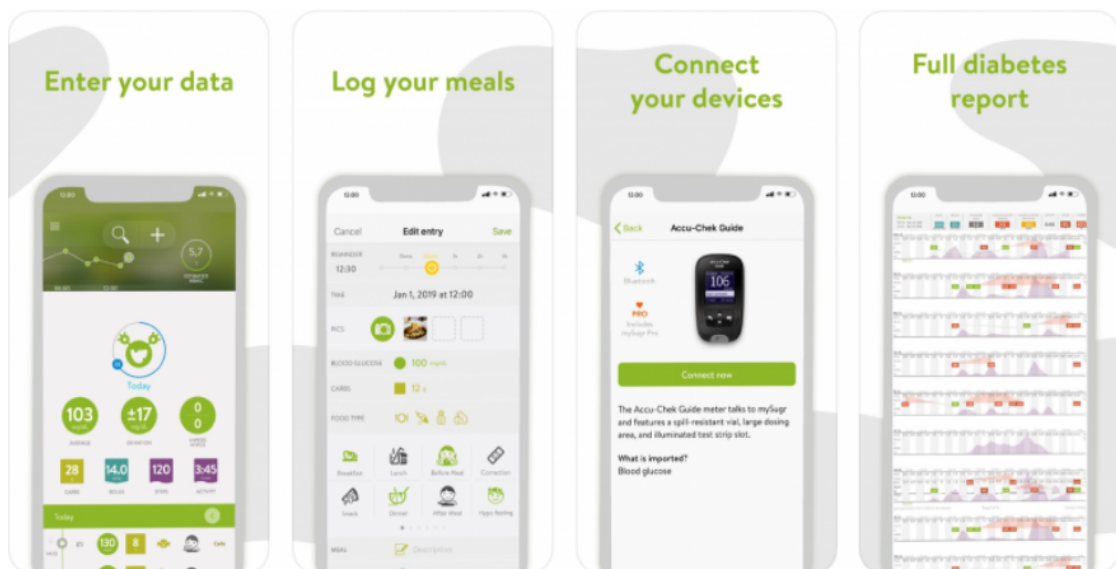


Figure 2.3: Functionality example of MySugr

2.4 Related Work

Some related work has already been addressed in the previous gamification section, but there still exist some work/solutions related to other aspects of the App, namely some approaches that condense different kinds of services into a single App or share similar ideas/technologies. Below we discuss some of this related work.

2.4.1 Design of a mobile service for the farming sector

This App was designed in a master's thesis and addresses an approach to develop an App for Agros collaborators, it condenses several services important to people in the agriculture business (namely the milk producing business) such as buying supplies needed and information relative to the business. It features a farmer's store that allows farmers to buy milk, to solicit accounting services, laboratory analysis, equipment maintenance, purchasing of equipment among other things (14) (figure 2.4).

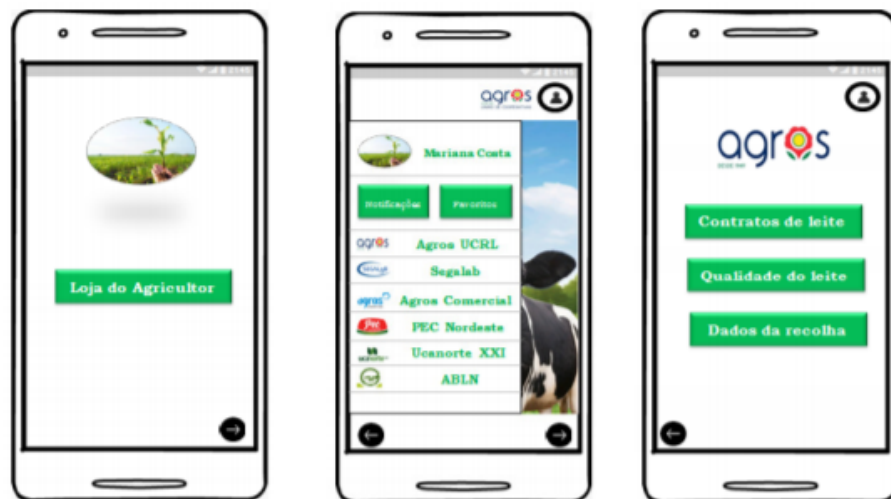


Figure 2.4: Mockups for the App: mobile service for the farming sector

2.4.2 Glovo and Uber Eats

Glovo and Uber Eats are Apps that condense several food services and small commerce into a place where they are easily accessible for the common consumer, they also deliver the goods to the doorstep of the client. They provide small businesses the opportunity of providing home delivery services without the need for them to put resources of their own in making the deliveries, being that, these are handled by Glovo and Uber Eats (15; 16). These Apps contain similarities to the project in question, seen as they condense several services in one place for the user. Figure 2.5 shows the front Pages of Uber Eats and Glovo.

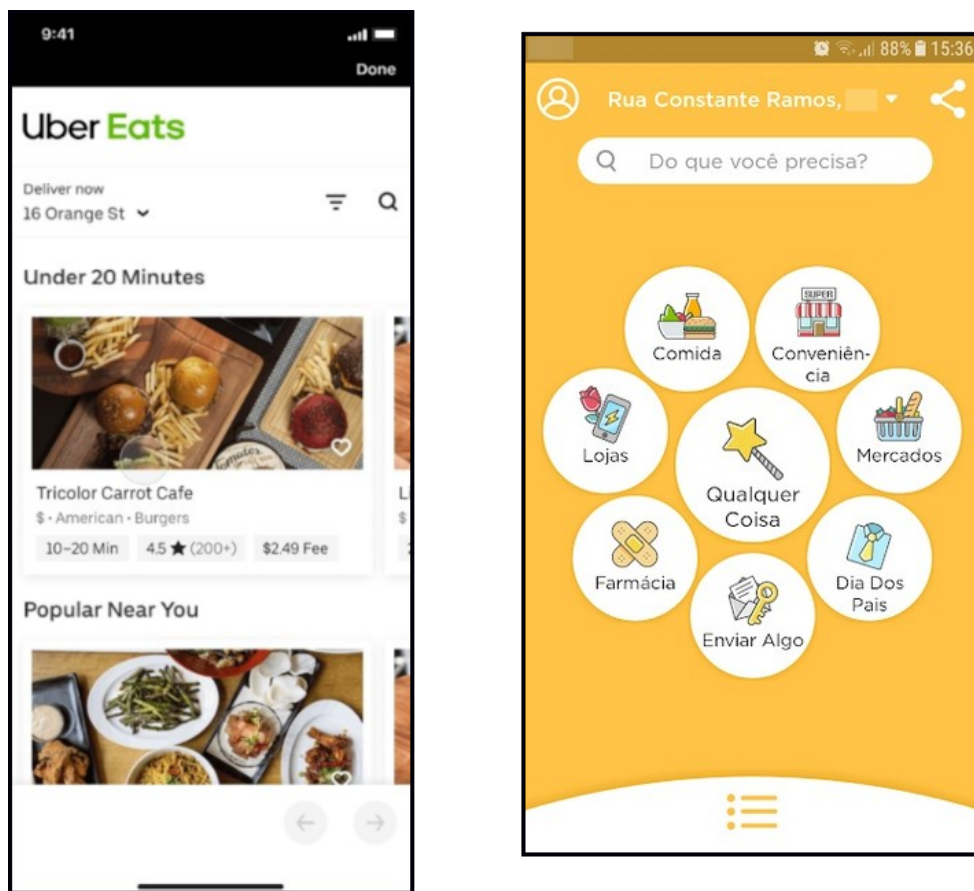


Figure 2.5: Front Pages of Uber Eats and Glovo

2.4.3 Gamified mobile application for creative introduction to programming

This thesis developed by Artur Sousa Ferreira consists on the development of a Web App to supplement school subjects and drive up engagement between students by gamifying the learning process by having challenges and quizzes to complete and rewarding students with points (17). The figure 2.6 shows an example of a mini-game from the App.

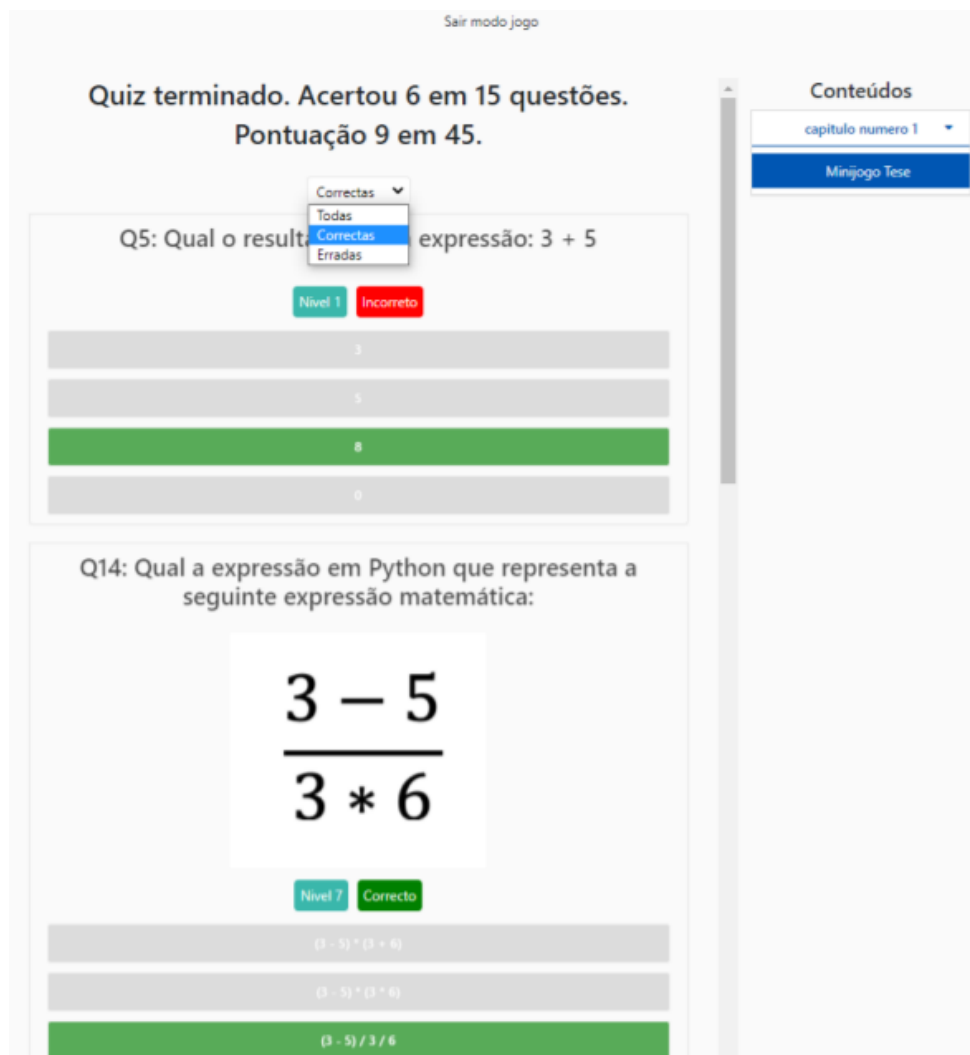


Figure 2.6: Example of a mini-game

The App developed is relevant in terms of architecture and technologies used, it uses a lot of the same technologies that we use in our App as well as its gamification component that uses a points system to reward students as well leaderboards, all things relevant to our gamified Admin component.

2.4.4 Zoho CRM

Zoho CRM is a customer relationship management software (available for desktop PCs and mobile devices), it allows the creation and customization of modules, automatizing sending of emails, assigning tasks, different kinds of graphics that allow business to better gauge their performance, implementation of an AI Zia, that analyses the performance of the business among other metrics to advise when to contact potential prospects, highlight irregularities in the business and make predictions based on the sales patterns, among other features (18). There are also features that let

the user edit data offline and submit that data once the device goes online again. The different features of Zoho come useful for the dissertation in question given the useful insight it provides on how to structure and what to include in the gamified Admin module, seen as it will function akin to a CRM. In figure 2.7 we can see a screenshot of the platform.

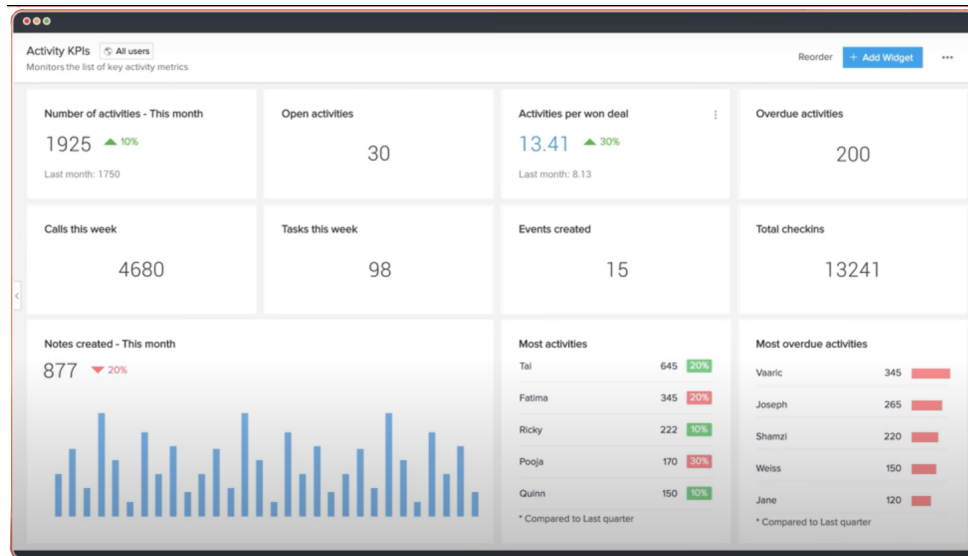


Figure 2.7: Example of a Zoho Dashboard

2.5 Technologies for App Development

In this chapter, we will explore the different technologies taken into account when researching the way to develop our App. Given the need to develop an application that will facilitate the business process of a professional in the hairdresser sector, give some metrics on the business and the need for it to be available on both Android and iOS, there were some technologies and approaches considered, developing a progressive Web App with React or Flutter, or developing two independent Apps, one for Android and another for iOS using Kotlin or Java and the Android Studio IDE, for Android, and the Swift coding language and the Xcode IDE for iOS. There are also some multiplatform tools like unity or Xamarin but we won't go into detail on them.

2.5.1 Android and iOS independent Apps

Developing for Android and iOS independently would pose its challenges, namely, the fact that 2 very similar Apps would need to be developed independently, from scratch (one for Android and another for iOS) significantly increasing the workload. Another obstacle would be the lack of equipment to develop the iOS App on since it can only be developed within the Mac environment, there isn't a tool that simply converts an Android App to iOS and vice-versa (19; 20)

2.5.1.1 Android Development

To develop specifically for Android, the easiest and most recommended tool that was found is Android Studio (figure 2.8) which is the official IDE for Android development based on IntelliJ IDEA (21).

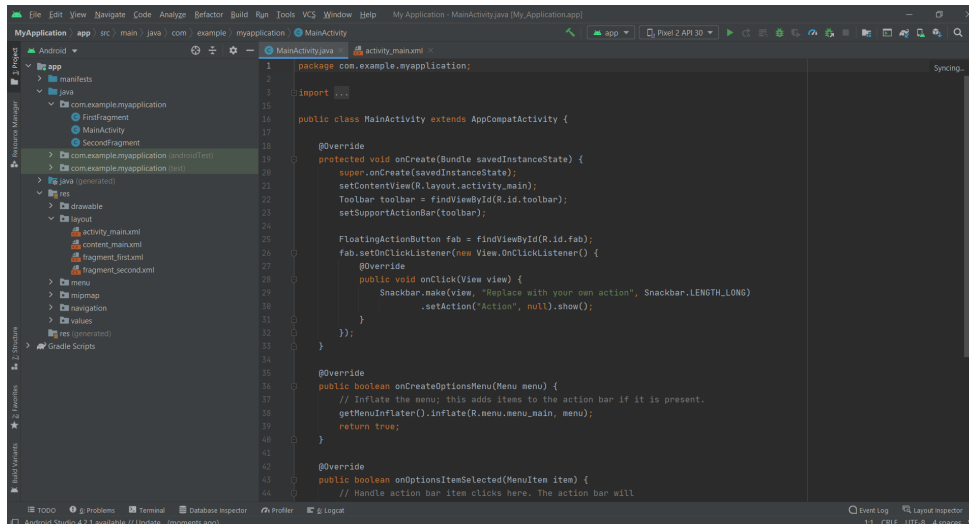


Figure 2.8: Android Studio development environment

Android Studio is a powerful tool to develop Android applications as it provides everything needed out of the box to develop them, such as (22):

- an integrated emulator that allows us to emulate with a specific kind of android device to test and visualize the app we are developing;
- a feature that lets us push changes to our app without the need to restart the emulator or the app;
- an intelligent code editor that offers advanced code completion, refactoring and code analysis;
- code templates and sample apps which makes the developer life easier by providing templates for certain things;
- testing tools and frameworks;
- version control system integration, such as, with GitHub that allows us to maintain an App with different versions;
- Firebase and cloud integration;
- a Layout Editor that allows us to visually edit the layout of the app in a graphical manner and with minimal code;

- lint checkers that allows us to keep the code clean;

2.5.1.2 iOS Development

Developing the App for iOS would provide its set of challenges seen as a Mac developing environment and the need to learn a new coding language (Swift) would be needed in order to develop our App (20). There exist some ways to develop an iOS App on a Windows PC although none of them are as simple as owning a Mac laptop with the Xcode IDE installed, they normally require renting services in the cloud or a virtual machine to emulate a Mac system in order to develop an iOS application (23). When it comes to IDEs, the one that is most mentioned is Xcode which is similar in functionality to Android Studio providing several beneficial features for development of iOS Apps, such as (24):

- a code editor with code completion, code folding, syntax highlighting, message bubbles that display warnings, errors and context-sensitive information;
- the assistant editor which splits the editor in two displaying files that are relevant to the part that is being edited at the moment;
- an interface builder that lets us build a design without code;
- an iOS simulator which lets us test our Apps in an emulated environment;
- the fix-it feature that alerts us to coding errors while writing code and allows us to fix them with a keyboard shortcut;
- a graphical debugger;
- the static analyzer that which can find potential bugs by exploring different paths in code;

2.5.2 Progressive Web App

A Progressive Web App or PWA is an application type of software commonly delivered through the Web that is intended to work with any platform that uses a standards-compliant browser (includes both mobile devices and desktop PCs), it is also commonly built using standard Web development technologies such as HTML, CSS and JavaScript, but other more advanced frameworks and libraries can be used to build them with better quality and or more easily, such as React or Flutter. PWAs mix what there is best of native Apps (applications installed and running on the device) and Web Apps (application software running on a Web server) as they offer the capabilities of native Apps and the reach of Web Apps (25).

“Progressive Web Apps (PWAs) are built and enhanced with modern APIs to deliver enhanced capabilities, reliability, and install-ability while reaching anyone, anywhere, on any device with a single code-base.”(25)

Figure 2.9 shows a comparison present in the web.dev website between PWAs, Native Applications and Web Apps.

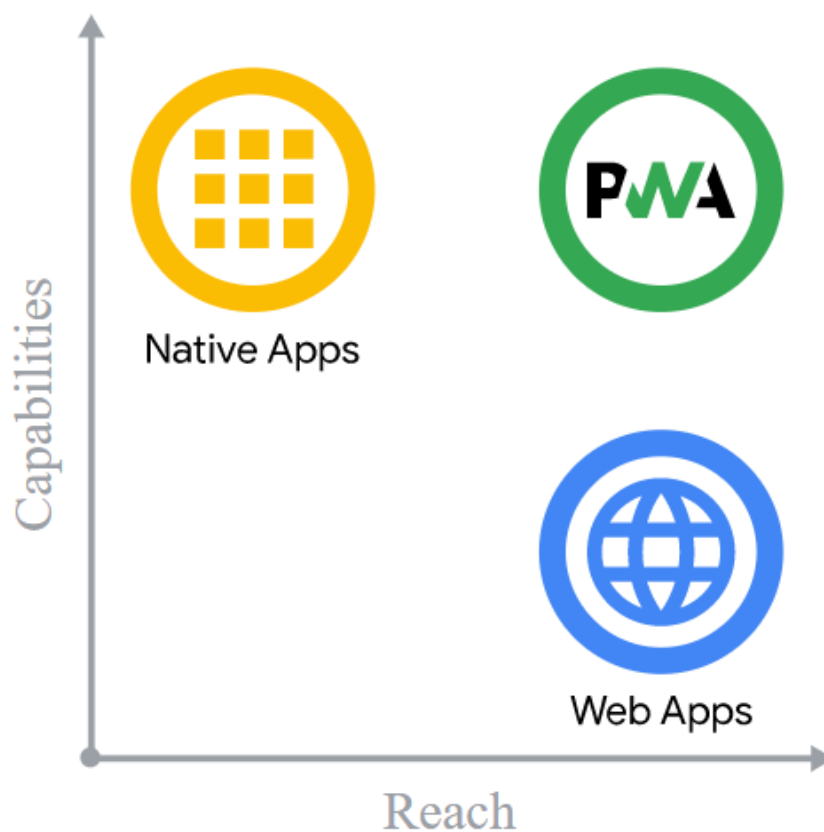


Figure 2.9: Comparison between Native Apps, PWAs and Web Apps. source: web.dev/what-are-pwas

2.5.2.1 React

React is a front-end JavaScript library for developing user interfaces created by Jordan Walke, a software engineer at Facebook (26; 27), it is a library with many features and qualities, such as being:

- Declarative, making it easier to develop interactive UIs as it will update and render only the needed/changed components;
- Component Based, allowing complex Web pages to be built using simpler components, that manage their own state;
- Learn Once, Write anywhere, allowing the writing of new features in react without the need of rewriting existing code;

In image 2.10 we can see an example of a complex React component composed of simpler components.

```

17 function Financial(props) {
18   return (
19     <div id="financialCore">
20       <div id="balanceSheet">
21         <AsyncEnhancedTable promiseFn={getBalanceSheet} Hcells=[
22           { id: 'Type', numeric: true, disablePadding: false, label: 'Type' },
23           { id: 'Designation', numeric: true, disablePadding: false, label: 'Designation' },
24           { id: 'Value', numeric: true, disablePadding: false, label: 'Value' },
25         ] Title="Balance Sheet" itemsPerPage="10"/>
26       </div>
27       <div id="resultsDemo">
28         <AsyncEnhancedTable promiseFn={getResultsDemo} Hcells=[
29           { id: 'Designation', numeric: true, disablePadding: false, label: 'Designation' },
30           { id: 'Value', numeric: true, disablePadding: false, label: 'Value' },
31         ] Title="Results Demonstration" itemsPerPage="10"/>
32       </div>
33     </div>
34   );
35 }
36
37 export default Financial;

```

Figure 2.10: Example of a complex React component composed of simpler components

2.5.2.2 Flutter

Flutter is a UI toolkit developed by Google for developing natively compiled applications for Web, desktop PCs and mobile devices, it can be used in conjunction with Android Studio or Visual Studio Code and uses the Dart programming language, which is tailored for client development and compiles to native machine code for mobile, desktop and JavaScript for Web (28). Flutter also uses widgets (screen and functionality templates) to make Apps easily.

2.5.3 Back-end technologies

For the project at hand we will need some sort of database be it to store users that use the App, services provided, among other things, this way some back-end technologies were considered:

- PostgreSQL for the database relational schema (29);
- Express for the back-end and handling HTTP requests (30);
- Firebase for the back-end and authentication (31);

2.5.3.1 Firebase

Firebase is a platform developed by Google for developing mobile Apps as well as web Apps. It was originally an independent company, Google acquired the platform, and it is now their flagship for App development (32). Firebase provides several services within their environment, namely Firebase Authentication for user authentication, and Cloud Firestore, which is a remote and schemaless back-end solution where data is organized into collections and documents which can contain complex nested objects in addition to subcollections. It also boasts support for offline work (33). It is a particularly interesting option given the fact that it is a remote solution that doesn't require additional hosting, and it is simple to use.

2.6 Conclusion

Business has always been a part of our society and with the advancement of the ages, ever expanding markets and challenges businesses need to adapt and adopt technology to keep up with the different and evolving markets. A CRM is a powerful tool to help businesses understand and better meet their costumers' needs. There are 3 different types of CRM focused on different facets of the business. There exist different and interesting work that can help the development of the App in question, namely the "Aplicação móvel ludificada para Introdução Criativa à Programação" which despite being with a different focus and for a different sector it uses many of the same technologies and architecture planned to be used in this application as well as an interesting gamification component similar to the one to develop in our App, "Glovo and Uber Eats" which come similar in the way they integrate several businesses in their App, and finally, the Zoho CRM which comes useful to the Admin part of the application, providing ideas on how to structure it, as well as, other functionalities. When it comes to technologies and approaches to use, we considered a diversity of selections, being either developing an android and iOS separate Apps using Android Studio and Xcode IDEs, or a PWA with React or Flutter. In conclusion, there exist a lot of ways to approach the development, design and conceptualization of this thesis and associated application, some related work worth considering taking inspiration from and providing some guidance, as well as a lot of technologies suited for this kind of undertaking.

Chapter 3

Specification and Early Prototypes

In this chapter we will provide a high level overview of the specification of the App, namely, the requirements analysis, technical architecture and user interaction.

3.1 High-level Overview

The App to be developed is a Progressive Web App that focuses on streamlining the business processes of small businesses, it provides a list of services, from which users can subscribe to the ones that will be helpful to their business, after which the user is contacted by the company that owns the App to agree on a subscription plan taking into account the number of services and type of services requested, After this step the users have access to the platform where they can request the services subscribed. There will also exist a gamified back office for Administrators of the system where they will be able to manage: users' registration requests, users' service requests and services. The administration back office as mentioned previously will be gamified in more concrete words Admins of the system will get points for actions within the system, such as:

- Dealing with users' service requests;
- Dealing with users' registration requests;
- Adding new services into the system;
- Assigning themselves to a service request;
- Assigning themselves to a registration request;

There will also exist a leaderboard where we will be able to see the standings of all the different Admins based on the amount of points each admin has collected and an achievements page where Admins are given challenges to complete (E.g. Dealing with x registration requests gives the admin y points). The application will be usable from a Desktop PC with access to the internet, as well as Mobile Devices. In figure 3.1 we can see some early prototypes of what the App could potentially look like.



Figure 3.1: Application screens example: Mockups for user Registration

3.2 Requirements Analysis

The solution needs to fulfill general requirements as well as some functional requirements. For this effect, we will define four actors within our system:

- guest: an unauthenticated user;
- user: an authenticated user;
- Admin: Administrator of the App, manages services, service requests and registration requests;
- Super Admin: Administrator of the platform that manages Admins and some gamification aspects;

3.2.1 Functional Requirements

For the actors mentioned above, these are the functional requirements:

- guest:
 - Create a registration request in the system;
 - Login into the system;
- user:
 - Request any of the subscribed services at a given time;
 - cancel requisition of a service;
 - Log out of the App;
- Admin:
 - Manage registration requests;
 - Manage service requests;
 - Manage services;
 - View the standing in the Admin Leaderboard;
 - View the status towards completion of active achievements;
- Super Admin:
 - Create new Admins;
 - Create new Super Admins;
 - Create new Achievements;
 - Reset accumulated Admin points;

- Refresh active achievements (changing them and resetting Admin progress towards old ones);

Figures 3.2, 3.3 and illustrate the use cases of functional requirements regarding an authenticated user and a guest (unauthenticated user) as well as an Admin and Super Admin of the system.

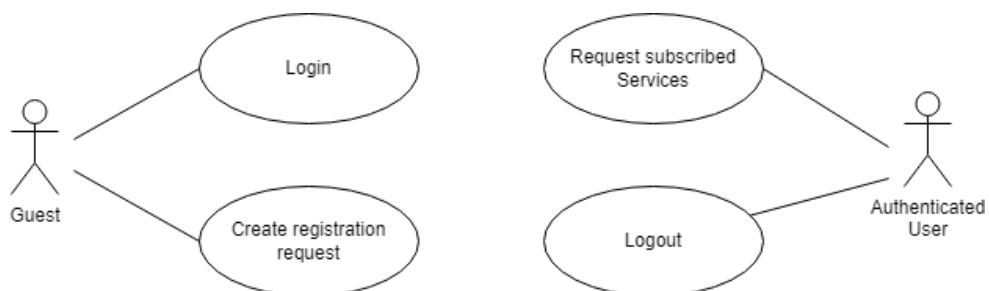


Figure 3.2: User and Guest use case diagram for functional requirements

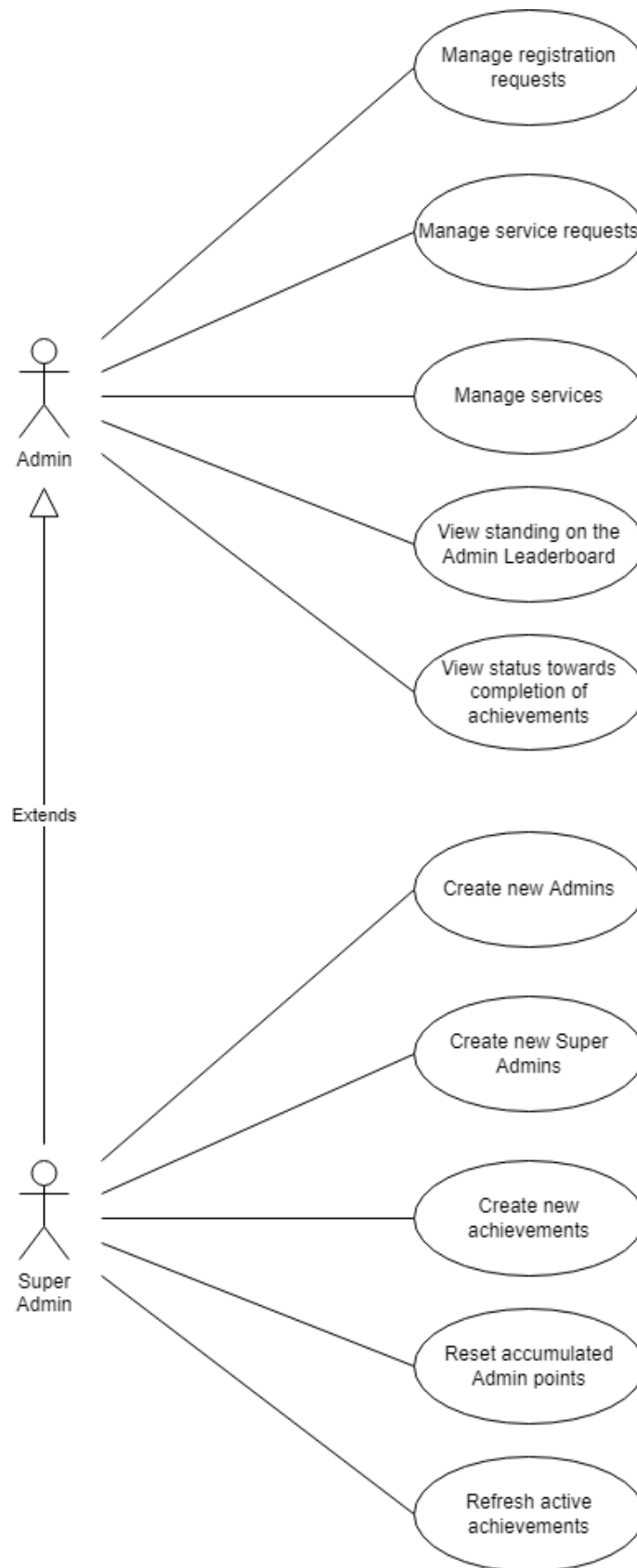


Figure 3.3: Admin and Super Admin use case diagram for functional requirements

3.2.2 Non-functional Requirements

The non-functional requirements of the application are described next. The application should:

- be accessible at all times;
- be intuitive and easy to use;
- function on different devices (PC, Android and iOS);
- be fast and responsive;
- be scalable and maintainable so that it can easily be adapted for different kinds of businesses;

The figure 3.4 further illustrates the non-functional requirements.

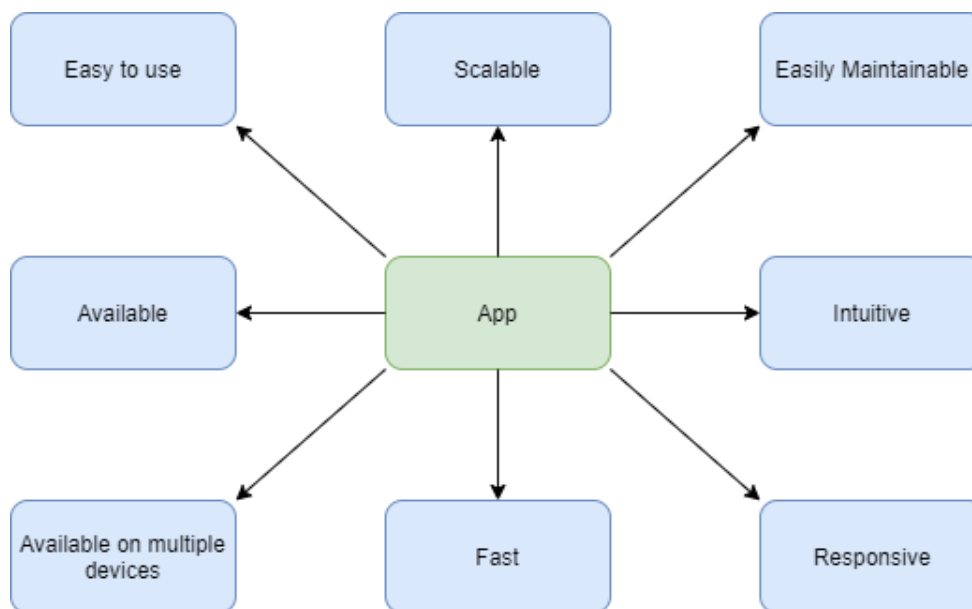


Figure 3.4: Non-Functional Requirements for the Application

3.3 Technical Architecture

For this kind of application we require the use of a database since we need to save entries (users, services, service requests, achievements, etc.), and we'll use controllers to be easier and safer to make database queries among other things. From the front-end side, we'll use React alongside Google's Material UI since it provides an intuitive way to make good-looking applications without much hassle, alongside Firebase Authentication for authentication and controllers using Firestore libraries to make requests to the back-end database hosted on Cloud Firestore. The PWA will be hosted on the Heroku platform for ease of use through the Web. The figure 3.5 further illustrates the architecture of the App.

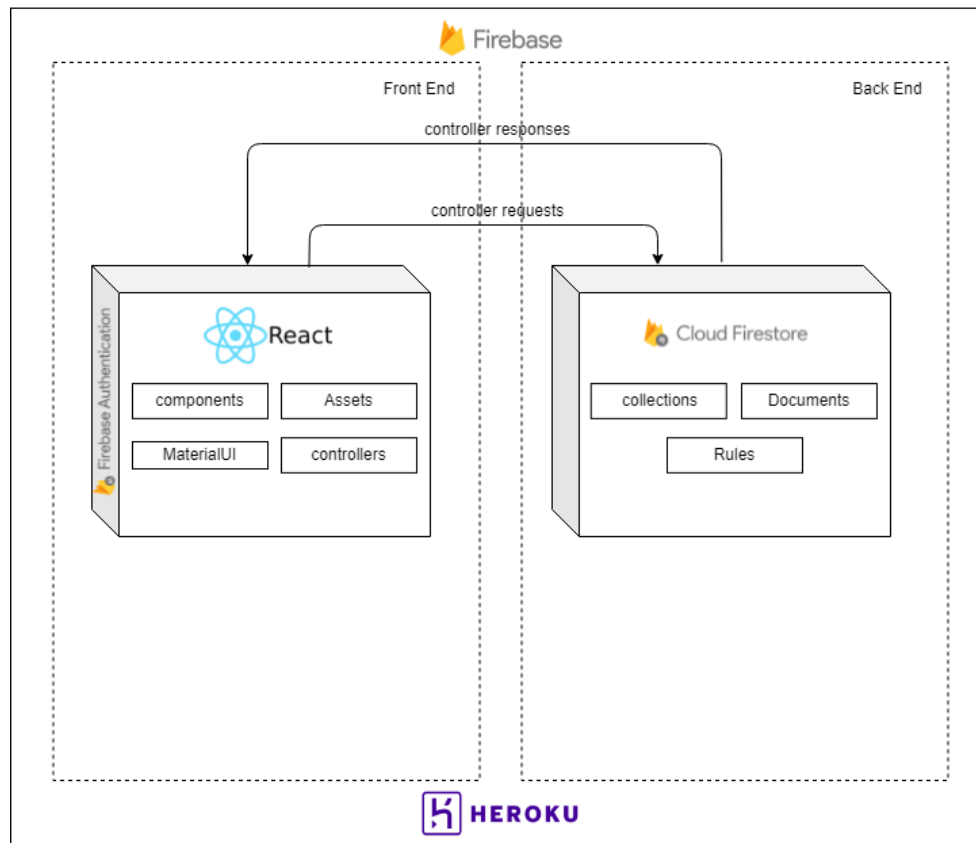


Figure 3.5: Architecture of the Application

3.4 Navigation

A user to be able to use the App needs to be logged in, if not they have to create a registration request and wait for an Admin to accept them into the system, after authentication or successful registration the user has access to their user environment where they can request services, in case of an Admin after login they have access to their Admin environment. The figure 3.6 illustrates the navigation flow within the application between different pages, as well some functionalities/fields within them.

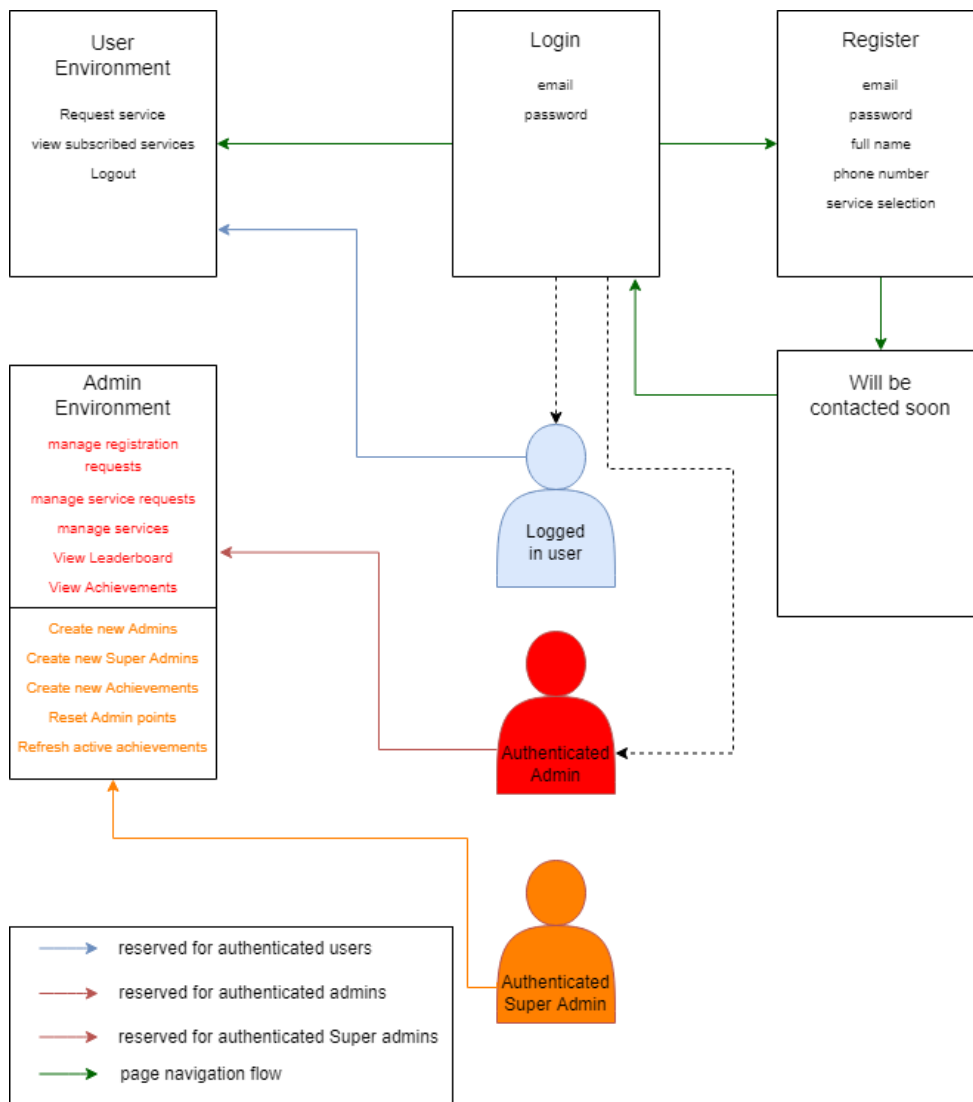


Figure 3.6: Navigation flow of the application

Chapter 4

Implementation of the solution

4.1 Project Organization

The goal of the implementation is to build a Progressive Web App with access and connection to a remote database fulfilling the mentioned necessities. The technologies used were: Google's Firebase, Cloud Firestore, Firebase Authentication, React and Google's Material UI, The project is organized in folders in order to encapsulate the different modules that constitute it. The general folder structure of the project can be seen in the figure 4.1 and table 4.1.

Folder/File	Description
components	Folder containing all the modular components that belong to the different pages of the App.
controllers	Folder containing all the controllers that communicate with the remote database.
views	Folder containing all the different views.
App.js	The main React component.
AppRouter.js	Router of the App with different routes for different pages.
db.js	Where the Firebase App and Authentication are initialized and connection to the database established.
index.js	Entry point of the App.

Table 4.1: File/Folder organization of the Project's src folder.

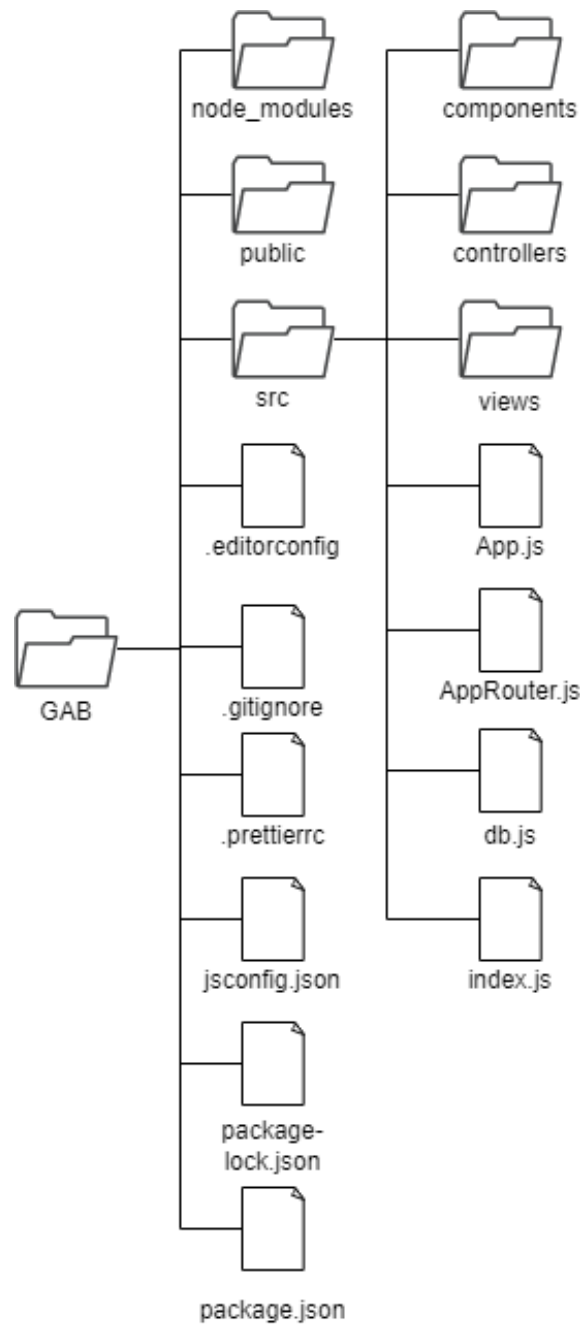


Figure 4.1: Project's relevant Folder Structure

4.2 Database

The database solution used for the project was Google's Cloud Firestore which is a schemaless and remote database approach where instead of tables and entries we have collections and documents, the relevant connections to the database are made in the db.js file inside the src folder. The database structure was projected to fulfill the needs of the project, in part by, having collections

to store all the needed information, such as: users, services, service types, service requests, admin achievements and respective progress, and achievement types. Some explanations about the purpose of the collections and some not so obvious aspects follow:

- The collection **users** stores information relating to all the different users of the system which are differentiated by the role attribute which can be user, admin and superAdmin, admins or super Admins need to allow users into the system.
- The collection **serviceTypes** serves to group services into standard types within our system for better specification;
- The collection **services**, serves to group different specific services identities and store them;
- The collection **serviceRequests** serves to store information of each individual request of a service, it saves information such as the id of the user who requested the service, the id of the admin that handled the request, the id of the type of service requested, as well as, the id of the service chosen for the user.
- the collection **adminAchievement** stores achievements of a given achievementType, the goal attribute represents the amount of times we need to perform said action to receive the specified points, the key attribute functions as a random string, in order to make a random selection of achievements, further explained a bit ahead;
- The collection **achievementTypes**, serves to group achievements into standard types, defined within the system for better specification;
- The **adminAchievementsProgresses** collection, serves to store the status of an admin, such as, the amount of service requests that specific Admin handled, or how many registration requests that admin handled, as well as, the IDs of the completed achievements, this information is used to keep track of the progression towards the unlocking of achievements. The specification of every field within this collection is as follows:
 - **id**: the id of the specific adminAchievementsProgress, which, coincides with the id of the admin, to which, this respective progress belongs to;
 - **regAssign**: a counter with the amount of registration requests the admin has assigned himself to;
 - **regReqsHandled**: a counter with the amount of registration requests the admin has handled/accepted into the system;
 - **serviceAssign**: a counter with the amount of service requests the admin has assigned himself to;
 - **serviceReqsHandled**: a counter with the amount of service requests the admin has handled;
 - **servicesCreated**: a counter with the amount of services the admin has created;

Despite the database solution being schemaless, a representation of the relevant data structures can be further examined in the UML scheme in image 4.2 and on the table 4.2.

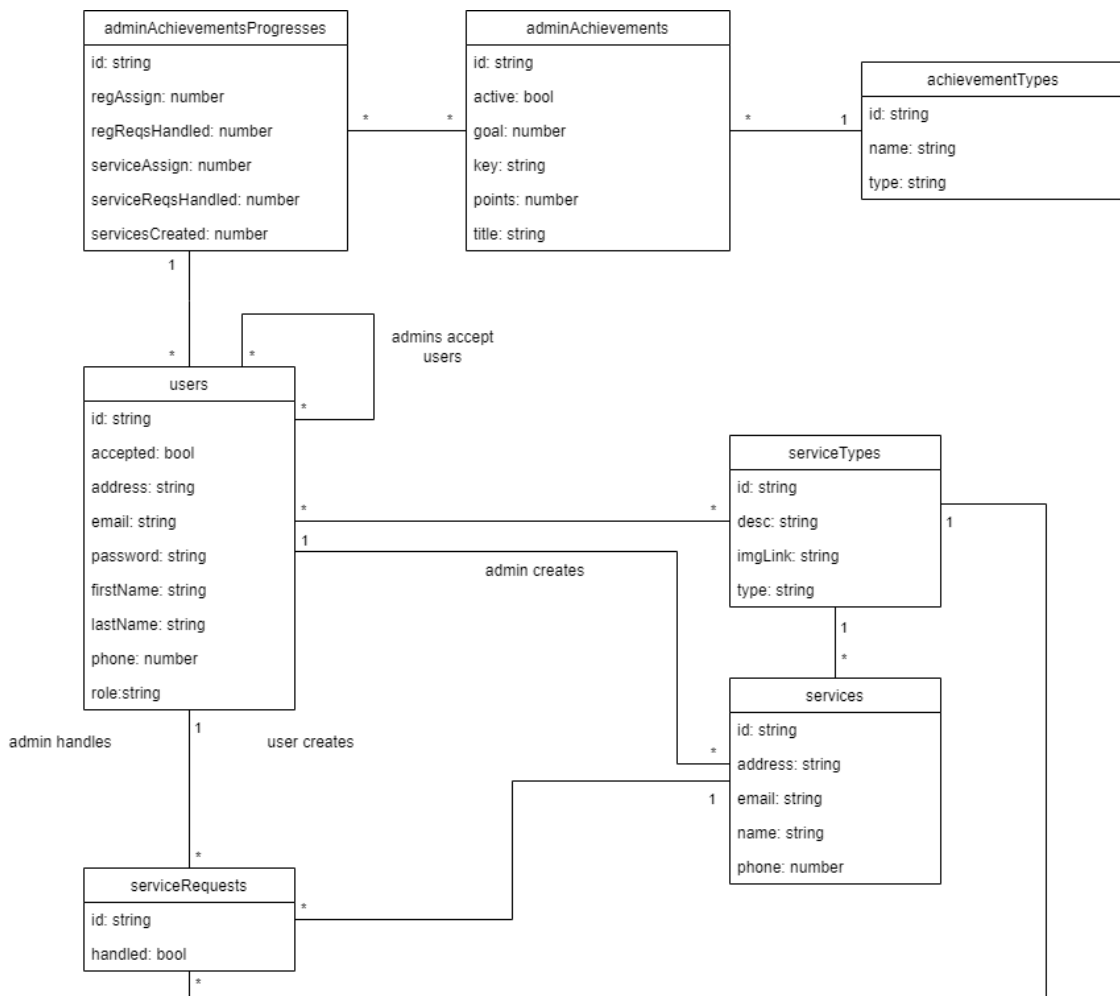


Figure 4.2: UML diagram for the database

Collection	Description
users	Users collection, where info relative to users and admins is stored.
serviceTypes	where information relating to the different service types is stored, for example: electrician, plumber, accounting, etc.
services	where information relating to specific services from a given type is stored.
serviceRequests	where information relating to specific user service requests is stored.
adminAchievements	where information relating to achievements for the admins are stored.
adminAchievementTypes	where information of different types of admin achievements is stored.
adminAchievementsProgresses	where information pertaining to the status of admin progress towards achievements, as well as, completed achievements is stored.

Table 4.2: Description of the various collections within the database

4.3 Register/Login

Register and Login function as they do in most applications with the exception that register takes the following parameters: first name, last name, phone number, address, email, password and a selection of services the user would like to subscribe to, for example: electrician, accounting, plumbing, etc. After filling the prerequisite fields, the user successfully creates a register request that will need to be processed by an Admin before the user has access to the system and can log in. In image [4.3](#) we can see some screenshots of the register page.

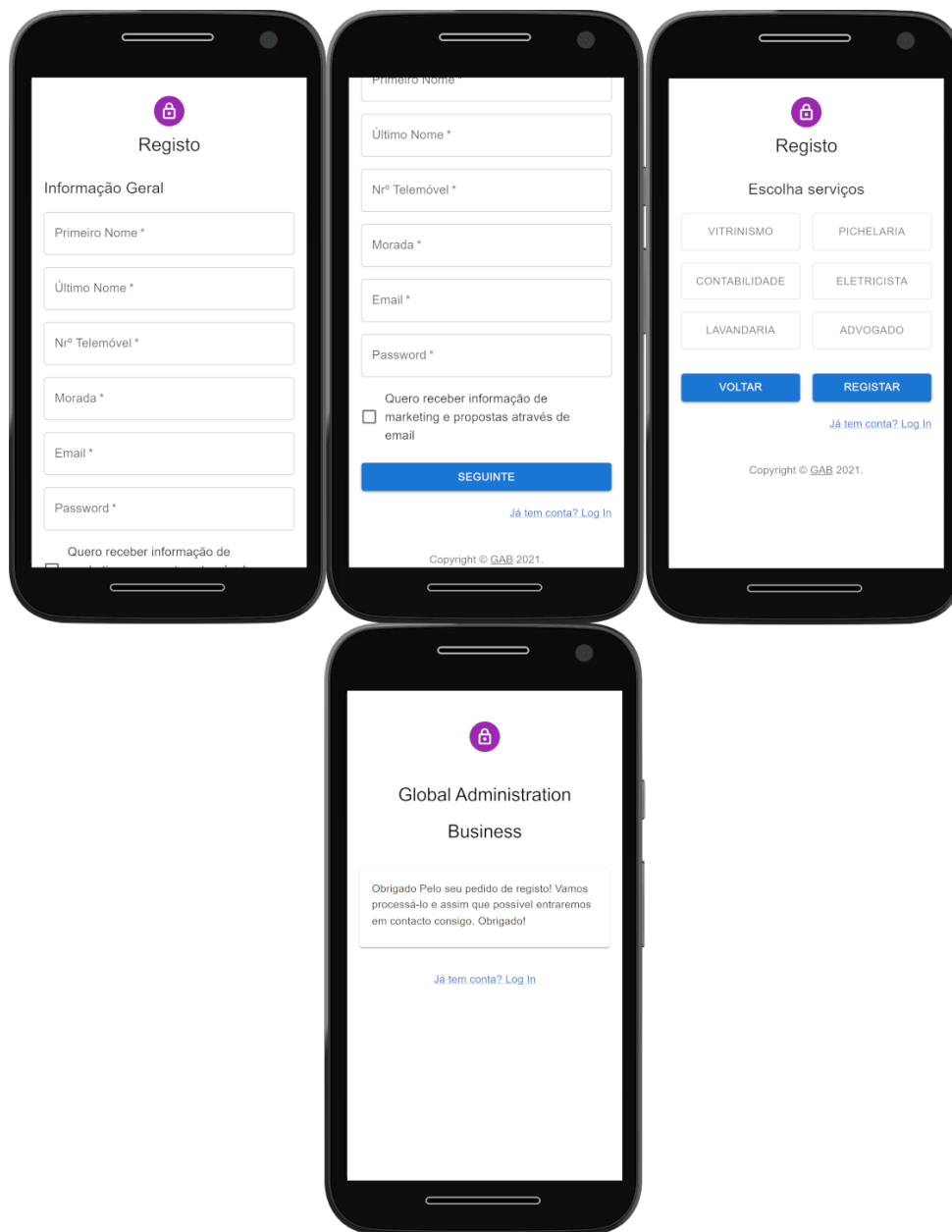


Figure 4.3: Registration screenshots

Login requires an email and password to login. In image 4.4 the login page is displayed.

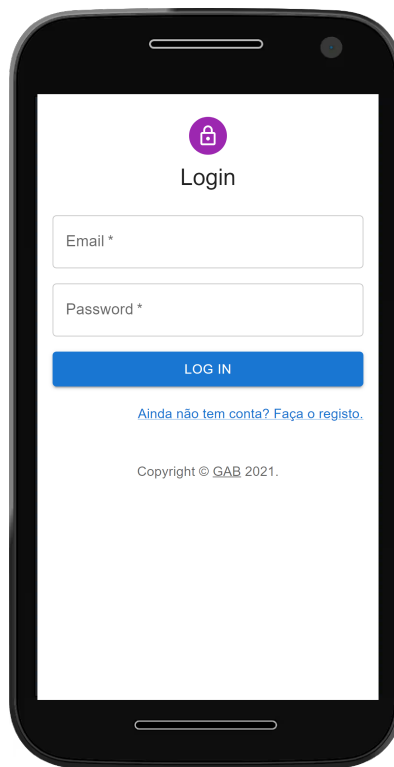


Figure 4.4: Login screenshot

4.4 User Environment

After successful login, a user is presented with a page with their subscribed services. These services can be requested at any time by pressing the "Chamar" button on the respective service card. In image 4.5 is displayed a screenshot of the user environment.

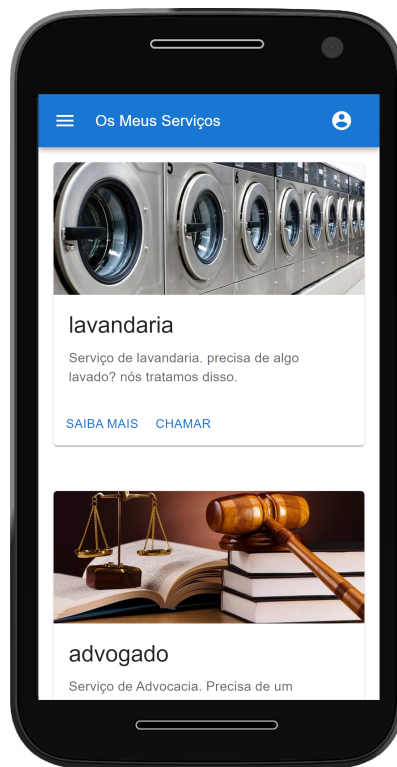


Figure 4.5: User services page

4.5 Admin Control Sections

The Admin control sections are where the Admins of the system can manage the system and answer user requests. They have access to a dropdown menu on the top left that allows them to navigate easily through their pages, this was designed with the intent of making navigation easier and better suited for the small screens of mobile devices, so the work environment wouldn't be cluttered. A screenshot of the navigational dropdown can be seen in image 4.6. The following subsections go in further detail about what each control section is and what it entails.

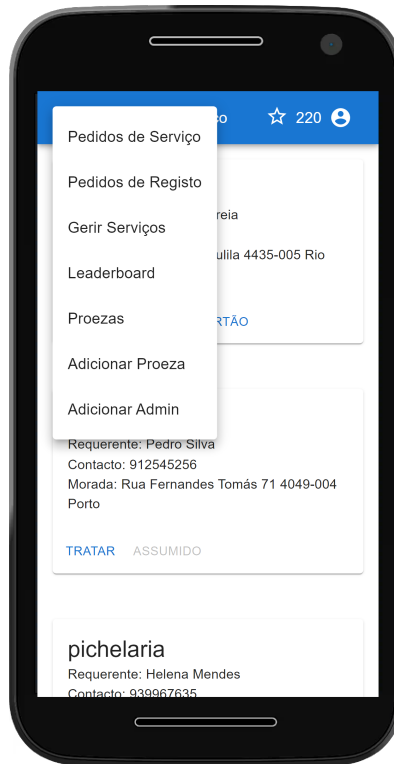


Figure 4.6: Navigational dropdown for Admins

4.5.1 Service Requests

This section translated into "Pedidos de Serviço", is where Admins manage users' service requests, when a user requests a service, a card appears in this page where the Admin can manage it by manually selecting a specific service to respond to the user's needs and making the necessary arrangements. To deal with a service request, an admin needs to assign themselves to the specific service request by clicking the button "ASSUMIR CARTÃO", which is a way to deal with concurrency in the sense that an admin presses said button and that card becomes unavailable for other admins (the "TRATAR" and "ASSUMIR CARTÃO" buttons become greyed out), symbolizing that that specific admin is handling that request. We can see the sequence of handling a service request in image 4.7, first by pressing "ASSUMIR CARTÃO", then pressing "TRATAR" and finally, choosing a service and making the needed arrangements.

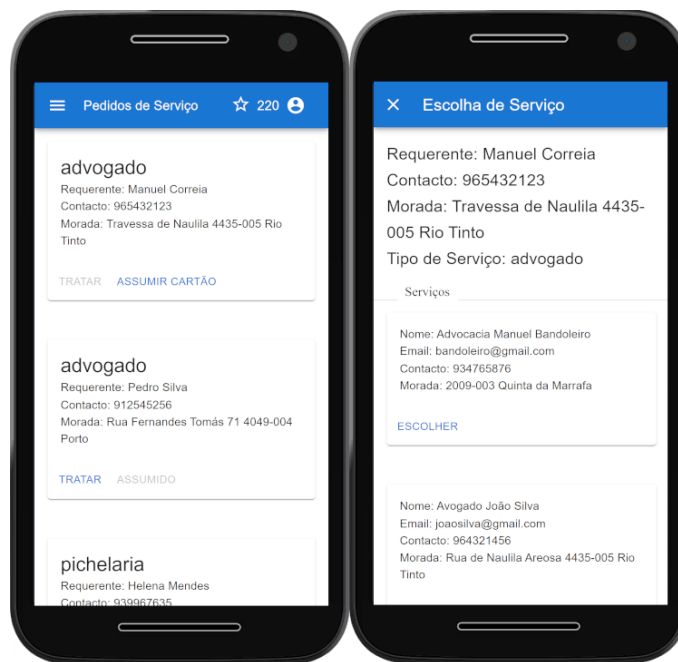


Figure 4.7: Handling of a service request

4.5.2 Registration Requests

This section translated into "Pedidos de Registo" is where Admins manage registration requests, when a user registers into the system through the register page, they aren't immediately accepted into the system, a registration request is created with their information and service types they would like to be subscribed to, and afterwards in this page an Admin manages the client's request and makes the necessary arrangements. In the image 4.8 we can see the screenshots involved in the process.

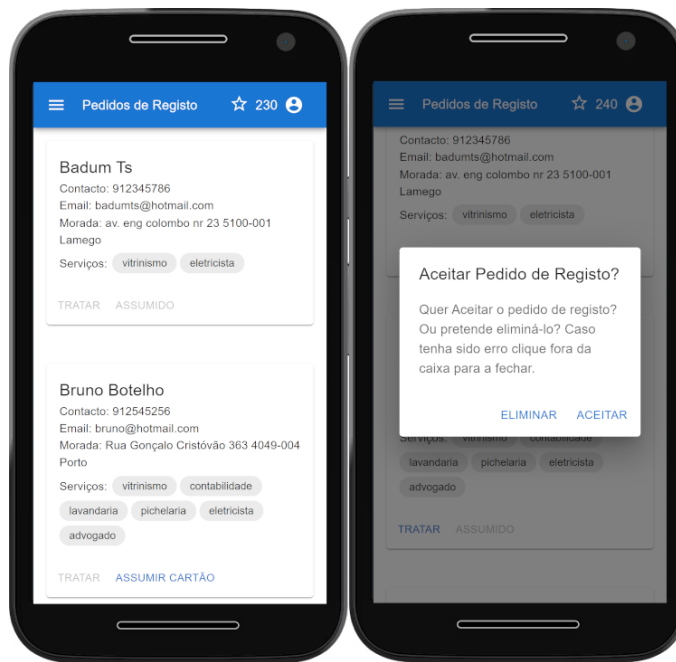


Figure 4.8: Handling of a registration request

4.5.3 Manage Services

This section translated into "Gerir Serviços" is where Admins can manage services existent within the system, this involves deleting services by pressing the button "APAGAR SERVIÇO" withing that service's card, and they can also add new services to the system by pressing the "CRIAR SERVIÇO" button at the top of the page. The composition of images 4.9 first shows the "Gerir Serviços" page, then the alert when clicking "APAGAR SERVIÇO" and finally, the form to create a new service by clicking the "CRIAR SERVIÇO" button.

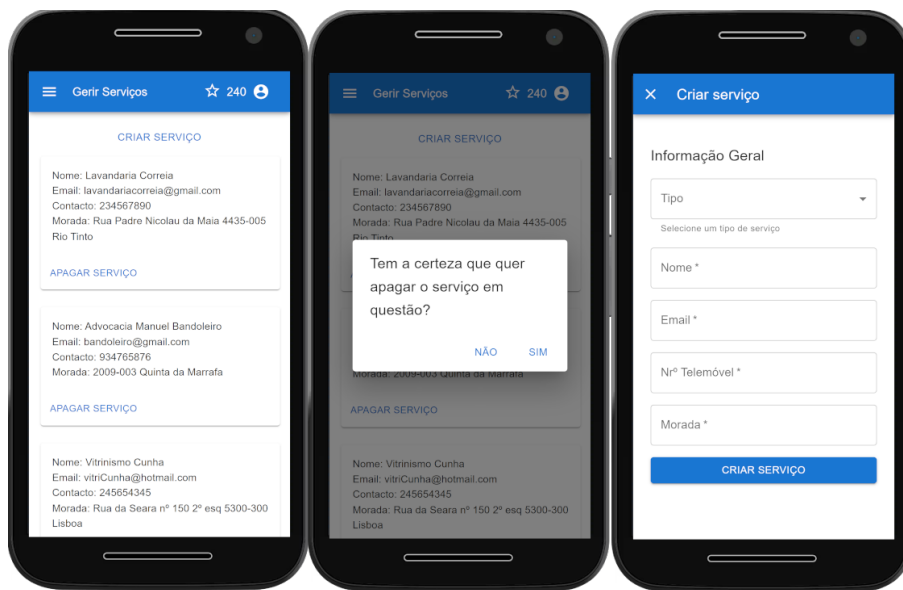


Figure 4.9: 1: "Gerir Serviços" page, 2: delete service alert, 3: create new service form.

4.5.4 Add Admin

This section translated into "Adicionar Admin", only accessible to Super Admins, is where they can create new Admins or Super Admins and add them to the system, similar to the regular registration, it requires some of the same parameters as well as some new ones, "Tipo" specifies the type of Admin, either Admin or Super Admin and the other ones are self-explanatory: First Name, Last Name, phone number, email and password. In image 4.10 we can see a screenshot of the form to create an Admin.

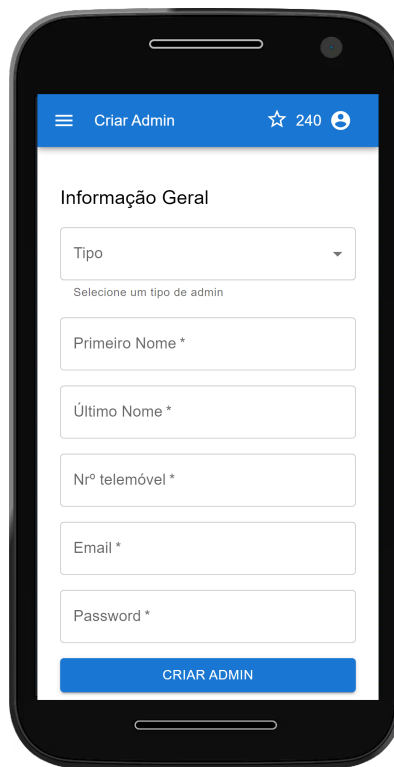


Figure 4.10: Form to create an Admin

4.6 Admin Gamification

The project features a gamification component, that was implemented as a way of making the Admins more engaged with the platform and to increase productivity by incentivizing them with points, completion of achievements, a leaderboard and potential rewards. This component was implemented in the previously mentioned Admin control sections, where each Admin, be it Super Admins or regular Admins earn points for actions within the system, these actions include:

- Assigning themselves to a registration request, which is worth 10 points;
- Assigning themselves to a service request, which is worth 10 points;
- Handling a registration request, which is worth 150 points;
- Handling a service request, which is worth 100 points;
- Adding a service to the system, which is worth 50 points;

The total points of each admin can be seen in the top right of most pages, for example in the image 4.6. This attribution of Points follows a given logic, namely, we tried to attribute more points to actions more important to the system. For example, handling a registration request is worth 150 points and handling a service request is worth 100 because despite both of these actions

being crucial to the system, we decided that accepting new costumers into the system was one step above in the ladder of priorities (table 4.3).

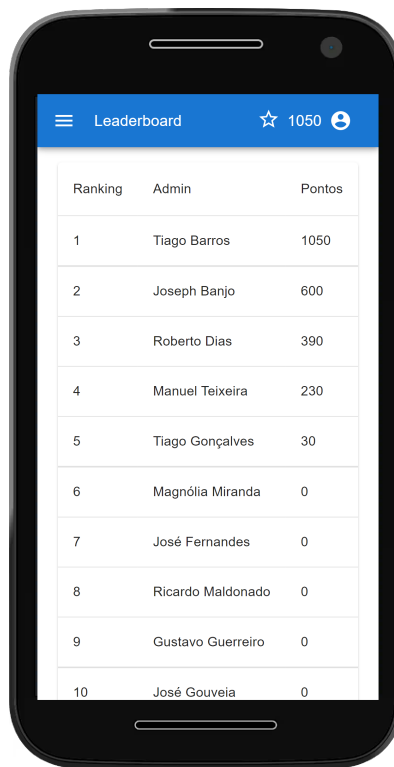
Points	Action
150	Handling a Registration Request.
100	Handling a Service Request.
50	Adding a service to the system.
10	Assigning themselves to a registration request.
10	Assigning themselves to a service request.

Table 4.3: Ladder of Priorities for Point attribution

Apart from the actions the Admins can realize in different pages of project giving them points, there also exist achievements they can earn giving point rewards as well as a leaderboard comparing the points of every admin. Below we further talk about these pages and concepts.

4.6.1 Admin Leaderboard

This page is dedicated to the leaderboard which lists all the Admins in the system ranked by their amount of points, this serves both as an incentive for those in the lower echelons to work harder and match those above, as well as, a form of recognition for those who sit at the top. Super Admins have the ability to at any point reset everyone's' points, thus restarting the process. In image 4.11 we can see a screenshot of the admin leaderboard.



Ranking	Admin	Pontos
1	Tiago Barros	1050
2	Joseph Banjo	600
3	Roberto Dias	390
4	Manuel Teixeira	230
5	Tiago Gonçalves	30
6	Magnólia Miranda	0
7	José Fernandes	0
8	Ricardo Maldonado	0
9	Gustavo Guerreiro	0
10	José Gouveia	0

Figure 4.11: Admin Leaderboard

4.6.2 Achievements

The Achievements page, translated into "Proezas", is where each Admin can see the status of their progression towards the completion of different achievements. Achievements are categorized into 5 distinct types, these types are:

- **regReqsHandled**, which pertains to the action of handling registration requests;
- **regAssign**, which pertains to the action of an Admin assigning themselves to registration requests;
- **servicesCreated**, which pertains to adding services into the system;
- **serviceReqsHandled**, which pertains to the handling of service requests by Admins;
- **serviceAssign**, which pertains to the action of an Admin assigning themselves to service requests;

In figure 4.12 we can see a screenshot of an Admin's achievements page.

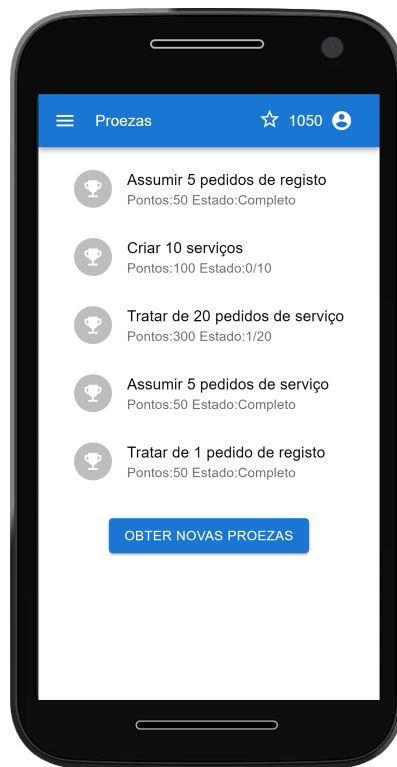


Figure 4.12: Admin Achievements Page

4.7 Controllers and API

As our solution resorted to a remote schemaless solution for the database by using Firebase and Cloud Firestore, there isn't the need for a back-end in the traditional sense, our requests to the database can be made almost directly through the front-end by using the Cloud Firestore and Firebase API. This way it allows us to streamline the development process, making change more instant. As such, controllers are basically functions that make calls to the remote database by using said API and interpreting the answers, they are kept in the controllers folder inside the src folder of the project. There are 3 main controllers that manage different information between the App and the database, pertaining information can be found in table 4.4.

Controller	Description
achievementsController	This controller handles all database information pertaining to achievements, including accessing the collections: achievementTypes, adminAchievements, adminAchievementsProgress
servicesController	This controller handles all database information pertaining to services and service requests, including accessing the collections: services, serviceTypes and serviceRequests
userInfoController	This controller handles all database information pertaining to users.

Table 4.4: Description of the various controllers within the App

4.7.1 Authentication

Within the App we have a need for authentication in order to access user specific content or just to securely save content within the App for this purpose user authentication is managed by Firebase Authentication which provides some useful features including UI libraries to authenticate users to the App, and it also supports several methods of authentication such as: passwords, phone numbers, popular federated identity providers like Google, Facebook and Twitter (34). User authentication is done in the login page, where it is required an email and password to effectively sign in and further access contents.

4.8 User evaluation

For the purpose of validating the developed App, a user evaluation/test is required to see how users adapt and work with the developed App. For this effect we chose the System Usability Scale (SUS) which provides a quick way to evaluate the usability of a system, it consists of 10 questions, each with 5 possible answers ranging from strongly disagree to strongly agree, the score of each question is put into a formula to calculate the SUS score, the higher the score the better, being 68 the baseline for an above average system in terms of usability, the SUS was originally created by John Brooke in 1986 (35). With the purpose of evaluating the App, a focus group meeting was arranged both with people well versed in the business world and some students, in this meeting everyone would test both sides of the App (the client side and the Admin side) as well as answer 2 SUS forms one for rating the client side and another to rate the Admin side. Due to the restrictions of the pandemic situation, the focus group allowed us to gather 10 answers (6 for the Client module and 4 for the Admin module). Although the sample is too small to be a proper user evaluation, we believe it still is capable of providing some valuable insight.

4.8.1 Client component evaluation

Question	Participant					
	1	2	3	4	5	6
1. I think that I would like to use this system frequently.	3	5	3	2	4	5
2. I found the system unnecessarily complex.	1	1	3	4	1	1
3. I thought the system was easy to use.	5	5	1	3	4	5
4. I think that I would need the support of a technical person to be able to use this system.	1	2	3	2	1	1
5. I found the various functions in this system were well integrated.	3	4	1	2	3	4
6. I thought there was too much inconsistency in this system.	3	2	3	4	2	2
7. I would imagine that most people would learn to use this system very quickly.	5	5	2	3	4	5
8. I found the system very cumbersome to use.	1	1	3	3	2	2
9. I felt very confident using the system.	3	4	4	2	3	4
10. I needed to learn a lot of things before I could get going with this system.	1	1	4	2	1	2
1. SUS score	80	90	37.5	42.5	77.5	87.5

Table 4.5: SUS scores for the client component

The average SUS score for the client component was 69.2 which is above average taking into consideration that 68 is an above average application in terms of usability. Some improvement suggestions from the participants were:

- not making the client select all the services they want to be subscribed to in the registration process, do it after;
- emails with dots before the @ are not accepted by the system;
- Being able to see the status of the requested service;
- service packs instead of selecting services clients want to be subscribed to individually;
- better feedback when selecting services to be subscribed to in the act of registration;

Some considerations regarding the suggested improvements:

- in regard to the service packs instead of selecting individual services to be subscribed to, this was the specification wanted for the project so that was how it was developed, not to say it can't be changed in the future, still a valuable insight;
- in regard to emails with dots not working at registration, this was a bug that was found in the system at the time of the focus group, it is an easy fix in a future iteration;
- In regard to being able to see the status of the service request being processed, I think it is a good and valuable suggestion that wasn't implemented due to the prototype nature of the project given that there were perhaps other features more important to implement in the given time, still it's a great idea that will certainly be a good contender to be implemented in a later build;

4.8.2 Admin component evaluation

Question	Participant			
	1	2	3	4
1. I think that I would like to use this system frequently.	2	5	4	5
2. I found the system unnecessarily complex.	4	1	2	2
3. I thought the system was easy to use.	3	5	3	4
4. I think that I would need the support of a technical person to be able to use this system.	2	2	1	3
5. I found the various functions in this system were well integrated.	2	2	3	4
6. I thought there was too much inconsistency in this system.	2	1	2	1
7. I would imagine that most people would learn to use this system very quickly.	3	4	4	5
8. I found the system very cumbersome to use.	3	1	2	1
9. I felt very confident using the system.	2	4	4	4
10. I needed to learn a lot of things before I could get going with this system.	2	2	1	3
1. SUS score	47.5	82.5	75	80

Table 4.6: SUS scores for the Admin component

The average SUS score for the Admin component was 71.3 which translates into a good score given that the base score for an above average system is 68. Some improvement suggestions from the participants were:

- Notifications for new requests entering the system;
- group service requests by client;

Some considerations regarding participants' suggestions:

- notifications for new requests entering the system is a good idea that wasn't implemented due to the prototype nature of the project, given that admins will spend a big part of the day looking at their control pages and processing requests in this initial development of a prototype, notifications weren't considered as a necessity, but they are a good contender to be implemented in the future.
- regarding grouping service requests by clients, it is an interesting suggestion we don't see too much value in implementing, given that keeping the requests atomic (1 service request corresponds to a request of one service only) will make the response to it faster instead of grouping more than one service in a request.

Despite not having garnered a big amount of participants, we still believe these evaluations provide somewhat of a valuable insight into the potential of the App and what ways to pursue it in order to improve it and make it a final product.

Chapter 5

Conclusions and Future Work

In the context of the small business industry, a professional has to deal with complex logistical aspects for contracting the various services required to run their business namely: electrician service, plumbing, accounting, showcase display services, among others. This need to manage and contract each service individually can be constraining and laborious. The implemented solution seeks to streamline and simplify this process by allowing a user to subscribe to the services they will need the most and call them whenever easily. For this to happen an administration back office needed to be implemented as well, where admins would be able to manage each user's service and registration requests among other aspects of the business, this back office contains some gamification elements such as admins gaining points for actions within the system, a leaderboard to compare performance and achievements/badges that provide points upon completion, this was implemented as a way to drive up engagement and Admin performance within the system. This work started as an App intended for the hairdresser sector as a way of facilitating business processes by taking away some menial attention and worries needed to manage a business, relegating some aspects of it to the App. Later the scope was broadened to include any kind of small business, given their equal need for some of the same services. The user part is simple as intended, it was designed, so it would bring as less hassle as possible to the consumer, the Admin back office is simple in design and provides a clear and workable environment with minimal clutter, where everything is easily reached and intuitive given the main intent for the App to be used in mobile devices.

5.1 Satisfaction of the Objectives

Confronting the initial objectives with the work developed, we can reach the following conclusions:

- Explore and study an approach and technologies to design a mobile application for business professionals;

- several approaches and technologies were studied in order to select the ones that best fit the work at hand, the possibility of developing an android application and iOS separately, developing a web App or a Progressive Web App, among many different technologies were considered such as, for iOS the Swift language, for android either Java or Kotlin, for a PWA, React and for back-end PostgreSQL and Firebase among others;
- Develop an App that facilitates the hiring of different services;
 - We believe that the developed App satisfies this point by providing a list of subscribable services at the point of registration that later can be requested at any time;
- Develop an intuitive and easy to use App
 - Steps were taken to fulfill this point, namely choosing React and Google’s Material UI to develop the front-end, designing the pages to look as less cluttered as possible and making sure the pages were responsive and provided a good level of usability at all times;
- Provide the ability for the App to be customized and suit different businesses;
 - This objective was accomplished given that the App was developed from the ground up for the business not to matter, the user is given the choice to subscribe to the services they deem most useful for their business thus eliminating the need to adjust to fit different businesses, the business owner already does that by selecting the services that are most useful to them;
- Validate the App and identify aspects to improve with the help of input from a focus group
 - The App was tested and validated with the help of a focus group. The SUS score for the client component was 69.2 and for the Admin component it was 71.3, this indicates a usability above average but with aspects to improve, many of the suggestions from the focus group focused on how the subscription of services was made;
- Provide a gamified back office for administrative efforts;
 - The App features a back office for Admins where they can manage the most relevant aspects of the business processes, like handling users’ registration requests, service requests, manage services, check leaderboard and achievements and add new Admins to the system;
 - The back office is gamified in which Admins earn points for different actions within the system as well as achievements and can compare their rankings on the leaderboard, super Admins can also reshuffle current achievements and reset points for every Admin;

5.2 Research Questions

In this section we make the analysis as to whether the research questions posed in chapter 1 were answered, as such we'll go through each one and give our answer:

- What is the best architecture to provide these services?
 - Extensive research was required to answer this question, either by finding different scientific articles, thesis works, real world applications in the market, as well as, exploring different technologies and approaches to tackle the issue. In the end we found the best architecture for our work as it was described before.
- What are the most fitting technologies to develop our App?
 - For this question we had to do a deep dive into what kinds of technologies and approaches existed to develop our solution, we found that a PWA using React, Google's MUI, Firebase for authentication and back-end and Heroku as a means to host the App were the most fitting technologies to develop our App.
- How can we gamify the App in an engaging way?
 - Research for this topic included, finding related concepts, articles, game elements and research on gamified Apps. We found that we answered this question and successfully used gamification with the implementation of a leaderboard, points and achievements on our solution.

5.3 Future Work

The project is currently in its first iteration and as such, there still exist plenty of features and improvements in order to fully realize the idea and integrate it in the market. The project also needs to be tested by a significant sample of users, which didn't happen given the pandemic situation we live in. As such, the possible improvements are divided into 3 categories, general improvements, improvements for the client component and improvements for the Admin component.

5.3.1 General Improvements:

- Bigger sample for testing;
- Further develop and improve the UI and UX;

5.3.2 Client Component Improvements:

- Add status progress of a given requested service;
- Handle service subscription after register;

- Separate service subscriptions into packs instead of individually;
- Add a user Profile page and notifications;

5.3.3 Admin Component Improvements:

- Add notifications for when new service requests or registration requests appear;
- Filter services to choose for a given service request by distance or quality to the client;
- Add a search bar to the multiple pages to make filtering information easier;
- Provide other admin functionalities, for example a super admin being able to remove other admins or users from the system;

References

- [1] “Andrew Mason - I didn’t realize how hard it was to run a...” [Online]. Available: https://www.brainyquote.com/quotes/andrew_mason_449370
- [2] M. Bildner, *Groupon: A Brief History of the Rise and Fall of Andrew Mason*. Daily Deal Media. [Online]. Available: <http://www.dailydealmedia.com/985groupon-a-brief-history-of-the-rise-and-fall-of-andrew-mason/>
- [3] “business | meaning of business in Longman Dictionary of Contemporary English | LDOCE.” [Online]. Available: <https://www.ldoceonline.com/dictionary/business>
- [4] S. Chaudhuri, U. Dayal, and V. Narasayya, “An overview of business intelligence technology,” *Communications of the ACM*, vol. 54, no. 8, pp. 88–98, aug 2011.
- [5] “Restaurant Tech: 8 Apps you Need to Add to your Kitchen Lineup | Lightspeed HQ.” [Online]. Available: <https://www.lightspeedhq.com/blog/restaurant-tech-8-apps-you-need-to-add-to-your-kitchen-lineup/>
- [6] H. A. Al-Homery, H. Asharai, and A. Ahmad, “The Core Components and Types of CRM,” *Pakistan Journal of Humanities and Social*, vol. 7, no. 1, p. 121. [Online]. Available: www.pjhss.com
- [7] “What is Gamification anyway? | HCI Games Group.” [Online]. Available: <https://hcigames.com/research/what-is-gamification-anyway/>
- [8] S. Deterding, D. Dixon, R. Khaled, and L. Nacke, “From game design elements to gamefulness: Defining “gamification”,” *Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments, MindTrek 2011*, pp. 9–15, 2011.
- [9] “What is Gamification.” [Online]. Available: <https://yukaichou.com/gamification-examples/what-is-gamification/>
- [10] K. Werbach and D. Hunter, “For the win : how game thinking can revolutionize your business,” p. 144, 2012. [Online]. Available: https://books.google.com/books/about/For_the_Win.html?hl=pt-PT&id=aL2tzgEACAAJ
- [11] “9 Examples of Apps Using Gamification for User Engagement.” [Online]. Available: <https://blog.getsocial.im/is-gamification-the-only-way-for-apps-to-survive/>
- [12] “Duolingo - A melhor maneira do mundo de aprender um idioma.” [Online]. Available: <https://www.duolingo.com/>
- [13] “Habitica - Gamify Your Life.” [Online]. Available: <https://habitica.com/static/home>

- [14] M. Brochado Costa, O. na FEUP, P. Teresa Sarmiento Orientador na Agros UCRL, and E. Jacinto Rui Santos, “Desenho de um serviço móvel no contexto do setor agropecuário,” Tech. Rep., sep 2017. [Online]. Available: <https://repositorio-aberto.up.pt/handle/10216/106792>
- [15] “What is Uber Eats? | Uber Eats - Uber Help.” [Online]. Available: <https://help.uber.com/ubereats/article/what-is-uber-eats-?nodeId=fbf73e2a-c21f-4a48-8333-c874ae195fd1>
- [16] “About us.” [Online]. Available: <https://about.glovoapp.com/en/>
- [17] A. S. Ferreira, “Aplicação móvel ludificada para Introdução Criativa à Programação,” Tech. Rep., apr 2021. [Online]. Available: <https://repositorio-aberto.up.pt/handle/10216/133698>
- [18] “Zoho - Cloud Software Suite and SaaS Applications for Businesses.” [Online]. Available: <https://www.zoho.com/>
- [19] “4-Step Guide to Convert an Android App to iOS and in Reverse.” [Online]. Available: <https://mlsdev.com/blog/how-to-convert-android-app-to-ios>
- [20] “Getting Started with iOS App Development | AWS.” [Online]. Available: <https://aws.amazon.com/mobile/mobile-application-development/native/ios/>
- [21] “Android Studio Overview | Android Developers.” [Online]. Available: <http://android.cn-mirrors.com/tools/studio/index.html>
- [22] “Android Studio features | Android Developers.” [Online]. Available: <https://developer.android.com/studio/features>
- [23] “Xcode for Windows (12 Ways to Build iOS Apps on PC).” [Online]. Available: <https://codewithchris.com/xcode-for-windows/>
- [24] “Xcode - Features - Apple Developer.” [Online]. Available: <https://developer.apple.com/xcode/features/>
- [25] “What are Progressive Web Apps?” [Online]. Available: <https://web.dev/what-are-pwas/>
- [26] “React – A JavaScript library for building user interfaces.” [Online]. Available: <https://reactjs.org/>
- [27] “The History of React.js on a Timeline | @RisingStack.” [Online]. Available: <https://blog.risingstack.com/the-history-of-react-js-on-a-timeline/>
- [28] “Flutter - Beautiful native apps in record time.” [Online]. Available: <https://flutter.dev/>
- [29] “PostgreSQL: About.” [Online]. Available: <https://www.postgresql.org/about/>
- [30] “Express - Node.js web application framework.” [Online]. Available: <https://expressjs.com/>
- [31] “Firebase.” [Online]. Available: <https://firebase.google.com/>
- [32] “Google Acquires Firebase To Help Developers Build Better Real-Time Apps | TechCrunch.” [Online]. Available: https://techcrunch.com/2014/10/21/google-acquires-firebase-to-help-developers-build-better-realtime-apps/?guccounter=1&guce_referrer=aHR0cHM6Ly9lbi53aWtpcGVkaWEub3JnLw&guce_referrer_sig=AQAAAEU51_90OHzw-fovKn1uEKggYEtR1a7rtfIkC-UbIYQ1LKLRftOURqhrzNwXhYDHNcB1m_4htMQtoieMOKa9zjpWo7HtJygo4GNSqon25h6DF-fL3mnGEU-jb2gf_J_rvn0qBDbnvShbd_ucT8w11-unIsOgVmuVNB6Et0YEaXqh

- [33] “Cloud Firestore | Firebase Documentation.” [Online]. Available: <https://firebase.google.com/docs/firestore>
- [34] “Firebase Authentication | Firebase Documentation.” [Online]. Available: <https://firebase.google.com/docs/auth>
- [35] A. S. f. P. Affairs, “System Usability Scale (SUS),” sep 2013.