Gym-Suite

An extensible platform for gyms management

Master degree in Computer Engineering-Mobile Computing

Royli Hernández Delgado

Leiria, November of 2021

Gym-Suite

An extensible platform for gyms management

Master degree in Computer Engineering-Mobile Computing

Royli Hernández Delgado

Project Report under the supervision of Professor Paulo Jorge Gonçalves Loureiro, and Professor Marco António de Oliveira Monteiro.

Leiria, November of 2021

Originality and Copyright

This project report is original, made only for this purpose, and all authors whose studies and publications were used to complete it are duly acknowledged.

Partial reproduction of this document is authorized, provided that the Author is explicitly mentioned, as well as the study cycle, Master degree in Computer Engineering-Mobile Computing, 2020/2021 academic year, of the School of Technology and Management of the Polytechnic Institute of Leiria, and the date of the public presentation of this work.

Dedication

A mi Padre por todo.

Abstract

The main objective of this work is to build a platform for gym management. The work includes a study of different Gym Management Systems to determine the essential elements of such kind of system. The study was carried out based on a group of indicators that allowed to identify the elements that should be considered essentials in a GMS. To study the relevance of the use of GMS in the district of Leiria, a survey was carried out with the owners and/or managers of gyms in the area. The study also served to contrast the study on the desired characteristics for a GMS and the perception that gym staff have about them.

Another of the elements included in the research work are the study of the main methodologies, tools, and technologies used in the implementation of a GMS. The system must be a solution that includes a web application and a mobile application. These technologies include Microsoft Azure, Terraform, Microsoft ASP NET Core and React. Scrum was used as a software development methodology.

The core concept of this architecture is the concept of Microservice. Microservices is an architectural style that structures an application as a collection of services. The idea is to group the businesses that look similar in the system and develop them as individual units. The development of a microservices system becomes to develop small system's units that are more manageable, maintainable, and scalable. The GMS modules were implemented according to the design of the application screens.

The implementation and testing stages demonstrated one of the main qualities of the GMS design, its extensibility. To test the correct functioning of Gym-Suite, unit, integration, and system tests were carried out. The tests results were satisfactory. The deployment of the Gym-Suit GMS also was successful.

Keywords: Gym Management System, extensible platform, mobile application, microservice, microfrontend.

Contents

List of Figures

List of Tables

List of Abbreviations and Acronyms

ASP	Active Server Pages
API	Application Programming Interface
CC	Cloud Computing
CD	Continuous Development
CI	Continuous Integration
CLI	Command Line Interface
CSS	Cascading Style Sheets
DB	Database
ESTG	School of Technology and Management
GACS	Gym Access Control System
GMS	Gym Management System
GPS	Global Positioning System
GUI	Graphical User Interface
HTML	Hipper Text Mark-up Language
HTTP	Hypertext Transfer Protocol
ICT	Information and Communications Technologies
IT	Information Technology
RDBMS	Relational Database Management System
RFID	Radio-frequency identification
SaaS	Software as a Service
SIT	System Integration Testing
SQL	Structure Query Language
UI	User Interface
UID	User Identification Number
URL	Uniform Resource Locator

1. Introduction

Most of the people take care of their health and looks in these days, watch what they do, eat, breathe, and drink. One of the most popular activities in the recent years is to go to the gym with the goal to feel and look good. Gyms have become one important industry in the leisure activities area. All around the world big and small gyms are popular, most of them use the Information and Communication Technologies (ICT) to manage their activities, the relationships with customers, the gym equipment, and other things.

Today ICT impact in different ways in the gym industry as in the rest of the human activities. Computer applications use different technologies and interact with the user using different interfaces. One kind of computer applications that become very popular in the last two decades are mobile applications, nowadays there is an app for nearly everything. There are currently more than two million apps on Google Play and on the Apple App Store. There are in these platforms a lot of mobile apps that offers different ways to achieve the goal of be fit or stay fit [1], some of them are linked to smart watches or to the gyms [2].

There are a diverse kind of computer systems that allows to control the different aspects of a gym, from the equipment, instructors, classrooms to the users [3] [4] [1]. Those systems are known under different names, but in the present research we will always be referring to them as Gym Management System (GMS). Even with different names, all systems assume the same goal: *"to control the resources and customers of the gym"*. In a gym, resources can be the time, equipment, instructors, venues, and other things that can be related with gym's customers. Relationships among resources and customers are a very important point and must be represented in the GMS.

In the present the GMS use different user interfaces, web, mobile and even desktop's User Interface. However, due to the increasing use of the mobile devices most of the studied GMS offer mobile applications to their users [5] [6] [4] [1]. One important characteristic of mobile apps in relationship with the gyms is the degree of customization that they allow. Mobile apps allow to the user perform different actions: booking a class, an exercise machine, or a venue; watch a demonstrative exercise video; pay the bills among many other things. Having a mobile app related with the GMS lets a more direct and private relationships between the

user and the gym from any place at any time and those are goal of any mobile apps, give the power to the user to interact with the institution.

According to the information in the TD Systems web site [2] *"To control the gym's access is a fundamental task for the gym's effective working. Is about to have the devices and the software for control the customer flow and to gather meaningful statistics. The collected statistics will help in the process of build the commercial strategies of the institution"*. We can infer that the gym access control is one of the most fundamental pieces in a GMS.

An access control system is a fundamental piece in every institution. A Gym Access Control System (GACS) provides an easy and convenient way for members to access the gym's facilities giving the gym's managers the security to control the movements of the gym´s members. GACS are a way to monitor who accesses the gym areas and who uses the gym equipment's. It is not only the software but also the gadgets that control the access. Usually, the Access control is made using different kind of physical devices, such as card readers, swivel, or biometric equipment [7]. Most of these kinds of equipment are ready to link with the GMS or can be programed using microcontrollers. Access control relies in the enrollment process and these two elements are in almost every commercial GMS. GMS are designed to make it easy to maintain detailed records of members and their memberships, book classes and trainers, process and track sales, and communicate with the right members at the right time.

Building a GMS as a whole application using the traditional software development methodologies can be a complex process. The microservice's ease of use and development can simplify the building process of a GMS. Other technologies like the Cloud Computing (CC) and the development under the Software as a Service (SasS) model can also reduce the development complexity. Using CC and SasS brings into the front line the desired characteristics in the Software Industry, modularity, and reusability. From the point of view of a GMS the software is made of small manageable units. In each unit the developer can enclose different views of the GMS and will offer to the users' different perspectives according to their needs.

Today several commercial GMS products include a mobile app that allows to the gym´s customers make actions like book and pay for classes, courses, appointments, and memberships, and connection with the social media platforms. An important characteristic of mobile apps is that they can connect people from anywhere, from their homes, workspaces

or other locations. Gym´s customers can use their time in a more efficient way. With mobile apps, the users do not need to go to the front desk to do the bookings and payments related with the Gims's services. These mobile apps also provide a good way and the tools that the managers need to engage new members, simplify the day-to-day, and outrival the competition.

1.1. Motivation and goals of the research

The main motivation for this work is to build a suite of applications that allows managing the main elements of a gym and that complies with the minimum core that an application of this nature must have. Within this suite there would be a mobile application for users from the gym can interact with the gym services.

The main goal of the work is ***to build an extensible platform to manage the basic services offered by gyms***.

In order to fulfil the main goal, the following particular goals must be met during the research:

- To establish a theoretical framework, identifying the basic concepts and technologies that will allow to build an extensible platform to manage the basic services offered by gyms.
- To design the extensible platform Gym-Suite.
- To implement a functional prototype of the extensible platform Gym-Suite.
- To test and to deploy the extensible platform Gym-Suite.

Among the methods of scientific work used in this research are: the systemic method for the development of computer systems and achieve that the elements that are part of the real application are a whole that works together as a unit; the historical-logical method for the critical study of the previous works, and to use these as a point of reference and comparison of the achieved results, the analytical-synthetic method by decomposing the research problem into separate elements and deepening the study of each of them, then to synthesize them in the proposed solution; the modeling method for the development of the extensible platform result of the research, a model of the extensible platform must be done to fulfill the main goal.

The research has among its objective to demonstrate that the application of cloud computing and an architecture based on microservices allows to facilitate the construction and / or extension of a platform for the management of gyms and other institutions with the same goals.

1.2. Document structure

The document is composed by an introduction chapter, which describes the work to be done, the structure of the research, what has been done and sets out the main goals and the expected results of the research.

The second chapter (A state-of-the-Art of Gym Management Systems) present the theoretical and practical elements that constitute the framework of the research, and that allowed to determine the essential elements of the work. In this chapter different GMS are studied using a set of indicators, the basic core of services that a system of this type must offer are determined. The second chapter also present the result of a survey applied to a group of gyms managers.

The third chapter (GMS´s implementation technologies) briefly characterizes the technologies that be used in the development of the proposed solution. These technologies are what allow you to build a SaaS platform that is extensible.

The fourth chapter (Gym-Suite Implementation) is dedicated to the design of the Gym-Suite platform. The chapter contains all the artifacts that allowed its implementation and the realization of the different kind tests to prove that the implemented solution is robust. The chapter also includes the description of the architecture, as well as the rest of the diagrams that allowed using an agile methodology the implementation of the extensible platform Gym-Suite.

The fifth chapter (Gym-Suite Testing and Evaluation) exposes the results of the Gym-Suite implementation and testing phases. The tests allowed in addition to testing the correct functioning of the platform the ease to add new functionalities to the suite, was taken as a base functionality the access control, and as additional functionalities the management of payments and the module of courses.

The document includes the conclusions, chapter six, derived from the study carried out and recommendations that contain elements to be studied and to put in practice.

A group of appendices that complement what is included and help to clarify all that has been done. Appendix A has the survey, Appendix B has the survey responses, Appendices C and D has some views of the Gym-Suite Platform and Appendices E and F the access control response test and the Arduino gadget to control the door.

2. A state-of-the-Art of Gym Management Systems

The goal of this chapter is to present the theoretical foundations of the research. It presents the study carried out on a group of GMS based on the definition of a group of indicators. Those indicators were defined to characterize the GMS. The analysis of these systems will make it possible to determine the basic functionalities that a GMS must offer. The chapter also present a survey made by some of the Gym's owners in the Leiria city.

2.1. Gym Management Systems

According to G2 [8], a company dedicated to reviews and comparisons of GMS, a Gym Management System is a software that is designed to help managers and owners to organize and to manage all aspects of this type of business. The authors of this definition, place in the center of the GMS the processes related with the access control, the enrollment, the financial management, and the communication with members.

GMS are implemented using a variety of technologies, from web portals to desktop apps, but it is important to think about the GMS as a collection of software and not as a monolithic solution. Sometimes the GMS include a portal that allows access to its members to make reservations of venues or other resources of the institution, make payments watch instructional videos. Some of these solutions provide a web application for members to interact with the institution in a more fluid way. It is possible also that the web application is a small part of a bigger portal.

To count with a platform ready to manage the gyms is a necessity for managers and/or owners of the gyms. These platforms avoid having to split the management of access control, payments, accounting, membership, using different systems.

For the analysis of the GMS, a group of applications were studied, using a set of indicators, with the aim of carrying out its characterization. This characterization allowed to know what services they offer, what interface they use, whether they are paid or not, whether they have a mobile application for the users.

The defined indicators shown in Table 2.1 have been defined as a way to identify the main characteristics that a GMS should have. Later, a table will be presented that summarizes each

of the platform's characteristics and identifies if the platform has or has not that characteristic.

Table 2-1. GMS's analysis set of indicators

Indicator	Description
User profiles (UP)	The GMS include facilities to keep personal information about the customers, and to track their data related to payments, workout routines, etc.
Access control (AC)	The access control system is included and is made according to the type of user or other relevant elements as payments.
Financial management (FM)	The system includes tools for financial management, payments, invoices, cash handling, and other financial information.
Activity planning and resource control (AP)	Tools are included for the planning of activities, enrollment in an activity, cancellation, and control of the resources of the institution (equipment, machinery, for example).
Notifications (NF)	Notifications are issued to the user, notifications can come from financial management, activity planning or other areas of the GMS.
Formal Training Sessions (CL)	It includes facilities to put activities of this type on the platform.
Mobile app (MA)	It has at least mobile app for users. The mobile app allows users to interact with the gym.

The following GMS were studied: GestiGym Cloud [9], GymMaster [3], Softar-gym [4], PerfectGym [6] and VirtualGym [10].

In each case a small analysis of the GMS was done to fill a summary table. The final table will help in the process to identify what are the elements that should have this kind of systems.

2.1.1. GestiGym Cloud

GestiGym is a company dedicated to the sale, installation and configuration of access control and sports management software for gyms, sports centers, personal training institutions, swimming pools, dance schools, martial arts, climbing walls and yoga centers. It provides its clients with the necessary tools to be able to carry out gym management and control of their clients with the support of a technical team [9].

This company offers the GMS GestiGym Cloud that allows you to control customers, financial management, gym staff, classes, and other services. It also includes the management of training routines; in each activity the system will collect data from and for users [9].

The system is offered in the form of packages as you can see in the Illustration 2-1.

Illustration 2-1. GestiGym's example package (15/5/2021)

As we can see in the Illustration 2-1 a package contains the physical devices, the applications it contains and the access cards for the members. Packages differ in the access control gadgets, number of users, available cards, and other items. The price of the packages offered by the company varies between 999 and 3,999 Euros.

The software has some modules for: customer management, payment management, issuance of monthly receipts, payment reporting, reservation management, warehouse and employee control, and training routines management [9] [1].

The company that offers the product has a wide range of solutions for access control, locks for lockers and printers. On the product site you cannot reach conclusions about some of the stated indicators.

2.1.2. Softar-gym

Softar [11] has built the GMS named Softar-gym [4]. The company promotes its product as a *"Softar Gym Management Software for gyms and sports centers"* emphasizing in its versatility and ease of use.

The software has among its characteristics the management of customer's records, planning of activities and resources, enrollment in planned activities, treasury, access control.

The company does not specify in its website whether it has a mobile application for subscriber access to the services, although it does interact with them using mail, the email templates can be customized.

The company includes in the initial offer a tourniquet with RFID reader and 250 cards for the price of € 1,800. Even though it is promoted as a GMS on the official website of the product emphasis is placed on access control. The company itself consider this system more a GACS than a GMS [4].

In the search carried out was not possible find images of this system, only the references and documents included here. The images in the document does not have sufficient quality to infer information from their analysis.

2.1.3. GymMaster

According to Gym Master Software advertisement products, "*GymMaster is gym software designed to make it easy to maintain detailed records of your members and their memberships, book classes and trainers, process and track sales, and communicate with the right members at the right time*." [5].

The system has the following modules: access control, customer management, finance, gym resources. It is a comprehensive software, and even has two applications for mobile devices one for the gym staff and another for its customers.

From the point of view of access control, the system proposes an application with availability of 24-7. The system has a reminder for customers' birthdays, among other strategies that promise to enhance customer retention.

Illustration 2-2 shows some views of the GYMASTER app. In the illustration 2-2 you can see some elements related to a user, for example, the schedule of classes and reservations made.

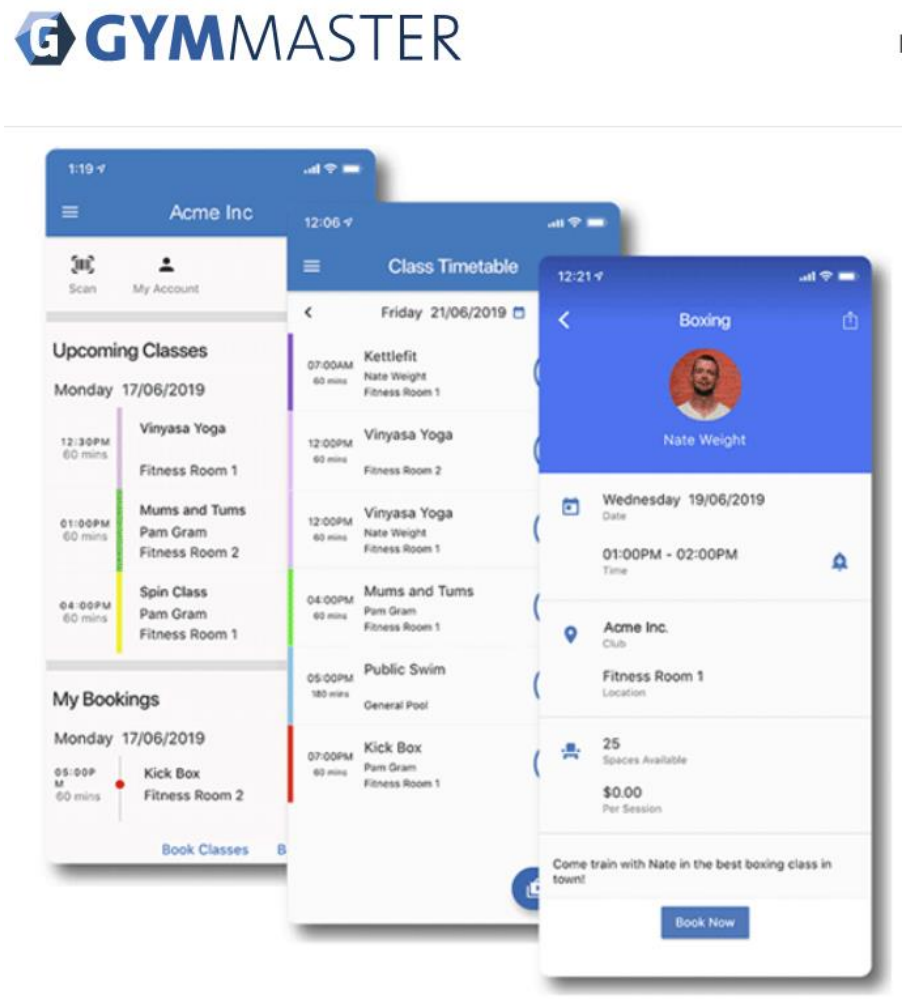


Illustration 2-2. GymMaster mobile app views [5]

By the information offered on the company's website and product this software meets most of the indicators defined for the GMS analysis. Prices for the solution are not listed in the company' web site.

2.1.4. PerfectGym

The company PerfectGym offers a product of the same name. According to PerfectGym it has been designed with the tools that are needed to manage a fitness club in an efficiently way [6].

The software has a portal for customers, a point-of-sale terminal, an access control module, a control panel, customer management, tools for automation and a mobile app.

Illustration 2-3 shows two different views of the application. Both views are of mobile devices, the first on a tablet and the second on a smartphone. The image shows the activities performed by the user and the schedule of the activities of the gym. The gym's activities are

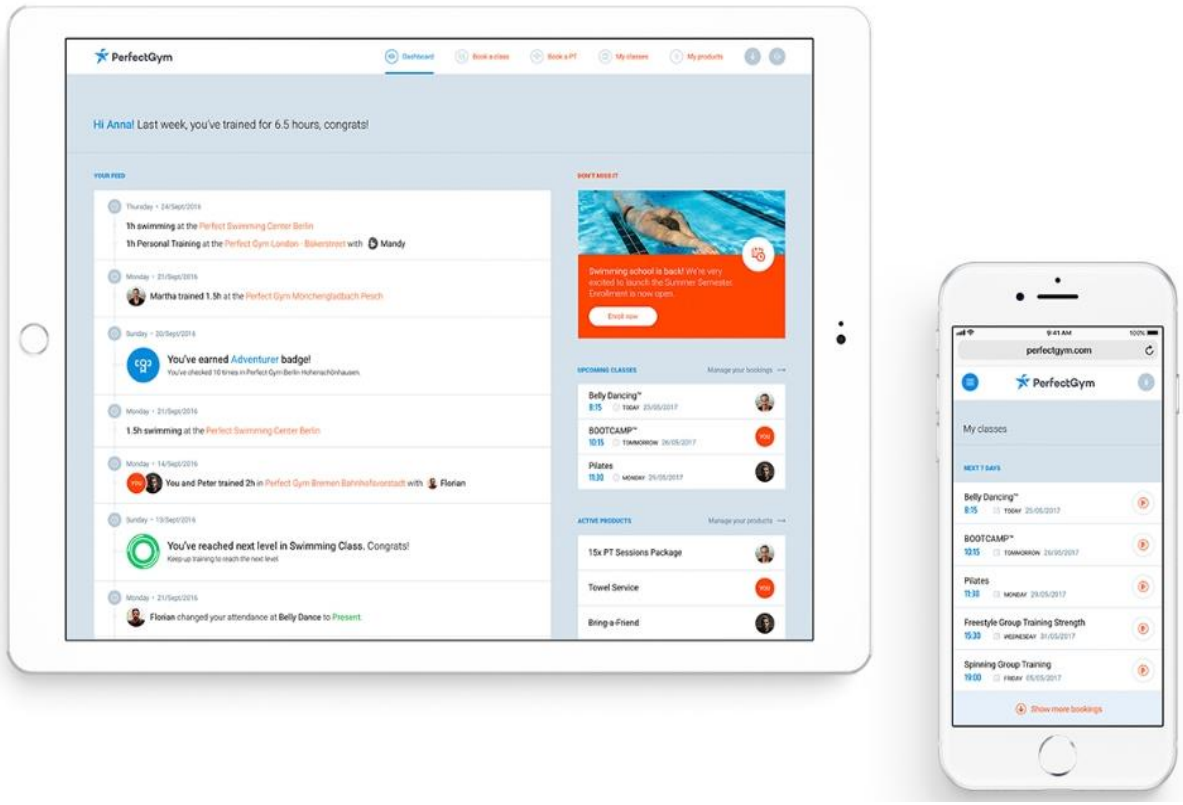


Illustration 2-3. PerfectGym in different mobile devices [6]

one of the elements that enhances the information on the company's website is that both access control and management software are integrated into a single application. As an added value the software has a data analysis tools which allows to the owners an analysis in search of customer retention. It is one of the GMS that declares to use the cloud computing and its services.

2.1.5. VirtualGym

VirtualGym is a cloud application available in multiple languages that offers an integrated solution for gyms and related institutions. Among its features it has the customer management, booking of equipment and classes, as well as tools for remote training [10].

This solution offers two applications, one for personal use that includes workouts and monitoring nutrition and a second application for the company's staff and include the tools that allows customers retention, financial managements among other elements [10].

Illustration 2-4 shows a calendar where a user's activities are found when accessed from a computer. The lower right corner shows the different activities that can be booked in the gym, the view is from the mobile app.

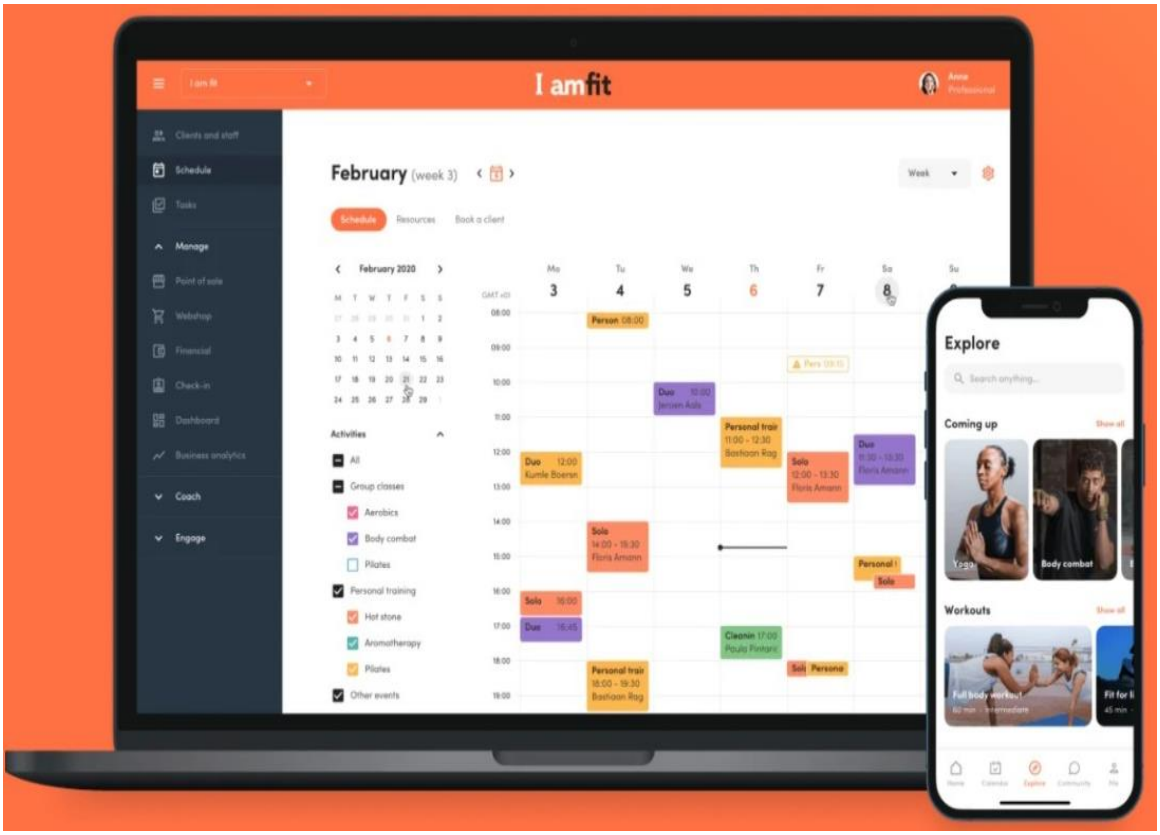


Illustration 2-4. Virtual Gym´s application images

This application has a control panel that allows companies to monitor the use of resources, and according to its creators is not focused only on the management of gyms. This solution seeks to introduce a remote solution to allow its customers to record their actions, so it seeks those coaches and customers also have their owns benefits.

2.2. Summary of the studied systems

The Table 2-2 summarize the indicator's value for every GMS that was analyzed. The value for the indicator will be YES when the described characteristic is met, NO otherwise and NULL if the product site or documentation is not explicitly stated. Table 2-1 shows the description of the indicators that were taken into account: User profile (UP), Access control (AC), Financial management (FM), Activity planning and resource control (AP), Notifications (NF), Formal Training Sessions (CL) and Mobile app (MA).

Table 2-2. Values of the indicators defined for each GMS studied

GMS	Indicator						
	UP	AC	FM	AP	NF	MA	CL
GestiGymCloud	YES	YES	YES	NULL	NULL	YES	YES
Softar-gym	YES	YES	YES	YES	NULL	YES	YES
GymmMaster	YES	YES	YES	YES	YES	YES	YES
PerfectGym	YES	YES	YES	YES	YES	YES	YES
VirtualGym	YES	NULL	YES	YES	YES	YES	YES

From the information in the Table 2-2, we can infer that the basic functionalities that a GMS must have, are access control, user profiles, and financial management. All the solutions analyzed have at least one application for mobile devices to allow access to users.

Only one of the systems, PerfectGym, states that it is extensible, being possible to interface with other systems. Although many of the companies that develop these products do not advertise the technologies used, their name and the existence of web interfaces means that many of them are made and supported cloud computing technologies.

2.3. The GMS's survey in Leiria Gyms

To complete the initial study on the research subject a survey was carry out. The goal of the survey it is to gather information about the characteristics and the use of GMS in a group of gyms of Leiria. Leiria is a city and a municipality in the Centro Region of Portugal and in the historical province of Beira Litoral. It is the capital of Leiria District. The population of the area is about 128,640 inhabitants according to the 2021 census [12].

According to Google Maps in the Leiria city exist a total 18 gyms, as shown in Illustration 2-5.

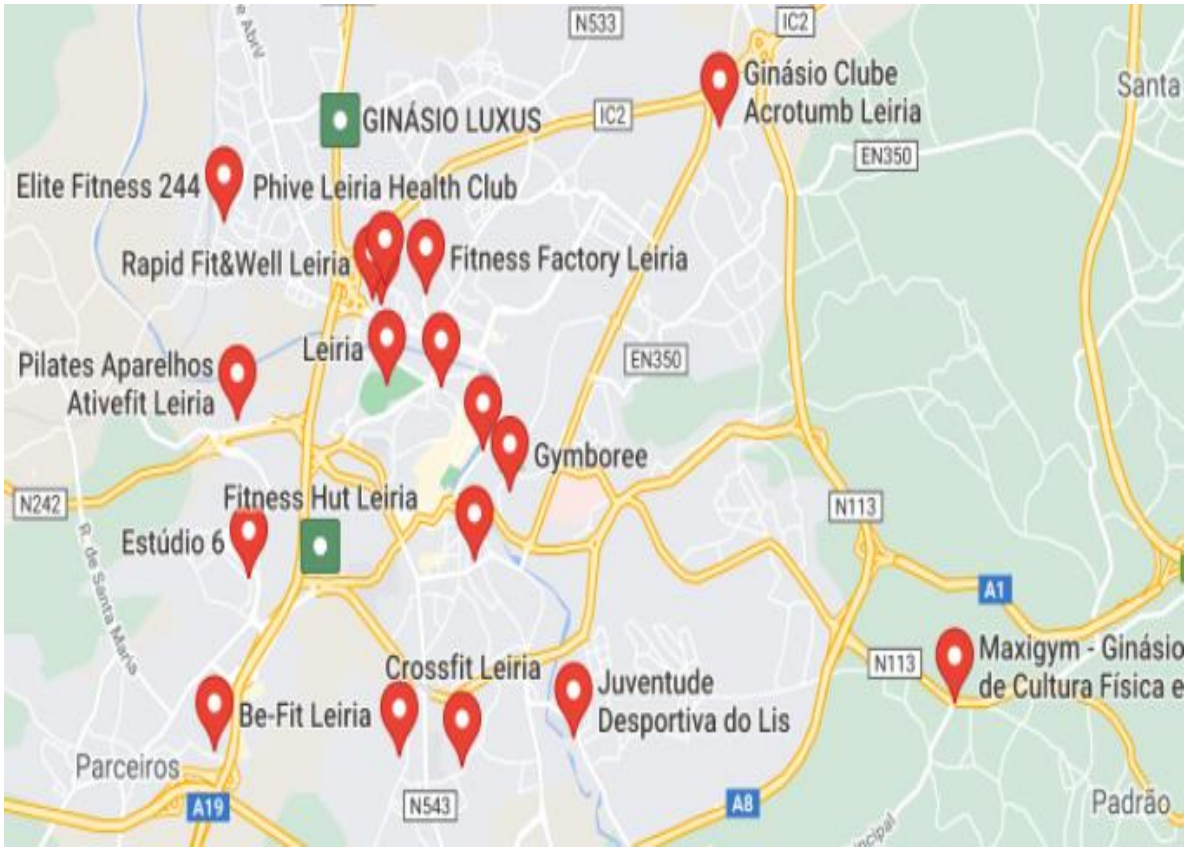


Illustration 2-5. Leiria' Gyms according to Google Maps

Table 2-3 shows the survey template and in Appendix A you can find the surveys answered by the gymnasiums. The survey has seven sections: scheduling, customer, administrative tools, back-end organization, notifications, and client portals. Each section has its own goals represented in the questions made.

Table 2-3. Survey (English-Template)

Gym:				
Clients	Response		Recommended	
	Yes	No	Yes	No
Customers have their own online profile				
Customers can apply online				
Customers can update their data				
Trainers can update training results in user profiles				
Customers can view their subscriptions and payments				

Customers can control their access to the gym from their mobile phone				
Schedules	Response		Recommended	
	Yes	No	Yes	No
The gym has an online class schedule				
Gym teachers plan class schedules				
Customers can view their class schedule on a web or mobile application				
Customers can schedule their training times				
Management	Response		Recommended	
	Yes	No	Yes	No
In the gym application, training equipment and tools are managed				
Gym workers are managed in the application.				
Employee payments are managed by the application.				
The gym application controls the collection of customer subscriptions (billing)				
The gym application allows you to pay online				
The application allows you to configure subscription plans				
Internal management	Response		Recommended	
	Yes	No	Yes	No
In the gym application, the frequency of clients is controlled				
In the gym application, the entry and exit of customers is controlled				
The gym application controls the number of customers who are in the gym.				
In the gym application, it is controlled which equipment each client uses to carry out their training.				
In the gym application, the workers' schedules are managed				
Notifications	Response		Recommended	
	Yes	No	Yes	No
Customers are notified about their classes				
Teachers are notified about classes				
Customers are notified of subscription payment date.				
Schedule changes are reported in the gym.				
Customers are notified of new plans and subscriptions.				
Customer portal	Response		Recommended	
	Yes	No	Yes	No
Advertising is displayed to users				
Images of the installations are displayed				
Free classrooms are displayed				
Gym subscription plans are published (plan prices)				
Gym class timetables are published				
Software you use:				

Software Hosting: Number of customers:

The first section is aimed at knowing elements related to customers. The section includes questions related to the possibility of online registration, access to payment information, the realization of subscriptions and the use of mobile devices to access the areas of the facility.

The section dedicated to administration, includes the elements that allow to know how the equipment, the payments and the information of the workers are handled. The internal management section is related to the previous one and includes the aspect such as the entry and exit of customers and workers. These elements in both cases are linked to the use or not of a mobile application.

The section dedicated to notifications aims to know how clients, trainers and gym staff are informed about the activities and resources of the institution. Among the elements that are considered important in this section are the notice about payments. The last section of the survey aims to know the features of the customer portal, in case, it is used. This section asks questions about the available information on the gym's web portal.

The possible answers are yes, no, recommended or not recommended. The first two answers are related to know if the featured is currently implemented in the GMS, if a GMS are implemented in the gym. The recommended or not recommended are intended to know the perception of the owners about the item if they consider the feature important or not.

The survey was sent to five (5) gym’s owners of which three (3) responded. A summary of the answers can be seen in the table shown in the Appendix B.

A small analysis of the survey is presented below.

- All gyms have software installed, there is no match between the application used by the gyms since the GMS they use are their own.
- Only the 33% allows the user to enroll and change the data on-line, but the 100% declared that every user has an on-line profile.
- There is not a coincidence in the management of gym’s resources using the installed software.
- Notifications is a big coincidence in all the software used but there are many differences in the way that notifications are delivered.

- The main recommendations are related with the client's on-line profiles and the management of gym's resources.

With the survey analysis we can reach the following conclusion there is a necessity of a software that can help in the management of gyms and the most important is to enhance not only the financial management but also model the customer relationship in this kind of organization.

2.4. Chapter Summary

The second chapter addresses the fundamental elements that allow to know the characteristics of the systems for the control of the gyms. We started by selecting a definition to gym use. A group of indicators are also defined in the body of the chapter to perform the analysis of a group of systems for the control of gyms.

The analysis of several GMS using the defined indicators is included. The chapter also includes a summary analysis of how these indicators behave for the systems analyzed. The chapter concludes with the presentation of the analysis of a survey conducted to several gyms in the Leiria city. The objective of the survey was to know the characteristics of the systems used by the gymnasiums and the perception of those who work with them about the set of characteristics they should have.

After the study made in this chapter the researcher reaches the following conclusions:

- A Gym Management System is a software that is designed to help managers and owners to organize and to manage all aspects of this type of business.
- The basic functionalities that a GMS must have are access control, user profiles, and financial management, the most recommended UI for a GMS are web and mobile.
- The result of the survey carried out to characterize the use of GMS in Leiria give a result that every gym uses a GMS with different characteristics and possibilities, the user´s on-line profile is an intersection point among the systems used but not all have the possibility of the on-line enrollment. Most of the gyms does not use the systems to manage the resources.

The following chapter will address the main technologies that are used for to develop GMS.

3. GMS´s implementation technologies

This chapter contains the analysis of some technologies that were used to build our extensible GMS. One of the goals is to allows users to interact with the GMS from anywhere at any time, and cloud computing mixed with mobile apps are a good choice to fulfill this goal. The technologies that are presented in this part allows to build an extensible GMS according to the elements that were studied earlier in this chapter.

Among the technologies used to develop the extensible GMS are the three base languages for the development of any web application: HTML as the standard markup language for creating and describing the structure of a Web page; CSS as the language to describe how the HTML elements should be displayed and JavaScript to implement the behavior of a web page.

Databases are an important part of any applications, to develop the solution we need database technologies to store the information. To determine the appropriate database technology the data must be characterize according to its level of structure, this characterization will allow to select between relational databases and not relational databases or a combination of these two technologies.

To develop the extensible GMS a combination of Microsoft SQL Server 2019 and Azure Cosmos DB was selected. Microsoft SQL Server is a one of the most used and stable RDBMS.

Azure Cosmos DB is the Microsoft fully managed NoSQL database for modern app development. The key benefits of Azure Cosmos DB are: the guaranteed speed at any scale, the simplified application development, it is mission-critical ready and is fully managed and cost-effective. According to Microsoft: *"Any web, mobile, gaming, and IoT application that needs to handle massive amounts of data, reads, and writes at a global scale with near-real response times for a variety of data will benefit from Cosmos DB's guaranteed high availability, high throughput, low latency, and tunable consistency"* [13].

To develop the extensible GMS we use C# for backend applications, the Microsoft object-oriented programming language. C# allow to use of the power of the .NET platform, and it is an easy and powerful language. Due to the fact that the extensible GMS has a web component the mentioned JavaScript and Typescript also are used. TypeScript is an open-

source language based on JavaScript, adding static type definitions. Adding static type definitions can save time catching errors and providing fixes before the code runs [14].

The analysis of the remaining technologies can be found in the following sections, these sections include only the technologies that the researcher consider most important to fulfill the goal of its research.

3.1. Microsoft Azure

Cloud computing is today one of the main ways to deliver and to consume services, the cloud has become not only popular but a necessary option in this pandemic era. Every major company in the computer business offers their own solution, Microsoft has Azure, Amazon offer Amazon Web Services, Google has Google Cloud Platform and in China Alibaba cloud offer by Alibaba is the primary cloud option.

Cloud computing in 2021 has become the go-to model for information technology as companies prioritize as-a-service providers over traditional vendors, accelerate digital transformation projects, and enable the new normal of work following the COVID-19 pandemic [15].

One of the major players in this technology is Microsoft, one of the biggest companies in the IT sector. Microsoft Azure is a public cloud platform offered by Microsoft. Azure provides a wide variety of services that can be used without purchasing and provisioning hardware. Azure enables the rapid development of solutions and provides the resources to accomplish tasks that may not be feasible in an on-premises environment. Azure's compute, storage, network, and application services allow you to focus on building great solutions without the need to worry about how the physical infrastructure is assembled [16].

Microsoft Azure has a group of core services that help organizations store and manage their data, create complex web apps, improve their cybersecurity and compliance practices, and much more. Azure´s users can scale up their infrastructure to align with their particular needs.

The main benefits of Microsoft Azure, among others are: is a perfect platform for small businesses and established enterprises, on-demand scalability, cost-effective subscription models, security, compliance, and disaster recovery, and high availability [17] [18] [16] [15].

The stated characteristics and the availability of Microsoft Azure are the fundamental reason to select it in the implementation of the proposed solution in this research.

3.2. Terraform and Kubernetes

Hashicorp Terraform[1] is an open-source tool for provisioning and managing cloud infrastructure. It codifies infrastructure in configuration files that describe the topology of cloud resources. These resources include virtual machines, storage accounts, and networking interfaces. The Terraform CLI provides a simple mechanism to deploy and versioning the configuration files to Azure [19].

Terraform is a tool that help in the deployment of a solution, is an easy tool and can be used with Azure. Terraform is also good at deploying an infrastructure across multiple cloud providers, it enables developers to use consistent tooling to manage each infrastructure definition. The previous characteristics were the reasons to select Terraform as a deployment tool.

Kubernetes is the option to deploy the solution. Kubernetes is an open-source container-orchestration system for automating computer application deployment, scaling, and management [20]. Kubernetes can support data center outsourcing to public cloud service providers or can be used for web hosting at scale.

Every system needs to be developed with a set of development frameworks. In this research the development frameworks are ASP.Net Core for backend applications and React for frontend applications. Each one offers the programmer a lot of features that help to simplify the building process, the next part of this chapter analyzes those tools.

3.3. Microsoft ASP.NET Core

ASP.NET is a popular web-development framework developed for Microsoft. ASP.NET is used for building web apps on the .NET platform. The ASP.NET Core is the open-source version of ASP.NET, that runs on macOS, Linux, and Windows. ASP.NET Core was first released in 2016 and is a re-design of earlier Windows-only versions of ASP.NET [21].

[1] https://www.terraform.io

ASP.NET Core is a cross-platform, open-source framework for the development of internet-connected, cloud-based modern web applications [22]. Among the main advantages of ASP.NET Core we can mention: is an open source and a cross-platform development framework, use the increased .NET Core security, allows to use a widely used programming languages as C# to develop a wide range of application types and enables flexibility in the software development process. Other important advantage is the overall performance this implies applications give better reaction times and require less calculation power [23] [24] [25] [26].

According to Microsoft, ASP.NET Core was designed with performance as one of the most important elements, this has been proved by The Web Framework Benchmarks[2] an independent site to assess the web frameworks performance.

As the applications developed ASP.NET Core allow Run web applications on .NET Framework or .NET Core, develop progressive and productive web applications and back-ends services, and IoT apps, an easy integration with the cloud of azure because they both belong to Microsoft will use this framework to develop the platform's backend services, enabling multiplatform services that easily integrate with the Azure Cloud and facilitate the deployment and continuous integration of services.

3.4. React and React Native

React is a JavaScript library for building user interfaces that is used mostly to build single page applications and allows us to create reusable UI components. React Native is a cross-development open-source mobile application framework created by Facebook, Inc. React Native runs on React, a popular open-source library for building user interfaces with JavaScript. [27].

React is component-based, build encapsulated components that manage their own state, then compose them to make complex users' interfaces [28]. React Native runs in a background process (which interprets the JavaScript written by the developers) directly on the end-device and communicates with the native platform via serialized data over an asynchronous and batched bridge [29].

[2] https://www.techempower.com/benchmarks/#hw=ph&test=plaintext

The working principles of React Native are virtually identical to React except that React Native does not manipulate the DOM via the Virtual DOM. React components wrap existing native code and interact with native APIs via React's declarative UI paradigm and JavaScript [29] [30].

Flexibility is one of the main advantages of React Native. The framework allows developers to write native code in languages such as Java or Kotlin for Android, Objective-C or Swift for iOS, and C++/WinRT or C# for Windows 10, which makes the software development process more flexible and faster [31] [32].

In the platform to be developed, React will be used for the creation of frontend applications since it will allow to develop single page applications, due to its popularity it has a great community and a great component and libraries that complement it, and it allows to reuse the logic and the components. To complete the applications of clients, React Native will be used to develop the mobile application since it allows to compile the application for iOS and Android and reuse components and services implemented with React for single page applications.

3.5. Chapter's Summary

In the chapter, an analysis of the main technologies for the construction of GMS was presented. The technologies and languages used to develop the extensible GMS with a web and mobile UI are HTML, CSS, JavaScript, TypeScript, C#, SQL. The databases are a combination of Microsoft SQL Server and Azure Cosmos DB as a way to get the best of the SQL databases and non-SQL Databases to store structured and unstructured data. The Microsoft ASP.Net Core, React and React Native are the core development elements for the web and mobile app that are part of the extensible GMS since ASP.Net Core will be used for the creation of the backend services, React for the part of web applications in the frontend and React Native for the mobile application. The deployment must be done using Kubernetes and Terraform.

In the next chapter se presents the architecture and design of the extensible GMS that aims at this research.

4. Gym-Suite Implementation

The goal of this chapter is to present the work that has been done in the Gym-Suite implementation phase. The chapter begins with the presentations of the tools and methodologies used in all related implementation processes, the design, programming, and testing processes. After the presentation of tools and methodologies the system´s architecture is presented. The programming process is explained according to first present the core unit of Gym-Suite and later the addition of new modules to the platform.

4.1. Tools and methodologies

A group of tools and methodologies was used to develop the Gym-Suite solution, this section briefly described each of them. A methodology is necessary to guide the software development process, Scrum was selected for this research. Several tools and technologies to build the Gym-Suit solution were used. Some of these tools and its relationships will be also described in this chapter.

4.1.1. Software Development Methodology

To develop the Gym-Suite platform Scrum framework was selected as the software development methodology. According to its creators *"Scrum is a framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value"* [33].

Scrum is an agile development methodology used in the development of software based on an iterative and incremental processes. Scrum is widely used and it is considered an agile framework that is adaptable, fast, flexible, and effective. It was designed to deliver value to the customer throughout the project development.

Agile methodologies aim to deliver the right product, with incremental and frequent delivery of small chunks of functionality, through small cross-functional self-organizing teams, enabling frequent customer feedback and course correction as needed. Agile software development methodologies are iterative, meaning the work is divided into iterations [34].

Illustration 4-1 shows the Scrum Framework artifacts and Events. Scrum's artifacts represent work or value to provide transparency and opportunities for inspection and

adaptation. Artifacts defined by Scrum are specifically designed to maximize transparency of key information so that everybody has the same understanding of the artifact. The Scrum Artifacts are: Product Backlog, Sprint Backlog and Increment. The Scrum Events are Sprint, Sprint Planning, Daily Scrum, Sprint Review and Sprint Retrospective [34].

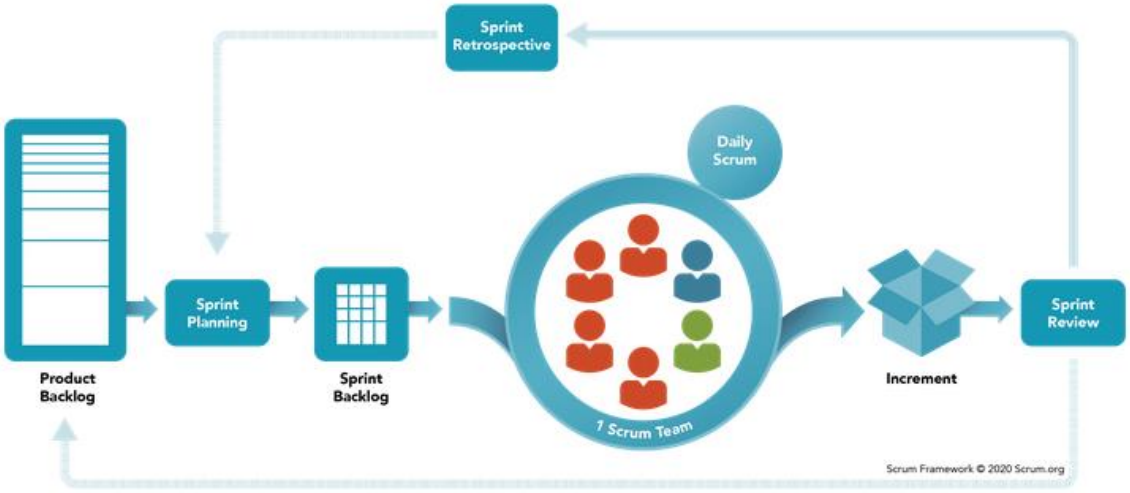


Illustration 4-1. Scrum Framework [33]

"The Product Backlog is an emergent, ordered list of what is needed to improve the product. It is the single source of work undertaken by the Scrum Team. Product Backlog items that can be Done by the Scrum Team within one Sprint are deemed ready for selection in a Sprint Planning event. They usually acquire this degree of transparency after refining activities. Product Backlog refinement is the act of breaking down and further defining Product Backlog items into smaller more precise items. This is an ongoing activity to add details, such as a description, order, and size. Attributes often vary with the domain of work" [34].

The Sprint Backlog is composed of the Sprint Goal (why), the set of Product Backlog items selected for the Sprint (what), as well as an actionable plan for delivering the Increment (how).

One of its main features is that it allows small team development where roles are well identified but can be exchanged. The main roles are the product owner, the scrum master, and the scrum team. An Increment is a concrete steppingstone toward the Product Goal.

Scrum itself can be explained by the following points:

- The product owner makes a priority wish list called the product backlog.
- During sprint planning, the team chose one of the items from the top order of the wish list and decided how they would run the piece.

- The team has sometimes, called the Sprint term (usually two to four weeks) to complete his work, but every day there will be checks to see the progress of the job (Scrum daily).
- Along the way, Scrum master makes the team stay focused on its purpose.
- At the end of the sprint, work must potentially be sent ready to be submitted to the customer, placed on the store shelf, or shown to stakeholders.
- The sprint ends with a sprint and retrospective review.
- As the next sprint starts, the team selects another item again from the product backlog and starts working again.
- This lasts until the project is considered complete, either because of the deadline and budget or by completing the entire list of specified items at the beginning.

The Illustration 4-2 shows Scrum roles and its relationship.

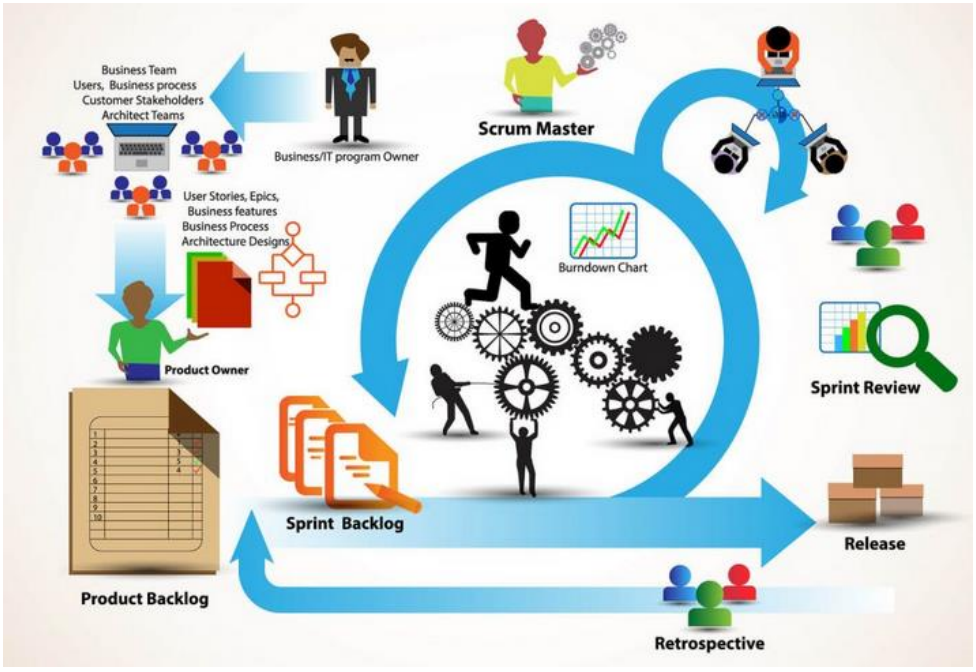


Illustration 4-2. Scrum roles and its relationship

In the context of the research, all the roles that define the methodology were performed by the researcher. According to the questions and answers of the reference web site of the methodology[3] this is possible if the person is an individual with multiple skills, that is, it is the so-called orchestra man. The situation is not an ideal situation mainly because the

[3] www.scrum.org

methodology was designed for team, one individual cannot be considered a team, but the work can be done.

4.1.2. Technologies, tools and frameworks

The implementation of Gym-Suite uses the following technologies, tools and frameworks, this section only includes a brief explanation of those that were not characterized in the previous chapter.

As a tool Visual Studio 2019, Visual Studio Code were used to implement de APIs, Azure functions and the web and mobile application.

Visual Studio 2019 and Visual Studio Code are the main Microsoft developments tools, the first is considered the faster and more reliable than ever. Develop cross-platform and cloud apps with .NET 6, Blazor. Use hot reload capabilities across .NET apps in your developer inner loop. Visual Studio Code is a lightweight development tool. It is a powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages (such as C++, C#, Java, Python, PHP, Go) and runtimes (such as .NET and Unity) [35]. Microsoft Visual Studio code has a marketplace where its users can find a lot of extensions to work with different languages, this marketplace grows every day. Visual Studio code is the programmer's preferred code editor [36].

To build the modeling artifacts Draw.io[4] was used. This is a simple tool to make the different kind of diagrams to represent the architecture, database and other diagrams that are needed to document the Gym-Suite development process. Some of these diagrams are included in the body of the actual chapter and in the appendices.

The framework used to build Gym-Suit were .Net Core, React and React Native. All are a part of the web application development and, they are used to develop the mobile application.

Microsoft Azure Cosmos DB and Microsoft SQL Server 2019 was used as a Database Management Systems. The role of the Microsoft SQL Server database is to serve as a container for the information that the Identity server needs to store.

[4] https://app.diagrams.net/

Microsoft Azure Cosmos DB is a fully managed NoSQL database service for modern app development. It is fast and allow a flexible application development, also it is scalable [13].

4.2. System´s architecture

The core concept of this architecture is the concept of Microservice. Microservices is an architectural style that structures an application as a collection of services. The idea is to group the businesses rules that look similar in the system and develop them as individual units. The development of a platform, becomes to develop small system's units that are more manageable, maintainable, and scalable. In the Gym-Suite solution.

Each microservice is independent and has an API that allows to respond to the requests of the client applications, database (Azure Cosmos DB or Microsoft SQL Server), the functions of Azure that perform the tasks in background and an App-Insights for monitoring. As the microservices are independent of each other to resolve the dependencies between them, a communication is established from events using the EventGrid where all the events of the platform are published.

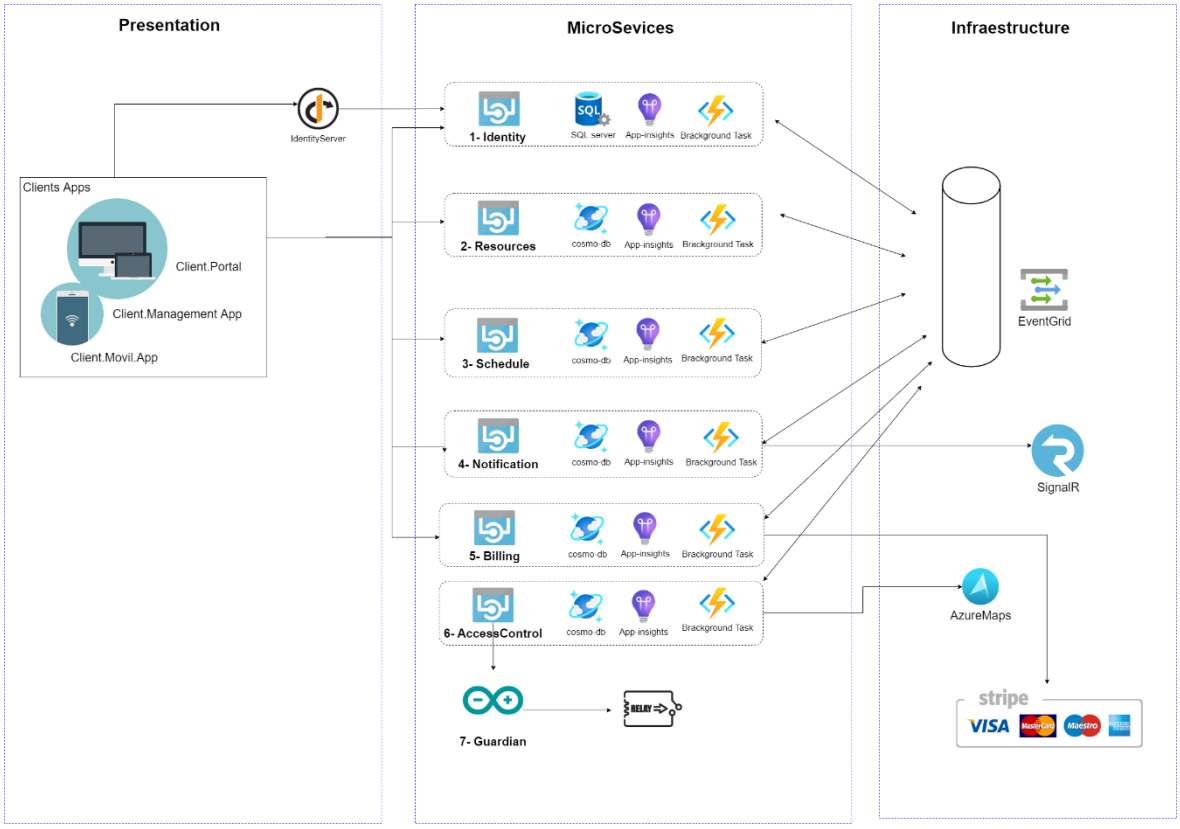


Illustration 4-3. Gym-Suite architecture

Illustration 4-3 shows the architecture proposed for the platform. This architecture groups its components into three layers: Presentation layer, Microservices layer and Infrastructure layer.

The **Presentation layer** contains the client applications that allow users to interact with the platform.

1. Client.Management.App - it is the administration application of the platform. This allows gym administrators to manage resources, schedules, and configuration of areas in which access is controlled.
2. Client.Portal - this application is a landing page that allows you to make visible the main gym offers.
3. Client.Movil.App - this is a mobile application for clients that allows them to interact with the functionalities provided by the gym.

The **Microservices layer** contains the microservices that support the functionalities that have been defined for the gym and is where it is made possible and is prepared for the future addition of new microservices that extend the business of the platform. To meet the needs detected from the survey carried out, the following microservices were defined

1. Identity - controls the identities of users, carries out the processes of registration, authentication, and security between applications.
2. Resources - control the resources of the gym, the clients and the workers of the gym.
3. Schedule: it carries out the control of the schedules, the planning of the classes, the calendar of the users. The planning includes customers and workers.
4. Billing - include control of invoices, payment dates, payment gateways that are used (currently with stripe), users' cards in case they need to be saved for another form of payment and billing addresses.
5. Access Control - is the module in charge of access control to the gym areas, it allows you to define the zones and the permissions on them using groups. It integrates with Azure Maps for the creation of the geofences and the verification of the user's location.
6. Notification - it allows the control of tasks in background belonging to the users, allows to include reminders of different types, birthdays, payment dates, among others.

7. Guardian - controls the relay that opens and closes the door, this is a code written in C ++ that is installed on an Arduino, allows you to run a web server to be able to link the different devices from the mobile application.

The **Infrastructure layer** contains the azure services that are used in the platforms that allow communication between the microservices of the Microservices layer, the notifications that are sent to the Presentation layer and the integration with Azure Maps services. Also, this layer contains the integration with the payment system with Stripe

1. Azure App Configuration - the module stores the Gym-Suite's global configuration. The information related with those configurations are accessible to the rest of the modules as they needed.
2. Key vault - is the module where the access keys are safely store. The module stores the SQL Server, Azure Cosmos DB and others access keys.
3. SignalR - azure service that enables real-time communication between client applications and architecture microservices.
4. EventGrid - event routing service that allows communication between the different microservices of the architecture. Enabling one microservice not to depend on the operation of the other to complete requests from client applications.
5. Stripe - payment gateway that allows to carry out the transactions of the payments made in by the clients.
6. Azure Maps - azure service that provides the necessary functionalities for working with geolocation and definition of geofences on the platform.

To explain the independence of the microservices in this architecture proposal for the platform we can see the example of the microservices Billing. This is a "microservice" that has its own database, its own model and does not depend on any other service. When dependencies exist, because the small units can be related among them, Azure Event Grid can be used to resolve the existing dependencies asynchronously.

Although this architecture is designed with the objective that each microservice is independent of each other, there is a microservice, the IdentityServer, which horizontally in the architecture guarantees the security of each microservice. This microservice controls user access with their credentials and requests from client applications to services through implicit flow.

4.2.1.Microfrontend

A Microfrontend is a piece of the frontend, which a team can independently develop, test, and deploy as a microservice. On the other hand, if we look at the Presentation layer, we have the Microfrontend, that is a microservice that exists within a browser [37]. They are related with the concept of a single page application, which sits on top of a micro service architecture.

Over time the frontend layer, often developed by a separate team, grows, and gets more difficult to maintain. However, is necessary to ensure that these pieces can be glue together to represent a single web application to the end-users.

Microfrontend allow a vertical organization of the web application development. The idea behind Micro Frontends is to think about a website or web app as a composition of features which are owned by independent teams. Each team has a distinct area of business or mission it cares about and specializes in. A team is cross functional and develops its features end-to-end, from database to user interface [37].

In an application built using microservices, they work together using events and background tasks to cooperate. This way of cooperating and work made the process a synchronic one. With Event Grid, event publishers are decoupled from event subscribers using a pub/sub model and simple HTTP-based event delivery allowing you to build scalable serverless applications, microservices, and distributed systems [38].

4.2.2.Communication between microservices

The Azure Event Grid is a tool to deliver events, the main concept used are events, event sources, topics, event subscription and event handlers. Azure Event Grid allows you to easily build applications with event-based architectures. First, select the Azure resource you would like to subscribe to, and then give the event handler or WebHook endpoint to send the event to. Event Grid has built-in support for events coming from Azure services, like storage blobs and resource groups. Event Grid also has support for your own events, using custom topics [38].

The illustration 4-4 shows an example of communication between microservices using EventGrid. The process starts with a patch on the zone endpoint to the

Gymsuite.AccessControl.Api, when this data is modified the ZoneUpdatedEvent event is launched towards the topic accesscontrol-egt. In this topic the AccessControlZoneUpdatedSusciption subscription receives the published message and stores it in the accesscontrol-zone-updated queue. To store a message in this queue triggers the azure function trigger ProcessZoneUpdated that is responsible for communicating with Azure Maps and sending the necessary data to obtain the OperationLocationUrl of the zone.

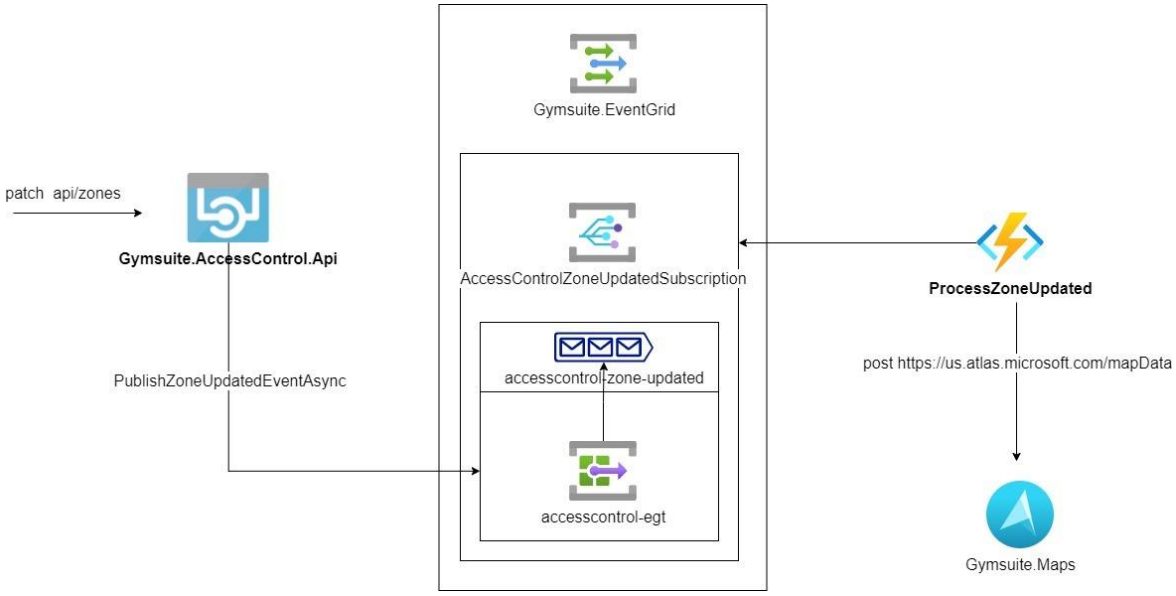


Illustration 4-4. Communication between microservices using EventGrid

4.2.3.Extensibility in Gym-Suite

Gym-Suite was designed as an extensible platform, the extensibility in Gym-Suite´s microservices architecture is shown in Illustration 4-5.

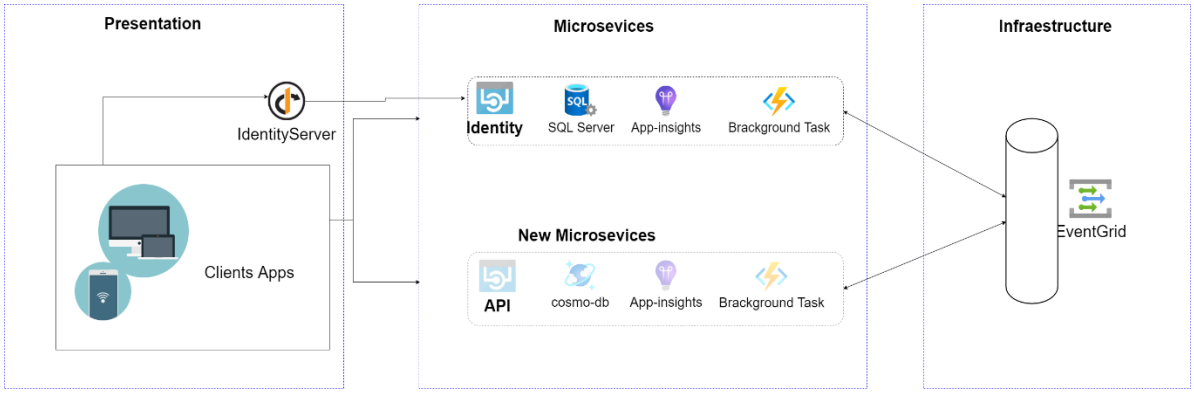


Illustration 4-5. Extensibility in Gym-Suite´s microfrotntend architecture

The architecture allows the addition of new microservices without affecting existing ones, ensuring the extensibility of the platform.

To add a new service, the following tasks must be executed:

1. Create the new related microservice (Illustration 4-5).
2. Create the new micro frontend (Illustration 4-6).
3. Register the URL where the new micro frontend is published.

To modify an existent service, the following tasks must be executed:

1. Modify the new related microservice.
2. Modify the new micro frontend.
3. Review the URL where the frontend is published, change it if it is necessary.

To remove a service/module, the following tasks must be executed:

1. Unregister the URL where the micro frontend of the service/module was published.
2. If there are dependencies in some microservices then review the dependencies and decide what to do.
3. Delete the micro frontend and the rest of the elements related with the microservices.

New microservices must use the IdentityServer for endpoint security. Communication with existing microservices must be done through the EventGrid by subscribing to the events that these services expose in its interface. In the implementation there should be no restrictions of technologies, they can use the database manager they need, if it would be important that they use the insights app to be able to monitor the microservice's behavior.

Illustration 4.6 shows how microfrontends are organized in the Presentation layer of the platform architecture, considering the responsibility of each microfrontend, and allowing the addition of new microfrontend when necessary, by the business. To meet these objectives, two groups were defined: one Global micro frontends and a second Parcel micro frontends.

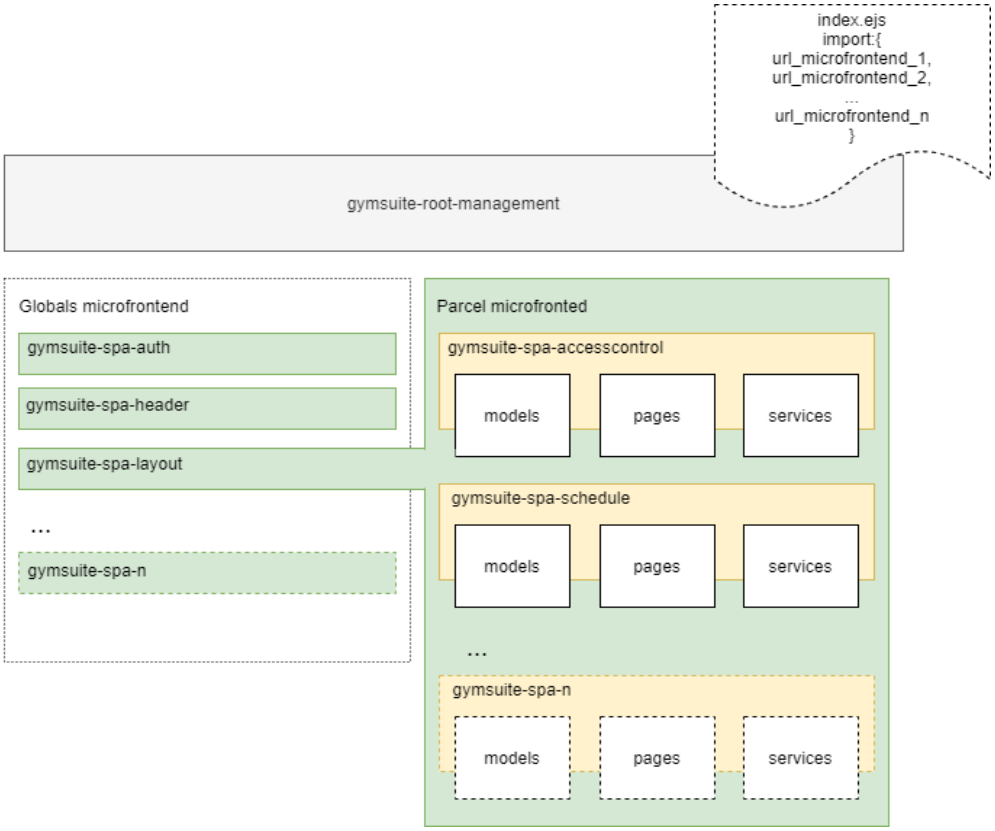


Illustration 4-6. Extensibility in Gym-Suite´s microfrotntend architecture

In **Global micro frontends** we can find the micro frontends that contain transversal responsibilities in the application. Here we find the authentication microfrontend "gymsuite-spa-auth", the microfrontend that manages the visibility of user data in the applications "gymsuite-spa-header" and the general layout of the application "gymsuite-spa-layout". In addition, in this group of micro frontends, utility modules type micro frontends would be added if necessary to handle common business logic.

In the **Parcel micro frontends** group, there are micro frontends that respond to specific business functionalities such as access control in "gymsuite-spa-accesscontrol" and functionalities related to schedule management in "gymsuite-spa- schedule". Also, in this group you can add new micro frontends that are needed in the application.

A microfrontend can use one or more microservices. The new microfrontend must be registered in the gymsuite-root-management, the file *index.ejs*, define the URL where it will be loaded in the gymsuite-root-management file *gymsuite-root-config.ts*.

The Illustration 4-7 shows how a new microfrontend is registered in the root of the application, for this the function *registerAplicaction* is used. This function receives the name

of the microfrontend, the content of the microfrontend that is going to load, the last parameter refers to the path of the application.

```
registerApplication({
  name: "@gimsuite/spa-layout",
  app: () => System.import("@gimsuite/spa-layout"),
  activeWhen: ['/'],
});
```

Illustration 4-7. Microfrontend register application code example

If it is not going to be loaded by URL and if by parcel the microfrontend must be imported into the parcel in the microfrontend that you want to show.

The Illustration 4-8 example shows an example of code that loads the microfrontend *@gymsuite/spa-resource* into the *@gymsuite/spa-layout*.

```
<Parcel config={() => System.import('@gymsuite/spa-resource')} />
```

Illustration 4-8. Example code for load a microfrontend

Illustration 4-9 shows the addition of the URLs to the menu in the *defaultProp.js* file.

```
        routes: [
             {
               path: '/',
               name: 'Dashboard',
               icon: <SmileOutlined />,
             },
             {
               path: '/schedule',
               name: 'Schedule',
               icon: <CrownOutlined />,
               routes: [
                 {
                   path: '/schedule/activities',
                   name: 'Activities',
                   icon: <CrownOutlined />,
                 },
                 {
                   path: '/schedule/goups',
                   name: 'Groups',
                   icon: <CrownOutlined />,
                 },
```

Illustration 4-9. Application's menu example

The code in the illustration 4-9 *defaultProp.js* contains a list of routes, where each route is composed of a path, a name, and an icon for display in the main menu (icon).

4.2.4.Gym-Suite.Management

It is the General Administration application, that will allow access to all entities in the domain. The domain entities are different and depend on the microservice. For example, in the access control the domains are Zone, Groups, Persons and other.

The main domain entities are:

- **User**: platform user.
- **Employee**: trainers and gym officials.
- **Client**: clients who come to train at the gym.
- **Zone**: Azure Maps geofence that defines a physical area with restricted access.
- **Group**: Groups of users who have access to a certain zone.
- **Permissions**: Defines a user group's permission to a zone.
- **Activities**: Classes and other types of events that you want to plan in the gym.
- **CalendarRecord**: Date of the activities planned for display in the calendar.
- **Product**: Classes, courses, workouts, and gym plans that are sold to customers.
- **Tax**: Taxes that apply to products that are sold.
- **Payment**: payments that are made on the platform through stripe.
- **Invoice**: invoice of payments made.

The Illustration 4-10 shows the set of domain entities and its relationships.

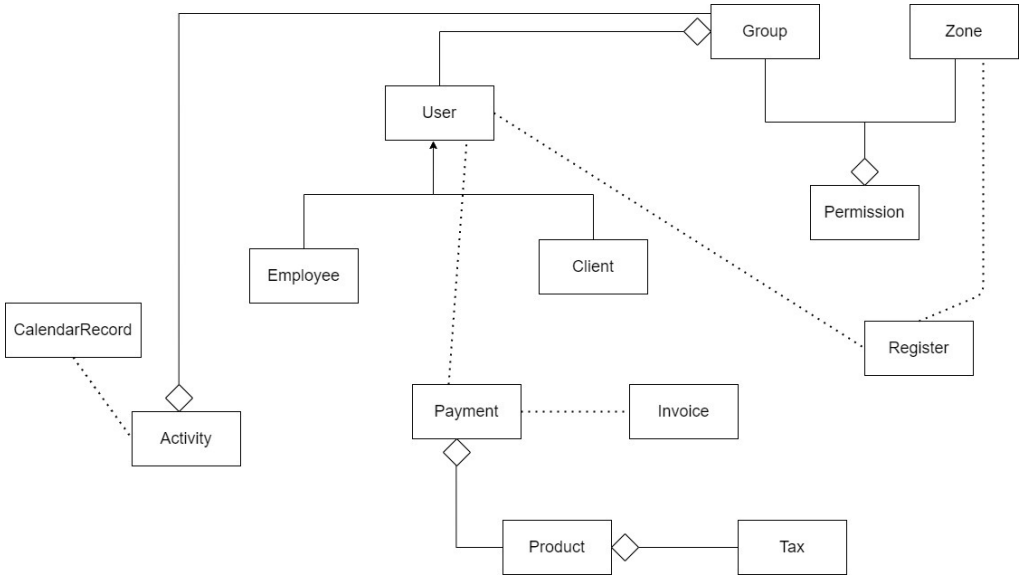


Illustration 4-10. Domain's entities

Although each is allowed by the architecture to use its own domain in a general way, the domain of the platform is conformed as follows:

- Users of the platform can be customers or employees. Employees can configure zones (physical spaces with restricted access) and assign these zones groups of customers with access permissions. Customers can access these areas where they have access permission and every time they enter or leave a record is saved.
- On the other hand, gym employees or trainers can create, and plan activities and clients can subscribe to these activities by incorporating the defined planning for the activity raised into their own calendar of activities.
- The activities that in practice can be courses or trainings are considered products that together with the gym plans are purchased by customers and generated the invoices of these purchases.
- Some entities are shared among services, but they are defined in the microservice where they are a main entity and not a supported one. One example of this kind of entity is Person, it main microservice is the Identity, but it is used in the rest of them, for example in Billing and Schedule.

Gym-Suite.Management is a web application implemented using React. It is based on microservices that control all the entities of the business; it also contains a dashboard. The dashboard allows access to the general and specific information of the gym. The dashboard shows: the people who are entering and leaving the premises, the number of people currently there. The dashboard also includes notifications about classroom schedules and the equipment usage.

Each module defined for the application (Access control, Billing, Schedule, and others) is also a microfrontend which is a separate application displayed at another URL. The microfrontend is loaded here with Parcel, which is the mechanism that the microfrontend uses to load one within another, the overall layout is another microfrontend, and allows to integrate several micro frontends.

4.2.5.Gym-Suite.Client

Gym-Suite.Client is a commercial web application for gym customers. It contains the publication of the plans that are sold, contact information, general information, prices of the gym, class schedules, available equipment, services among other elements.

Gym-Suite.Client is a Simple Page Application that only displays Gym´s information as advertising. Gym-Suite.Client is a way also to connect with the Gym's potential customers. To use this application, it is not necessary to authenticate, the contents that are published in this application will be managed from the main application.

This application has the following sections:

- **Home**: This section shows general information about the gym.
- **Mobile App**: This section shows information about the mobile application, some of its views and the main characteristics for which it should be used.
- **Testimonials**: This section informs future gym clients the opinion of other clients.
- **Pricing**: Shows the plans that are offered in the gym and their main characteristics.
- **Teams' members**: This section shows a photo and the name of the team members who work in the gym.

Illustration 4-11 shows the "Home" section where the mobile application is promoted and access to the application links is found.

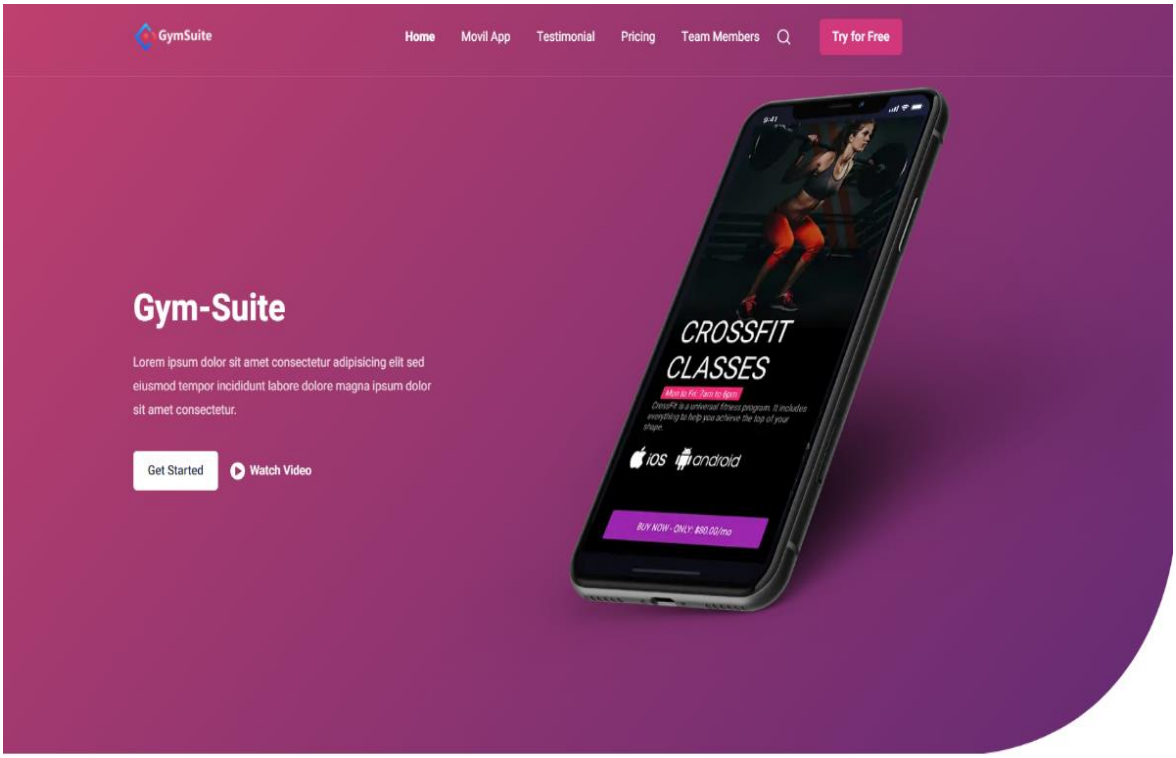


Illustration 4-11. Gym-Suite.Client - home section

Some views of the Gym-Suite.Client application are included in the Appendix C.

4.2.6.Gym-Suite.Client.App

Gym-Suite.Client.App is the mobile application of Gym's customers. Customers can manage their profile, purchase plans and services, make payments, book classrooms and equipment, view their gym calendar and calendar, and access to gym areas.

Gym-Suite.Client.App must be developed with a technology that is platform independent. In the case of mobile devices there are different technologies such as iOS and Android. To avoid having to develop applications for each operating system React Native was used as a framework and Expo as a development tool.

Expo is a framework and a platform for universal React applications. It is a set of tools and services built around React Native and native platforms that help you develop, build, deploy, and quickly iterate on iOS, Android, and web apps from the same JavaScript/TypeScript codebase. [39] Expo provides a more robust and developer-friendly workflow.

In Illustration 4-12 the views to start the application are shown, the home view of the application where the access button to the gym is seen, the view to purchase courses and the user menu.

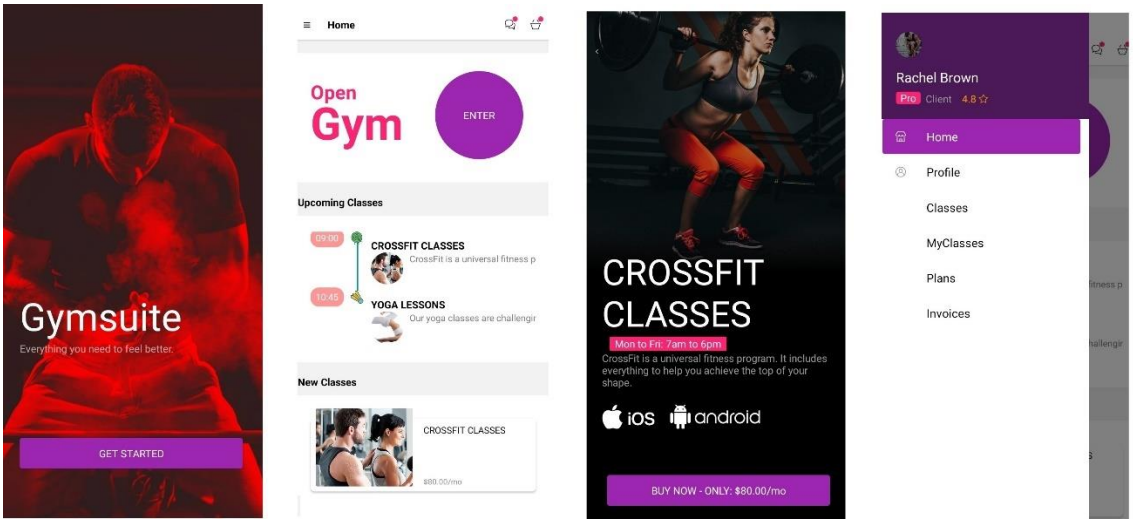


Illustration 4-12. Gym-Suite.Client.App

There are two tools that are needed to develop apps with Expo: a local development tool and a mobile client to open your app. As a local development tool, we use Expo CLI. In addition to the command-line interface (CLI) it also has a web-based graphical user interface (GUI) that pops up in your web browser when you start your project [40].

They are several advantages of use React Native and Expo, they are code reusability, Pre-built components, real-time feedback, and performance.

The Gym-Suite.Client.App comes with the following options:

- **User's registration** - allows to the client of the Gyms register and to set its initial data, full name, picture, payment information, billing address and other useful information.
- **Access to the user's profile** - this option was design to change the user's data in any time.
- **Activities enrollments** - show the user its training activities, the full schedule, the payment can be done in this option or using the Invoices option.
- **My trainings and activities** - this option show the user the activities that are currently in offer, the user can enroll in those activities in a very simple way.
- **My Invoices** - contains all the user's invoices, the user can see the detail of each invoice and the related payment.
- **Gym's Plan** - shows the active Gym's Plan and its characteristics, cost, activities, machinery. The user can acquire, pay, and customize the plan.
- **Access control** - among its functionalities are open the door, give access to the Gym's areas, and some notifications.
- **Schedule** - shows the machinery, venues and activities belonged to the user.

Some views of the mobile application are included in the Appendix D.

4.3. Microservices implementation strategy

The microservices were implemented according to the design of the application screens. In the implementation of each of the screens we tried to have a minimum functionality fully implemented. Each functionality has its own microservice. In the process of implementing a functionality, if necessary, the characteristics of that functionality were improved. These new functionalities sometimes required the implementation of new microservices and features. This process demonstrates the extensibility of the platform itself that, starting from minimum characteristics, grew until a product with multiple views was obtained.

The first module implemented was the IdentityServer. For its implementation, the code base was downloaded, and some APIs were added that allow exposing the roles and users of the

platform. This action obtained the API and therefore the microservice of the IdentityServer. This microservice ensures security when users access through the mobile application or through the Web application. Security is also guaranteed in the exchange of information between the rest of the microservices and their applications.

Once you have the IdentityService you move to the implementation of the API that manages the resources, this is the Resource API. In this version the Resource API handles employees, assigning them their roles. Employee management triggers events, and it is these events that ensure that employees are registered and with the necessary access to the system using the assigned roles. In the future, the Resource API could manage equipment or other resources that you want to control.

Another of the functionalities implemented was the Schedule, which is where the activities with its characteristics are defined. Activities are planned to run only once or may be periodic. In the case of single activities, it is defined when they are to be carried out. Periodic activities, among other elements, define their start and end date. All activities are planned for groups of users. The addition of users to groups can be done asynchronously using the web application or by the user himself through the mobile application.

The application for billing management was also implemented. This application has its microservice that interacts with Stripe, generating the payment order and the associated invoice. This application is used when buying activities by gym users and these activities are paid for. Among its facilities are the consultation of the payments made and their details.

One of the most important APIs that was implemented is the API for access control. This API and its corresponding microservice allow the configuration of the Geofences. This API then allows users to gain access to the different areas of the gym.

Finally, the notification microservice was implemented. In the implementation of all microservices, the workflow explained for the implementation of the IdentityServer was followed.

Once microservices began to be in place, microfrontends began to be implemented. These microfrontends respond to each of the microservices. The workflow followed was different from that of implementing microservices.

It was necessary first to implement the user authentication process, so that users could authenticate to the platform, then the guide to capture user data and then the layout.

The three microfrontends mentioned above are the global microservices of management. They allowed the implementation of microfrontend Resources and Access Control. The first of them to register users and employees, the second to define the Geofences.

The process was, to implement the microservice, the microfrontend that responded to it was implemented. This workflow made it possible to test all the implemented microservices and microfrontend. Once they were tested then the mobile application was implemented, which is nothing more than the union of all these elements. At this point in the development the mobile application is the access point to the microfrontend and microservices of Gym-Suite.

The next section explains the implementation of the flows of the main microservices. These microservices respond to the mandatory characteristics of a GSM.

4.4. Microservices Implementation

Implementation is an important step in the application's building process, not all the elements in this part are extensible discussed here, just the most important elements.

4.4.1. Access control

Access control is a very important part in the Gym-Suite platform, the Illustration 4-13 shows the workflow used by the platform to open an access, a normal door, and a small door, or an access control's gadget.

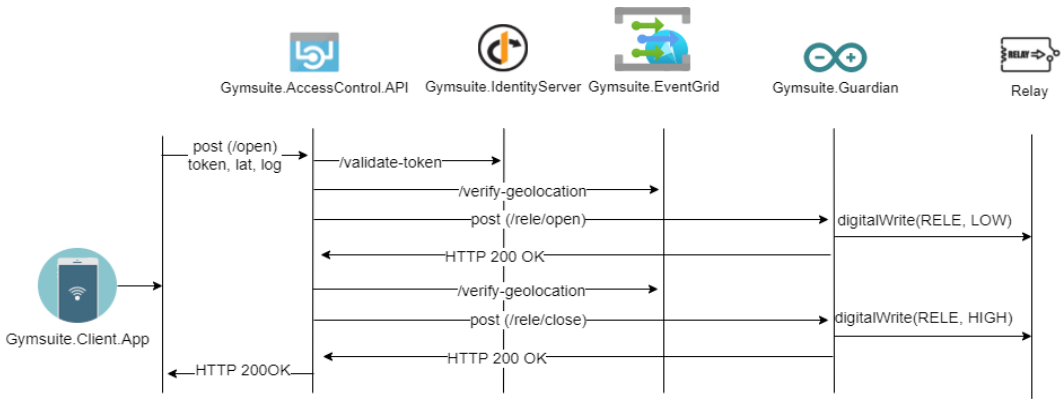


Illustration 4-13. Access control's flow

The workflow in the illustration 4-13 starts with an access request made by the mobile app, this request include the position (a GPS coordinate), the gym's zone where the user want to

gain access and the user's security token. Once that a request is made it is sent to the control access' API. This request tries to validate that the user can access to the area, and it is close enough to it. The AC's API sent a validation position request to the Azure Maps component. The validation of the closeness is made using a threshold. If the user is close enough and can access the zone, the Arduino component receive a signal to open the control access gadget. The Arduino closed the door if the door is open more than a specified time.

Event Grid is a component of Azure to handle the events. The general idea is there are services/apps that produce events and others that consume those events. Consumer functions needs to be subscribed to the events that they want to process. When a function makes a subscription need two main thing the topic and the queue where the function is going to store the message related to the event. When an event of this type is detected, this function is executed and produce a message that is stored in the related queue. The consumer functions are always listening in order to process the events.

The Illustration 4-14 shows the configuration zone´s flow. The interaction in this flow happened between the different microservices components that represent, the API, the EventGrid and the asynchronous functions ProcessZoneUpdate and ProcessOperationLocation. These functions are event triggers. ProcessZoneUpdate is subscribed to the ZoneUpdate event. This message contains the GPS position, this message is captured and processed. The Azure Map used among others to paint the zone, and store the position in Cosmos DB. When this operation finishes, a message is sent to let it knows that the Geofence can be used. The final step is executed when Azure Maps has ready the Geofence, this operation needs the Geofence's UID.

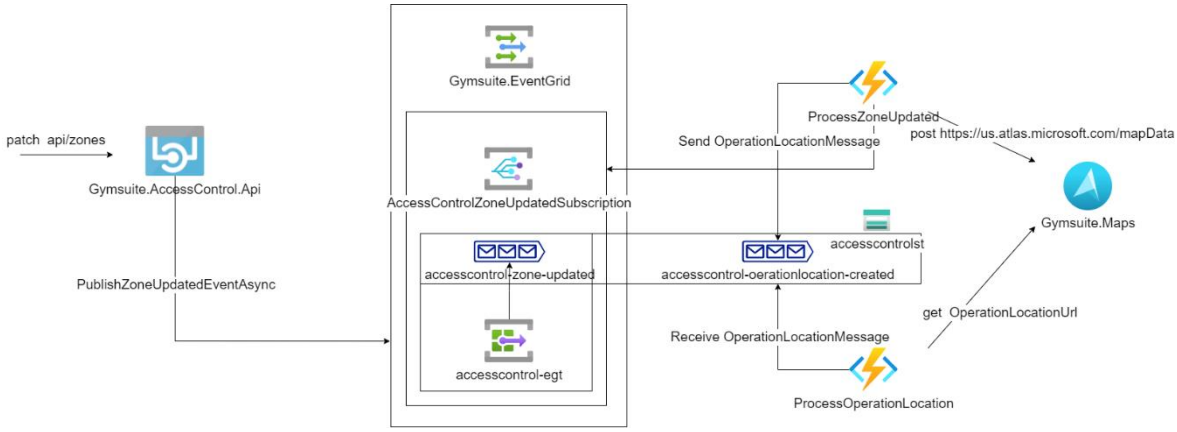


Illustration 4-14. Configuration zone´s flow

Illustration 4-15 shows an application view where a Geofence is defined, specifically the Zone configuration form, where you can define a polygon on a map using the services provided by Azure Maps.

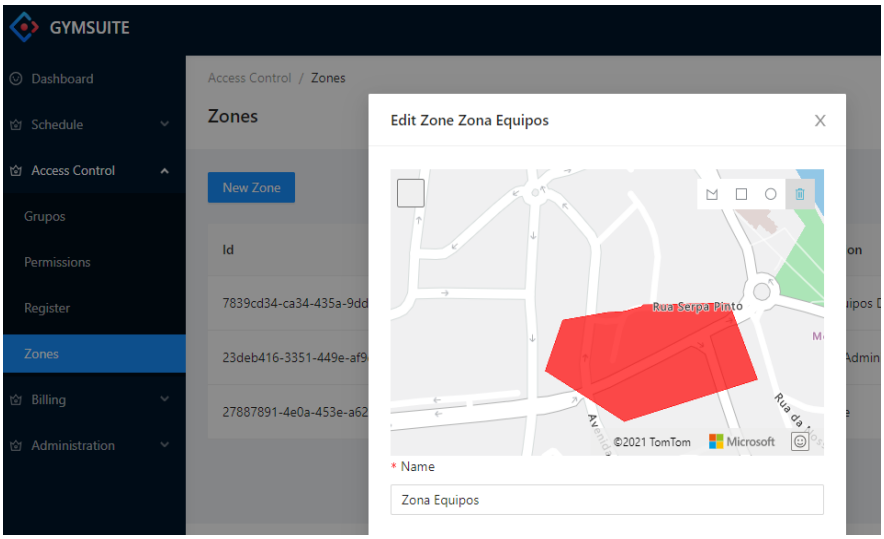


Illustration 4-15. Geofence definition in Gym-Suite

4.4.2.Payments

The payment flow begins when the user wants to pay for some service, from any of the ways that can be used for it. This payment request triggers the creation of a payment method using the Stripe API, this API uses a public key to create the payment method. The flow of this process can be seen in the illustration 4-16.

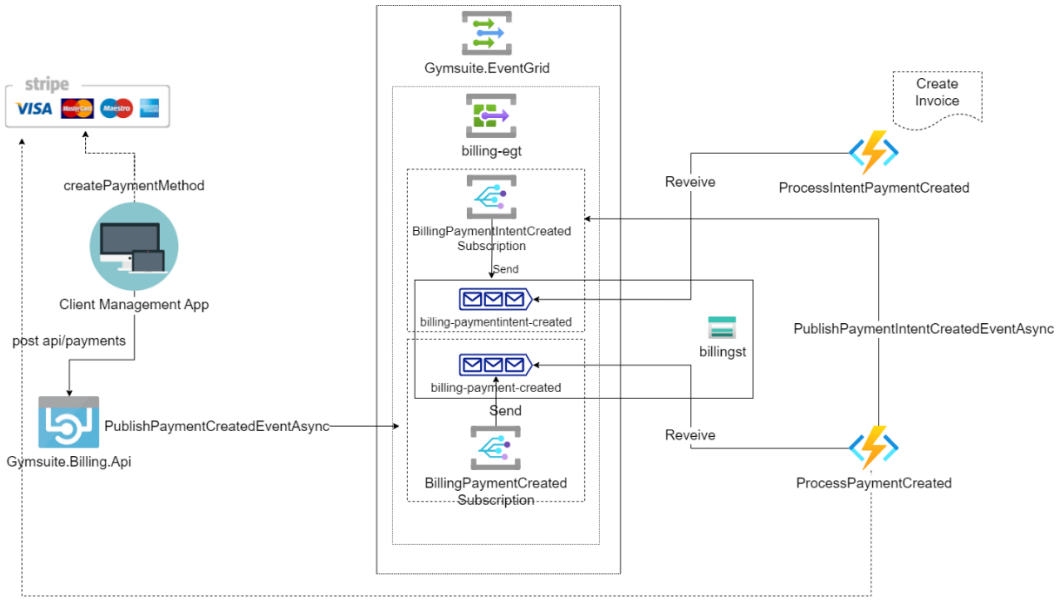


Illustration 4-16. Payment´s workflow

The response from Stripe API, the payment method, is sent as a request to the Billing App API, which creates the payment, using the private key and stores it in the Cosmos DB. This process is done asynchronously. The next step is to create a payment event (PaymentCreateEvent), this event is published in the Event Grid, in the Billingst GT app. Stripe has a process that is running in background and subscribed to this event. Events in the queue are processed by the ProcessPaymentCreate function.

Processing each of these events creates an event of type ProcessPaymentIntentCreate, which represents a payment attempt. If the payment attempt is successful, the money is deducted from the payment method, and a ProcessPaymentCreate event is published. The processing of this event is what triggers the creation of the invoice and its storage in the database. Module's implementation and extensibility.

4.4.3.Notifications control

The notifications microservice is responsible for issuing the different toast notifications. Notifications are related to users' access to the gym. These notifications are displayed in a dashboard that is updated automatically. For the update SignalR is used, it performs a communication using Sockets and update the dashboard. The dashboard is kept up to date and displaying the generated notifications.

Illustration 4-17 shows the SignalR notification flows.

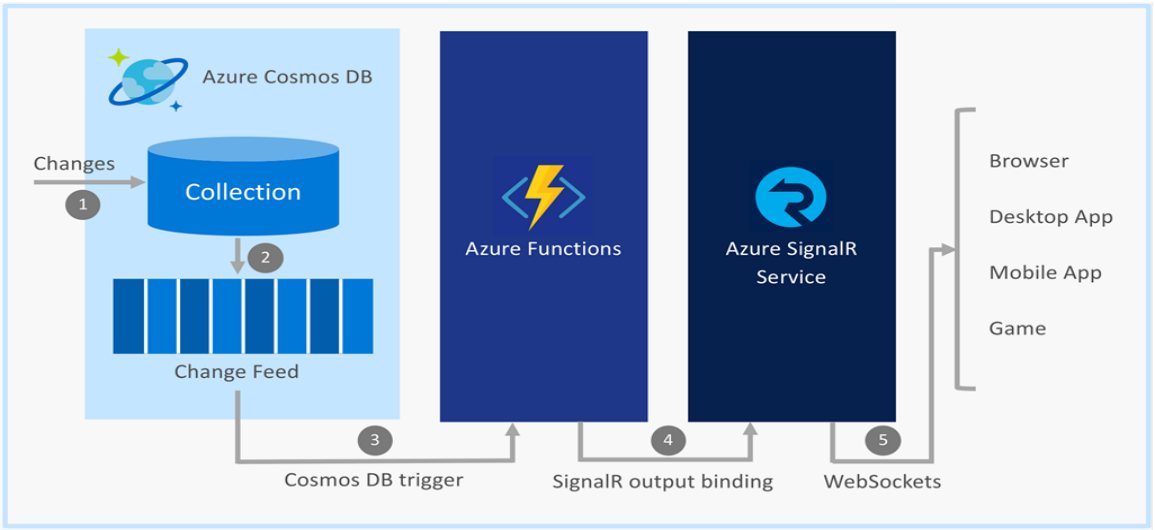


Illustration 4-17. SignalR notifications flows

The SignalR use the following workflow in the notification process:

1. The notification API (Gymsuite.Notification.Api) receive the messages and stored it in the Cosmos DB database.
2. The function MessageCollectionTrigger detect when the Cosmos DB is modified and sends a SignalRMessage (gymsuite-notify) to be processes. The function MessageCollectionTrigger belongs to GymSuite.Notification.Functions.
3. The clients that are subscribed to the SignalRMessage establish a connection with the gymsuite-notify's hubs calling the NegotiateFunction with "notification" as the value of HubName parameter. This step guarantee that clients receive the messages published in the Azure SignalR for the hub passed as a parameter using web sockets.

4.5. Gym-Suite General evaluation

After the implementation of the Gym-Suite solution the conclusions are:

- Based on the characteristics of the Gym-suite platform, a combination of methodologies and tools was used that allowed to carry out an agile process, where iterations are focused on the delivery of functional requirements.
- The architecture selected and implemented for the Gym-suite platform allows it to be an extensible platform, to which new modules can be added that respond to new functional requirements, its core concept is the concept of microservice.
- Each application has the same structure and interacts in a general way following the same pattern. In this way each application has its own database, its models, its microfrontend and its microservices. If there are dependencies between the applications, they are resolved using the Event Azure Grid.
- Algorithms were designed for the addition, modification, and removal of applications to the platform. These algorithms were tested when new functional requirements were added to the platform.
- Unit, integration, and system tests were carried out. Each type of tests carried out from its objectives managed to demonstrate that the platform obtained is a system that meets the defined requirements.

- The application deployment process is an automated process that is performed when modifications in the code that is uploaded to the repository are detected. In this process two pipelines are used, one of these pipelines builds the artifacts that the other needs to deploy the solution in Dockers and Terraform.

5. Gym-Suite Testing and Deployment

To prove that the gym-suite application was well built and works fine different kinds of tests were carrying out. This part describes the test and how they were applied. The objectives of this chapter are to demonstrate the operation of the system and the realization of a deployment of the same in the selected platform.

5.1.1. Unit testing

A unit test is a way of testing a unit - the smallest piece of code that can be logically isolated in a system [41]. In most programming languages, that is a function, a subroutine, a method or property. The isolated part of the definition is important, the unit need to be correct and work in the way that was defined.

A unit can be almost anything -- a line of code, a method, or a class. Generally, though, smaller is better. Smaller tests give you a much more granular view of how your code is performing. There is also the practical aspect that when you test very small units, your tests can be run fast; up to thousands of tests per second.

The unit tests made to the system was done in the Visual Studio, using the tools that the IDE provide. These unit tests are performed on the application layer of the microservices. It is in this layer where the business logic is implemented. Therefore, tests are carried out on the microservices, each one or of them. These tests are run first to ensure that the code that is uploaded to the repository is error-free and runs correctly and then in the pipeline to ensure that the artifacts of the platform, its applications and/or microservices are obtained error-free.

The illustration 5-1 and illustration 5-2 show the Gymsuite.AccessControl application layer and the tests designed for it.

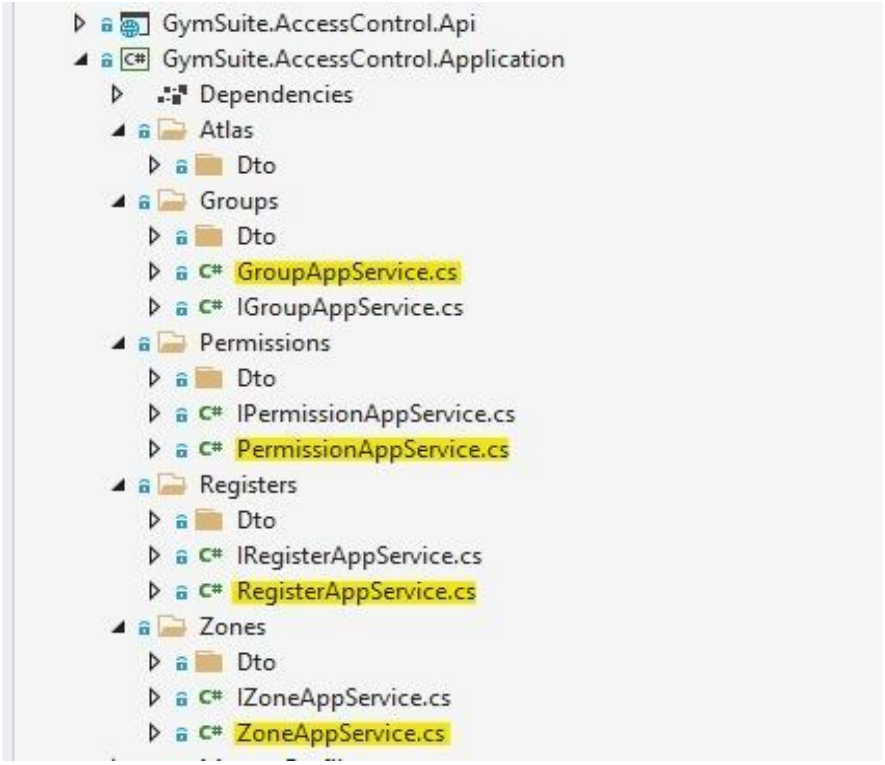


Illustration 5-1. Gymsuite.AccessControl Application Layer

The marked elements in yellow are the elements that will be tested.

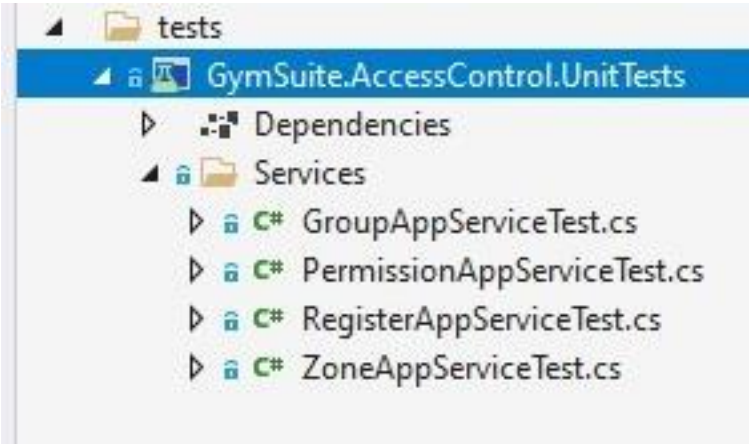


Illustration 5-2. Unity test example

Once the tests are defined on the application layer, they are executed, and their result is analyzed. Until the test result is satisfactory, the code is not uploaded to the repository and the process is not continued.

The following illustrations 5-3 and 5-4 show two runs of the tests, the first was unsatisfactory and the second satisfactory.

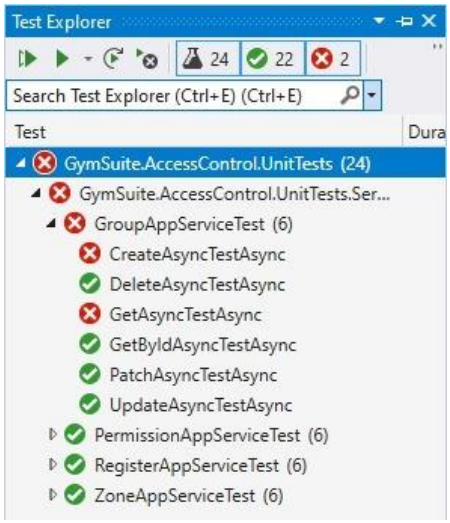


Illustration 5-3. Non-satisfactory execution of the tests

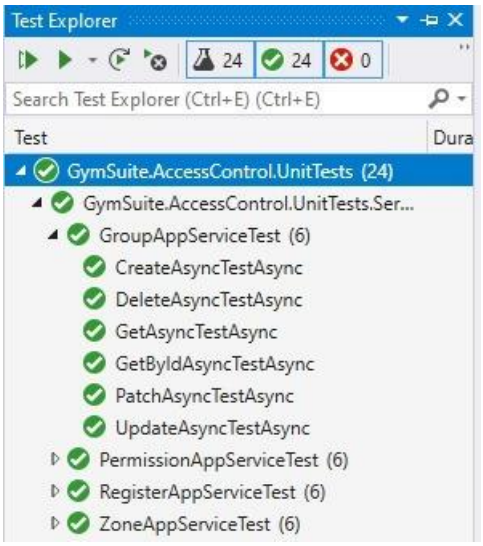


Illustration 5-4. Satisfactory execution of the tests

Once the tests are executed satisfactorily the code is uploaded to the repository. When the code is uploaded to the repository and changes are detected in it, a pipeline is activated that allows obtaining the artifacts of the platform, its applications and / or microservices. This pipeline contains the configuration of the actions to be executed automatically in it, within these actions are the unit tests.

Illustration 5-5 shows the pipeline configuration to compile and run the Gymsuite.AccessControl.Api tests. In it you can see a Unit Test name state that contains a job with two tacks; the first *DotnetCoreCLI@2* that has the responsibility of executing the test project and the second *PublishCodeCoverageResult@1* that saves the result of the tests in an XML for later use.

```yaml
stages:

- stage: UnitTest
  displayName: Unit test
  jobs:
  - job: UnitTest
    displayName: Unit test
    steps:
    - task: DotNetCoreCLI@2
      displayName: 'dotnet test'
      inputs:
        command: 'test'
        projects: 'test/*Tests/*.csproj'
        arguments: '--configuration $(buildConfiguration) /p:CollectCoverage=true /p:CoverletOutputFormat=opencover'

    - task: PublishCodeCoverageResults@1
      displayName: 'publish code coverage result'
      inputs:
        codeCoverageTool: 'Cobertura'
        summaryFileLocation: 'test/*Tests/coverage.opencover.xml'
```

Illustration 5-5. Configuration of the pipeline that runs the tests

The illustration 5-6 shows an example of the GymSuite.AccessControl pipeline where the tests have been executed satisfactorily and where this has allowed to move to the state where the Docker images are generated. These images are the ones that will later be displayed.

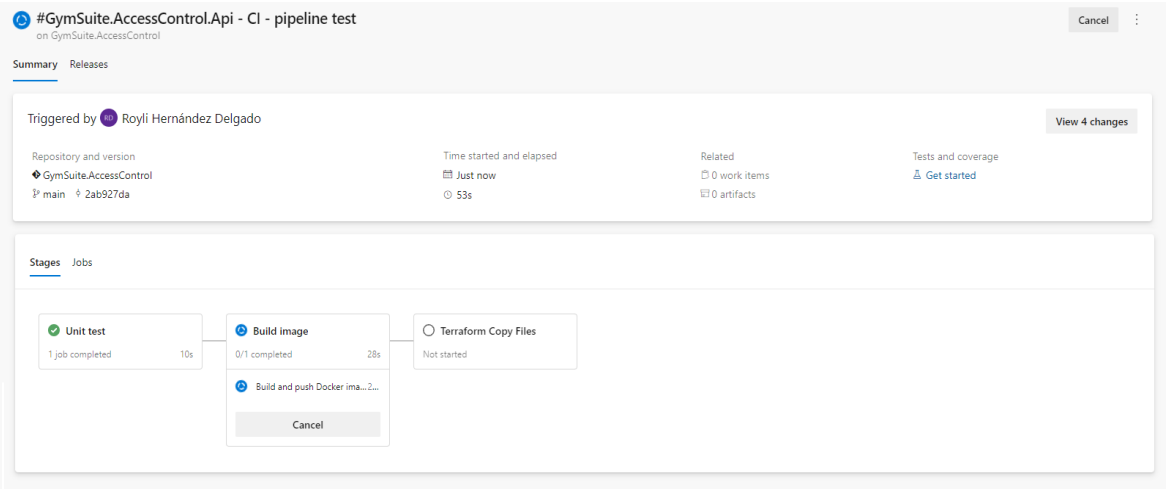


Illustration 5-6. An example of the GymSuite.AccessControl pipeline

The same process was applied to all applications and/or microservices that are part of the Gym-Suite platform.

5.1.2. System integration testing

System Integration Testing (SIT) is performed to verify the interactions between the modules of a software system. It deals with the verification of the high and low-level software requirements [42].

Cypress was used for the execution of the integration tests. Cypress framework is a JavaScript-based end-to-end testing framework built on top of Mocha – a feature-rich

JavaScript test framework running on and in the browser, making asynchronous testing simple and convenient. It also uses a BDD/TDD assertion library and a browser to pair with any JavaScript testing framework.

To use Cypress, it is necessary to define the general project where the tests are going to be added. The tests are run by the developer. The developer designs the tests considering the business logic or functional requirements. These tests allow to test the operation of the elements of the platform once it has been deployed.

The illustration 5-7 shows a part of the Cypress project with its tests, it shows the implementation of the test for the permission creation flow that defines which group of users can enter a zone, from line 4 to line 13 the different actions that would be performed in user to create a permission from the application are shown. The application translates them into a more natural language would be:

1- visit the permission creation URL,
2- click on the add button,
3- select the name field, write "Unit test" as the name of the permission,
4- select the zone field,
5- select one of the values of the zone select,
6- select the group field,
7- select one of the groups from the select groups,
8- enable the Enabled checkbox and finally
9- click on the OK button.

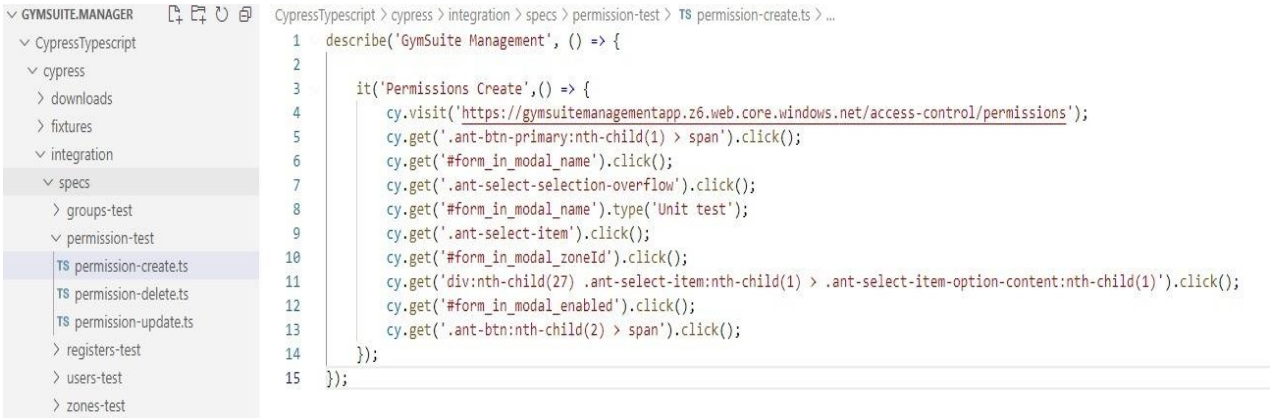


Ilustração 5-1. Cypress Project

After this flow the list with the new permission will appear as shown in the Illustration 5-9.

Illustration 5-8 shows the selection of the tests you want to run. Shows the interface of Cypress for the tests to be executed, in this case the test *permission-create.ts* appears in the spec folder.

Illustration 5-7. Selection of the test to run with Cypress

Illustration 5-9 shows the execution of the permission-create.ts test in Cypress where it is observed that the test has been executed correctly and the test permission that it has created in its execution.

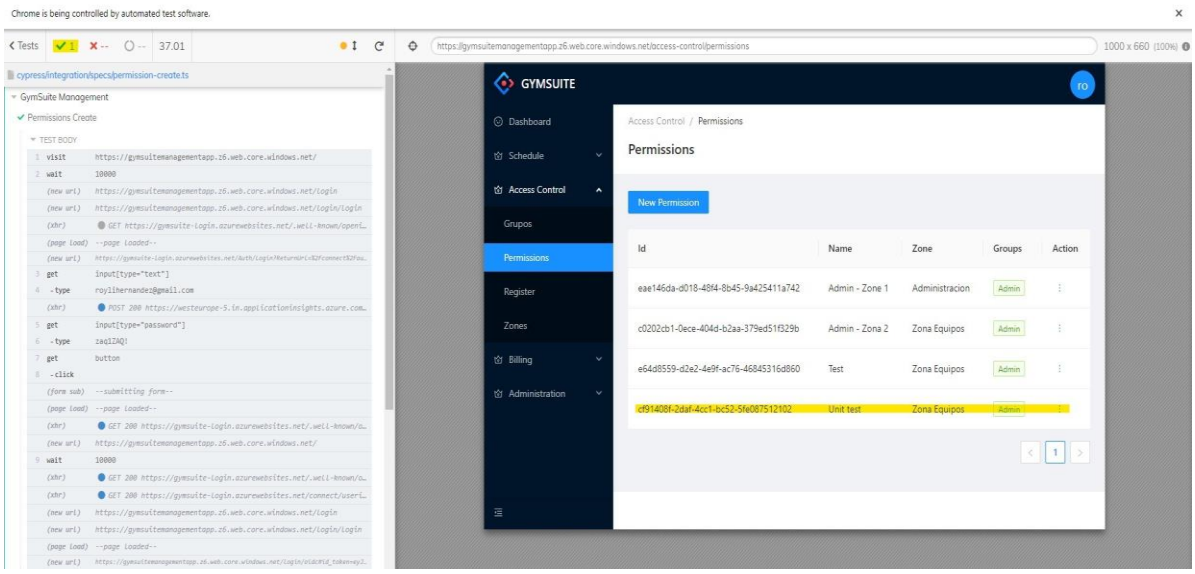


Illustration 5-8. Result of tests with Cypress

The same process was applied every time that in the Gym-Suite platform the process of include a new functional requirement finished its implementation.

5.1.3. System testing

System Testing is done after Integration Testing. This plays an important role in delivering a high-quality product.

During the implementation of the Gym-Suite platform, extensive system tests where not done. System testing is going to focus on some aspects that are considered the most important.

The first of the important elements to test is the flow of access control, one of the most important flows in the system. It is necessary to guarantee this flow offers a good response so that the rest of the flows can work. This flow interacts with some physical devices using Arduino. In this case it is necessary to know the action response time of the devices. This action time includes interaction with the device and notification to the dashboard. These actions include open the doors and to store the information in the Azure Cosmos DB.

For these tests the response value 12 seconds is considered a good reference value, equal or lower response times are considered good. Smaller response values mean better response of the system.

During the tests, 4 iterations were performed, and the calculated average response time was 14 seconds. Response time combines the door opening time and the time used for to inform the user. When the calculated value is compared with the reference value, it can be said that the response of the system is good.

5.2. Deployment

The Gym-Suite's deployment uses a strategy knows as Continuous integration (CI) and continuous development (CD) embody a culture, set of operating principles, and collection of practices that enable application development teams to deliver code changes more frequently and reliably. The implementation is also known as the CI/CD pipeline.

In the CI/CD pipeline the integration occurs in a small scale. These small changes are check into the repository frequently. This continuous integration brings multiple benefits to the development process, the most important is integrate different tools ad platforms used in the process. Other important benefit is the time is not spent in the debugging process because is easier to identify the error source in a small change made to the system.

Gym-Suite's CI/CD strategy uses Visual Studio development tools, a repository, Azure DevOps, and two pipelines. One of the pipelines is used to generate the resources and the other to build the Docker images that contain the API Service and the Azure function. The interaction of all these elements allows to obtain in an automated way all the elements of an application or a microservice. All this corresponds to the "infrastructure as code" strategy.

Illustration 5-10 shows the Gym-Suite CI/CD workflow.

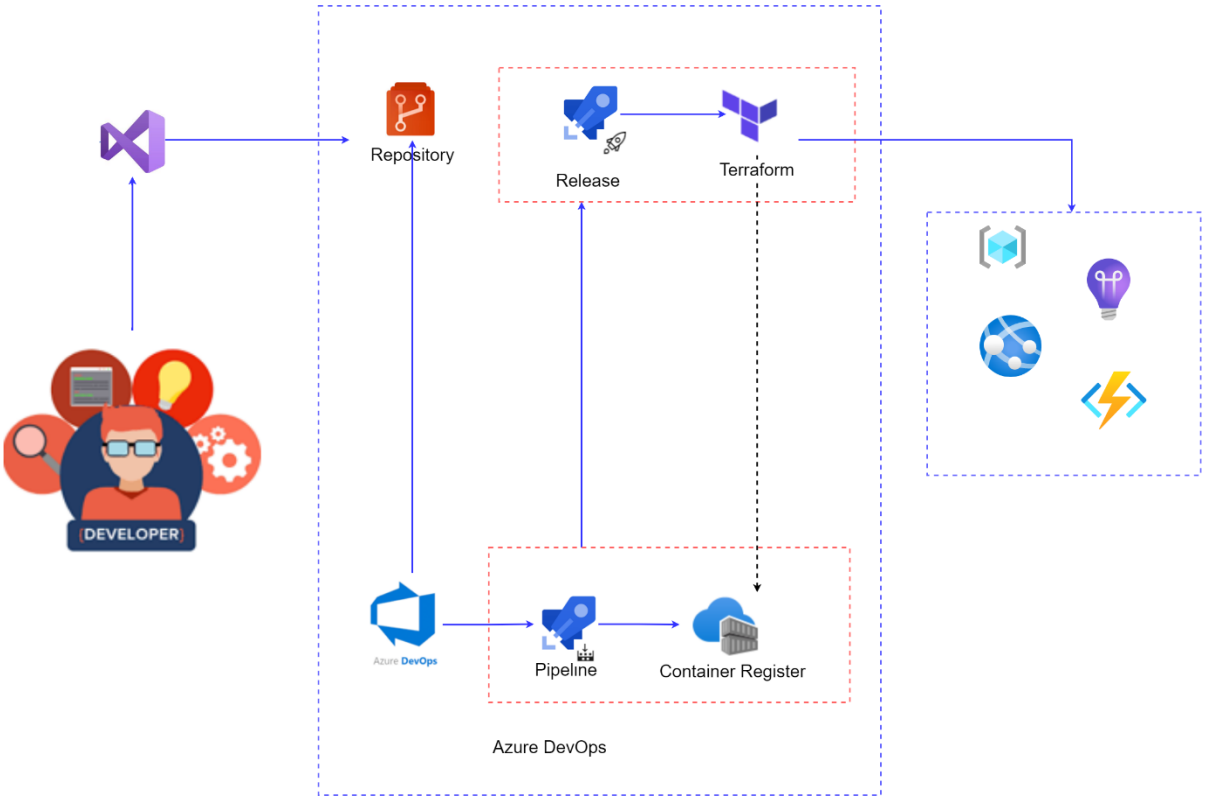


Illustration 5-9. Gym-Suite CI/CD workflow

The first pipeline is launched when changes are detected in the repository. In this pipeline the tests on the code are executed, the Docker image is created that is published in the Azure Container Register and an artifact is created with the Terraform files.

The second pipeline uses the Terraform artifact to create the application resources or microservices in the Azure, and this is the output that is obtained from the workflow execution.

The Illustration 5-11 below shows an example pipeline for the Access Control API.

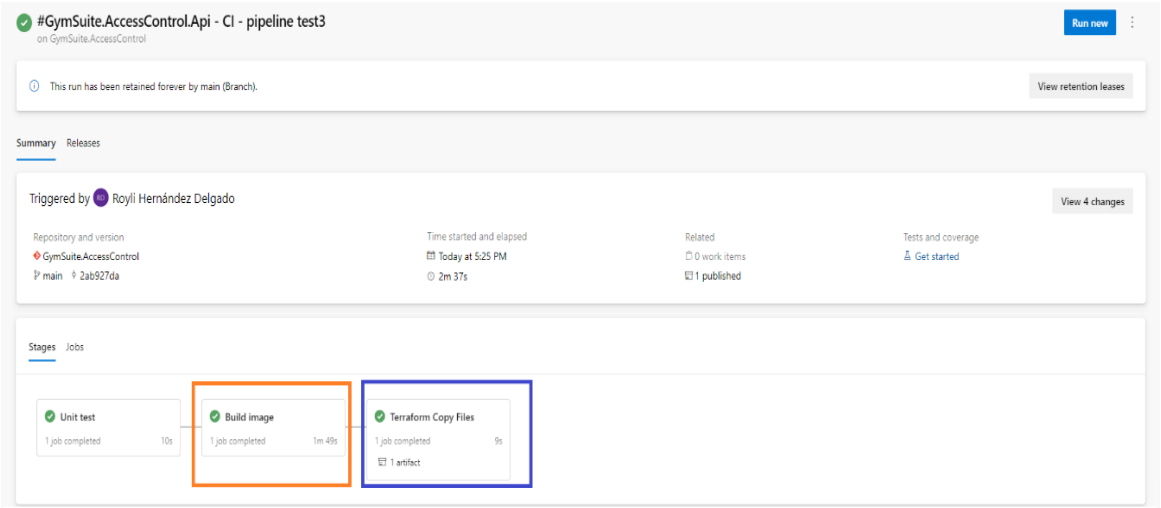


Illustration 5-10. Access Control´s example pipeline

Once the images are built and published, they need to be deployed, this is the continuous deployment part of the workflow. In this part the Azure´s resources need to be created and the Docker´s image needs to be deployed. This process is fully automated and used a release pipeline. This release pipeline executes the Terraforms task to create the resources in the Azure DevOps and to deploy the compiled images in the Azure Container Register. Illustration 5-12 shows the AccessControl microservice continuous deployment release pipeline.

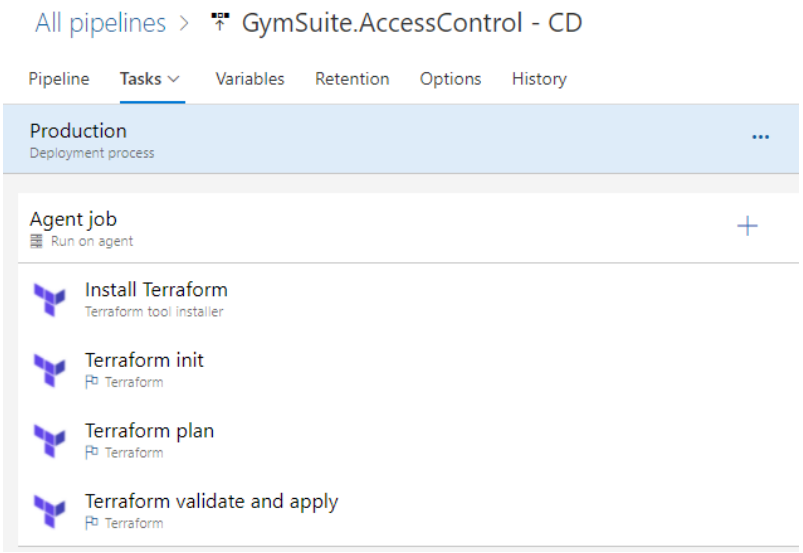


Illustration 5-11. GymSuite.AccessControl CD release pipeline

The same process is applied to manage all the components of the Gym-Suite platform.

5.3. Chapter Summary

After the test, evaluation and deployment of the Gym-Suite solution the conclusions are:

- Unit, integration, and system tests were carried out. Each type of tests carried out from its objectives managed to demonstrate that the platform obtained is a system that meets the defined requirements.
- The application deployment process is an automated process that is performed when modifications in the code that is uploaded to the repository are detected. In this process two pipelines are used, one of these pipelines builds the artifacts that the other needs to deploy the solution in Dockers and Terraform.

6. Conclusions and Future work

The conclusions derived from the present research are:

- A Gym Management System is a software that include access control, user profiles, and financial management among other functionalities. This type of software is designed to help managers and owners to organize and to manage all aspects of this type of business. Most of the Gym's owners in the Leiria district, that respond to the survey, agree in the core functionalities and the need of use of automated systems to manage their business.
- The architecture designed for the platform allows to add, to modify and to remove applications from it. The inclusion of new applications is possible because each application has a similar structure, and this simplifies the test and deployment processes. The central concept is the concept of microservice, and these are oriented to the functional requirements defined for the applications.
- The deployment of the application is done in an automated way in the cloud when the tests have been carried out on the system. Automation also allows to include tests that guarantee the quality of the same and the transparency to the user when working with technologies that are complex in their nature and in their interaction.

A limitation of this research is that the developed system can be used in only one institution. The limitation is due to the way the deployment is made, which uses a single environment and a single database oriented to that institution. This limitation is based on the use of the Identity Server that is prepared for a single client. However, Azure Cosmos DB does come prepared for the inclusion of multiple companies.

6.1. Future work

As future and as a short-term tasks we recommend:

- Study how to manage multiple gyms through the Gym-Suite platform. This task includes the adaptation of the Identity Server.
- To include a greater range of physical access devices, interfacing them with Arduino.

- Conduct a more in-depth study of the response time of the access control component.
- Perform the acceptance tests of the Gym-Suite platform.

Regarding long-term tasks, we propose:

- Carry out the necessary studies to deploy the platform in other environments.
- To add to the Gym-Suite platform the ability to consider each service as an independent product.

Bibliography

[1] gestigym, “Software de Gestión GestiGym,” *gestigym.com*, pp. https://gestigym.com/gestigym-software, 2020.

[2] “TD Sistemas,” TD Sistemas, [Online]. Available: https://www.tdsistemas.com/sistema-control-acceso-gimnasios/. [Accessed 01 04 2021].

[3] Gymmaster, “24/7 Gym Access Control,” pp. https://www.gymmaster.com/gym-access-control/.

[4] Softar-Gym, “Control de accesos en instalaciones deportivas,” pp. https://softar-gym.com/control-accesos/.

[5] Gym Master Software, Running a Successful Gym, Gym Master Software, 2021.

[6] PerfectGym, “PerfectGym,” PerfectGym, [Online]. Available: https://www.perfectgym.com/es/fitness-software. [Accessed 25 03 2021].

[7] “Enzocard Research & Development,” Enzocard Research & Development, [Online]. Available: http://www.enzocard.eu/90/software-para-gimnasio-y-club-deportivo/control-de-acceso-a-sala-de-gimnasio.html. [Accessed 01 04 2021].

[8] “G2.com,” G2, 2021. [Online]. Available: https://www.g2.com/categories/gym-management#learn-more. [Accessed 15 05 2021].

[9] “GestiGym,” [Online]. Available: https://gestigym.com/. [Accessed 2021GestiGym 02 05].

[10] VirtulaGym, “VirtualGym,” [Online]. Available: https://virtuagym.com/es/. [Accessed 22 05 2021].

[11] “Softar,” softar, [Online]. Available: https://softar-gym.com/. [Accessed 07 04 2021].

[12] nstituto Nacional de Estatística, “Statistics Portugal,” CSC, 2021. [Online]. Available: https://www.ine.pt/xportal/xmain?xpid=INE&xpgid=ine_main. [Accessed 15 06 2021].

[13] Microsoft, “Azure Cosmo DB,” Microsoft, 2021. [Online]. Available: https://docs.microsoft.com/en-us/azure/cosmos-db/introduction. [Accessed 12 05 2021].

[14] Microsoft, “TypeScript,” 2021. [Online]. Available: https://www.typescriptlang.org/. [Accessed 16 05 2021].

[15] ZDNet, “Top cloud providers in 2021,” 2021.

[16] M. Collier and R. Shahan, Fundamentals of Azure, Redmond, Washington 98052-6399: Microsoft Press, 2015.

[17] A. Solutions and M. Moore, “What are the business benefits of Microsoft Azure Cloud?,” 2021.

[18] Pro Tech, “Top 5 Benefits of Microsoft Azure for Bussiness,” 2021.

[19] Microsoft Corp, “Terraform with Azure,” 2019. [Online]. Available: https://docs.microsoft.com/en-us/azure/developer/terraform/overview. [Accessed 16 05 2021].

[20] The Kuberbernetes Authors, “Kubernetes,” The Kuberbernetes Authors, 2021. [Online]. Available: https://kubernetes.io/. [Accessed 12 04 2021].

[21] Microsoft, “https://dotnet.microsoft.com,” Microsoft, 01 06 2021. [Online]. Available: https://dotnet.microsoft.com/learn/aspnet/what-is-aspnet-core. [Accessed 31 26 2021].

[22] Cyber-Infraestrcture India, “CIS,” CIS, 2020. [Online]. Available: https://www.cisin.com/coffee-break/technology/benefits-of-asp-net-core-to-develop-robust-web-applications.html. [Accessed 31 06 2021].

[23] NMG Technical Blog, “NMG,” NMG, 2020. [Online]. Available: https://nmgtechnologies.com/blog/advantages-of-asp-net-core.html#_Toc33524964. [Accessed 25 06 2021].

[24] Ace Infoway, “Ace Infoway,” 2021, [Online]. Available: https://www.aceinfoway.com/blog/asp-dot-net-core-advantages. [Accessed 15 06 2021].

[25] C. Bilgin, Mobile Development with .NET: Build cross-platform mobile applications with Xamarin.Forms 5 and ASP.NET Core 5, 2nd Edition, Packt Publishing, 2021.

[26] K. Y. Fukizi, J. De Oliveira and M. Bruchet, Learn ASP.NET Core 3: Develop modern web applications with ASP.NET Core 3, Visual Studio 2019, and Azure, 2nd Edition, Packt Publishing, 2019.

[27] w3schools, “w3schools,” 2021. [Online]. Available: https://www.w3schools.com/react/default.asp. [Accessed 09 06 2021].

[28] C. Ripon, ASP.NET Core 5 and React: Full-stack web development using .NET 5, React 17, and TypeScript 4, 2nd Edition, Packt Publishing, 2021.

[29] Facebook Inc., “reactnative.dev,” Facebook, 2021. [Online]. Available: https://reactnative.dev/docs/intro-react. [Accessed 17 05 2021].

[30] D. Ward, React Native Cookbook - Second Edition, Pack Publishing, 2018.

[31] P. Akshat and N. Abhishek , React Native for Mobile Developmen, Apress, 2019.

[32] N. Dabit, React Native in Action: Developing iOS and Android Apps with JavaScript, Manning, 2019.

[33] Scrum.org, “Scrum.org, The home of Scrum,” scrum.org, 2021. [Online]. Available: https://www.scrum.org. [Accessed 2021 07 01].

[34] digité, “digité, How work really gets done,” digité, 2021. [Online]. Available: https://www.digite.com/agile/agile-methodology/#scrum. [Accessed 2021 06 15].

[35] Microsoft Inc, “Visual Studio Code,” Microsoft Inc., 2021. [Online]. Available: https://code.visualstudio.com/docs. [Accessed 25 05 2021].

[36] 1105 Media Inc., “Visual Studio Magazine,” 1105 Media Inc., 1996-2021. [Online]. Available: https://visualstudiomagazine.com/articles/2021/01/13/wakatime-vs-code.aspx. [Accessed 17 06 2021].

[37] Micro Frontends, “micro-frontends.org,” Micro Frontends, 2021. [Online]. Available: https://micro-frontends.org/. [Accessed 07 06 2021].

[38] Microsoft, “Azure Event Grid,” Microsoft, 2021. [Online]. Available: https://azure.microsoft.com/en-us/services/event-grid/#features. [Accessed 12 06 2021].

[39] Expo Dev, “Expo,” 2021. [Online]. Available: https://docs.expo.dev/. [Accessed 16 07 2021].

[40] T. Botelho, “www.imaginarycloud.com,” 2021. [Online]. Available: https://www.imaginarycloud.com/blog/how-to-accelerate-your-development-with-react-native-and-expo/. [Accessed 01 08 2021].

[41] SmartBear, “SmartBear,” SmartBear Software. , 2021. [Online]. Available: https://smartbear.com/learn/automated-testing/what-is-unit-testing/. [Accessed 2021 06 15].

[42] Guru99, “Guru99,” 2021. [Online]. Available: https://www.guru99.com/user-acceptance-testing.html. [Accessed 2021 6 5].

[43] Enzocard, “Proveedor global de soluciones de identificación y de control,” p. http://www.enzocard.eu/company.html.

[44] Saltoks, “The app. SALTO KS is the solution for access control anywhere, any time,” pp. https://saltoks.com/salto-ks, 2021.

[45] B. Wilder, “Cloud Architecture Patterns: Using Microsoft Azure,” 2012, p. x.

[46] M. B. Jason De Oliveira, in *Learning ASP.NET Core 2.0: Build modern web apps with ASP.NET Core 2.0, MVC ...*, pp. 15, 16.

[47] C. Nagel, in *Professional C# 6 and .NET Core 1.0*, p. 27.

[48] C. Macrae, in *Vue.js: Up and Running: Building Accessible and Performant Web Apps*, p. VII.

[49] O. Filipova, in *Learning Vue.js 2*, p. 45.

[50] C. M. Ricardo and S. D. Urban, Databases illuminated, New York: Jones & Bartlett Learning, LLC, 2017.

[51] C. Musciano and B. Kemedy, HTML, La guía completa, Mexico, D.F.: McGRAW-HILL INTERAMERICANA EDITORES, S.A. de C.V., 1999.

[52] J. P. Mueller, Beginning Programming with Python For Dummies, New Jersey: John Wiley & Sons, Inc., 2014.

[53] A. Lock, ASP.NET Core in Action, Second Edition Annotated Edition, Manning Publications, 2021.

[54] I. d. l. R. A. C. J. M. D.-P. J. L. Y. Hüseyin Uzunboylu, “Innovation and Creativity in EducationProject–based learning in engineering higher education: two decades of teaching competences in real environments,” *Procedia - Social and Behavioral Sciences*, vol. 2, no. 2, pp. 1368-1378, 2010.

[55] A. Helland, V. M. Durano and E. Price, ASP.NET Core 5 for Beginners: Kick-start your ASP.NET web development journey with the help of step-by-step tutorials and examples, Packt Publishing , 2020.

[56] M. L. Gillenson, Fundamentals of database management systems, John Wiley & Sons, Inc., 2012.

[57] J. D. Gauchat, El gran libro de HTML5, CSS3 y Javascript, Barecelona: MARCOMBO, S.A., 2012.

[58] A. Freeman, Pro ASP.NET Core 3 (Develop Cloud-Ready Web Applications Using MVC 3, Blazor, and Razor Pages) 8th ed. Edition, Apress, 2020.

[59] R. Elmasri and S. B. Navathe, Fundamentals of Database Systems, Hoboken: Pearson Education, 2016.

[60] “Statista,” [Online]. Available: https://www.statista.com/topics/1141/health-and-fitness-clubs/. [Accessed 04 04 2021].

[61] M. J. Price, C# 9 and .NET 5 – Modern Cross-Platform Development, Packt Publishing, 2020.

[62] S. Norberg, Advanced ASP.NET Core 3 Security, APress, 2020.

Appendices

Appendix A Survey (English-Template)

The purpose of this research is to know there is a need for gyms to be automated in software. The result of this investigation will be applied to the final investigation project of Master degree in Computer Engineering-Mobile.

Gym:				
Clients	Response		Recommended	
	Yes	No	Yes	No
Customers have their own online profile				
Customers can apply online				
Customers can update their data				
Trainers can update training results in user profiles				
Customers can view their subscriptions and payments				
Customers can control their access to the gym from their mobile phone				
Schedules	Response		Recommended	
	Yes	No	Yes	No
The gym has an online class schedule				
Gym teachers plan class schedules				
Customers can view their class schedule on a web or mobile application				
Customers can schedule their training times				
Management	Response		Recommended	
	Yes	No	Yes	No
In the gym application, training equipment and tools are managed				
Gym workers are managed in the application.				
Employee payments are managed by the application.				
The gym application controls the collection of customer subscriptions (billing)				
The gym application allows you to pay online				
The application allows you to configure subscription plans				
Internal management	Response		Recommended	
	Yes	No	Yes	No
In the gym application, the frequency of clients is controlled				
In the gym application, the entry and exit of customers is controlled				
The gym application controls the number of customers who are in the gym.				
In the gym application, it is controlled which equipment each client uses to carry out their training.				
In the gym application, the workers' schedules are managed				
Notifications	Response		Recommended	
	Yes	No	Yes	No

Customers are notified about their classes				
Teachers are notified about classes				
Customers are notified of subscription payment date.				
Schedule changes are reported in the gym.				
Customers are notified of new plans and subscriptions.				
Customer portal	Response		Recommended	
	Yes	No	Yes	No
Advertising is displayed to users				
Images of the installations are displayed				
Free classrooms are displayed				
Gym subscription plans are published (plan prices)				
Gym class timetables are published				
Software you use: Software Hosting: Number of customers:				

Comments or notes

Thank you very much...

Appendix B – Survey responses

Gym: Gym 1

Gym-Suite

Projeto Plataforma Extensível para Gestão de Ginásio

Sondagem

O objetivo desta pesquisa é saber há necessidade dos ginásios serem automatizadas em software. O resultado desta investigação será aplicado ao projeto de investigação final do Mater - Engenharia Informática-Informática Móvel.

Ginasio:				
Clientes	Resposta		Recomendado	
	Sim	Não	Sim	Não
Os clientes têm o seu próprio perfil online	X			
Os clientes podem inscrever-se online		X		
Os clientes podem atualizar os seus dados	X			
Professores podem atualizar resultados de formação em perfis de utilizadores	X			
Os clientes podem ver as suas subscrições e pagamentos	X			
Os clientes podem controlar o seu acesso ao ginásio a partir do seu telemóvel	X			
Horários	Resposta		Recomendado	
	Sim	Não	Sim	Não
O ginásio tem um horário de aula online		X		
Professores de ginásio planeiam horários das aulas		X		
Os clientes podem ver a sua agenda de aulas numa web ou aplicação móvel	X			
Os clientes podem agendar os seus horários de formação	X			
Administração	Resposta		Recomendado	
	Sim	Não	Sim	Não
Na aplicação do ginásio são geridos os equipamentos e ferramentas de treino		X		
Os trabalhadores do ginasio são geridos na aplicação	X			
Os pagamentos dos trabalhadores são geridos pela aplicação				
A aplicação de ginásio controla a recolha de subscrições de clientes (faturação)	X	X		
A aplicação de ginástica permite-lhe fazer pagamentos online		X		
A aplicação permite configurar planos de subscrição		X		
Gestão interna	Resposta		Recomendado	
	Sim	Não	Sim	Não
Na aplicação do ginásio é controlada a frequencia dos cliente	X			
Na aplicação do ginásio, é controlada a entrada e saída dos clientes	X	1 hora aula		
A aplicação do ginásio controla o número de clientes que estão no ginásio		X		
Na aplicação do ginásio é controlado que equipamento cada cliente usa para realizar o seu treino		X		
Na aplicação do ginásio, são geridos os horários dos trabalhadores		X		
Notificações	Resposta		Recomendado	
	Sim	Não	Sim	Não

Gym-Suite

Projeto Plataforma Extensível para Gestão de Ginásio

Os clientes são notificados sobre as suas aula	X			
Os professores são notificados sobre as aula	X			
Os clientes são notificados da data de pagamento da subscrição		—		
As alterações de horário são reportadas no ginásio	X			
Os clientes são notificados sobre novos planos e subscrições.	X	Sem opção recomend.		
Portal do cliente	Resposta		Recomendado	
	Sim	Não	Sim	Não
A publicidade é exibida aos utilizadores	X	Propor		
São exibidas imagens das instalações	X			
São exibidas as salas de aula gratuitas	X			
São publicados planos de subscrição de ginásio (preços de plano)		X		
São publicados horário da aula do Ginásio	X			
Software que utiliza: Hospedagem de softwar Número de clientes:				

Comentários ou notas

vilão

Muito obrigado...

Gym: Gym 2

Gym-Suite

Projeto Plataforma Extensível para Gestão de Ginásio

IPL escola superior de tecnologia e gestão

Sondagem

O objetivo desta pesquisa é saber há necessidade dos ginásios serem automatizadas em software. O resultado desta investigação será aplicado ao projeto de investigação final do Mater - Engenharia Informática-Informática Móvel.

Ginasio:				
Clientes	Resposta		Recomendado	
	Sim	Não	Sim	Não
Os clientes têm o seu próprio perfil online	X			
Os clientes podem inscrever-se online	X			
Os clientes podem atualizar os seus dados		X	X	
Professores podem atualizar resultados de formação em perfis de utilizadores	X			
Os clientes podem ver as suas subscrições e pagamentos	X			
Os clientes podem controlar o seu acesso ao ginásio a partir do seu telemóvel	X			
Horários	Resposta		Recomendado	
	Sim	Não	Sim	Não
O ginásio tem um horário de aula online	X			
Professores de ginásio planeiam horários das aulas	X			
Os clientes podem ver a sua agenda de aulas numa web ou aplicação móvel	X			
Os clientes podem agendar os seus horários de formação	X			
Administração	Resposta		Recomendado	
	Sim	Não	Sim	Não
Na aplicação do ginásio são geridos os equipamentos e ferramentas de treino		X		
Os trabalhadores do ginasio são geridos na aplicação	X			
Os pagamentos dos trabalhadores são geridos pela aplicação		X		
A aplicação de ginásio controla a recolha de subscrições de clientes (faturação)	X			
A aplicação de ginástica permite-lhe fazer pagamentos online	X			
A aplicação permite configurar planos de subscrição	X			
Gestão interna	Resposta		Recomendado	
	Sim	Não	Sim	Não
Na aplicação do ginásio é controlada a frequencia dos cliente	X			
Na aplicação do ginásio, é controlada a entrada e saída dos clientes		X		
A aplicação do ginásio controla o número de clientes que estão no ginásio	X			
Na aplicação do ginásio é controlado que equipamento cada cliente usa para realizar o seu treino		X		
Na aplicação do ginásio, são geridos os horários dos trabalhadores		X		
Notificações	Resposta		Recomendado	
	Sim	Não	Sim	Não

Gym-Suite

Projeto Plataforma Extensível para Gestão de Ginásio

Os clientes são notificados sobre as suas aula		X		
Os professores são notificados sobre as aula	X			
Os clientes são notificados da data de pagamento da subscrição	X			
As alterações de horário são reportadas no ginásio	X			
Os clientes são notificados sobre novos planos e subscrições.	X			
Portal do cliente	Resposta		Recomendado	
	Sim	Não	Sim	Não
A publicidade é exibida aos utilizadores	X			
São exibidas imagens das instalações	X			
São exibidas as salas de aula gratuitas	X			
São publicados planos de subscrição de ginásio (preços de plano)	X			
São publicados horário da aula do Ginásio	X			
Software que utiliza: Hospedagem de software: Número de clientes:				

Comentários ou notas

Codiga QR desde el Movil

Muito obrigado...

Gym: Gym 3

Gym-Suite

Projeto Plataforma Extensível para Gestão de Ginásio

Sondagem

O objetivo desta pesquisa é saber há necessidade dos ginásios serem automatizadas em software. O resultado desta investigação será aplicado ao projeto de investigação final do Mater - Engenharia Informática-Informática Móvel.

Ginasio:				
Clientes	Resposta		Recomendado	
	Sim	Não	Sim	Não
Os clientes têm o seu próprio perfil online	X			
Os clientes podem inscrever-se online		X		
Os clientes podem atualizar os seus dados		X		
Professores podem atualizar resultados de formação em perfis de utilizadores		X		
Os clientes podem ver as suas subscrições e pagamentos		X		
Os clientes podem controlar o seu acesso ao ginásio a partir do seu telemóvel		X		
Horários	Resposta		Recomendado	
	Sim	Não	Sim	Não
O ginásio tem um horário de aula online	X			
Professores de ginásio planeiam horários das aulas		X		X
Os clientes podem ver a sua agenda de aulas numa web ou aplicação móvel	X			
Os clientes podem agendar os seus horários de formação	X			
Administração	Resposta		Recomendado	
	Sim	Não	Sim	Não
Na aplicação do ginásio são geridos os equipamentos e ferramentas de treino	X			
Os trabalhadores do ginasio são geridos na aplicação		X		
Os pagamentos dos trabalhadores são geridos pela aplicação		X		
A aplicação de ginásio controla a recolha de subscrições de clientes (faturação)		X		
A aplicação de ginástica permite-lhe fazer pagamentos online		X		
A aplicação permite configurar planos de subscrição		X		
Gestão interna	Resposta		Recomendado	
	Sim	Não	Sim	Não
Na aplicação do ginásio é controlada a frequencia dos cliente	X			
Na aplicação do ginásio, é controlada a entrada e saída dos clientes	X			
A aplicação do ginásio controla o número de clientes que estão no ginásio		X		
Na aplicação do ginásio é controlado que equipamento cada cliente usa para realizar o seu treino		X		
Na aplicação do ginásio, são geridos os horários dos trabalhadores		X		
Notificações	Resposta		Recomendado	
	Sim	Não	Sim	Não

Gym-Suite

Projeto Plataforma Extensível para Gestão de Ginásio

Os clientes são notificados sobre as suas aula	X			
Os professores são notificados sobre as aula		X		
Os clientes são notificados da data de pagamento da subscrição		X		
As alterações de horário são reportadas no ginásio	X			
Os clientes são notificados sobre novos planos e subscrições.		X		
Portal do cliente	Resposta		Recomendado	
	Sim	Não	Sim	Não
A publicidade é exibida aos utilizadores		X		
São exibidas imagens das instalações		X		
São exibidas as salas de aula gratuitas		X		
São publicados planos de subscrição de ginásio (preços de plano)		X		
São publicados horário da aula do Ginásio	X			
Software que utiliza: Hospedagem de software: Número de clientes:				

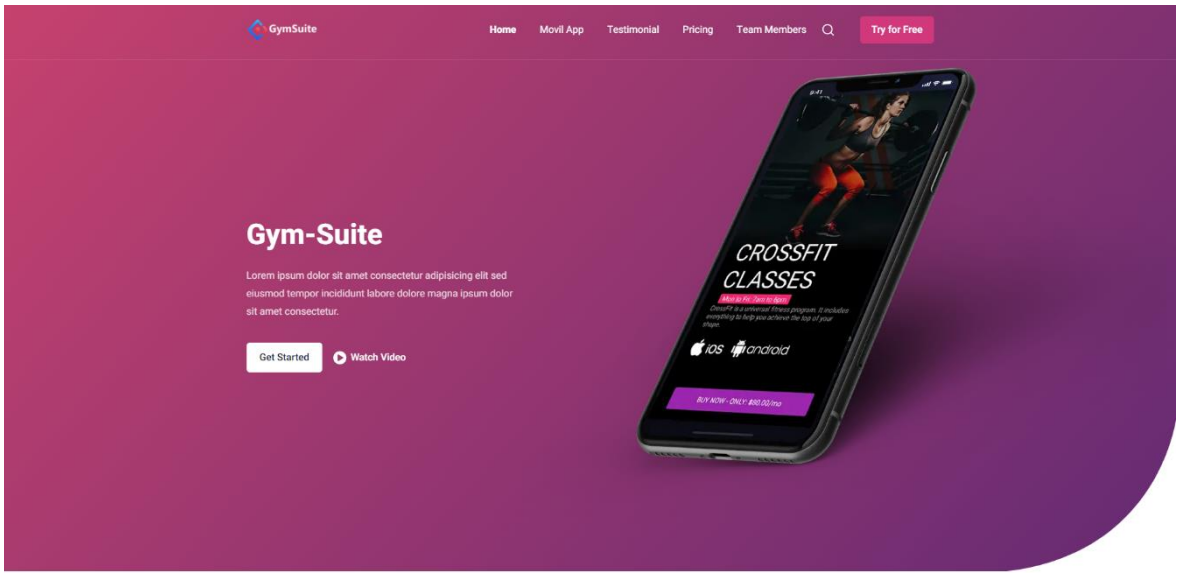
Comentários ou notas

código numerado

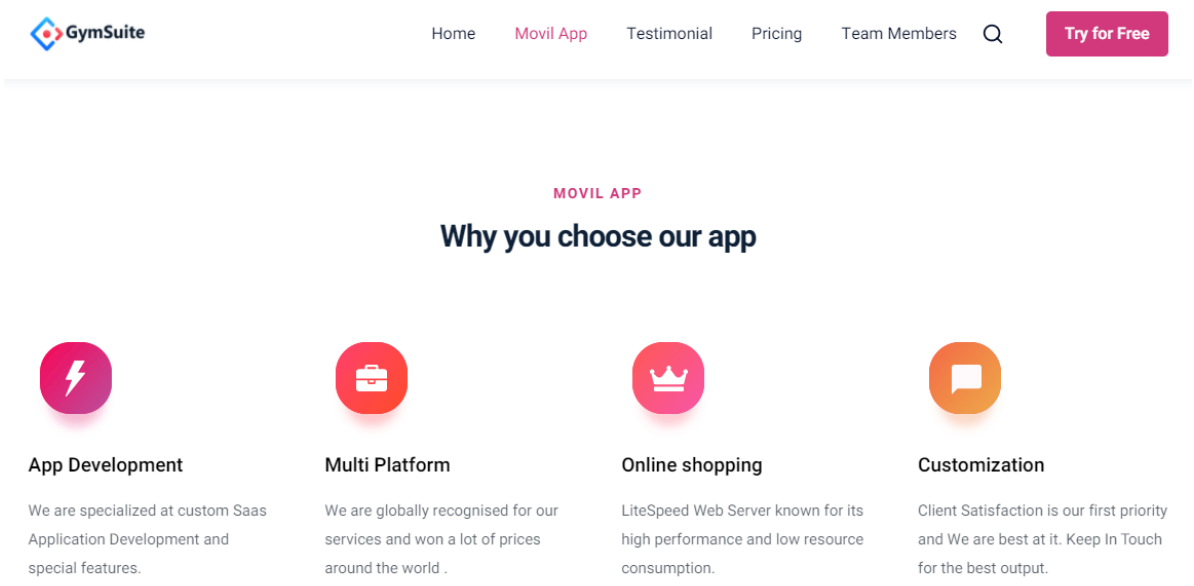
Muito obrigado...

Appendix C - Gym-Suite.Client

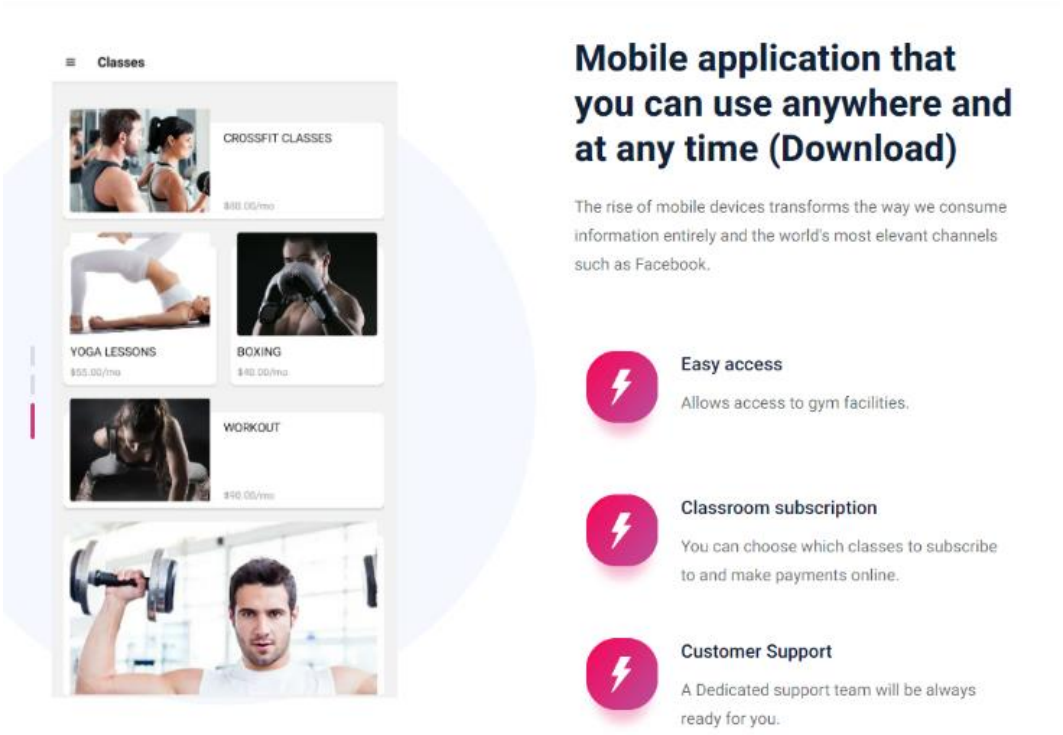
Home section



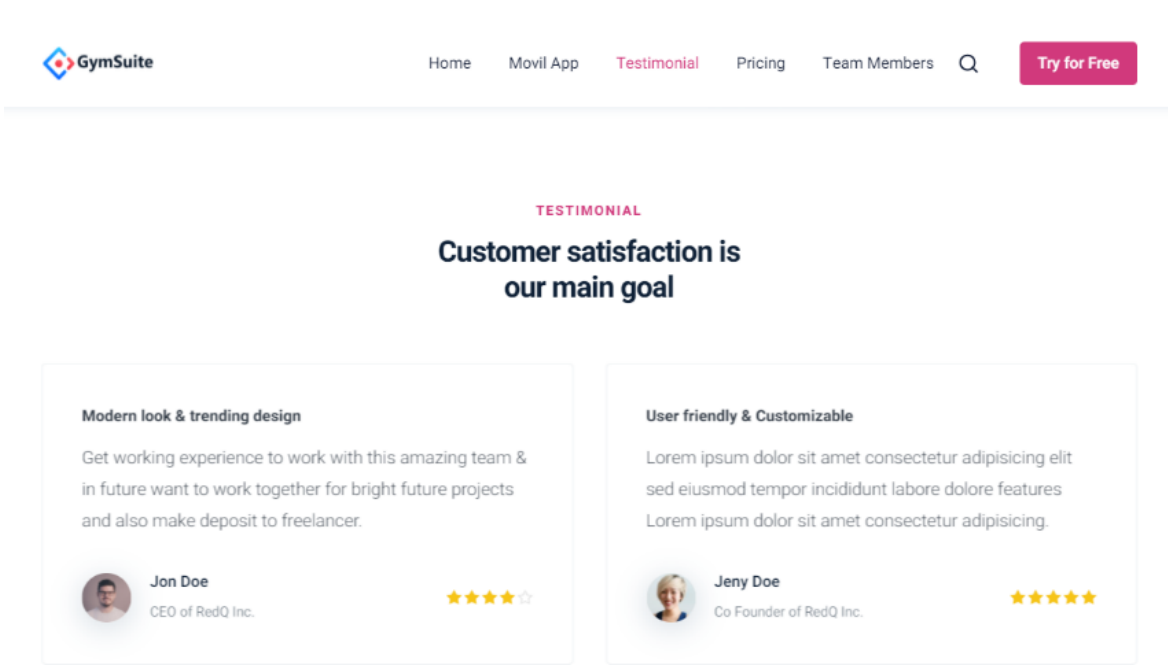
Mobile App Section



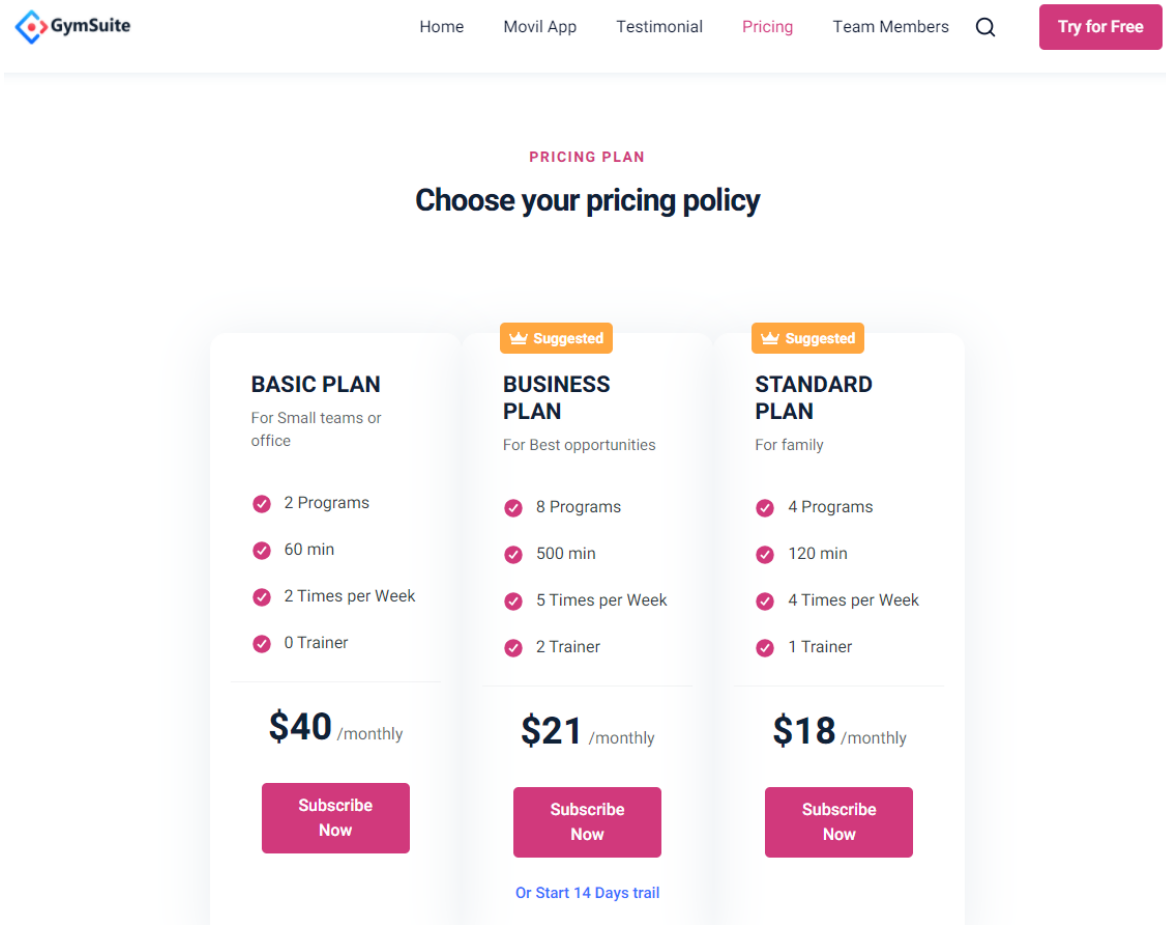
Mobile App Section



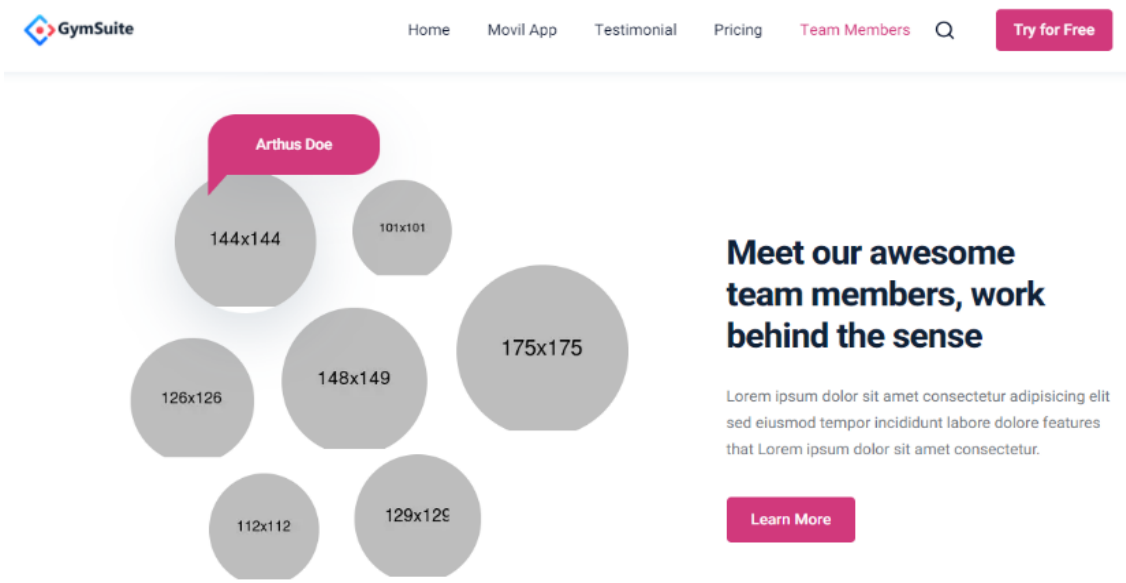
Testimonial Section



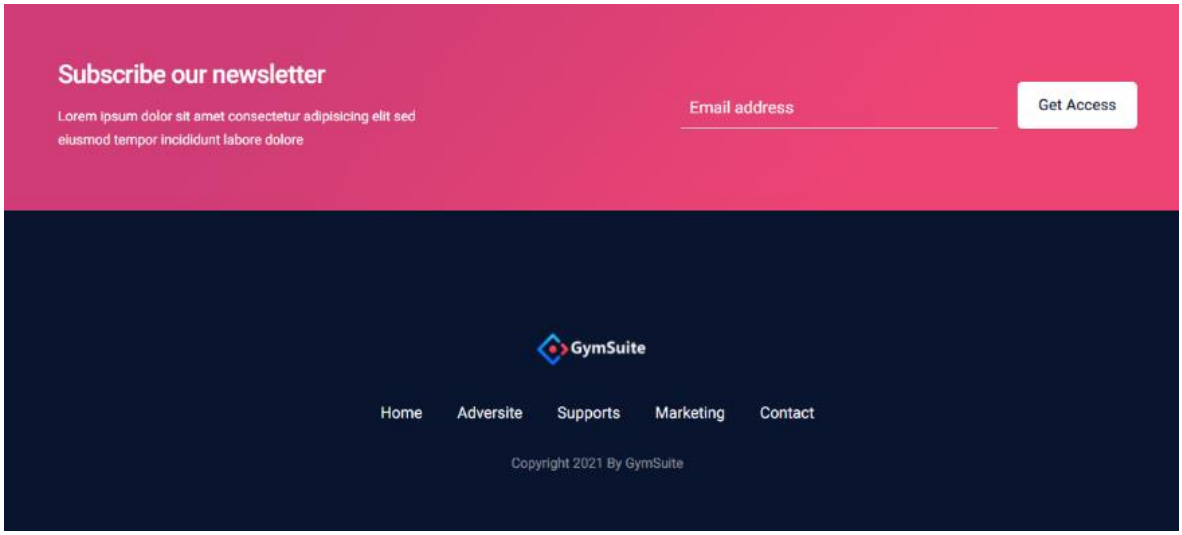
Pricing Plan Section



Our Team Section

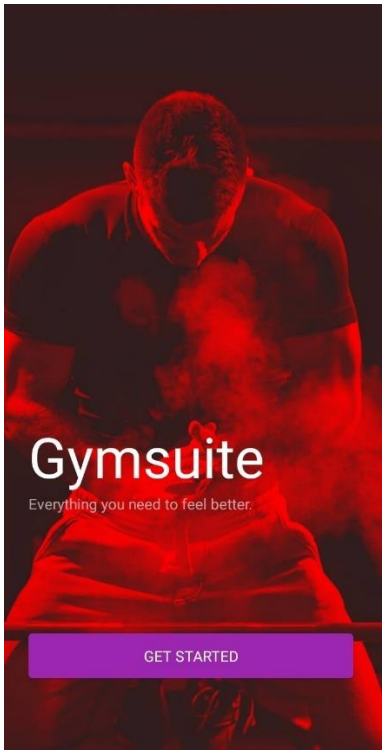


Web Site Footer Section

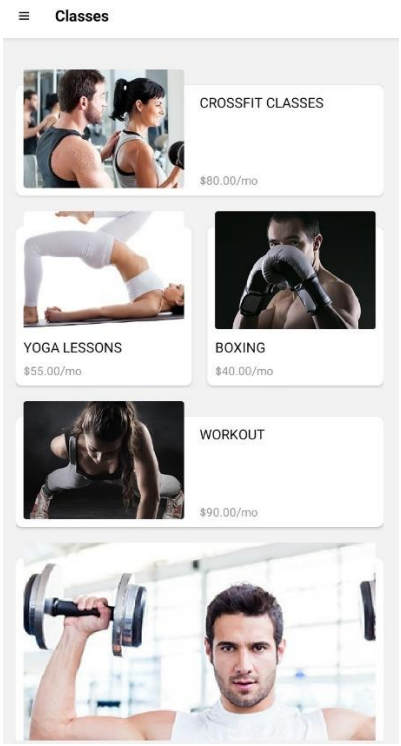


Appendix D - Gym-Suite.Client.App

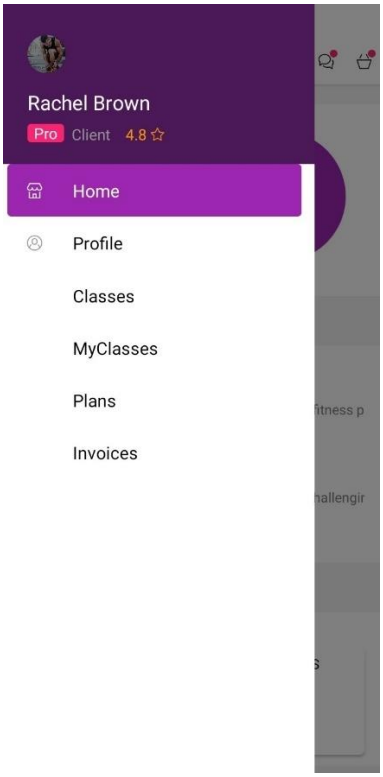
Mobile App Initial Screen



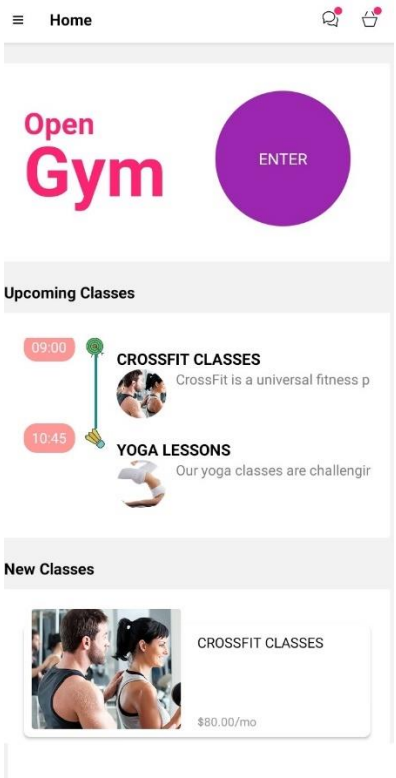
Mobile App Home Screen



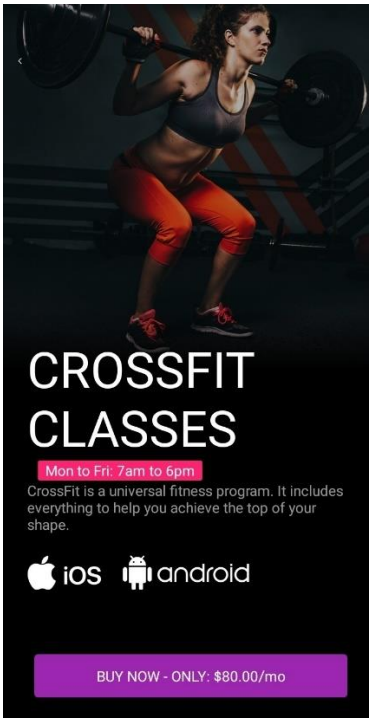
Main Menu



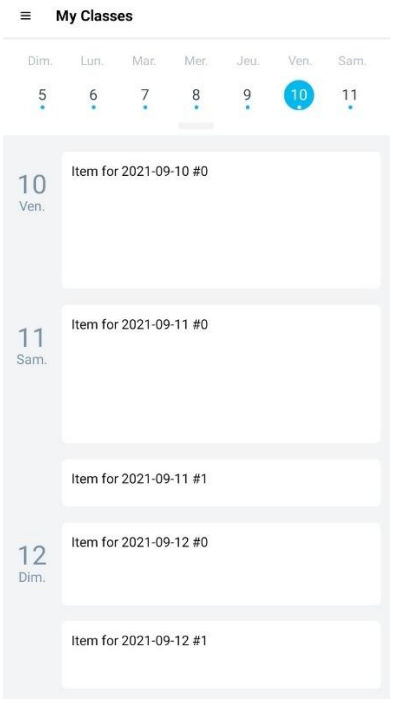
Class Detail Screen



Class Detail Screen



Calendar Screen



Appendix E – Access Control Response Test

Mobile app	Gymsuite.AccessControl.Api	Arduino	Notification
14:47:30	14:47:30	14:47:35	14:47:50
14:45:22	14:45:22	14:45:25	14:45:45
14:48:15	14:48:15	14:48:22	14:48:25
14:50:32	14:50:32	14:50:33	14:50:37

Time it takes to open the door (TD)
0:00:05
0:00:03
0:00:07
0:00:01

Time it takes to notify the user (TU)
0:00:15
0:00:20
0:00:03
0:00:04

Combined response time (TR)
0:00:15
0:00:20
0:00:03
0:00:04

Total Time it takes to open the door (TTD)	0:00:04
Total Time it takes to notify the user (TTU)	0:00:10
Total Combined response time (TTR)	0:00:10

Appendix F – Arduino