



PLANNING, MANAGEMENT & DEVELOPMENT OF A CLIENT AREA APPLICATION

Diego Sales Bellés

Professor: Antonio León Fernández

Final Master's Project presented at the
Higher Technical School of Telecommunication Engineers
of the Polytechnic University of Valencia,
to obtain the master's degree in Telecommunication's
Engineering

Course 2020-21

Valencia, February 22, 2021



Resumen

Un área de Cliente es una aplicación web desarrollada para los clientes de una empresa, con la finalidad de darles acceso a cierta información relevante y personal sobre un servicio contratado con una empresa.

Una aplicación web es un software que se ejecuta en un servidor web. El usuario accede a estas aplicaciones con una conexión activa a Internet, utilizando un navegador web. Normalmente, estas aplicaciones se basan en una estructura cliente-servidor, donde el cliente o usuario recibe servicios de un servidor externo alojado por un tercero.

Una aplicación web se puede utilizar con muchos propósitos, como un correo web, banca en línea o áreas de clientes.

En este Trabajo de Final de Master, presentamos información sobre la planificación, gestión y desarrollo de una aplicación de área de cliente, centrándose especialmente en el software back-end necesario para el proyecto. Los puntos principales del documento serán la planificación y gestión del área de clientes, el desarrollo del software del lado del servidor y su gestión y mantenimiento como administradores.

Resum

Una àrea de client és una aplicació web desenvolupada per als clients d'una empresa, amb el propòsit de donar-los accés a informació rellevant i personal sobre un servei contractat amb una empresa.

Una aplicació web és un programa que s'executa en un servidor web. L'usuari accedeix a aquestes aplicacions amb una connexió a Internet activa mitjançant un navegador web. Normalment, aquestes aplicacions es basen en una estructura client-servidor, on el client o l'usuari proporcionen serveis per un servidor extern que allotja un tercer.

Una aplicació web es pot utilitzar amb molts propòsits, com ara correu electrònic, banca en línia o àrees de client.

En aquest Treball de Final de Master, es presente informació sobre la planificació, la gestió i el desenvolupament d'una aplicació d'àrea de client, centrant-se especialment en el programari de fons necessari per al projecte. Els punts principals del document seran la planificació i la gestió de l'àrea del client, el desenvolupament del programari del servidor i la seva gestió i manteniment com a administradors.



Abstract

A Client area is a web application developed for the clients of a company, with the purpose of giving them access to some relevant and personal information regarding a hired service with a company.

A web application is an application software that runs on a web server. These applications are accessed by the user with an active Internet connection, using a web browser. Normally, these applications are based in a client-server structure, where the client or user is provided services by an off-site server hosted by a third party.

A web application can be used with many purposes, like a web-mail, online banking or client areas.

In this Final Master Thesis, we present information regarding the planning, management and development of a client area application, focusing specially on the back-end software needed for the project. The main points of the document will be the planning and management of the client area, the development of the server-side software and its management and maintenance as administrators.



Index

Chapter 1. Introduction, objectives and methodology.....	10
1.1. Project's introduction.....	10
1.2. Project's objectives	10
1.3. Project's methodology	11
1.4. Knowledge necessary for the work's comprehension	11
Chapter 2. Project's planning and management	13
2.1. Introduction	13
2.1.1. Application area	13
2.1.2. Project proposal.....	13
2.1.3. Business case	13
2.2. Scope of the project	14
2.2.1. Specification.....	14
2.2.2. Project organization.....	15
2.2.3. Lifecycle of the product	18
2.2.4. Work Breakdown Structure (WBS)	19
2.2.5. List of project activities	20
2.2.6. Format and procedure for requesting changes in the process.....	21
2.3. Temporal planning	22
2.3.1. Identification of precedence.....	22
2.3.2. Resources distribution and duration of activities	22
2.3.3. Calendar	23
2.4. Economic planning	24
2.4.1. Estimation of individual costs of activities.....	24
2.4.2. Elaboration of a budget	26
2.5. Uncertainty analysis	27
2.5.1. Environmental analysis	27
2.5.2. Internal analysis: SWOT Analysis	28
2.5.3. Risk and opportunities treatment.....	28
2.6. Closing of the project	29
2.6.1. Conclusions	29
2.6.2. Future lines of action	29
2.6.3. Learned lessons.....	30
Chapter 3. Customer Relationship Management Software Development.....	31
3.1. Definition of minimum requirements.....	31
3.2. Lead creation proposed workflow	31
3.3. Development of the lead creation process	32
3.4. Development of an energy consumption calculator.....	33
3.4.1. Integromat	34
3.4.2. Docsumo	37
3.4.3. Columns	38
3.5. Clients spreadsheet.....	38
3.6. Proposal spreadsheet.....	39



3.7.	Fee spreadsheet	40
3.8.	Bills spreadsheets	40
3.8.1.	Payments	40
3.8.2.	Consumptions	40
3.8.3.	Periods	40
3.9.	Benefits spreadsheet	41
3.10.	Taxes spreadsheet	41
3.11.	Aggregate spreadsheet	41
3.12.	Test	41
Chapter 4. Back-end development of the project		43
4.1.	Definition of minimum requirements for first version	43
4.2.	Definition of the product areas	43
4.2.1.	Identification area	44
4.2.2.	Services area	44
4.2.3.	Main menu area.....	45
4.2.4.	Emissions area	45
4.2.5.	Fees area	46
4.2.6.	Consumptions area	46
4.2.7.	Savings area	47
4.2.8.	Communication between front-end and back-end area	47
4.2.9.	Database area	47
4.3.	Definition of the work environment	48
4.4.	Definition of the startup file	49
4.5.	Definition of the appsettings file	50
4.6.	Definition of data model	51
4.7.	Databases treatment	52
4.7.1.	DbContext	52
4.7.2.	Migration	52
4.7.3.	Filling.....	53
4.8.	Definition of services, interfaces and infrastructure	53
4.8.1.	Authentication classes	54
4.8.2.	Account managing classes	56
4.9.	Definition of controllers	57
4.9.1.	Authentication controller	57
4.9.2.	Client controller	58
4.10.	Server publication	60
4.11.	Maintenance	61
4.11.1.	Automation of databases filling	62
4.11.2.	User's addition	65
4.12.	Results	66
4.12.1.	Workflow.....	66
4.12.2.	Testing.....	67
Chapter 5. Conclusion		73
Chapter 6. References		75



5.	Annex A	79
6.	Annex B	80
7.	Annex C	81
8.	Annex D	83
9.	Annex E	86
10.	Annex F	88
11.	Annex G	91
12.	Annex H	92



Index of figures

FIGURE 1. QR CODE WITH ACCESS TO THE GOOGLE DRIVE SHARED FOLDER, WITH CONTENT OF THE PROJECT. [20] [21]	21
FIGURE 2. GANTT DIAGRAM OF THE PROJECT.....	23
FIGURE 3. FLOWCHART OF SCENARIO A.....	34
FIGURE 4. FLOWCHART OF SCENARIO B.....	36
FIGURE 5. FLOWCHART OF SCENARIO B.....	36
FIGURE 6. QR CODE TO ACCESS THE SPREADSHEET	38
FIGURE 7. DOCSUMO’S REVIEW MODE.....	42
FIGURE 8. CLIENT AREA WORKFLOW	43
FIGURE 9. AUTHENTICATION WEBPAGE	44
FIGURE 10. SERVICES AREA WEBPAGE.....	44
FIGURE 11. MAIN MENU WEBPAGE.....	45
FIGURE 12. EMISSIONS AREA WEBPAGE	45
FIGURE 13. FEES AREA WEBPAGE.....	46
FIGURE 14. CONSUMPTIONS AREA WEBPAGE	46
FIGURE 15. SAVINGS AREA WEBPAGE	47
FIGURE 16. ENTITY FRAMEWORK CORE SCHEMA	49
FIGURE 17. ENTITIES SCHEMA	51
FIGURE 18. AUTHENTICATION PROCESS SCHEMA.....	54
FIGURE 19. MODIFY PASSWORD’S PROCESS SCHEMA	55
FIGURE 20. AUTHENTICATION’S ENDPOINT ROUTE	57
FIGURE 21. AUTHENTICATION’S ENDPOINT JSON BODY STRUCTURE	57
FIGURE 22. SUCCESSFUL AUTHENTICATION’S REPLY	57
FIGURE 23. START MODIFYING PASSWORD’S ENDPOINT ROUTE.....	58
FIGURE 24. FINISH MODIFY PASSWORD’S ENDPOINT ROUTE	58
FIGURE 25. GET ALL THE MANAGED HOUSE’S ENDPOINT ROUTE	58
FIGURE 26. GET ALL THE USER’S DATA ENDPOINT ROUTE	59
FIGURE 27. GET ALL THE BILL’S DATA ENDPOINT ROUTE.....	59
FIGURE 28. GET ALL THE USER’S SERVICES DATA ENDPOINT ROUTE	59
FIGURE 29. POST A NEW USER ENDPOINT ROUTE	59
FIGURE 30. DEPLOYED DOCKER CONTAINERS IN OUR SERVER	61
FIGURE 31. INTEGROMAT’S WORKFLOW	62
FIGURE 32. INTEGROMAT’S WORKFLOW	64
FIGURE 33. INTEGROMAT’S WORKFLOW	64
FIGURE 34. INTEGROMAT’S WORKFLOW	65
FIGURE 35. INTEGROMAT’S WORKFLOW	65
FIGURE 36. TESTING WORKFLOW.....	66
FIGURE 37. RESULT OF PHASE 1	67
FIGURE 38. BILL ANALYSIS ON OUR OCR	68
FIGURE 39. RESULT OF PHASE 2	68
FIGURE 40. RESULT OF PHASE 3	68
FIGURE 41. START PROCESS.....	69
FIGURE 42. RECEIVED EMAIL.....	69
FIGURE 43. END PROCESS	70
FIGURE 44. LOGIN REQUEST	70
FIGURE 45. REFRESH TOKEN COOKIE.....	70
FIGURE 46. UNAUTHORIZED PETITION	71
FIGURE 47. USER SERVICES REQUEST	71
FIGURE 48. BILLS FROM HOUSE REQUEST	72
FIGURE 49. DIAGRAM OF PRECEDENCIES	79
FIGURE 50. QUESTION 1.....	81
FIGURE 51. QUESTION 2.....	81
FIGURE 52. QUESTION 3.....	81
FIGURE 53. QUESTION 4.....	82
FIGURE 54. QUESTION 5.....	82



FIGURE 55. QUESTION 6.....	82
FIGURE 56. QUESTION 7.....	82
FIGURE 57. PAGE 1 OF A TEST BILL.....	83
FIGURE 58. PAGE 2 OF A TEST BILL.....	84
FIGURE 59. PAGE 3 OF A TEST BILL.....	85



Index of tables

TABLE 1. GANTT DIAGRAM	11
TABLE 2. HUMAN RESOURCES NEEDED.....	15
TABLE 3. TANGIBLE RESOURCES NEEDED.....	16
TABLE 4. INTANGIBLE RESOURCES NEEDED	16
TABLE 5. DELIVERABLE'S DESCRIPTION	17
TABLE 6. PROJECT BUDGET	26
TABLE 7. PESTEL ANALYSIS.....	27
TABLE 8. SWOT MATRIX [22]	28
TABLE 9. TYPEFORM QUESTIONNAIRE	33
TABLE 10. TABLE WITH THE PROJECT PLANNING.....	80
TABLE 11. FORMULAS OF THE CALCULATOR SPREADSHEET	86
TABLE 12. FORMULAS OF THE CLIENT'S SPREADSHEET	87
TABLE 13. FORMULAS OF THE PROPOSAL'S SPREADSHEET	87
TABLE 14. FORMULAS OF THE PROPOSAL'S SPREADSHEET	92

Chapter 1. Introduction, objectives and methodology

1.1. Project's introduction

A Client area is a web application developed for the clients of a company, with the purpose of giving them access to some relevant and personal information regarding a hired service with a company.

A web application is an application software that runs on a web server. These applications are accessed by the user with an active Internet connection, using a web browser. Normally, these applications are based in a client-server structure, where the client or user is provided services by an off-site server hosted by a third party. [1]

A web application can be used with many purposes, like a web-mail, online banking or client areas.

This project will detail the planning, development and management of the first version of a client area developed during an internship done at a Start-Up called Onerz. When the project was developed, the start-up work as an intermediary between clients and electric companies, helping clients to find out cheaper services and also reduce their carbon fingerprint.

Also, this project includes the development of a Customer Relationship Management (CRM from now onwards) program, developed on Google Sheets, with the purpose of helping the operations and sales staff of the company in their daily work, helping them to make automatically electricity studies for clients, as well as organizing properly their list of clients.

The software of this work has been developed using .NET Core 3.1. (based on C#), Entity Framework Core as the Object Relational Mapper (ORM) [2] and PostgreSQL as the open-source relational database provider [3].

1.2. Project's objectives

The main goals of this work can be divided into three big groups:

- Planning of the project:
 - Establishment of a realistic calendar.
 - Effective management of the available resources.
 - Risk management.
 - Supervision of the daily work.
- CRM development:
 - Automatize electricity studies.
 - Provide company staff a helpful tool for organizing clients and leads.
- Back-end development of the project:
 - Give to clients the possibility of accessing their data and electric bills.
 - Show clients their last month's data regarding consumption, generated CO2 and money savings compared to their old electric rates.

1.3. Project's methodology

The project will follow these stages:

- Firstly, there will be a brief investigation on some basic concepts that will be necessary for the development of the work. This stage will take one week.
- Secondly, the student will analyze some important data for the project, like the needs, resources or communications with the CEO of the company regarding the monitorization of the project. This stage will take one week.
- Then, the planning of the project will be developed. This includes the analysis of resources, definition of some deliverables, the definition of a list of activities to do and which resources will be used in each activity, the definition of a budget, the definition of a calendar and the definition of risks of the project. This stage will take three weeks.
- Once the project is fully defined, it will be time for the development of the project, that will include:
 - Development of a CRM for the company staff. This stage will take three weeks.
 - Creation and filling of databases with fake data (start) and real data (end). This stage will take two separate weeks.
 - Development of API software. This stage will take four weeks.
 - Development of automations. This stage will take two weeks.
- Finally, there will be a conclusion, where the whole work will be analyzed, in order to see if the goals of the work where met. This stage will take one week.
- In the end, the redaction of the project will be done. This stage will take four weeks and will be done from all the data collected during the work.

According to all this information, we think the work will be done in 21 weeks. All this information is summarized in a Gantt diagram, available in the figure 1.

Week	1	2	3	4	5	6	7	8	9	#	#	12	13	14	15	16	17	18	19	20	21	22	23	
Basic concepts investigation	█																							
Market analysis		█	█																					
Project planning				█	█	█																		
Software development						█	█	█	█	█	█	█	█	█	█	█								
Testing																	█							
Publication																		█						
Conclusion of the project																			█					
Redaction of the document																				█	█	█	█	

Table 1. Gantt diagram

1.4. Knowledge necessary for the work's comprehension

There is some knowledge that could be interesting to be aware of for the complete comprehension of the work. This is detailed as it follows.

- Relational Database: digital database based on the relational model of data, which is an approach to managing data using a structure and language consistent



in the first-order predicate logic, consistent on the use of quantified variables over non-logical objects. [4][5][6]

- Application Programming Interfaces (API): computing interface that defines the interactions between multiple software intermediaries. An API can define the kind of calls or requests that can be made or the way they can be made. [7] In this work, there will be two kind of APIs, one for the development branch, and another one for the production branch. This one will be the branch clients will be requesting through the web service application.
- HTTP Methods: those are the kind of requests a user can make to an API. Those are:
 - HTTP GET: used for retrieving data only. A GET request cannot change the state of the resource.
 - HTTP POST: used for creating new subordinate resources in a database table. It is normally used for adding a new row in a database table.
 - HTTP PUT: used for updating an existing resource in a database table.
 - HTTP DELETE: used for deleting resources.
 - HTTP PATCH: used for making partial updates on a resource. It can also be used for a total update, like a PUT request. [8]
- Object Relational Mapper (ORM): coding technique that transform data used in an oriented-to-objects coding language into a relational database as a persistence engine, creating a virtual oriented-to-objects database over the relational database, providing benefits like heritage and polymorphism. [9]
- Customer Relationship Management Software (CRM): software made with the purpose of being a helpful tool for manage relationships between clients and sales and marketing departments, compiling data from clients and leads. [10]

Chapter 2. Project's planning and management

2.1. Introduction

2.1.1. Application area

Onerz is a new-borned start-up with the mission of offering sustainable alternatives for the daily services that clients enjoy every day. The main goal is to help in the process of turning the world into a more sustainable and fairer place. [11]

As one of their first services, the company has been offering clients a change to a new electricity company, offering them as well saving yearly money and to reduce their carbon fingerprint. As an exchange, the company gets a monthly fee for their services. With this action, the company has been working as an intermediate. But the company also needs to offer their clients information regarding their bills, consumptions, savings, etc.

2.1.2. Project proposal

Once the context has been shown, the company needs to develop:

- An informatic tool for helping the company to manage the relationships with clients, compiling data from clients and leads. This part of the project will improve the way of working of the company, as well as gives the chance to scale the business.
- A web application for giving clients information regarding their hired services with the company. This part of the project will help to win trust with clients, providing them information in a clear way.

2.1.3. Business case

With this proposal, we pretend to provide a solution that improves the way of working of the coworkers, as well as we would provide a solution to the company for giving clients relevant information.

With the development of the CRM, we can make automatically electric studies, so our leads can be turned into clients in much less time than before. Also, this CRM can make easier for the company to manage the relations with the clients, which will give a better inside organization, avoiding possible mistakes.

Regarding the client area, this will provide essential information to clients regarding their services with the company, ensuring in that way one of the values of the company, transparency. This web application is expected to provide a good image of the company to the clients, and this good image can be turned into the emergence of new leads, that can be turned into clients if our offers are of their interest.

This project will be the first stage of development of the company. According to the success of the project, the expansion of the company into new services (like water supply or gas supply) will be studied.

2.2. Scope of the project

2.2.1. Specification

In order to provide clients and staff the projected service, we need to develop and implement the next elements:

- Development of the communication channels:
 - These communication channels are:
 - Questionnaires through which leads will be able to give us relevant information for making their studies.
 - Emails through which staff will be able to communicate with leads.
- Development of an electric calculator:
 - In this electric calculator, electrical studies for leads will be done, in order to see if we can offer a better alternative to their current energy supplier.
 - This electric calculator will include all the extracted information from a bill of the interested lead, that will send us by taking a picture or sending us a document by filling a questionnaire.
 - After configuring the calculator, we will be able to check the savings in money and emissions per year if the lead hired our services.
- Development of the CRM
 - The CRM will include a different page for:
 - Calculator: contains all the electrical studies for each lead.
 - Clients: contains the state of every lead, differentiating between a discarded lead, a lead that could be recovered, a client, a lead that is pending to accept our offer, a lead that still needs a human approval for his study, and a lead that couldn't be studied with the information he provided to us.
 - Proposals: contains the final proposals made to leads.
 - Quotes: contains the monthly quote of each client for hiring our services.
 - Payments: includes the monetary amount of every bill.
 - Periods: includes the start and ending of the period of every bill.
 - Consumptions: includes the consumption (in kWh) of every period, divided into sections, according to their hired services.
 - Benefits: includes the expected benefit from each client per month.
 - Taxes: includes the expected payment of the company because of taxes, per client. It's a register of the VAT from each client.
 - Aggregate: contains a calculation of benefits and losses of the company per client.
 - Savings: contains a calculation of benefits and losses for the client.
- Development of a database:



- This database will store all the relevant information of clients, like personal data, bills, consumptions, savings, data from their old bills, data from different electric companies and hired services.
- Development of the API:
 - The back-end will define the different HTTP Requests that will be necessary for accessing all the information from the databases, as well as will calculate savings of emissions and money for clients.
- Development of a front-end application:
 - The front-end part of the web application will be making request to the API, in order to get the information that is requested by clients.
 - Also, it will provide an interface for clients to interact through their web navigator.
- Integrations for automatizing the filling of databases:
 - These processes will connect the databases and the CRM, so when a lead is turned into a new client, his user will be created. In that way, he will be able to connect to the client application.
 - Also, the bills that the company receives from clients will be stored and classified according to their house and interested client in the databases automatically.

2.2.2. Project organization

As in any project, the organization is of vital importance for the success or failure of this. In addition, since it is a start-up, it does not yet have the experience or clearly defined organizational models, so it is essential to establish a good foundation from the beginning. Obviously, the more solid the foundation, the higher the quality of service can be given to the customer, who in turn will be more satisfied.

2.2.2.1. Resources analysis

For the development of the project, we have identified three big resources groups, that are the human resources, tangible resources and intangible resources.

Human resources

All the companies are sustained on people. In our case, due to the start-up was recently created, we need to define precisely the amount of people we need to work for this project. We estimate that we will need:

	Nombre del recurso	Iniciales	Grupo	Tipo	Etiqueta de	Iniciales	Capacidad	Tasa	Tasa horas	Costo/U.	Acumu	Calendario
1	Project Manager	PM	Project Manager	Trabajo		PM	100%	8,00 €/hora	8,00 €/hora	0,00 €	Prorrateo Estándar	
2	Developer 1	D1	IT	Trabajo		D1	100%	5,75 €/hora	5,75 €/hora	0,00 €	Prorrateo Estándar	
3	Developer 2	D2	IT	Trabajo		D2	100%	5,75 €/hora	5,75 €/hora	0,00 €	Prorrateo Estándar	
4	Developer 3	D3	IT	Trabajo		D3	100%	5,75 €/hora	5,75 €/hora	0,00 €	Prorrateo Estándar	
5	Development Manager	DM	IT	Trabajo		DM	100%	8,00 €/hora	8,00 €/hora	0,00 €	Prorrateo Estándar	

Table 2. Human resources needed

Tangible resources

Tangible or material resources are those quantifiable and measurable assets belonging to our start-up, and which are necessary for the development of the project. The following table shows the tangible resources that have been foreseen as necessary:

Resource	Units	Unitary Cost (€)	Total Cost (€)
Computers [12]	5	549	2745

Table 3. Tangible resources needed

Intangible resources

Intangible or immaterial resources are those assets without physical presence. In our case, we will need:

Resource	Units	Unitary Cost (€/month)	Months	Yearly Total Cost (€)
G Suite [13]	5	4,62	12	277,2
Cloud Server [14]	1	2,178	12	26,136
OCR [15]	1	8,33	12	99,96
Integromat [16]	1	7,71	12	92,52
Typeform [17]	1	50,62	12	607,44
Kantree [18]	5	7	12	420

Table 4. Intangible resources needed

This is the first technological project done by the start-up. Normally, start-ups lack of resources from the origin. In our case, the most part of the capital of the company comes from investments made by a start-up accelerator, Zubi Labs, providing the possibility of developing the project. This fact gives the chance to develop and maintain the whole project.

Even though the company has this great economic help, we need to distribute our resources in the most efficient way possible.

2.2.2.2. *Deliverable's description*

During the project's development, the workers will be posting some deliverables, in order to give visibility to the state of the project, giving a real time vision of itself, as well as making it easier to analyze the meeting of the initial calendar.

The deliverables have to include, at least, the following information:

- Done activities every week.
- Intended activities to be done every week.
- Time used for each activity.
- Time forecast for each activity.

All these deliverables will be discussed with the project manager every Monday, by using some platforms, like Kantree [18], and online meetings (by using Google Hangouts). With this information, the project manager will make a weekly summary to our client, in this case, the CEO of the start-up. In case of changes in the planning are needed, these will be made by the project manager, and negotiated with the CEO.

The most important deliverables are defined in the next table:

Task	Deliverable
Planning	<ul style="list-style-type: none"> • Economic report • Action plan • Deadline analysis
CRM Development	<ul style="list-style-type: none"> • Requirements report • Workflow definition report • Model definition report • First version sketch • Development report • Testing report • Manual of instructions
Software development	<ul style="list-style-type: none"> • Functional needs report (software) • DB modeling report • Report with a first sketch of the front-end of the App (customer + employee) • App development report • Code accessible in different repositories • Private • Back-front integration report • Test report • Employee use manual • Customer use manual
Release	<ul style="list-style-type: none"> • Communication campaign
Ending of project	<ul style="list-style-type: none"> • Report of results, lessons learned and project closure

Table 5. Deliverable's description

2.2.2.3. *Assumptions and restrictions*

Before starting any project, it is important to study the assumptions and restrictions made in order to have identified the main weaknesses and, in this way, try to prevent them from becoming long-term problems.

The start-up developers were able to focus their whole work schedule to the development of the project.

Also, we assumed that the company shuttle could help us financing the project, as one of the creators of the start-up.



Also, we assumed that only one developer (Developer 1) will develop the back-end side of the application, help by another developer (Developer 2) with the DevOps functions, while there will be one developer (Developers 3) working on the front-end side of the application.

Finally, we assumed that this project would provide an important service for both staff of the start-up and clients, making easier for the company to provide of information to clients, as well as making easier to win new clients.

But this project has also some restrictions. The project was developed during the global pandemic of 2020, during an emergency state in Spain. So, the whole project was developed during a time of isolation at home for the population. So, mainly, it was developed by using remote work tools, instead of working together in the same space.

2.2.3. Lifecycle of the product

2.2.3.1. *Phases of the project*

We can divide the project into the next phases:

- **Project management:** in this phase, the project will be defined, and will be traced. This phase will last for the whole development of the project, due to the necessity of a constant monitoring and management.
- **Market analysis:** in this phase, we pretend to get a global vision of the environment where the project will be developed, defining the main advantages and disadvantages, and taking some successful and similar projects as guides for us. This will be the first phase to develop, apart from the project management.
- **Acquisition of the needed material:** in this phase, the company will buy all the needed material and resources (like extra computers, subscriptions or licenses) for starting the activity, and will start after we have a clear idea of what are we going to do.
- **Software development:** in this phase, all the software will be developed (both back and front-end). This will start after acquiring all the materials.
- **Testing:** in this phase, the web application will be tested, in order to detect and fix mistakes and bugs. This phase will start after the whole development is finished.
- **Publishing:** this phase will consist on the deployment of the final web application and announce it to our clients.
- **Ending of the project:** this phase will consist on the analysis of the fulfilled objectives, as well as the final results of the project will be analyzed. This is the last step of the project.

2.2.3.2. *Key checkpoints*

The key checkpoints will help us to visualize if we are reaching our goals with the estimated quality. Because we work using the Scrum technique (a technique for collaboration in a team when developing complex products) [19], every week the work will be planned and supervised.



Also, it will help the development of the project defining some intermediate milestones, in order to control the project progress, and check its correct timing. These milestones are directly related to the phases of the project, and are the next ones:

- Milestone 1: Presentation of a market analysis report.
- Milestone 2: Software needs report.
- Milestone 3: Completion of hardware and licenses acquisition.
- Milestone 4: Completion of the CRM.
- Milestone 5: Completion of the back-end development.
- Milestone 6: Completion of the front-end development.
- Milestone 7: Service available for our customers.
- Milestone 8: End of project.

Fulfilling these requirements will be a signal that the project is going well, meanwhile, if we don't get the milestones on time, we will have to analyze the causes and establish correcting measures for avoiding this to happen again.

2.2.4. Work Breakdown Structure (WBS)

Like in every project, is necessary to divide and structure the different tasks to do. In this case, the structure of the project will be:

1. Project "First Phase of Onerz Development"

1.1. Project Management.

1.1.1. Supervision and management of the project.

1.2. Market analysis.

1.2.1. Identification of the needs of our clients.

1.2.2. Analysis of technological solutions.

1.2.3. Research and analysis of similar cases and potential competitors.

1.2.4. *Milestone 1 – Presentation of a market analysis report.*

1.3. Acquisitions.

1.3.1. Analysis of material needs.

1.3.2. Analysis of required licenses and software.

1.3.3. Acquisition of needed resources.

1.3.4. *Milestone 3 – Completion of hardware and licenses acquisition.*

1.4. Software development.

1.4.1. Analysis of functional needs.

1.4.2. *Milestone 2 – Software needs report.*

1.4.3. Development of the CRM

1.4.3.1. Creation of tables.

1.4.3.2. Definition of functions.

1.4.3.3. Coding of functions.

1.4.3.4. Tests.

1.4.3.5. Connection with sources of information.

1.4.3.6. *Milestone 4 – Completion of the CRM.*



- 1.4.4. Web app design.
 - 1.4.4.1. Web app sketch (front-end).
- 1.4.5. Web app development.
 - 1.4.5.1. Back-end development.
 - 1.4.5.1.1. Creation of Docker containers in a server
 - 1.4.5.1.2. Definition of models and entities.
 - 1.4.5.1.3. Definition of databases context.
 - 1.4.5.1.4. Migration to databases in a server.
 - 1.4.5.1.5. Population of databases with test data.
 - 1.4.5.1.6. Definition of services.
 - 1.4.5.1.7. Definition of API controllers.
 - 1.4.5.1.8. Upload API image into development container.
 - 1.4.5.1.9. Tests with server in development.
 - 1.4.5.1.10. Upload API image into production container.
 - 1.4.5.1.11. Tests with server in production.
 - 1.4.5.1.12. Population of databases with real data.
 - 1.4.5.1.13. Elaboration of a manual.
 - 1.4.5.1.14. *Milestone 5 - Completion of the back-end development.*
 - 1.4.5.2. Front-end development.
 - 1.4.5.2.1. Authentication page
 - 1.4.5.2.2. Index page
 - 1.4.5.2.3. Emissions page
 - 1.4.5.2.4. Expenses page
 - 1.4.5.2.5. Consumption page
 - 1.4.5.2.6. Savings page
 - 1.4.5.2.7. Interconnection back-front end.
 - 1.4.5.2.8. Upload to development
 - 1.4.5.2.9. *Milestone 6 – Completion of the front-end development.*
- 1.5. **Final tests**
 - 1.5.1. Testing in development
 - 1.5.2. Testing in production
- 1.6. **Project publication**
 - 1.6.1. Publication of the final web application (front-end production).
 - 1.6.2. *Milestone 7 – Service available for our customers.*
 - 1.6.3. Communication to clients.
- 1.7. **End of project**
 - 1.7.1. Closing document.
 - 1.7.2. *Milestone 8 – End of project.*

2.2.5. List of project activities

Due to, our EDT if composed of a lot of activities, we have decided to upload a screenshot of every card, made with Kantree, in a Google Drive shared folder, accessible easily by scanning the next QR Code.



Figure 1. QR Code with access to the Google Drive shared folder, with content of the project. [20] [21]

2.2.6. Format and procedure for requesting changes in the process

During the development of the project, there can be some new circumstances that may make us think about changing some aspects of the project. In this case, there should be, at first, a request to make changes in the project. In order to make these requests, we defined a procedure, that will be explained next to.

Also, it's important to note that everyone involved in the project will be able to make these requests, listening in that way all the opinions of the people involved, improving our way to progress with the project.

The procedure consists on the next steps:

1. Elaboration of a modification report: anyone that thinks the project needs changes in their work area, has to make a brief document explaining the reasons of them.
2. Sending of the report: the interested has to send the document to the project manager.
3. Analysis of the modification: the project manager has one day to study the report. Once the project has been studied, the project manager will discuss it with the one that wrote the report and the person in charge of the affected area of the project (in case they are not the same person) in an online meeting. During this meeting, they will analyze the impact that the modification will have in the project and its viability.
4. Acceptation or rejection: as a result of the meeting, there will be a conclusion, where the modification will be accepted or rejected.
 - i. Acceptation:
 1. Final report, made by the project manager, available for everyone involved in the project.
 2. Execution of the modification, reporting to all the people involved.
 3. Control and follow up of the modification.
 - ii. Rejection:
 1. If the result of the analysis is negative for the request of modification, the writer of the report will be able to modify its request and start again the process.

2.3. Temporal planning

Now, it's time to analyze the timing of the project. In order to do that, we will use the computer program Microsoft Project. The final WBS is defined in the Annex B. There, we can see that our project will take 50 days, starting on mid-April of 2020, and ending on 1st July 2020. Despite that, the effort put in the project is equivalent to 124 days of work.

As we know, this is only the first phase of development of the Start-up, so, when the project is finished, the activity will not finish. Despite, we have defined an end of activity for the project, where we can analyze if the final result meets the needs of the project.

2.3.1. Identification of precedence

The project is divided into different quantifiable tasks. All those tasks are related, and they follow the same goal: deliver a project that meets the initial requirements. In order to do all the tasks effectively, we will need to do them following a specific order. We cannot test the software without developing it first.

In the annex A, there is a diagram explaining the order of all the tasks defined in our Work Breakdown Structure. This diagram explains, globally, all the tasks of the project and its dependencies, with their identifier and the duration of each activity.

2.3.2. Resources distribution and duration of activities

For assigning the necessary resources to each activity, we know that we only have a back-end engineer, a front-end engineer, and a DevOps engineer. According to that, we have to administrate the resources of each activity, according to the nature of the activity.

In our case, this administration has been done manually on a Microsoft Project file, so we have a better control on the resource's distribution.

The duration of each activity depends on the effort and the amount of resources we put in each one. Explaining this with an example, on the task 1.4.1. Analysis of functional needs, we estimate that the effort will be of 3 days of 8 hours each one ($8 \times 3 = 24$ hours). But there are two workers that will carry with this task on a full-time basis (Developer Manager and Developer 1). So, the duration of the task is $3/2 = 1,5$ days.

It is possible to check the Project file on the annex B, where it is displayed the whole WBS, as well as the estimated effort of each activity ("Trabajo"), the duration of each activity ("Duración"), the start and end date, the resources that will be used in every task, and the precedent activities.

2.3.3. Calendar

After configuring every task of the WBS manually, we can check valuable data in a Gantt diagram, regarding calendar and timing of the project.

Using the Gantt diagram tool, we can check the temporal distribution of each task, as well as the assigned resources of each activity.

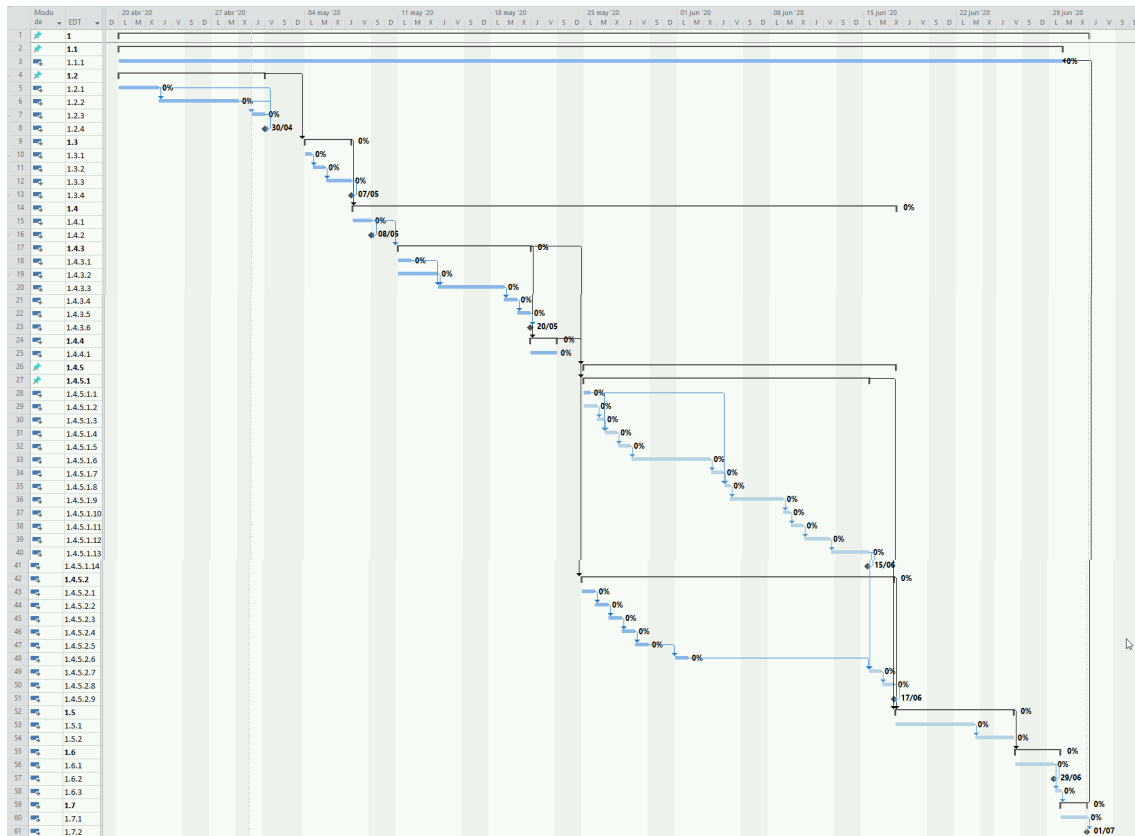


Figure 2. Gantt diagram of the project

2.4. Economic planning

2.4.1. Estimation of individual costs of activities

Next on, we can see the individual cost that can be assigned to each activity. These costs had been calculated using our Microsoft Project file.

1. Project “First Phase of Onerz Development”

1.1. Project Management.

- 1.1.1. Supervision and management of the project.
 - Wages: 3.072,00€

1.2. Market analysis.

- 1.2.1. Identification of the needs of our clients.
 - Wages: 192,00€
- 1.2.2. Analysis of technological solutions.
 - Wages: 256,00€
- 1.2.3. Research and analysis of similar cases and potential competitors.
 - Wages: 64,00€

1.3. Acquisitions.

- 1.3.1. Analysis of material needs.
 - Wages: 32,00€
- 1.3.2. Analysis of required licenses and software.
 - Wages: 64,00€
- 1.3.3. Acquisition of needed resources.
 - Wages: 128,00€
 - Computers acquisition: 2.745,00€
 - Google Suite year license acquisition: 277,20€
 - Scaleway (cloud server) year license acquisition: 26,14€
 - Docsumo (OCR) license acquisition: 99,96€
 - Integromat year license: 92,52€
 - Typeform year license: 607,44€
 - Kantree year license: 420€

1.4. Software development.

- 1.4.1. Analysis of functional needs.
 - Wages: 165,00€
- 1.4.2. Development of the CRM
 - 1.4.2.1. Creation of tables.
 - Wages: 46,00€
 - 1.4.2.2. Definition of functions.
 - Wages: 330,00€
 - 1.4.2.3. Coding of functions.
 - Wages: 276,00€
 - 1.4.2.4. Tests.
 - Wages: 110,00€
 - 1.4.2.5. Connection with sources of information.
 - Wages: 46,00€
- 1.4.3. Web app design.



1.4.3.1. Web app sketch (front-end).

- Wages: 220,00€

1.4.4. Web app development.

1.4.4.1. Back-end development.

- 1.4.4.1.1. Creation of Docker containers in a server
 - Wages: 23,00€
- 1.4.4.1.2. Definition of models and entities.
 - Wages: 46,00€
- 1.4.4.1.3. Definition of databases context.
 - Wages: 23,00€
- 1.4.4.1.4. Migration to databases in a server.
 - Wages: 46,00€
- 1.4.4.1.5. Population of databases with test data.
 - Wages: 46,00€
- 1.4.4.1.6. Definition of services.
 - Wages: 184,00€
- 1.4.4.1.7. Definition of API controllers.
 - Wages: 46,00€
- 1.4.4.1.8. Upload API image into development container.
 - Wages: 23,00€
- 1.4.4.1.9. Tests with server in development.
 - Wages: 92,00€
- 1.4.4.1.10. Upload API image into production container.
 - Wages: 23,00€
- 1.4.4.1.11. Tests with server in production.
 - Wages: 46,00€
- 1.4.4.1.12. Population of databases with real data.
 - Wages: 92,00€
- 1.4.4.1.13. Elaboration of a manual.
 - Wages: 46,00€

1.4.4.2. Front-end development.

- 1.4.4.2.1. Authentication page
 - Wages: 46,00€
- 1.4.4.2.2. Index page
 - Wages: 46,00€
- 1.4.4.2.3. Emissions page
 - Wages: 46,00€
- 1.4.4.2.4. Expenses page
 - Wages: 46,00€
- 1.4.4.2.5. Consumption page
 - Wages: 46,00€
- 1.4.4.2.6. Savings page
 - Wages: 46,00€
- 1.4.4.2.7. Interconnection back-front end.
 - Wages: 92,00€
- 1.4.4.2.8. Upload to development



- Wages: 46,00€

1.5. Final tests

1.5.1. Testing in development

- Wages: 368,00€

1.5.2. Testing in production

- Wages: 184,00€

1.6. Project publication

1.6.1. Publication of the final web application (front-end production).

- Wages: 46,00€

1.6.2. Communication to clients.

- Wages: 32,00€

1.7. End of project

1.7.1. Closing document.

- Wages: 128,00€

2.4.2. Elaboration of a budget

In the next table, we calculate the necessary amount of money for developing our project and maintain it working for a year. This includes different concepts, like wages or tangible/intangible acquisitions.

Resources	Cost (€)
Wages	6.909,00 €
Project Manager	3.200,00 €
Development Manager	1.248,00 €
Developer 1	1.334,00 €
Developer 2	345,00 €
Developer 3	782,00 €
Material	2.745,00 €
Computers	2.745,00 €
Licenses	1.523,26 €
G Suite	277,20 €
Cloud Server	26,14 €
OCR	99,96 €
Integromat	92,52 €
Typeform	607,44 €
Kantree	420,00 €
TOTAL AMOUNT	11.177,26 €

Table 6. Project budget

2.5. Uncertainty analysis

In order to be able to evaluate the conditions related to the project, first, we need to make a global analysis of the project, macroeconomically and micro economically. With that information in mind, we can identify precisely the risks and opportunities of the project.

2.5.1. Environmental analysis

To analyze the macroenvironment, we can make a PEST analysis, while for analyzing the microenvironment, we should make a Porter's five forces analysis.

2.5.1.1. PESTEL

PESTEL analysis involves the study of Political, Economic, Sociocultural, Technological, Ecological and Legal factors. A PESTEL analysis for the company would be as follows:

EXTERNAL FACTORS	OPPORTUNITIES	THREATS
Political-Legal	Political proposals encouraging sustainability Favourable regulations towards green companies	Negacionism towards climatic change COVID-19
Economical	Economic helps to clean companies	Economic crisis due to COVID-19
Sociocultural	Evolution towards cleaner society Increase of committed customers with environment	Lack of recognizable brand (trust) Low number of committed customers with environment
Technological	Improvement of processes through ICT High use of phone apps in society	Informatic attacks Continuous change & obsolescence
Ecological	Great interest in avoiding climatic change in society Forecasts of positive legislation for sustainability	Could be late for avoiding climatic change Generating more harm than benefit with our actions

Table 7. PESTEL analysis

2.5.1.2. Porter's five forces

To analyze the microenvironment, is more interesting to use the Porter's five forces, because it allows us to deep into the study object, in this case, providing electricity services.

- Negotiation power with clients: this is referred as the capacity of the client to put under pressure the company. As it stands, there are companies that offer similar services. So, the power of the clients is high.
- Providers negotiation power: regarding the electricity service, providers offer the service to our clients. So, they are important in the chain, but they are also easily replaceable. So, we could say they have a medium power.
- New incoming competitors' threat: sustainability is a trending topic, and electricity is one of the markets where this topic can work. So, it is expected that new companies appear offering the same services, as comparers or as intermediaries, like in our case.
- Substitutive product threat: regarding electricity service, could be self-consumption installs and services. It's not a big problem because the company also offers this as a different service.



- Competitors rivalry: there are some competitors that provide similar services and is expected to increase because of the sustainable and ecologic trends in our society.

2.5.2. Internal analysis: SWOT Analysis

After studying the environment where we work, it's time to analyze the strengths, weaknesses, opportunities and threats our company has. This will help us to get in touch with the qualities of the project, but also to identify the weaknesses, so we know where we can improve:

Weaknesses	Threats
No experience	Economic recession (COVID-19)
Limited resources	Competitors
No margin for mistakes	Negacionism towards climatic change
Lack of famous brand	Low commitment with environment from society
Objective public is not too big	Informatic attacks
Strengths	Opportunities
Processes through ICT	Sustainability trend
Sustainable services	Favorable regulations
Benefits for consumer	Increase of commitment from society
Investors supporting project	High use of phone apps

Table 8. SWOT matrix [22]

2.5.3. Risk and opportunities treatment

Once we have all this data, we are able to establish measures for treating these risks and opportunities:

- Risks:
 - Economic recession: this is the biggest risk for the company due to the situation. We have to be aware of the economic evolution, as well as political, thinking about the measures that governments implement. Also, in these times, people save more money than in economic expansion times, so economic goals have to be lowered, thinking more on surviving the recession rather than having big benefits immediately.
 - Experience: the company is mainly formed by students, so there is a lack of working experience. Learning as fast as possible, as well as being open to receive feedback from coworkers are good ideas to face this risk. Also, in new borne startups, everyone gets involved in a big number of topics, so the learning is about many topics, instead of specialization.
 - Lack of resources: as a new company, incomes and resources are low. But, the company has also investors and is offering to the company resources from their own, so this may be fixed.
 - Lack of famous brand: when talking about sustainability, people tend to follow gurus or leading brands, due to the trust they have. The company is new, and still has a long path to run for reaching this level of trust. So, it will be important to do things properly, thinking on the consumer's satisfaction.



- Small group of objective public: unfortunately, there is not a big group of people committed to sustainability.
- Competitors: there are competitors, and new companies providing the same service may appear. In order to do that, the company should be distinguished with a perfect service for clients.
- Leaves: due to it's a small company, any leave, due to sickness, vacation or leave of the position, hurts the company. So, the boss should treat well the employees, trying to retain them as much as possible, while they work properly according to the standards of the company.
- Opportunities:
 - Sustainability trend: due to the climatic change, people are more aware and committed to sustainable ways of life, and it's a trend that is forecasted to be increasing. The company pretends to flag sustainability ways of life by its services. For making people aware of this, marketing and advertising is key.
 - Favorable legislation: due to climatic change, legislation is changing, in order to ease the change into a more sustainable way of living. Green companies like ours should try to take advantage of this.
 - High use of phone apps: society uses smartphones every day for everything. Our company, as IT company, should try to take advantage of this.

2.6. Closing of the project

2.6.1. Conclusions

The company is in an early stage in the moment of the project and needs to develop itself before trying to reach high standards. A key action, in order to save time in these early stages, as well as winning internal organization, is to create a tool for company workers, in order to manage efficiently and quickly new leads before offering them new services, until a more specialized tool like Odoo can be developed for them.

Apart from this, one of the values of the company is transparency. Our relations with clients must be transparent. It's one of the key values of the company and provides an added value to our services. In order to do that, it is necessary to develop a tool like the proposed in this project, so clients have all their data regarding their services easily accessible.

These are projects that add value to the company and their services, despite they might not increase the number of customers directly. Its success will be analyzed according to the time needed for treat every lead, and the use of the personal area done by clients.

2.6.2. Future lines of action

In this part of the document, we describe the first part of the company technological development. But new stages will come after, like extending the company offers to other services, like gas, water, diet or solar panels installs. For that, we will need to



develop new processes, that may be supported in new CRM specially done for those services.

Meanwhile, with growth of the number of new services, the personal area must be adapted to those, so all clients can access to their data. During this process, code could be refactored, to make it more efficient, clean and easy to understand.

2.6.3. Learned lessons

Finally, it's time to extract some learned lessons from planning this project. These lessons are related to time estimations, distribution of resources, AGILE methodologies, establishing realistic checkpoints or managing human resources and conflict resolution.

Also, we have done an estimation of the needed materials for the project, as well as the costs of the project, in order to carry it out.

Finally, some concepts that we consider key in the development of any project are the following:

- Planning is a very important stage in any project. An unrealistic planning will lead to not reaching the established checkpoints and will risk the development of a project.
- Human resources are the ones that carry out the development of a project. It's very important to make a good planning of them, as well as guarantee their wellness.
- It's also important to make an environment study, before taking any decision regarding the project.
- It's important to define adequate protocols before asking for changes in the project.
- Finally, the project manager has to do a close monitoring and control of the development of the project if success is intended, being comprehensive as well with the employees.



Chapter 3. Customer Relationship Management Software Development

3.1. Definition of minimum requirements

In this part of the project, we are going to talk about the development of the first company's CRM. This software will be an important tool for qualifying and organizing all the leads related to the electric service.

This CRM will be a helpful tool for studying, organizing, qualifying and managing a lead or client. It will provide:

- A list of leads.
- Studies about the company's services that could be interesting for both company and client. These studies should be done automatically and sent to the lead once it has been accepted by the company. As input for making these calculations, the lead will provide us an electric bill. They should calculate:
 - Power prices with the company's provider.
 - Consumption prices with the company's provider.
 - Average yearly cost of the customer regarding this service with their actual provider.
 - A proposal with:
 - House power.
 - Price of a yearly subscription with the company's service.
 - Economic savings.
 - Emissions.
- A list of final proposals.
- A list of clients of this service.
- Information related to finances, like:
 - Subscriptions of each client related to the service.
 - Payments done by each client.
 - Revenues
- Information regarding our service, like:
 - Periods of time of each bill.
 - Monthly electric consumptions.

This CRM will be designed in a Google Sheets spreadsheet, including different pages for each CRM's service, like Calculator, Clients, Proposals, etc. These will be discussed from the point 3.4 until the point 3.11.

3.2. Lead creation proposed workflow

In order to treat each lead, we propose the following workflow:

- First, the interested person will enter in a form, where the person will introduce some required information about the service. The required information will be:
 - Name



- Email
- Address of the place we have to analyze.
- Electric bill (file).
- Acceptation of the privacy policy.
- Phone number.
- Time of submission.
- Token of the form.
- Once the interested person has introduced its data, this data will be written in a spreadsheet. Also, the bill file will be sent to our OCR, where it will be analyzed, and the interesting information will be returned to the spreadsheet. The data to retrieve from the OCR will be:
 - Details about the studied place:
 - ZIP Code.
 - City.
 - Province.
 - Details about the incumbent:
 - Name.
 - ID number.
 - CUPS (identifier for an electric or gas installation) [23].
 - Details about the electric consumption:
 - Electric toll.
 - Hired power (peak, valley, super valley).
 - Company.
 - Is a regulated tariff?
 - Month consumption (peak, valley, super valley).
 - Prices for all kind of powers and consumption installed.
 - Extra charges.
 - Start and end date of the bill.
- When the relevant information has already been placed in the spreadsheet, the relevant parameters of the study will be calculated automatically. Also, the lead will be qualified.
- After an employee of the company has validated the study, an email will be sent to the interested person, detailing the study made.
- From this point, all the changes in the state of a lead/client will be done manually by the company employees, due to signing in process is still manual in this stage of development of the company.

3.3. Development of the lead creation process

The first step in our development will be develop the information input process for being able of making electric studies for our potential clients. For that, we need a form page. Due to the quick needs of development, developing a whole web application in order to get this information is not possible.

So, our best option is to use a form page creator tool, so we can get this page in minutes. In our case, we used the tool Typeform [24].



Typeform is a powerful tool for creating forms, in order to receive information from users. In there, developers can configure questions based on the type of expected responses, introducing validators for those answers, as well as some extra details, like pictures, videos or text. Another helpful feature of Typeform is the possibility of adding hidden values and logics to the defined questions.

In our case, we will use those hidden values in the case our lead is already known by us, and comes from a known media, like it could be Facebook or an email campaign. So, in the case we already know the name or email of the customer, we will skip those questions. In this form, the hidden values are related to the name and email of the person.

Id	Question	Type of answer	Logic		
			If	Do	
1	Name	Short text	Hidden Value Email	Contains @	Jump to 3
			Rest of cases		Jump to 2
2	Email	Email address	All cases		Jump to 3
3	Address	Long text	All cases		Jump to 4
4	Bill Address 1	File	All cases		Jump to 5
5	Bill Address 2	File	Answer null is possible		Jump to 6
6	Phone	Phone Number	All cases		Jump to 7
7	Privacy Policy	Boolean	All cases		Jump to 8
8	Final screen	No answer			

Table 9. Typeform questionnaire

In the annex C, there are available some pictures regarding how to fill the designed questionnaire.

3.4. Development of an energy consumption calculator

Once we receive the information from a lead, we need to make a study about their electricity consumption. In order to make it straightforward, it would be interesting to automatize it as much as possible.

That's why we need to perform a tool that can help workers of the company in this task. Because of that, we developed a spreadsheet that works as a calculator.

This calculator will store the information given by the user through the previously filled form. Then, an automation will be triggered, where the stored file (electricity bill) will be sent to our OCR, called Docsumo [25]. These automations are made thanks to a software called Integromat [26].

The OCR will read the data from our file, and through Integromat, will be sent back to the spreadsheet, and stored in some reserved cells. According to the values of these cells, the different designed formulas will be triggered, providing the results to the staff of the company. They will check that data is accurate before sending to the lead an email with the made study.

In the annex D, there's an example electricity bill, that will be used to test the whole software.

3.4.1. Integromat

Integromat is an online automation software. It allows automatize different processes from different services into one automation, without introducing any code, by using those services APIs. It's a very helpful tool for creating/automatizing processes easily and quickly.

For our development, we have hired a Basic Subscription, that allows us to manage 10.000 operations and transfer 1GB of data per month [16] per \$9 (already considered in our budget, 2.4.2).

In this part of the software development, we created three different scenarios, for fulfilling all the needs of the electricity studies.

3.4.1.1. Scenario A. Files saving and processing

This first scenario includes the saving of the data introduced through Typeform by the user, and the processing of these data. Here we can see the flowchart.

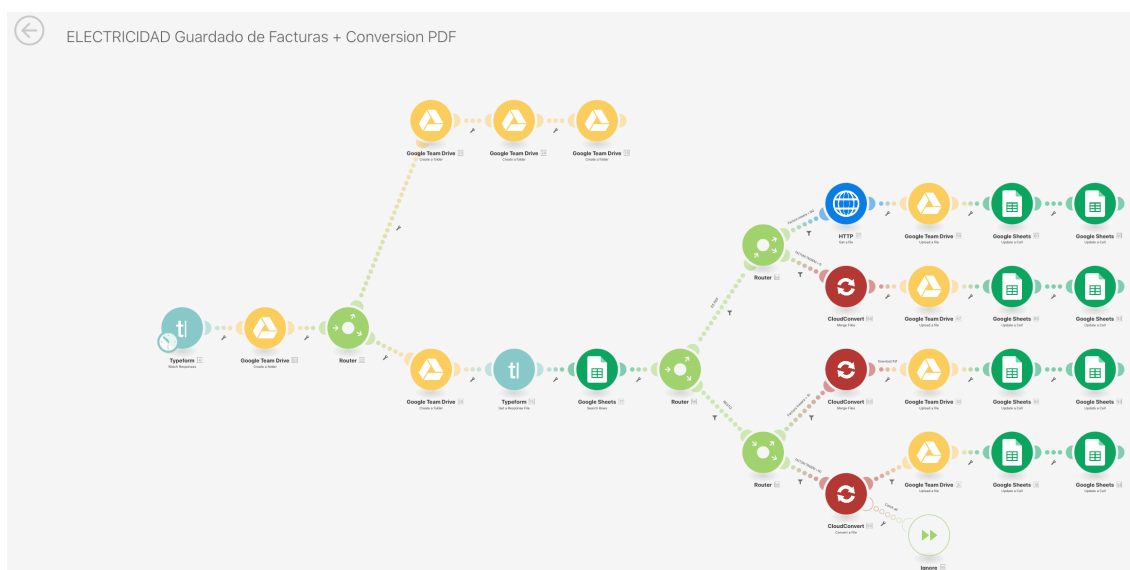


Figure 3. Flowchart of scenario A.

As we can see, we make calls to the APIs of Typeform, Google Team Drive, Google Sheets and CloudConvert.

The scenario starts when we receive a new filled form through Typeform. Then, we create a new folder in our Google Team Drive account, named with the day of the study (when the lead fills the form), name of the lead and street where we make the study. Once this folder has been created, we start two independent processes.



The first (upper branch) one creates more folders inside the one we already created, one for saving contracts, another one for saving Bills, and the last one, for saving important files related to managing the user services. These folders will be useful if the lead turns into a client.

Meanwhile, another folder is created, for saving all the data related to the electricity study. Then, we retrieve the link where the first file has been saved (normally in a private Typeform bucket storage). Then, we search the Google Sheets row (in the calculator) where we have to save the link.

Then, it's time to process the file. The process can be summed up into that we need to convert every kind of file into PDF format (by using CloudConvert) and upload the resulting file to the Studies folder of Google Drive. But, previously, we have to consider two aspects of the uploaded information, that are if the lead uploaded one or two files, and if those files are already PDFs or another kind of file. So, we have 4 different cases, with its corresponding branch in the scenario:

- Branch 1: only one file has been uploaded and is a PDF. In this case, we just download the file, upload it to Google Drive, and save the link in the calculator spreadsheet.
- Branch 2: two files had been uploaded and both are PDF. In this case, we have to merge both files into one with PDF format (thanks to CloudConvert), upload the resulting file to Google Drive, and save the access link of the resulting file in the calculator spreadsheet.
- Branch 3: two files had been uploaded and, at least one, is not a PDF. The workflow is similar to the branch 2, but has been separated from the branch 2 due to configuration parameters in the call made to the CloudConvert API.
- Branch 4: only one file has been uploaded and is not a PDF. In this case, we have to turn it into a PDF (through CloudConvert), upload the resulting file to Google Drive, and save the access link of the resulting file in the calculator spreadsheet.

In case the lead uploads files with a format we are not ready to process, in order to avoid the process to stop, we have added an ignore function, that only will be triggered in this case.

3.4.1.2. Scenario B. Docsumo uploading

This second scenario consists on uploading the file we obtained in the scenario A to Docsumo. Due to Docsumo needs some manual approval before the AI learns how to read the file accurately, the data retrieving process has been divided into two processes (this scenario and scenario C). We can see the workflow in the next diagram.

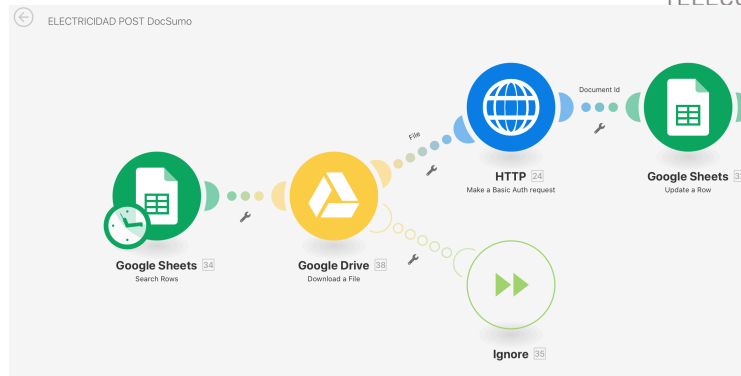


Figure 4. Flowchart of scenario B.

This scenario is triggered every 60 minutes. In it, we search the rows where the file hasn't been uploaded yet, download that file from our Google Drive account (through the link we have saved previously in that row), and through an HTTP request, we upload it to Docsumo. Finally, Docsumo will return as an answer of the HTTP call an ID for the file, that we store in the row of the spreadsheet the file is.

In order to identify which files to upload to Docsumo, Integromat checks which rows have a link to a PDF file stored in our Google Drive (all those links are saved in a specified column of our spreadsheet), as well as if the Docsumo File ID has been saved (if it hasn't been stored yet means the file hasn't been uploaded yet).

In order to make the HTTP call to Docsumo API, we configured it as a POST method, with a private API key sent in the parameters (authentication), and a multipart/form-data body, sending the file (as a file) and the filter we want to use in Docsumo (to see later).

3.4.1.3. Scenario C. Docsumo downloading

Once the file has already been approved, we need to retrieve all that data, and store it in our spreadsheet. For that, we follow the next workflow.

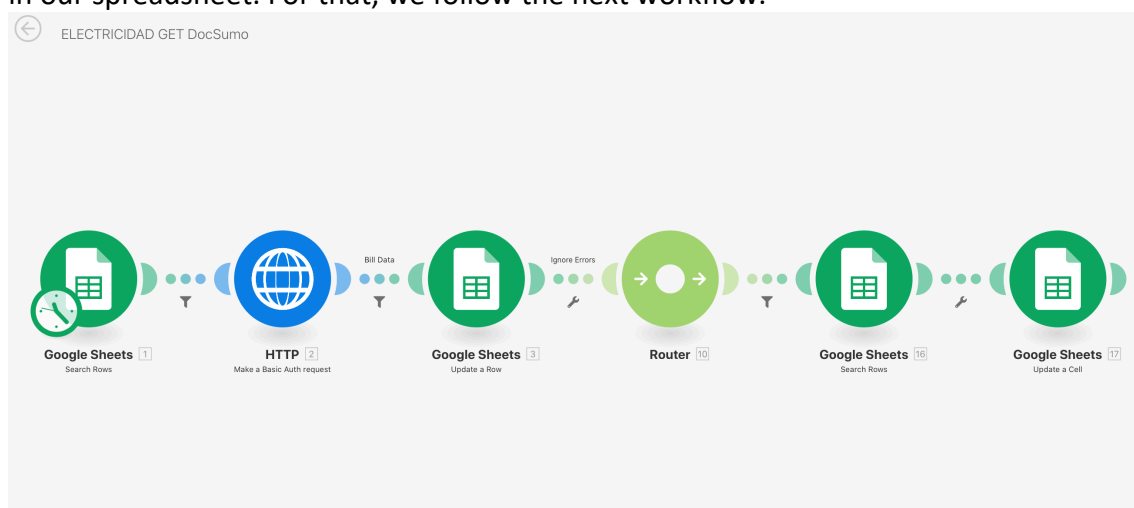


Figure 5. Flowchart of scenario B.

This scenario is also triggered every 60 minutes. If a row in the spreadsheet has a Docsumo File ID stored, but the cells reserved for Docsumo are empty, the process is

not stopped. Then, we make a HTTP call to Docsumo's API, where we retrieve the information from the sent file in a JSON file. Integromat is able to process information that comes in a JSON, and classify it into fields, so we can fill the Docsumo cells in our spreadsheet with this information. In the end, we overwrite the address of the study, in order to have more accurate information.

3.4.2. Docsumo

Docsumo is an AI software with an intelligent OCR technology, that helps users to convert unstructured documents like invoices and bank statements to actionable data [25]. Compared to others OCR software, Docsumo allows to access to a Rest API, with easy-to-understand documentation (including a Postman library), from which we can retrieve all the data analyzed by the software, all this with a relatively low price per document. Also, allows user to develop their own document formats. This feature allows to adapt all the bills into a precise filter, so results are more accurate.

In order to use Docsumo, we need to pay a fee as exchange, even though a free trial is included. There are three available pricing plans:

- Blocks of 100 credits (one credit is equivalent to \$1, every managed file is equivalent to 0,2 credits).
- Up to 10.000 documents per month.
- More than 10.000 documents per month [15].

We pretend to use Docsumo to other services in the future, but because we are starting our services and the volume of files to analyze is not too big, we prefer to launch our service with blocks of 100 credits, meaning we can analyze up to 500 documents plus the free trial, giving us the chance of analyzing 1.000 documents during the first year. All this data has been considered previously in our budget (2.4.2).

In our case, we developed a generic filter, able to process the main features of every bill in Spain. This filter checks the following values:

- Owner - string
- National ID - string
- Address - string
- ZIP Code - string
- City – string
- Province - string
- CUPS - string
- Peak power – double
- Valley power - double
- Supervalley power - double
- Peak power price - double
- Valley power price - double
- Supervalley power price - double
- Power discount - double
- Peak consumption - double

- Valley consumption - double
- Supervalley consumption - double
- Cost of service - double
- Company - string
- Period start date - date
- Period end date - date
- Amount - double
- PVPC – boolean (in case the bill is a PVPC tariff, true, opposite false)

Once the file has been analyzed and approved, when making a GET API call, a JSON file with all this information will be returned.

3.4.3. Columns

Once our OCR has analyzed the data, our integrations will fill all the reserved for Docsumo spaces in our spreadsheet. When we have this information filled, our defined formulas will get all the data necessary for making the study.

Those studies are made estimating the data for a period of one year, using the data from the period of data (generally a month) from the bill given by the lead.

But, it's important to mention that our proposals are limited to only one provider. In case we cannot offer a better offer with our pricing, the system will detect it, and push a review signal. Mostly, when this happens, means that the only profitable change for the user is changing his electricity service to a PVPC pricing.

In the annex E, we can find a table with an explanation of how to get each value for every column.

The created spreadsheet can be accessed through the following QR Code.



Figure 6. QR Code to access the spreadsheet

3.5. Clients spreadsheet

The client's spreadsheet will be developed in a new page inside the CRM's spreadsheet, with the name "3_Clientes".

The functions of this page are:

- Keeping information regarding the lead. We should keep information like identification data, address, owner data, banking, etc.



- Keeping information about the state of the lead. We should qualify the lead according to the processes he has completed. The important dates are:
 - Registry date.
 - Bill reception date.
 - Sent proposal date.
 - Second proposal date (in case the first proposal has been rejected).
 - Call date (in case it's necessary).
 - Subscription date.
 - Intranet subscription date.
 - Removal date.

In order to qualify a lead, we defined 7 different states. This qualification will be defined in the first column of the client's and proposal's pages.

The meaning of these states are:

- Number 0: we still need to receive a bill to make the study.
 - Lead that arrived through third channels
- Number 1: Pending study
- Number 2: Pending to accept a proposal (made offer)
- Number 3: Client
- Number 4: Recoverable client (rejected the first offer, but a second offer may be made)
- Number 5: not interesting lead
- Number 6: lost client

The state of a lead is determining, according to a registry of dates with each event, available on the same spreadsheet.

The spreadsheet can be accessed through the Figure 6 QR Code. Also, as we discussed previously, a brief description of all the columns developed in this page is included in Annex E.

3.6. Proposal spreadsheet

The proposal spreadsheet will be developed in a new page inside the CRM's spreadsheet, with the name "4_Propuesta".

This part of the spreadsheet is designed as a summary of the made studies. In it, we can see data regarding those studies, like the previous yearly cost, the yearly cost of hiring our services for the client, and the savings data; apart from the commission the company would receive from the operation.

In order to modify some aspects of the studies, these should be made in the calculator spreadsheet. Also, the emails coworkers will send with the studies to the leads will use data from this spreadsheet.

As we already commented previously, we can find an explanation of each column's meaning in the annex E.

3.7. Fee spreadsheet

The fees spreadsheet will be developed in a new page inside the CRM's spreadsheet, with the name "5_Cuotas".

This spreadsheet notes the fees the company charges to customers monthly. It's divided into months and covers from the arrival of the first client (September 2019) until the end of 2021. Also, a summation per year is made.

All this data should be introduced manually by a coworker (except names and yearly summations, done by the spreadsheet), once the client has been subscribed.

3.8. Bills spreadsheets

In this part of the CRM, three spreadsheets should be included.

3.8.1. Payments

The payments spreadsheet will be developed in a new page inside the CRM's spreadsheet, with the name "6_Pagos".

Following the structure of the fee spreadsheet, this spreadsheet includes the cost of each received bill from a client. These values have to be written manually by a coworker, month per month.

3.8.2. Consumptions

The consumptions spreadsheet will be developed in a new page inside the CRM's spreadsheet, with the name "6.1_Consumos".

This spreadsheet includes the electric consumptions per house and month, divided per sections (peak, valley, super valley), with a similar structure as the fee spreadsheet.

3.8.3. Periods

The periods spreadsheet will be developed in a new page inside the CRM's spreadsheet, with the name "6.2_Periodo".

This spreadsheet includes the start and end date of each received bill from each house. Follows a similar structure as the previous spreadsheets.

3.9. Benefits spreadsheet

The benefits spreadsheet will be developed in a new page inside the CRM's spreadsheet, with the name "8_Beneficio".

This spreadsheet includes the monthly benefit before taxes the company gets from each house per month. These calculations are made based on the data introduced in the Fees and Payments spreadsheets. Follows a similar structure as the previous spreadsheets.

3.10. Taxes spreadsheet

The taxes spreadsheet will be developed in a new page inside the CRM's spreadsheet, with the name "9_Impuestos".

This spreadsheet calculates the taxes the company has to pay because of every subscription the company sells. This calculation is done per months, has been automatized and uses data from the benefits spreadsheet. Follows a similar structure as the previous spreadsheets.

3.11. Aggregate spreadsheet

The aggregate spreadsheet will be developed in a new page inside the CRM's spreadsheet, with the name "10_Acumulado".

This spreadsheet calculates the benefits (after taxes) that the company gets per house and month. This calculation has been automatized and uses data from the benefits and taxes spreadsheet. Follows a similar structure as the previous spreadsheets.

3.12. Test

Now, we are going to test our whole spreadsheet by using the test bill we can find in the annex D.

After loading the lead data in the Typeform form, Integromat's scenario A will activate, loading data in our Calculator spreadsheet, like name, surname, address, email and a link to the file in Google Drive.

Later, the scenario B will activate, loading the file into Docsumo.

Once it's uploaded, the file will be available for a manual review. Later, we click on approve (review is not a 100% necessary step, but it's recommendable, specially in the first tries of the software). In the next figure, we can see a screenshot of the review mode in Docsumo, where we can see that the software recognizes text.

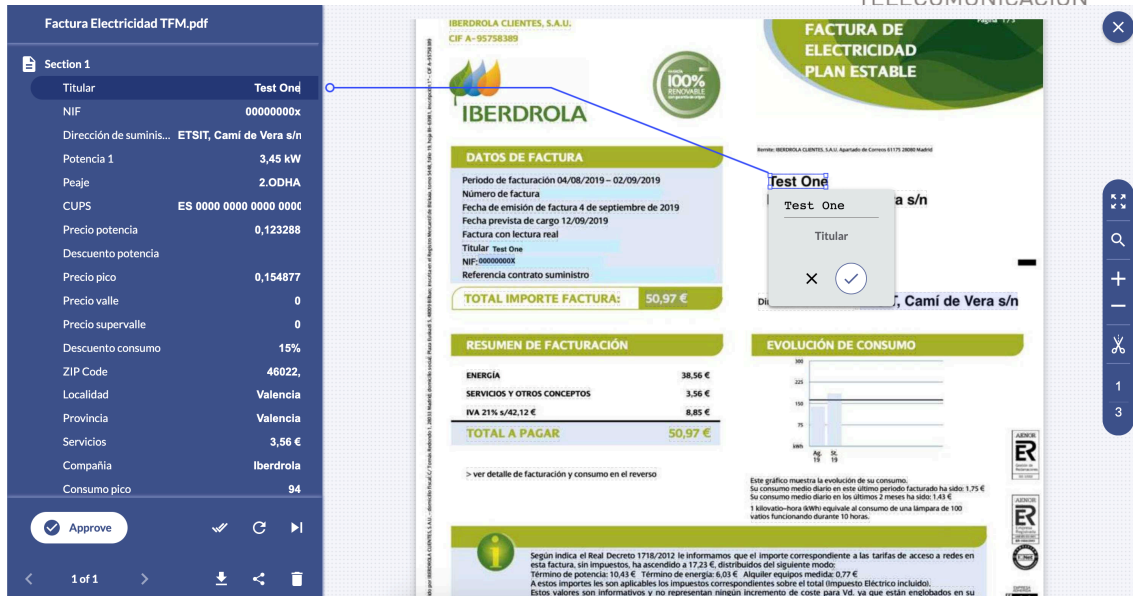


Figure 7. Docsumo's review mode

When the scenario C is activated, all this information will be retrieved in a JSON (available on annex F), and copied in our calculator spreadsheet, making the study ready to review. In this case, due to all the information can be seen clearly in the file, the retrieved data is accurate, so the automatic version of the study is accepted.

Meanwhile, the other designed spreadsheets also apply their own formulas. In the client's spreadsheet, we have to mark the current date in the study column, so the lead state changes from 1 to 2.

In the proposal spreadsheet, the values are searched and copied, so the data is shown clearly. Also, from this data, a coworker can communicate the data to the client.

Then, in order to continue the workflow, let's suppose the leads is turned into a client, so we note the subscription date in its client's column. Then, the code changes to 3, meaning it's a registered client. So, the client starts appearing on the quotes, payments, consumptions, periods, benefits and taxes spreadsheets.

Chapter 4. Back-end development of the project

4.1. Definition of minimum requirements for first version

In this part of the project, we are going to explain the development of the back-end side of our client's private area web application. This side will:

- Receive and treat requests from the front-end side.
- Search information through databases.
- Make calculations.
- Send information to the front-end side of the web application.

The main goal of the web application is to show to clients, in a comprehensible way, relevant information regarding their electricity bills like consumptions, emissions, money spent, savings hiring our services or bills. In order to do that, we define the following requirements:

- Petitions will require authentication.
- Some clients will be able to access to other people's accounts, because they manage those accounts, even though they are not the owners.
- The area will be ready to implement new services in the future.
- The area must allow users to access their bills and information related to them.
- A request for changing passwords will be created. This will be used only by the company.

4.2. Definition of the product areas

We have defined the following flowchart for our client area, marked in green the parts planned for this project.

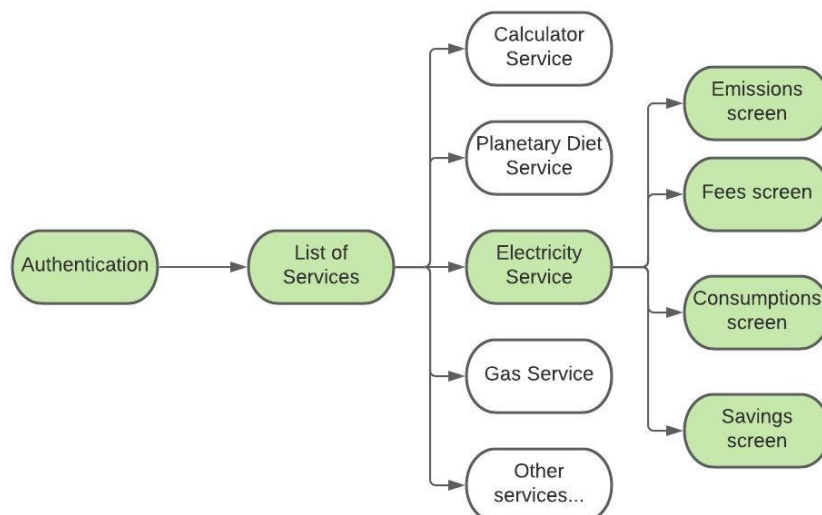


Figure 8. Client area workflow

4.2.1. Identification area

This area will allow users to identify themselves, so they can access to their private area. It will consist on an authentication petition, where the email or the phone number, and a password will be sent. In case the credentials match, a bearer token will be sent back, following the JSON Web Token standard. With this token, the user will be able to do petitions during the following 24 hours.

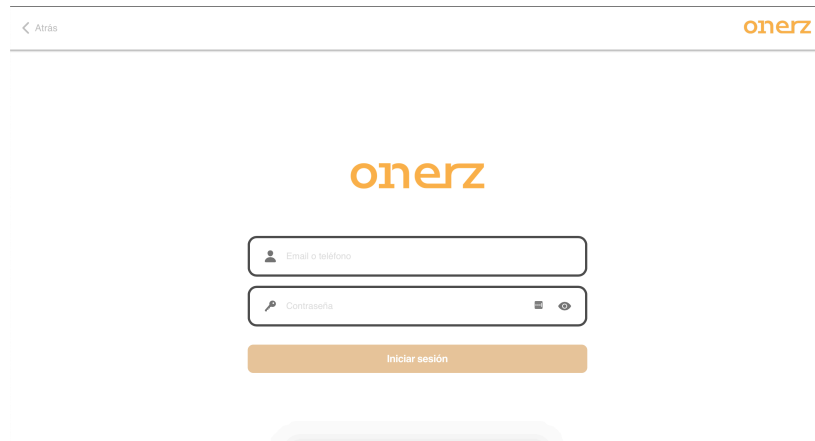


Figure 9. Authentication webpage

4.2.2. Services area

This area will allow users to get their available services. Requests regarding this area are necessary to access all private information, because it will provide some necessary identification numbers for accessing other information.

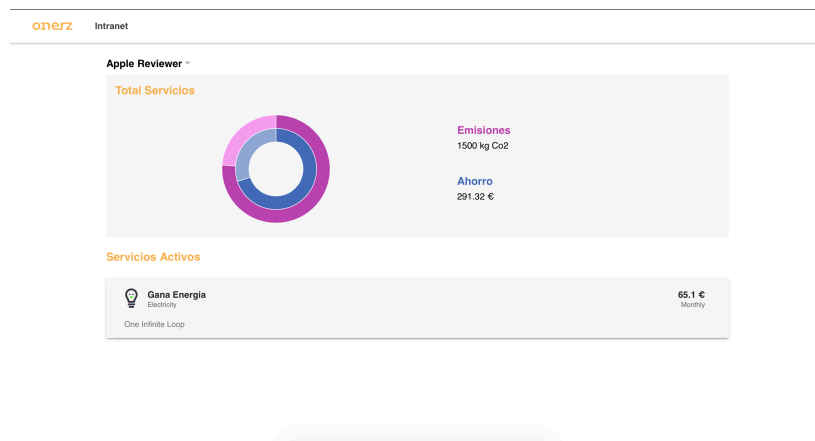


Figure 10. Services area webpage

4.2.3. Main menu area

In this area, the emissions, fees, consumptions and savings areas are accessible. This area is not relevant for the back-end.

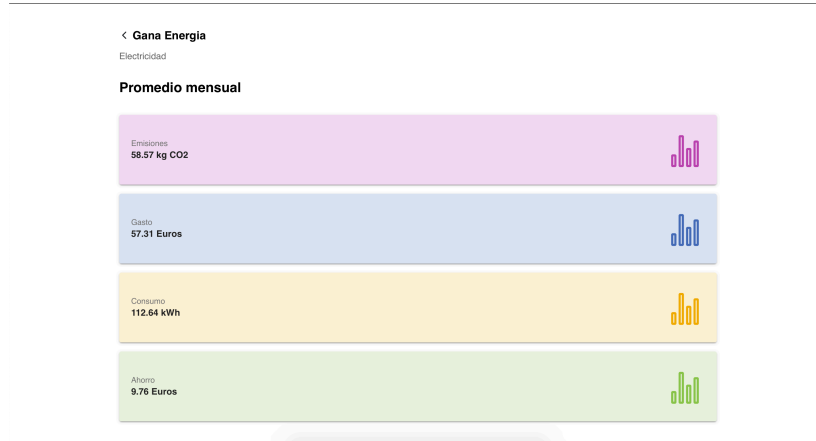


Figure 11. Main menu webpage

4.2.4. Emissions area

This area shows the estimated emissions from the user per month.

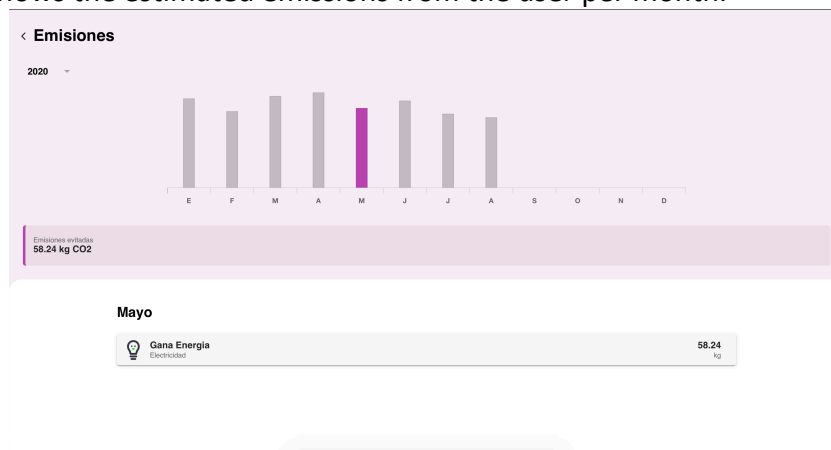


Figure 12. Emissions area webpage

4.2.5. Fees area

This area shows two fees:

- The fee paid from the user to our company (line chart).
- The fee paid from the company to the electricity provider (bar chart).

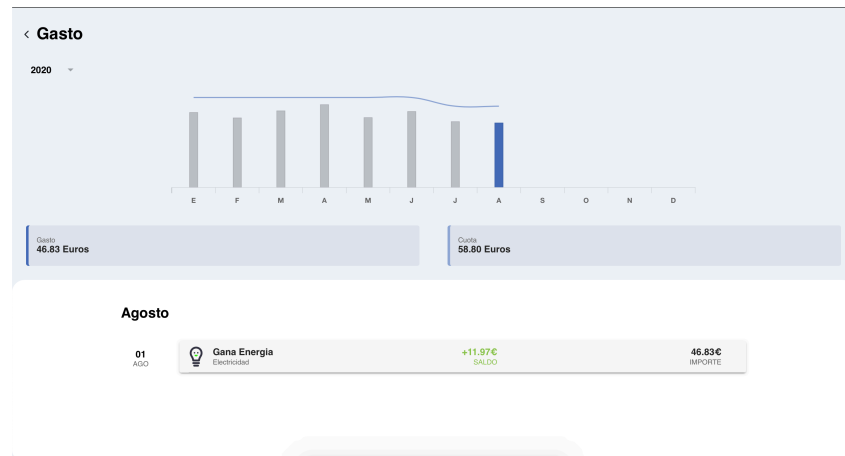


Figure 13. Fees area webpage

Also, by clicking on the card, we are able to access to the related electricity bill, using Dropbox services.

4.2.6. Consumptions area

Through this area, users are able to see a bar chart showing:

- Their peak consumption.
- Their valley consumption.
- Their supervalley consumption (in case they have hired a fare with this service).

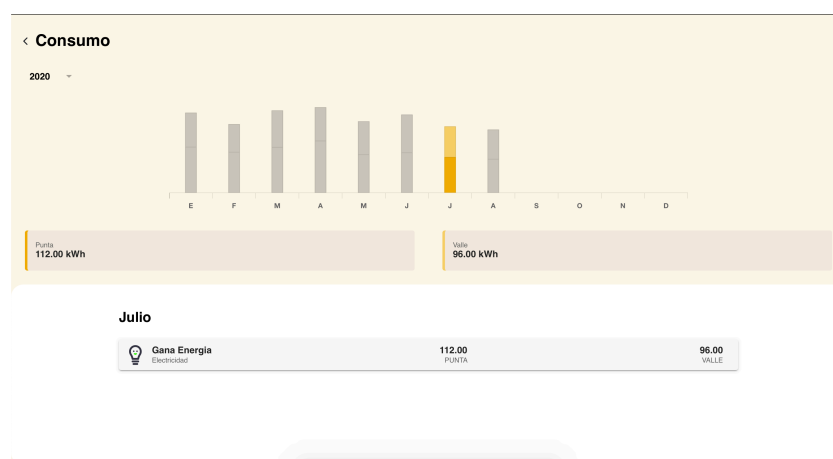


Figure 14. Consumptions area webpage

4.2.7. Savings area

This area shows an estimation of the amount of money (per month) the user is saving with our services, compared to the money he would be paying with his old company.

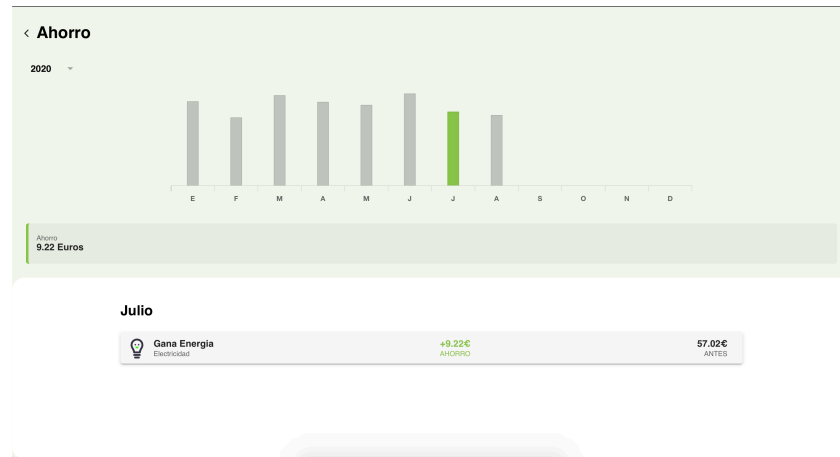


Figure 15. Savings area webpage

4.2.8. Communication between front-end and back-end area

One of the key points of the product will be the communication of the front and back-end sides. Both front and back-end sides will be developed as independent (but related) projects that will communicate (the front-end will send petitions, and the back-end will reply these petitions).

We have already mentioned that these petitions will need an authentication token for being accepted, that will come as a header of the petition. With the reply, a status code will be returned, giving information regarding the state of the reply.

- Status code 100-199: informative reply.
- Status code 200-299: satisfactory reply.
- Status code 300-399: redirect reply.
- Status code 400-499: Client error.
- Status code 500-599: Server error [27].

Also, in order to reply the petitions, a body will be sent (with relevant data), using an application/json content type value. A Json file is an open standard file format, that uses human-readable text, in order to store and transmit data objects that consist on attribute-value pairs and array data types [28].

4.2.9. Database area

In order to provide all the relevant information, first we will need to store it somewhere, so our back-end application can access to it. In order to do that, we will use relational databases.



These databases are organized in one or more tables (relations) with columns and rows, and a unique key identifying each row. Each table represents an entity, that can be related to a product or service [29].

In our case, we will use PostgreSQL as the manager of these tables. We will create a database for our development environment, and another database for our production environment. In these databases, some tables will be created, storing relevant information regarding:

- Authentications.
- Bills details.
- Companies.
- Consumptions.
- Old bills.
- Services.
- Users

4.3. Definition of the work environment

For developing the back-end side of our web-application, we are going to use Entity Framework Core 3.1, a free open-source managed computer software framework developed by Microsoft, based in C# [30].

In the next figure, we can see a schema of how an API with Entity Framework Core works. The client communicates with HTTP Requests with some endpoints defined in a controller. In the controller, sent/received data is mapped and serialized/deserialized using the defined data models.

This controller calls the services, that may request also an infrastructure (developed for communicate with APIs from 3rd parties). Usually, these services need to do some queries or data transfers with a database. For that, we need to use as an intermediary a DbContext.

All these parts of the API are working because are included in the Program and Startup files. These are the files that are running in the server.

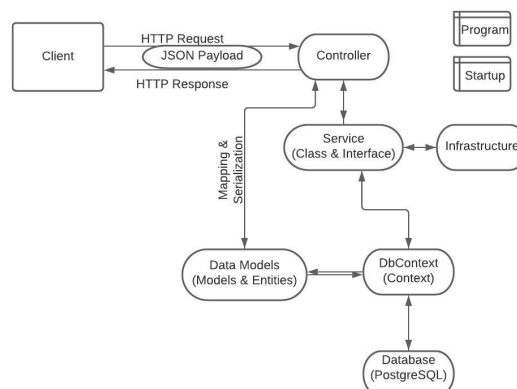


Figure 16. Entity Framework Core schema

This software, as we said before, will be connected to a relational database, managed by using PostgreSQL and it will also have connections with Integromat and our OCR (Docsumo), in order to automatize operations. This software will be deployed in Docker containers placed in a server [31], managed by Scaleway [14], and shared by the start-up and the launcher company.

In order to guarantee a proper development and performance of the software while not impacting the normal use of the web application by users, two environments will be deployed in our server, in separate containers.

- One environment will be called “dev” and will be used for the development environment of the software. This means that all the development and new functionalities will be first developed in this environment.
- Another environment will be called “prod” and will be used for the production environment of the software. This means that users will be able to access to this environment.

So, in order to guarantee this proper performance, the IT team will be working in the development environment, while users will be using the production environment. This last environment will be only modified when the new software has been tested and approved in the development environment.

Finally, all the developed code will be uploaded to a private repository using Git, so it's safely saved, and other coworkers can also work with it now or in the future.

In order to develop the software, we use the Code-First work mode, meaning that first we will develop our data models, and from them, we will generate our databases.

4.4. Definition of the startup file

The Startup class (named that way because of convention) configures services and works as the app's request pipeline. It optionally includes a Configure Services method to configure the app's services, that will be consumed across the app via dependency injection.

It also includes a Configure method to create the app's request processing pipeline.

This class is typically specified in the host builder, *WebHostBuilderExtensions.UseStartup<T>*, available on the Program class.

The Startup class only accepts the injection in its constructor of services related to the Host Environment or configuration, which are:

- IWebHostEnvironment
- IHostEnvironment
- IConfiguration

Services in the application are not available until the Configure method inside the Startup class is called [32].

In our case, we will configure in our startup class:

- In the ConfigureServices method:
 - Authentication configuration: it configures the authentication method, as well as the handling of tokens and signing keys and validators.
 - Swagger: helpful tools for developing RESTful APIs [33].
 - Cors policy: mechanism that allows the request of restricted resources in a webpage from a different domain than the one that served the first resource [34].
 - AutoMapper: a library that helps to model data [35].
 - Services
 - Authentication service: explained on 4.8.1.
 - AccountDetails service: explained on 4.8.2.
- In the Configure method:
 - Environment builder: parametrized according to the selected environment when executing the application. In our case, it also chooses an appsettings file (next point).
 - Swagger activation.
 - Routing activation
 - Cors policy activation
 - Authentication activation
 - Authorization activation
 - Endpoints: controllers mapping, explained in 4.9.

4.5. Definition of the appsettings file

The appsettings file is a JSON file used by the web application with the purpose of storing information related to the configuration of the web application, like the connection with the database or secret keys.

In this file, we collect all the information that we don't want to show directly in our code because of security reasons.

It is recommendable to configure an appsettings file per execution environment because some connection keys may change according to the environment. The system will know which file to use according to the execution environment and the environment builder we configured in our Startup class.

In our case, we create two appsettings file, called:

- Appsettings.Development.json
- Appsettings.Production.json

Both files will store similar information:



- Connection string with the database
 - Will be different between development and production, because we don't use the same database for both environments.
- A secret key used for our token generation and password generation (same in both environments).
- Connection with Integromat (same in both environments).

4.6. Definition of data model

Our first step in the development of this software will be defining the data model that we will be using. For that, we define Entities, that will be used in order to define and make transfers of information with our databases; and models, that will be used in order to manage data inside the software. We will define also a primary key, an attribute (column) that identifies our records.

In our software, we will define 8 different entities:

- Authentications: this entity will define data related to the authentication process.
- Users: this entity will define data related to the owner of the service.
- BillsDetails: this entity will define data related to each uploaded bill into the database.
- Consumptions: this entity will define data related to the consumption data of each uploaded bill. Depends on the BillsDetails entity.
- Houses: this entity will define data related to the house we are providing service.
- ServiceModel: this entity will define data related to the service we are providing to each house.
- OldBills: this entity will define data related to the information extracted from the bills we studied before the client hired our services.
- Companies: this entity will define data related to electricity companies, like emissions.

Authentication	
Userid	int
Phone	string
Email	string
Password	string

BillDetails	
Billid	int
Houseid	int
Consumptionid	int
Onerz	bool
BillTitle	string
FacturationStart	string
FacturationEnd	string
Amount	double
Difference	double
Co2Save	double
RadioSave	double
Link	string
CompanyName	string
Houses	House
Consumptions	Consumptions

Consumptions	
Consumptionid	int
Houseid	int
Billid	int
Days	double
MeasureDate	string
Power1	double
Power2	double
Power3	double
PeakConsumption	double
ValleyConsumption	double
SupervalleyConsumption	double
Co2Waste	double
RadioactiveWaste	double

Companies	
Companyid	int
Name	string
IkgCo2Kwh	double
mgRadioKwh	double

House	
Houseid	int
Address	string
ZipCode	string
City	string
Region	string
Country	string
oldBill	OldBill
Userid	int[]
Companyid	int

OldBill	
oldBillid	int
Houseid	int
PreviousPower1Price	double
PreviousPower2Price	double
PreviousPower3Price	double
PreviousPower1Discount	double
PreviousPower2Discount	double
PreviousPower3Discount	double
PreviousPeakPrice	double
PreviousValleyPrice	double
PreviousSuperValleyPrice	double
PreviousConsumptionDiscount	double
Services	double
OldCompany	string
PreviousPower1	double
PreviousPower2	double
PreviousPower3	double

ServiceModel	
Serviceid	int
Userid	int
Services	string
Company	string
Payment	double
Time	string
Generatid	int
PreviousEmissions	double
AvoidedEmissions	double
PreviousAnnualCost	double
ActualYearlyCost	double
SavingWithOnerz	double

Users	
Userid	int
Name	string
Surname	string
Email	string
PhoneNumber	string
Houses	List<House>

Figure 17. Entities schema

In all our entities, our primary key will be the Id value related to that entity. This means that the first attribute of our entities will identify all the records of our tables.



Also, we will work with relational databases. This means that our data models will be interrelated. Our entities will be related in the following way:

- A User will have one authentication, and one authentication will be related to one user.
- A Bill will be related to one Consumption, and a Consumption will be related to a Bill.
- A Bill will be related to one House, while a House can be related to many Bills.
- A Bill will be related to one Company, and a Company can be related to many Bills.
- A House will be related to one OldBill, and one OldBill will be related to one House.
- A House will be related to a User, but a User can be related to many Houses.
- A House will be related to a Company, and a Company can be related to many Houses.
- An OldBill will be related to a Company, and a Company can be related to many OldBills.
- A Service will be related to a User, and a User can be related to many Services.

4.7. Databases treatment

4.7.1. DbContext

Once we have designed our data entities, we can define a DbContext instance. This represents a session with the database which can be used to query and save instances of the entities to a database. These allows us to:

- Manage our database connections.
- Configure models and their relationships.
- Querying database.
- Saving data in the database.
- Transaction management [36].

Thanks to our DbContext instance, we will be able to interact from our software with our databases and PostgreSQL.

4.7.2. Migration

By using the CodeFirst workflow, the recommended way to develop an application's database is through a migration. A migration is a set of tools that allow to:

- Create an initial database that works with an Entity Framework model.
- Generate migrations to keep track of changes made to an Entity Framework model.
- Keep a database up to date with those changes.

In order to use migrations, we first need to define our DbContext, the data models we want to create databases from, and activate the interaction with Entity Framework Core,



ensuring that the defined databases exist (and in the case these don't exist, will be created in a migration) [37].

Once we have done this, it is necessary to interact with the Package Manager Console, available on Visual Studio, following the next steps:

- First of all, it is necessary to enable migrations, using the command *Enable-Migrations*.
- Then, it's time to apply the migration into PostgreSQL, with the command *Add-Migration [name]*. This command adds a Migration folder in the project, with two files:
 - Configuration class: this class allows to configure how Migrations behave in our context. Usually, when creating the migration, this file is generated automatically.
 - [Time Code]_[Name] file: this file represents the created objects for our database. It's written in SQL, and will be applied to the database in order to generate the new databases.
 - An example of file name is "20200727112424_Initial.cs".
- Once a SQL file has been generated, it's time to apply the migration to the database, using the command *Update Database*. [38].

4.7.3. Filling

Once we have created our tables, these are empty at first, so, in order to work with them, it is necessary to fill them.

There are different procedures, like typing manually through a database manager like pgAdmin (PostgreSQL), uploading csv files with a similar structure to the tables, or through predesigned HTTP POST requests.

In our case, due to the big amount of information to upload, uploading csv files that we can create using Microsoft Excel is easier and time-saving. Also, most of the needed information can be easily copied and pasted from our CRM, developed in Google Sheets. Because of this, we will use this option.

Also, in our first stages, we will be developing in the development environment. So, we will need to fill first the tables that work in this environment. A good procedure to follow is to fill this tables at first with test data, and only when the software is finally developed and ready to be deployed, fill the development and production environment with the real data.

4.8. Definition of services, interfaces and infrastructure

In our software, in order to manage our controllers, we will need to create some classes, grouped in services. In .NET Core, controllers request dependencies explicitly via constructors (Dependency Injection) [39].

Services are added as a constructor parameter, and these are defined using interfaces. The methods developed in a service will be used later in a controller.

Meanwhile, in case a .NET Core application requires using endpoints from 3rd parties' APIs, a good practice is to develop a service class (and its interface) separated from the own services, as an infrastructure. So, each time our application needs to do a HTTP Request to one of those 3rd parties' APIs, it will call an equivalent class inside the infrastructure (by using the dependency injection), that will be in charge of doing the communication with the endpoint.

In our application, we will need to develop two different services (with their own interface) for:

- Authentication classes.
- Account managing classes.

4.8.1. Authentication classes

This class will define functions about:

- Acceptance/rejecting authentication pair value.
- Encryption/decryption of password.
- Bear token generation

In our authentication process, we follow the JSON Web Token standard (RFC 7519). This standard defines a compact and self-contained way for transmitting information safely between parties as a JSON object. Due to it is digitally signed, this information can be verified and trusted, by using a secret, using public/private key pair [35]. All the encryption and decryption in the software are done using SHA-256.

Once the authentication process has been completed successfully, this token will be necessary for calling the other endpoints and exchange information successfully.

4.8.1.1. User authentication

The authentication process follows the next workflow:

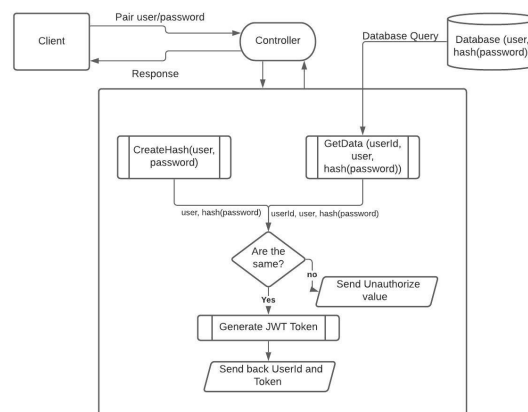


Figure 18. Authentication process schema

If we examine the authentication service, when being called by the controller, two functions are called at first. One of them creates a hash with the received inputs, that correspond to the user and the password, while the other function makes a query in the database, in order to retrieve the authentication data of the user, which are userId, email, phone and password. The saved value as password in our database is not the explicit password, but it's a hash created with the user's email and his password.

Then, in another function, the software checks if user and password's hash coming from the controller and from the database are the same. In that case, a token is created using the software's private key, and it's sent back to the controller with the user's userId value.

4.8.1.2. *Modifying a user's password*

In order to modify a user's password, we will need a new endpoint, that will modify the password's hash stored in a table with a new hash formed from the new password. In order to do that, we will follow the following workflow:

- First, the user will ask to change its password. We will ask for his email. As a result, and in the case this email exists in our databases, an email will be sent to the user, containing a link.
 - In order to increase this operation security, we create a process for handling it. This means that a new table in the database will be created, storing a process number, userId, an UUID identifying the process, a start date and an end date.
 - The operation will only be valid if the whole process it's finished before the defined end date. For this process, we have defined a short period, 5 minutes.
- The user will access through the link to a page prepared for handling this operation. In this page, he will introduce his email and the new password (twice). If it checks, the password's hash will be modified in the authentication's table, and the process will be finished.

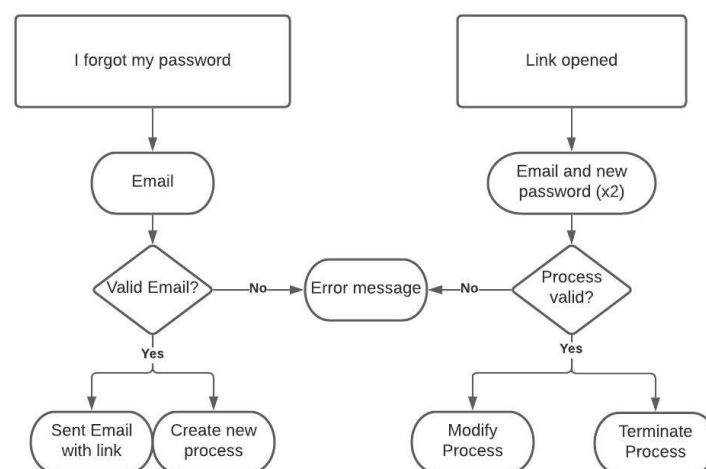


Figure 19. Modify password's process schema

4.8.2. Account managing classes

In this class, we are going to develop functions that will perform:

- Retrieving information from databases.
- Writing information in the databases.
- Modifying information in the databases.
- Calculate savings and emissions from data stored in the database.

4.8.2.1. *Retrieving information functions*

These functions will search and get information in the tables. In order to do that, we will use LINQ queries.

LINQ (Language Integrated Query) is a uniform query syntax integrated and used in C# and .NET used for saving and retrieving information from different sources. This syntax would work as a translator between C# code and SQL code [36]. The main advantages of using LINQ are:

- Developers don't need to learn a new querying language.
- Reduces the amount of code needed.
- It makes code more readable and understandable.
- Standardizes data consulting processes.
- Compiles and verifies the object types.
- Includes IntelliSense for generic collections.
- Allows retrieving data in different models.

In our case, we would use LINQ in order to query in our defined tables in the database through Entity Framework.

All these calls are going to be called from other functions or by controllers. The ones called by a controller will be explained in the part 4.7.2. and will be part of a GET request.

4.8.2.2. *Writing information functions*

These functions will add new lines in our tables and will be part of POST requests. These functions will use the commands from Entity Framework `Add(Entity)` and `SaveChanges()`.

4.8.2.3. *Modifying information functions*

These functions will query information in the tables, applying retrieving functions, and modify some attribute of them, saving these changes in the end. It is a necessary requirement that the requested information to modify has to exist in the tables of the database previously.

4.8.2.4. Calculations functions

These functions will do mathematic operations that will calculate monetary savings and emissions of our client's services, from information queried previously from the databases.

4.9. Definition of controllers

In a Controller class that invokes a ControllerBase class, we define our API's endpoints clients will interact with. Here, we will define the routes and the service's methods will be called from here.

As we did in our services part, we will make a separation between the authentication part of the workflow and the rest of the available operations.

Our controllers will have a common route in their URL, that will depend on the environment we are working with:

- Development environment: <https://dev-intranet-api.servest.app>
- Production environment: <https://intranet-api.servest.app>

4.9.1. Authentication controller

4.9.1.1. Authentication endpoint

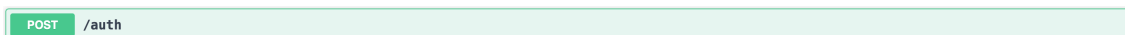


Figure 20. Authentication's endpoint route

This endpoint is a POST HTTP Request and is used for authenticating the user when trying to access our API. It activates the authentication service functions we explained previously in the point 4.6.1.1.

The petition requires a JSON body with the following structure, where the values marked as "string" are supposed to be changed by the input values.



Figure 21. Authentication's endpoint JSON body structure

In case it is successful, we will receive a 200 OK answer like the following:



Figure 22. Successful authentication's reply



4.9.1.2. *Modify password*

This process can be divided, as we saw previously in the workflow shown in the point 4.8.1.3) into two different processes, with their respective endpoints.

4.9.1.2.1. *Start modifying password's process endpoint*

```
POST /auth/startChange/{email}/{processId}
```

Figure 23. Start modifying password's endpoint route

This endpoint is a POST HTTP Request and is used for starting a process for changing a password. It sends an email to the specified email (in case it exists in the database) and start a process of 5 minutes for finishing the procedure (explained above).

This endpoint doesn't require any body of other information apart from an email and a UUID, generated from the end-point.

4.9.1.2.2. *Finish modifying password's process endpoint*

```
PATCH /auth/modifyPassword/{processId}
```

Figure 24. Finish modify password's endpoint route

This endpoint is a PATCH Request and is used for modifying the password in our databases. It needs a body with email and password and a processId (UUID) as a parameter of the petition.

In case the procedure has been completed successfully, a 200 OK answer will be received. But, in another case, an error message will be shown.

4.9.2. *Client controller*

All the endpoints defined in this controller will require an authentication token as a header of the request, following the structure "Bearer "+token. Now we can see all the endpoints used in our web application

4.9.2.1. *Get all the managed houses*

```
GET /api/user/usuarios/casa/{customerId}
```

Figure 25. Get all the managed house's endpoint route

Using this endpoint allows us to get all the houses where a user is suscribed. In order to execute it, it is necessary to introduce the User Id value as a parameter, and as a successful reply, the API will send back the data from that house.

The only needed header in this endpoint is the previously introduced Authorization header.



4.9.2.2. *Get all the user's data*

```
GET /api/user/cuenta/{customerId}
```

Figure 26. Get all the user's data endpoint route

Using this endpoint will allow us to get all the data related to the user defined through the user Id value introduced as a parameter of the endpoint. Also, it will include information about his registered houses.

The only needed header in this endpoint is the previously introduced Authorization header.

4.9.2.3. *Get all the bill's data*

```
GET /api/user/facturas/casa/{houseId}
```

Figure 27. Get all the bill's data endpoint route

Using this endpoint will allow us to get all the data related to a specified house's bills. As an input, we need to introduce as a parameter of the endpoint a house Id value, and if the call is successful, it will return a list with all the house's bill's, its respective consumption values, information about the house and its old bill. This is the most complete get endpoint in the API.

The only needed header in this endpoint is the previously introduced Authorization header.

4.9.2.4. *Get all the user's services data*

```
GET /api/user/servicios/{userId}
```

Figure 28. Get all the user's services data endpoint route

Using this endpoint will allow us to get all the data related to the hired services by a user. As an input, it will require the user Id value, and as an output, if the call is made successfully, will return a list with all the hired services by the user.

The only needed header in this endpoint is the previously introduced Authorization header.

4.9.2.5. *Post a new user*

```
POST /api/user/new/user
```

Figure 29. Post a new user endpoint route

Using this endpoint will allow us to create a new user in our tables. As an input, it will require the following data, written in a JSON file in the body:

- Name - string
- Surname - string
- Phone - string
- Email - string
- Password - string
- Service hired - string



- Company that offers the service - string
- Payment – double
- Periodicity of payment – string
- Name of the address – string
- Zip code – string
- City – string
- Region – string
- Country – string
- Data about the old bill (in case it is electricity, other case is null)
 - Price of the previous power 1 - double
 - Price of the previous power 2 - double
 - Price of the previous power 3 - double
 - Discount on the previous power 1 - double
 - Discount on the previous power 2 - double
 - Discount on the previous power 3 - double
 - Price of the peak consumption – double
 - Price of the valley consumption - double
 - Price of the supervalley consumption - double
 - Discount on consumption – double
 - Price of the extra services - double
 - Previous power 1 – double
 - Previous power 2 - double
 - Previous power 3 - double
 - Name of the old company – string

The only needed header in this endpoint is the previously introduced Authorization header. It will be used only by the IT team of the company; in case it is necessary to add a new subscriber.

4.10. Server publication

Once the application has been tested running it in a computer, it's time to upload it to a container in a server, where it will run, giving the possibility to the front-end application to connect to the back-end, and continue with the web application development.

In order to run our applications in a server, we will use an OS-level virtualization open-source code project called Docker, that works as a container's engine.

An OS-level virtualization is a method of virtualization where, over the OS core, a virtualization layer is executed, allowing the existence of multiple isolated instances of user spaces. These instances are called containers, and work as a real server from the user-side's point of view [40].

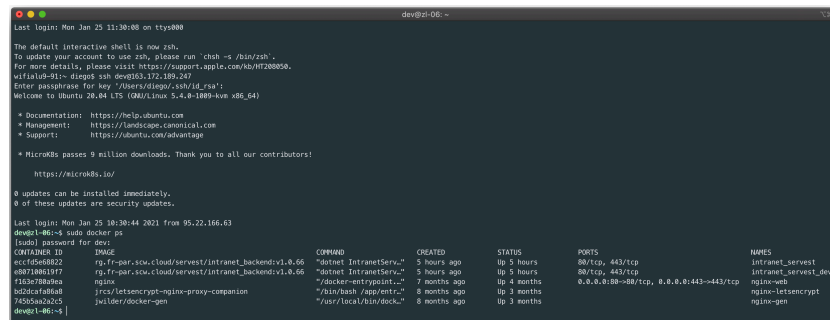
In our project, we decided to develop containers for the development software, the production software, and for a NGINX server, all of them running in an Ubuntu environment. This server will be protected with a public-private key system.

A NGINX server is a fast and reliable static web server, also used (as in our case) as a reverse-proxy used in front of the APIs, allowing a high concurrency, high performance and a low memory usage [41].

In order to upload our images to their respective container, we developed a script, available in the annex G, reducing all the tasks with Docker to a couple of commands. These commands are:

- Login in Docker
 - `docker login SERVERNAME/ SERVERNAMESPACE -u nologin -p PASSWORD`
- Creating an image of the software in the server (using the script `docker-upload.sh`)
 - `./docker-upload.sh IMAGENAME vVERSIONNAME`
- Connect to the server using SSH:
 - `ssh dev@SERVERIP`
 - Introduce passwords
- Upload image to containers
 - `sudo docker stop CONTAINERNAME && sudo docker container rm CONTAINERNAME && sudo ./docker-deploy.sh DOMAINNAME EMAIL CONTAINERNAME IMAGENAME:VERSIONNAME ENVIRONMENT`
 - `./docker-deploy.sh` is another script saved in the server OS, helpful for deploying the applications.

In our containers, we configured two ports for communications, using TCP. These are 80 and 443.



```
dev@21-06:~$ sudo docker ps
CONTAINER ID        IMAGE                                     COMMAND                  CREATED            STATUS            PORTS                               NAMES
eccc159e0822      rg-1-rgar-sbu-cloud/servevt/intranet_backend:v1.0.66  "dotnet IntranetServ..."  5 hours ago       Up 5 hours       80/tcp, 443/tcp                    intranet_servest
087f308e137f      rg-1-rgar-sbu-cloud/servevt/intranet_backend:v1.0.66  "dotnet IntranetServ..."  5 hours ago       Up 5 hours       80/tcp, 443/tcp                    intranet_servest_dev
f13c3e78a9bea     nginx                                     "docker-entrypoint..."  7 months ago      Up 4 months      0.0.0.0:80->80/tcp, 0.0.0.0:443->443/tcp  nginx-web
76d5c4ef8d68     jrcal/letsencrypt-nginx-proxy-companion              "bin/bash /app/ent..."  8 months ago      Up 3 months      0.0.0.0:80->80/tcp, 0.0.0.0:443->443/tcp  nginx-letsencrypt
74505a2a262c     juilder/docker-gon                                   "/usr/local/bin/dock..."  8 months ago      Up 3 months      0.0.0.0:80->80/tcp, 0.0.0.0:443->443/tcp  nginx-gon
```

Figure 30. Deployed docker containers in our server

4.11. Maintenance

Once the software is developed, it is important to do some maintenance works on the software, in order to fix bugs, adding information, adding new features, etc.

Some activities, like the addition of new features, can be developed as separated projects (as it is planned with the new company services). But in this project, we have to define procedures for adding automatically electricity bills to our database.

4.11.1. Automation of databases filling

The main goal of this activity is to make available for users all their electricity bills and, when a new one is generated by our provider (or ourselves), make it available for them as soon as possible, automatically.

For this part of the project, we will use the previously mentioned no-code tool called Integromat. This part will need three phases:

- Phase 1
 - Upload bills to our OCR.
 - Bill reading by the OCR.
- Phase 2
 - Download data from our OCR.
 - Database filling (consumptions table).
- Phase 3
 - Dropbox storage.
 - Database filling (bill details table).

Phase 1

As explained above, in this phase, we have to get a bill, classify it according to its origin (own source or provider) and send it to our OCR. In the end, we will approve the reading done by the OCR.

We propose the following workflow in Integromat.



Figure 31. Integromat's workflow

We receive four different kind of bills from three different providers. In order to initialize the process, we will upload all the documents to a folder made specially for that purpose.

Due our providers follow personalized naming structures, we can filter which kind of bills we are dealing with.

Then, we download the file from this folder, so we can get extra information, like a File ID used by Google, so we can identify that file in the next phases of the process. We store that ID in a Google Sheets file created as a registry of the bills uploaded to the database. In the Annex H, there's a brief explanation of this registry.

Then, it's time to filter which kind of bill we are working with. Our software is able to process three types of bills:

- **Electricity bill.**
 - This bill provides information about a period's consumption, expenses, etc. It's generated by our providers ("Holaluz" and "Gana Energia" paths).
- **Down payment rights bill.**
 - This bill is generated because of hiring our provider's services and is based on the work made in the change of operation ("Derechos de Enganche" path).
- **Quote bill.**
 - This bill provides information about the service we provide to the client and is generated by a third party we hire for it ("Onerz" path).

All the bills will follow a similar path, but we divided the process into three streams because we need to do changes in the filename according to the kind of bill. The name code of the bills is defined as follows:

- Electricity bill -> MM-YYYY Provider Name.pdf
- Down payment rights bill -> MM-YYYY Derechos de enganche.pdf
- Quote bill -> MM-YYYY Onerz.pdf

Being MM and YYYY the month and year we uploaded the bill to our table.

Finally, we should approve the reading done by our OCR in their web application. This step is recommendable, especially in the first stages, but not mandatory.

Phase 2

In this part, we need to download the taken data by our OCR, process it and store part of it in our consumption's table. In this process, we will define two different workflows, according to the origin of the analyzed bill.

First, we are going to analyze the designed workflow for the electricity bills and the down payment rights bill.

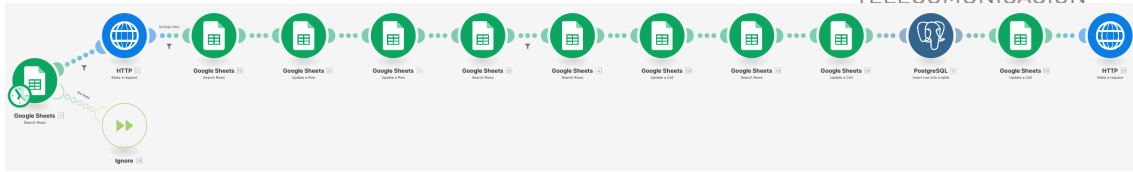


Figure 32. Integromat's workflow

First of all, in our bill registry, we search those bills that have been uploaded to our OCR, but the taken data hasn't been written in the registry. We can find that because the column D is filled, but the column E hasn't.

From those lines, we take the value stored in the column D, which is the ID used for uploading the file to our OCR. Thanks to this value, we can retrieve the information from our OCR.

Then, we write the retrieved information in our registry, both bill data and consumptions data.

Also, we need to perform a query, in order to write in our registry, the owner ID of the bill. This means, to which house is the bill assigned to. For that, we have another Google Sheet page with information related to all the assigned IDs to that house, street name, electricity CUPS of the house, NIF of our house's owner, and a Dropbox route of the folder that stores the bills of each client. For this workflow, we can do our query with the electricity CUPS.

Finally, once we have processed and stored all this information in our registry, we can upload it to our table in the database.

Once the information has been uploaded successfully, we mark a flag in our registry, and activate a webhook to a new automated scenario, part of the phase 3.

In the case we are analyzing a quote bill, the workflow is a bit different. These bills only show the quote paid by the client to the company, so we don't need to store any consumption. This is the used workflow:

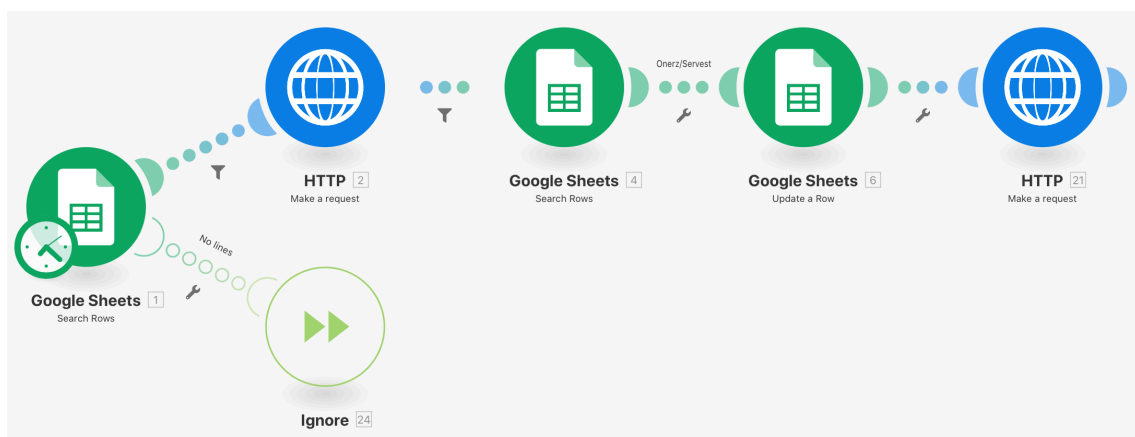


Figure 33. Integromat's workflow

The first part of the process is similar to the other workflow of this phase. We query in our registry and download the data from our OCR. But, in this case, if the process in our OCR has been done successfully, we only store data in our registry, and use as a consumption a reserved consumption for these cases. (with all values equal to 0). In order to identify the owner of the bill, we use a client Id number shown in the bill.

In the end, a flag is activated, as well as a webhook to the last process (phase 3).

Phase 3

This last phase, as mentioned above, will finish the bill storing process, and upload the bill to Dropbox, so users can access to it through a generated shared link with viewer permissions. The workflow is shown in the next figure.

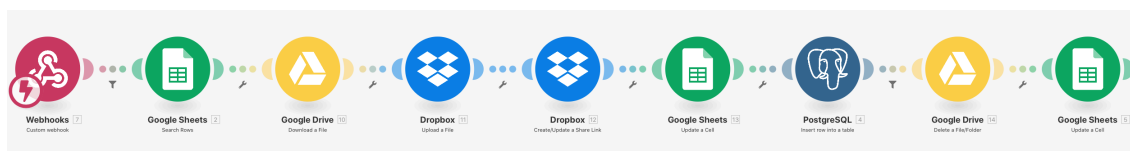


Figure 34. Integromat's workflow

The process starts when the webhook is activated (done by both scenarios in phase 2). Then, the line with the stored information of the stored bill is search (thanks to the flag we marked in the previous phase). Then, we get the stored file in our Google Drive folder, using it's file Id (stored in the link cell) and upload it to Dropbox. Then, we generate the shared link, and overwrite it on the link cell (where the Google's File ID was stored). Then, the bill information is written in our BillDetail's table in the database, and the file is deleted from the Google Drive folder. In the end, a final flag is marked, showing that the process has been finished.

4.11.2. User's addition

The main goal of this activity is to generate new accounts for the new users of the electricity service. As we said in the CRM part of this project, the subscription of our service has to be approved manually. But this record is written in our CRM, so the addition to our client area can be handled automatically, using Integromat and our developed API. The followed workflow is the following:

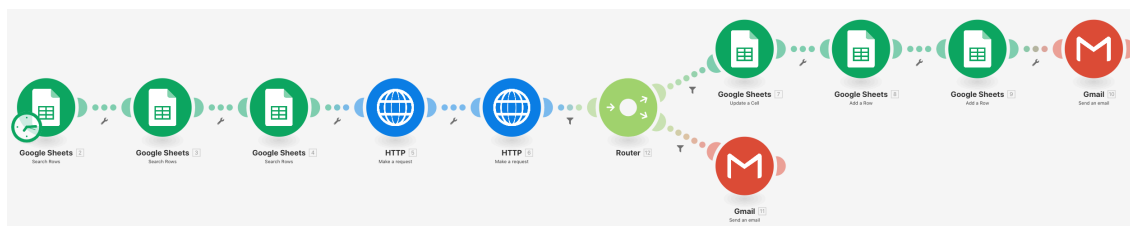


Figure 35. Integromat's workflow

First, we detect a new client in our CRM. A client will be a row in the page "Clients" with a subscription date and an empty private area subscription date. Once this new client

has been detected by the software, it will search more relevant info in our CRM, in the pages "Calculator" and "Quotes" and will perform two petitions to our API. The first one will be an authentication (with a reserved user and password), so we can get a token, which will be used as an authorization for our subscription petition (explained previously at the point 4.9.2.5). This request will be performed with the retrieved information from the CRM.

Then, some relevant information will be stored in a new spreadsheet (containing IDs and identity information of each client, only for operative purposes).

If the request has been done successfully, the private area subscription date will be filled with today's date. In case the request is replied with an error code, an email will be sent to the IT team of the company, in order to check the issue.

4.12. Results

Finally, it's the moment to test that our software works properly. In order to do that, we will start defining an example workflow for this test. Once we have defined the steps we will test and its order, we will make all the defined requests. Finally, we will comment the obtained results.

4.12.1. Workflow

First of all, we need to define a proper workflow, in order to test properly our whole software. We propose the following workflow.

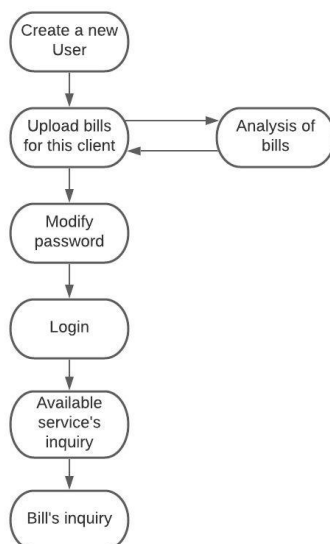


Figure 36. Testing workflow

First of all, we will create a new user, in order to carry our tests. Once the user has been created, we will upload some bills for him. We will analyze them in our OCR. In this moment, the account will be ready for our hypothetical client.

Then, we will change our hypothetical client's password. After this step, we will log in the account, so we can get all the private information of the client.

Finally, we will check the client's available services and its bills.

4.12.2. Testing

In order to start our testing, we will use the information we used when we tested our CRM (3.12). When the Integromat's scenario explained in the point 4.11.2 is activated, our signup procedure starts.

After a run to this scenario, we see that it has been completed successfully. The date has been written in our CRM, and the new user has been added to our database, with the following information:

- Username: diesabel@ade.upv.es
- Password: 1gAYn5nOpi

Now, it's time to upload a bill to his client area.

The process starts with an email to our billing email. Once our software detects this new email, it will start the phase 1, getting as a result that our bill is available on our OCR.

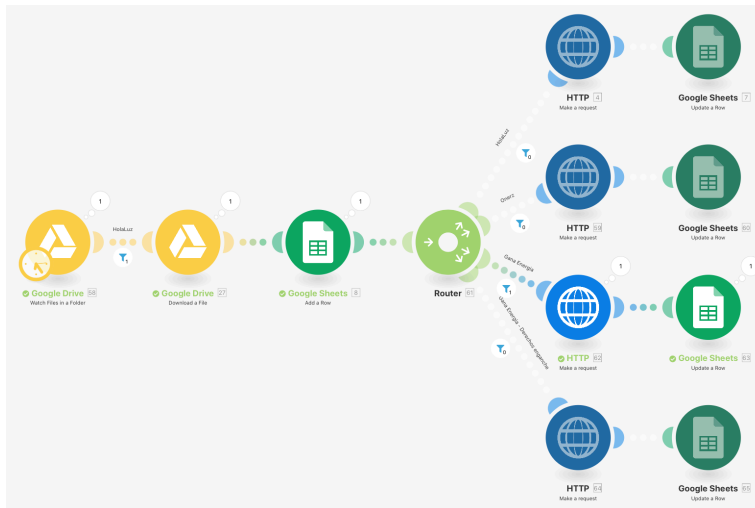


Figure 37. Result of phase 1

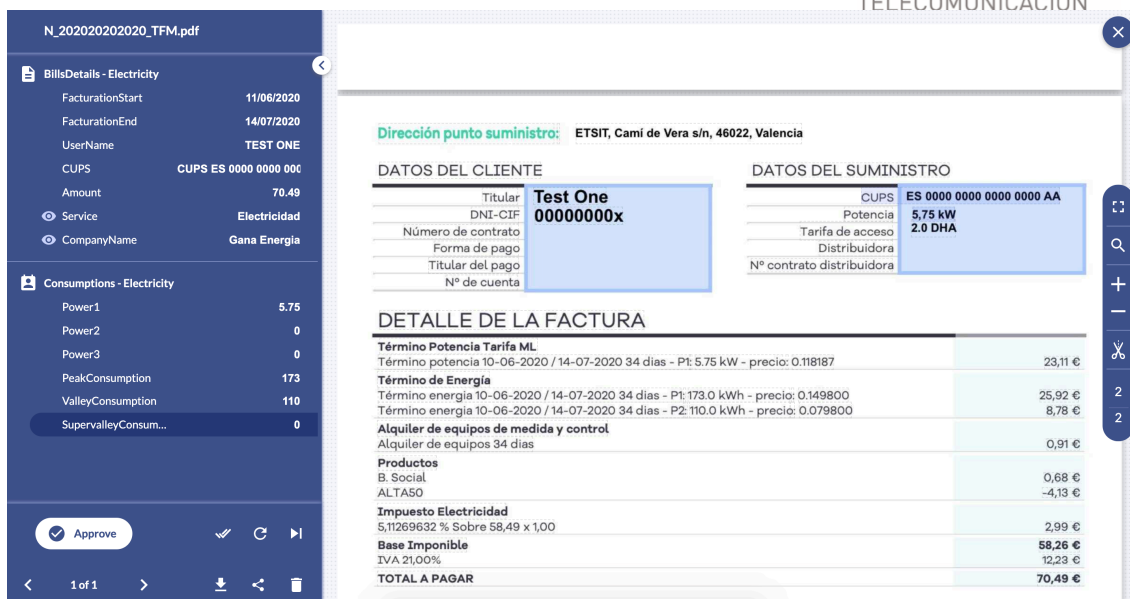


Figure 38. Bill analysis on our OCR

Once the bill has been approved, the phases 2 and 3 are ready to be activated. These procedures upload the bill to the database, making it available.

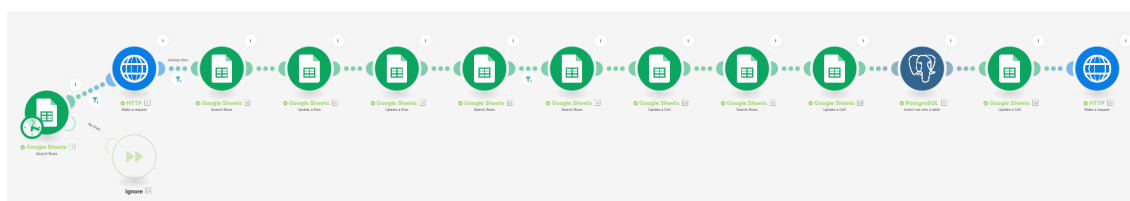


Figure 39. Result of phase 2

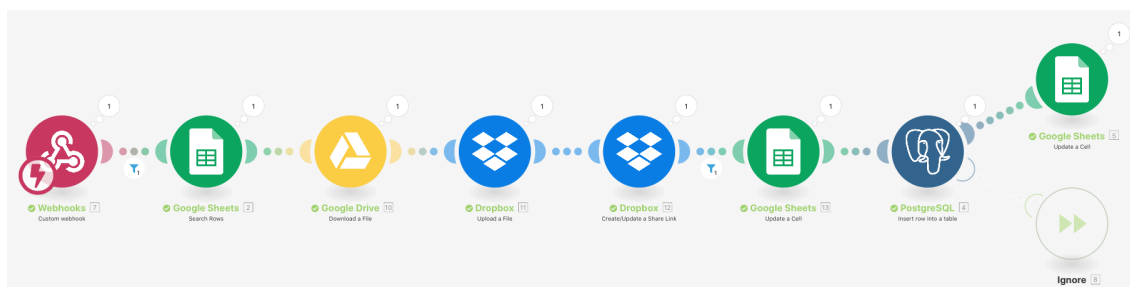


Figure 40. Result of phase 3

Now, it's time to change our passwords, due to our current password has been generated automatically, and it is recommendable to modify it for security and functional reasons.

From now on, we will use Postman as a client, in order to communicate with our designed API.

As we have explained in the point 4.9.1.2, the modifying password's process is divided into two parts:



- First, we have to make a HTTP petition, asking for starting a new petition to modify the user's password. It also sends an automatic email, with a link, from which users can modify it.
- Last, there's a second HTTP petition, which modifies the password.

In the next figure, we can see the request and result of this first petition.

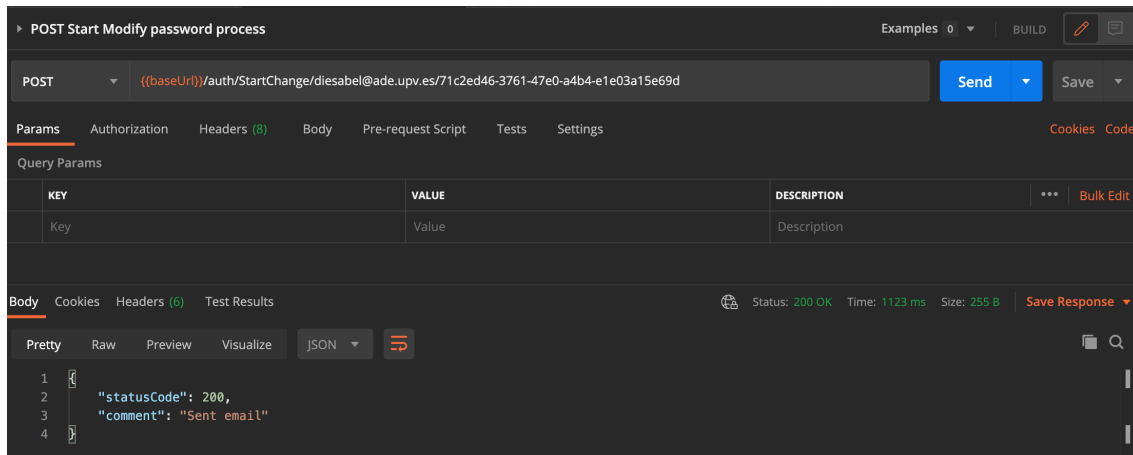


Figure 41. Start process

We receive a 200 OK reply, meaning that the process has been created, confirming that the email has been sent, and we have 5 minutes to finish the process.

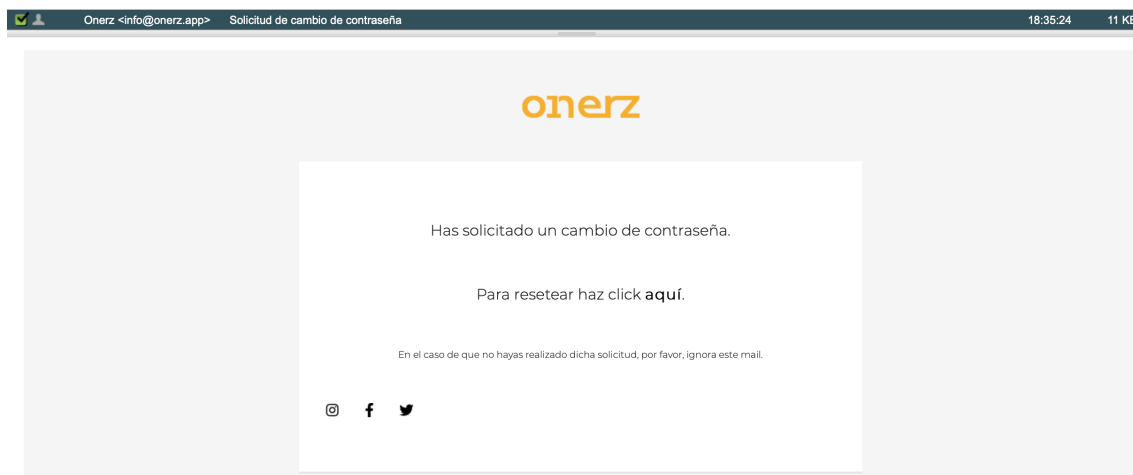


Figure 42. Received email

Then, we call the second endpoint, a PATCH petition, sending our new password. The HTTP request is shown in the next figure:

Now, it's time to call our hired services. At first, in order to try the software's security, we try to make the request with an expired token, getting the following result:

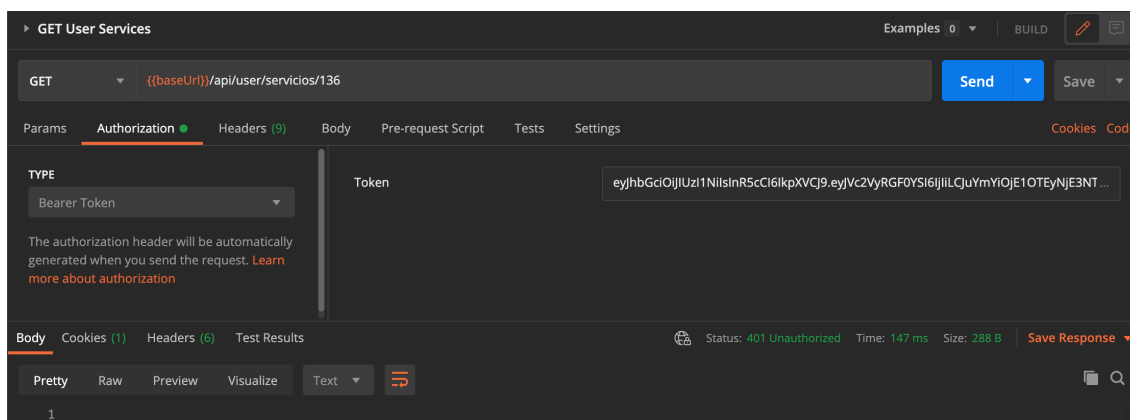


Figure 46. Unauthorized petition

As we can see, it returns a 401 Unauthorized response, meaning that our token is not valid and, even though the system understands the petition, is denied due to an error in the authorization stage.

Now, trying with the token we received when login into the system:

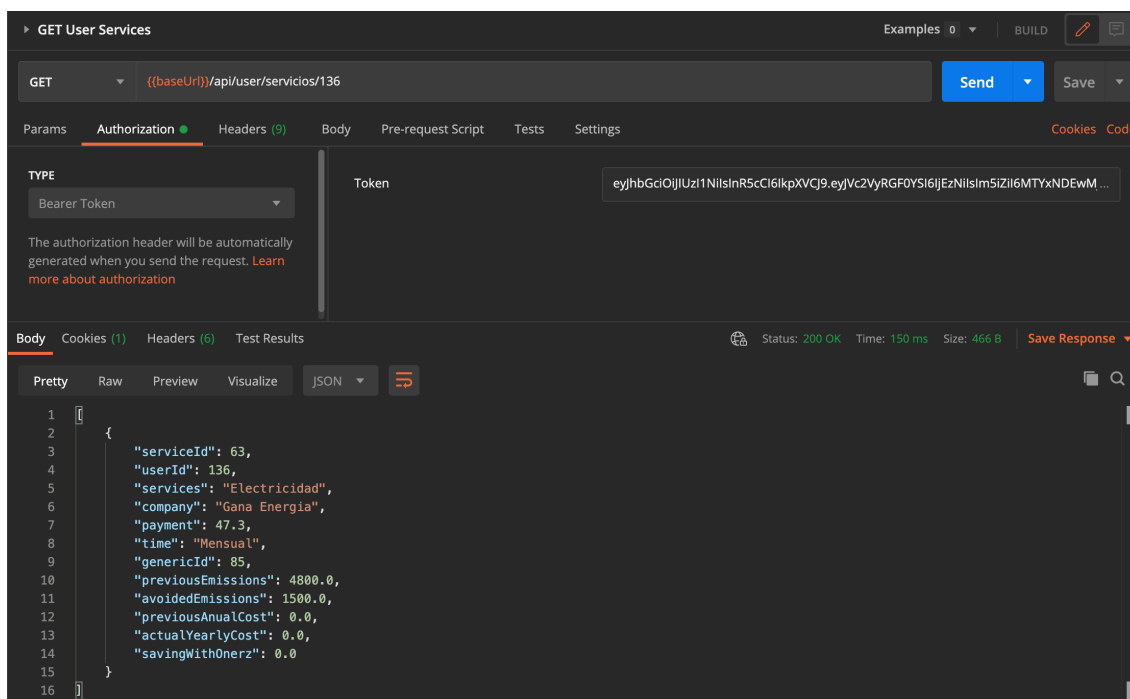


Figure 47. User services request

Using the token we received previously, we see that the response is a 200 OK, with the expected JSON file as a body of the response.

Finally, let's check if we can get the bill we uploaded to our client. For that, we do the following petition.

```
GET (baseUrl)/api/user/facturas/casa/85
Status: 200 OK Time: 727 ms Size: 145 KB Save Response
Body Cookies (1) Headers (6) Test Results
Pretty Raw Preview Visualize JSON
1 {
2   "billId": 1264,
3   "houseId": 85,
4   "consumptionId": 788,
5   "services": false,
6   "billTitle": "Factura Electricidad 2021-02",
7   "facturationStart": "11-06-20",
8   "facturationEnd": "14-07-20",
9   "amount": 78.49,
10  "difference": -766.2382880787879,
11  "co2Save": 73.58,
12  "radioactiveSave": 144.33,
13  "link": "https://www.dropbox.com/s/gcowr9xvdcqgg/2021-02%20Gana%20Energia%20%28%29.pdf?dl=1",
14  "companyName": "Gama Energia",
15  "houses": {
16    "houseId": 85,
17    "address": "ETSIT, Cami de Vera s/n",
18    "zipCode": "46022",
19    "city": "Valencia",
20    "region": "Valencia",
21    "country": "España",
22    "oldBill": {
23      "oldBillId": 63,
24      "houseId": 85,
25      "previousPower1Price": 0.123288,
26      "previousPower2Price": 0.0,
27      "previousPower3Price": 0.0,
28      "previousPowerDiscount": 0.0,
29      "previousPower2Discount": 0.0,
30      "previousPower3Discount": 0.0,
31      "previousPeakPrice": 0.154877,
32      "previousValleyPrice": 0.123288,
33      "previousSuperValleyPrice": 0.0,
34      "previousConsumptionDiscount": 15.0,
35      "services": 3.56,
36      "oldCompany": "Iberdrola",
37      "previousPower1": 3.45,
38      "previousPower2": 0.0,
39      "previousPower3": 0.0
40    },
41    "userId": 136,
42    "userId2": 0,
43    "userId3": 0,
44    "companyId": 2
45  },
46  "consumptions": {
47    "consumptionId": 788,
48    "houseId": 85,
49    "days": 33.0,
50    "measureDate": "14-07-2020",
51    "power1": 5.75,
52    "power2": 0.0,
53    "power3": 0.0,
54    "peakConsumption": 173.0,
55    "valleyConsumption": 118.0,
56    "superValleyConsumption": 0.0,
57    "co2Waste": 73.58,
58    "radioactiveWaste": 144.33,
59    "billId": 1264
60  }
61 }
62
63 }
```

Figure 48. Bills from house request

As we can see, we receive a 200 OK response with information in its body regarding the bill we just uploaded, the house it comes from (including information about the old billing of that house) and the consumption of that bill.

Chapter 5. Conclusion

This document has been divided into three differentiated parts.

In the first part of the project, we developed a project **planning document**. It contained a brief introduction of the project, a specification of the phases of development of the project, a temporal and economic planning, an uncertainty analysis and the closing documentation of the project.

This part of the project had some objectives to meet at the end of it, defined in the point 1.2 of this document.

The first of them was **establishing a realistic calendar**. We did a realistic calendar in the firsts phases of the project, establishing that this project could be done in 50 days, even though the effort was equivalent to 124 days of work. This is a long period for this project, but due to the low amount of resources, the precedent tasks and the numerous details of the project, it's realistic. Also, the final amount of time spent on developing this project for the company (in real life) was very close to 50 days.

The second of them was **managing effectively our available resources**. As said before, the whole project has been developed by 5 workers (counting the project manager), not all the workers could take part in all the parts of the project, due to lack of knowledge. This increased the difficulty level of the project, so managing effectively was mandatory. With the proposed resources distribution, the project could be developed effectively, even though some workers were on some free periods, but as a result of this managing, an effort of 124 days could be reduced to 50 days.

The third of them was **managing risks**. In order to manage risks, we made previously an uncertainty analysis, evaluating the conditions related to the project. We analyzed the macroenvironment and microenvironment of the project through a PESTEL analysis, the Porter's five forces, a SWOT analysis and evaluating the treatment of risks and opportunities. With all this analysis, we could be ready to face all the risks this project had with it.

The last of them was **establishing mechanisms of supervision of the daily work** during the development of the project. This has been solved introducing deliverables for all the activities done every week, about the finished activities during the week, and planning the activities to do during the next week, specifying the amount of time spent on all of them. Also, online meetings are another procedure of checking the daily work, through daily meeting or weekly meetings.

Jumping into the second part of the project, we **developed a CRM** for our coworkers. The main duty of this document was to ease our coworkers' duties. We divided this main goal into two more defined objectives.

The first objective was to **automatize electricity studies**. We could do that creating a sheet working as a calculator, where, after analyzing an electricity bill through our OCR, we could make studies regarding emissions, spending estimations and economic savings for our clients automatically.

Also, we had as an objective providing our coworkers of a **helpful tool for organizing clients and leads**. Using a numeric code system, our coworkers can organize the state of a lead or a client, and any modification was automatically shown in all the created sheets. And apart from the calculator sheet, we created other sheets, like a client's register, a quotes register, and other tools explained between the points 3.5 and 3.11, developed according to a predefined lead workflow, and hearing to the demands of our coworkers.

Finally, in the last part of the project, we developed a client area application. In this document, we explained the **development of the back-end side of the web application**. In this development, we created a new API with all the needed endpoints for meeting the initial requirements of the web application, as well as created two databases, one per development environment.

Through the developed endpoints, our clients could **access to their data and their electricity bills**. Also, these endpoints included **data regarding their electric consumptions, their emissions and savings**. All this information has been stored in our databases, and the API requests have been secured following an authentication process.

Also, in order to provide our clients with their electric data, we have created automations, easing the process of processing and storing new bills.

All this software has been tested, in order to grant both workers and clients of a reliable software, increasing that way the trust into the company.

As a result of all this, we can say that the objectives we defined at the start of the document have been met. Also, due to this project was published for the clients some months ago, we have seen that clients are satisfied with the product (according to their feedback). Even though, some refactoring and additions are planned for the future.



Chapter 6. References

- [1] Chaffee, A. "What is a web application (or "webapp")?". Jguru.com <http://www.jguru.com/faq/view.jsp?EID=129328>. July 2020.
- [2] Overview of Entity Framework Core - EF Core. (2020). Retrieved 17 September 2020, from <https://docs.microsoft.com/en-us/ef/core/>
- [3] PostgreSQL: The world's most advanced open source database. (2020). Retrieved 17 September 2020, from <https://www.postgresql.org>
- [4] Ambler, Scott. "Relational Databases 101: Looking at the Whole Picture". Retrieved 17 September 2020, from <http://www.agiledata.org/essays/relationalDatabases.html>
- [5] Codd, E.F (1969), Derivability, Redundancy, and Consistency of Relations Stored in Large Data Banks, Research Report, IBM.
- [6] Hodgson, Dr. J. P. E., "First Order Logic", Saint Joseph's University, Philadelphia, 1995.
- [7] Fisher, Sharon (1989). "OS/2 EE to Get 3270 Interface Early". Google Books. Retrieved 17 September 2020, from https://books.google.es/books?id=YToEAAAAMBAJ&pg=PA6&dq=application+programming+interface&redir_esc=y#v=onepage&q=application%20programming%20interface&f=false
- [8] Fielding, R., Irvine, U., Gettys, J., Mogul, J., Frystyk, h., & MIT et al. (1999). Hypertext Transfer Protocol -- HTTP/1.1. Retrieved 17 September 2020, from <https://www.w3.org/Protocols/rfc2616/rfc2616.txt>
- [9] Object-relational Mapping Revised - A Guideline Review and Consolidation. M Lorenz, G Hesse, JP Rudolph. International Joint Conference on Software Technologies (ICSOFT), 157-168
- [10] "Management Tools - Customer Relationship Management - Bain & Company". Retrieved 17 September 2020, from www.bain.com.
- [11] Onerz Technologies. (2020). Business Plan. Valencia: Zubi New Ventures SL.
- [12] Acer TravelMate P2410-G2. PcComponentes [Online]. Disponible en: <https://www.pccomponentes.com/acer-travelmate-p2410-g2-intel-core-i5-8250u-4gb-500gb-14>. Fecha de última visita: 24/09/2020.
- [13] Planes de precios | G Suite. (2020). Retrieved 24 September 2020, from <https://gsuite.google.com/intl/es-419/pricing.html>



- [14] Cloud, Compute, Storage and Network models and pricing - Scaleway. (2020). Retrieved 16 November 2020, from <https://www.scaleway.com/en/pricing/>
- [15] Docsumo | Simple pay-as-you-go pricing. (2020). Retrieved 24 September 2020, from <https://docsumo.com/pricing/>
- [16] Integromat | Pricing. (2020). Retrieved 24 September 2020, from <https://www.integromat.com/en/pricing>
- [17] Plans & Pricing | Typeform. (2020). Retrieved 24 September 2020, from https://www.typeform.com/pricing/?tf_campaign=brand_10147552538_v2&tf_source=google&tf_medium=paid&tf_content=99999583605_438052422454&tf_term=typeform%20pricing&tf_dv=c&tf_matchtype=e&tf_adposition=&gclid=CjwKCAjwh7H7BRBBEiwAPXjadnkZEcELqXAVY8pYIXtckV2FOakNPG_iQZpLV4GNonSzkJdKgVp0BoCOIlgQAvD_BwE
- [18] A simple pricing | Kantree, the truly flexible work management platform. (2020). Retrieved 28 September 2020, from <https://kantree.io/pricing>
- [19] What is Scrum?. (2020). Retrieved 13 October 2020, from <https://www.scrum.org/resources/what-is-scrum>
- [20] The QR Code Generator. (2020). Retrieved 5 November 2020, from <https://www.the-qr-code-generator.com>
- [21] Sales, D. (2020). List of Activities – Google Drive. Retrieved 5 November 2020, from <https://drive.google.com/drive/u/0/folders/1IK8ZkwaU4K47sUwdeuCz11JqrJOSu7ra>
- [22] Zubi Labs. (2020). Business Plan - Onerz [Ebook] (1st ed.). Valencia.
- [23] Qué es el CUPS y dónde encontrarlo. (2020). Retrieved 17 December 2020, from <https://www.endesa.com/es/conoce-la-energia/blog/cups-donde-encontrar>
- [24] Typeform: People-Friendly Forms and Surveys. (2020). Retrieved 17 December 2020, from <https://www.typeform.com>
- [25] Docsumo (2020). Retrieved 31 December 2020, from <https://docsumo.com>
- [26] Integromat. (2020). Retrieved 31 December 2020, from <https://www.integromat.com/en>
- [27] Developer.mozilla.org. n.d. Códigos De Estado De Respuesta HTTP - HTTP. [online] Available at: <<https://developer.mozilla.org/es/docs/Web/HTTP/Status>> [Accessed 9 January 2021].



- [28] Javascript-coder.com. n.d. A Modern Reintroduction To AJAX. [online] Available at: <https://www.javascript-coder.com/tutorials/re-introduction-to-ajax/> [Accessed 9 January 2021].
- [29] "A universal relation model for a nested database", The Nested Universal Relation Database Model, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 109–135, 1992, ISBN 978-3-540-55493-6, retrieved 9 January 2021.
- [30] "Download .NET Core". microsoft.com. Microsoft. Retrieved 13 January 2021.
- [31] Docker (2021). Retrieved 13 January 2021, from <https://www.docker.com>
- [32] Anderson, R., Dykstra, T. and Smith, S., 2019. Inicio De La Aplicación En ASP.NET Core. [online] Docs.microsoft.com. Available at: <<https://docs.microsoft.com/es-es/aspnet/core/fundamentals/startup?view=aspnetcore-5.0>> [Accessed 22 January 2021].
- [33] Swagger.io. n.d. API Documentation & Design Tools For Teams | Swagger. [online] Available at: <<https://swagger.io>> [Accessed 22 January 2021].
- [34] Ranganathan, A., «cross-site xmlhttprequest with CORS». [Accessed 22 January 2021].
- [35] Bogard, J., n.d. Automapper. [online] Automapper.org. Available at: <<https://automapper.org>> [Accessed 22 January 2021].
- [36] Entityframeworktutorial.net. n.d. Entity Framework Core Dbcontext. [online] Available at: <<https://www.entityframeworktutorial.net/efcore/entity-framework-core-dbcontext.aspx>> [Accessed 13 January 2021].
- [37] Lambson, B., 2018. Create and Drop API: EF Core. [online] Docs.microsoft.com. Available at: <<https://docs.microsoft.com/es-es/ef/core/managing-schemas/ensure-created>> [Accessed 16 February 2021].
- [38] Docs.microsoft.com. 2016. Migraciones De Code First: EF6. [online] Available at: <<https://docs.microsoft.com/es-es/ef/ef6/modeling/code-first/migrations/>> [Accessed 14 January 2021].
- [39] Namrouti, S., Anderson, R. and Smith, S., 2019. Inserción De Dependencias En Controladores En ASP.NET Core. [online] Docs.microsoft.com. Available at: <<https://docs.microsoft.com/es-es/aspnet/core/mvc/controllers/dependency-injection?view=aspnetcore-5.0>> [Accessed 15 January 2021].
- [40] Gil, H., 2019. La “Contenerización” De Aplicaciones - Computadores Plotandesign. [online] Computadores Plotandesign. Available at:



<<http://www.plotandesign.com/sistemas/contenerizacion-de-aplicaciones/>>
[Accessed 25 January 2021].

[41] Hub.docker.com. n.d. Nginx. [online] Available at:
<https://hub.docker.com/_/nginx> [Accessed 25 January 2021].

5. Annex A

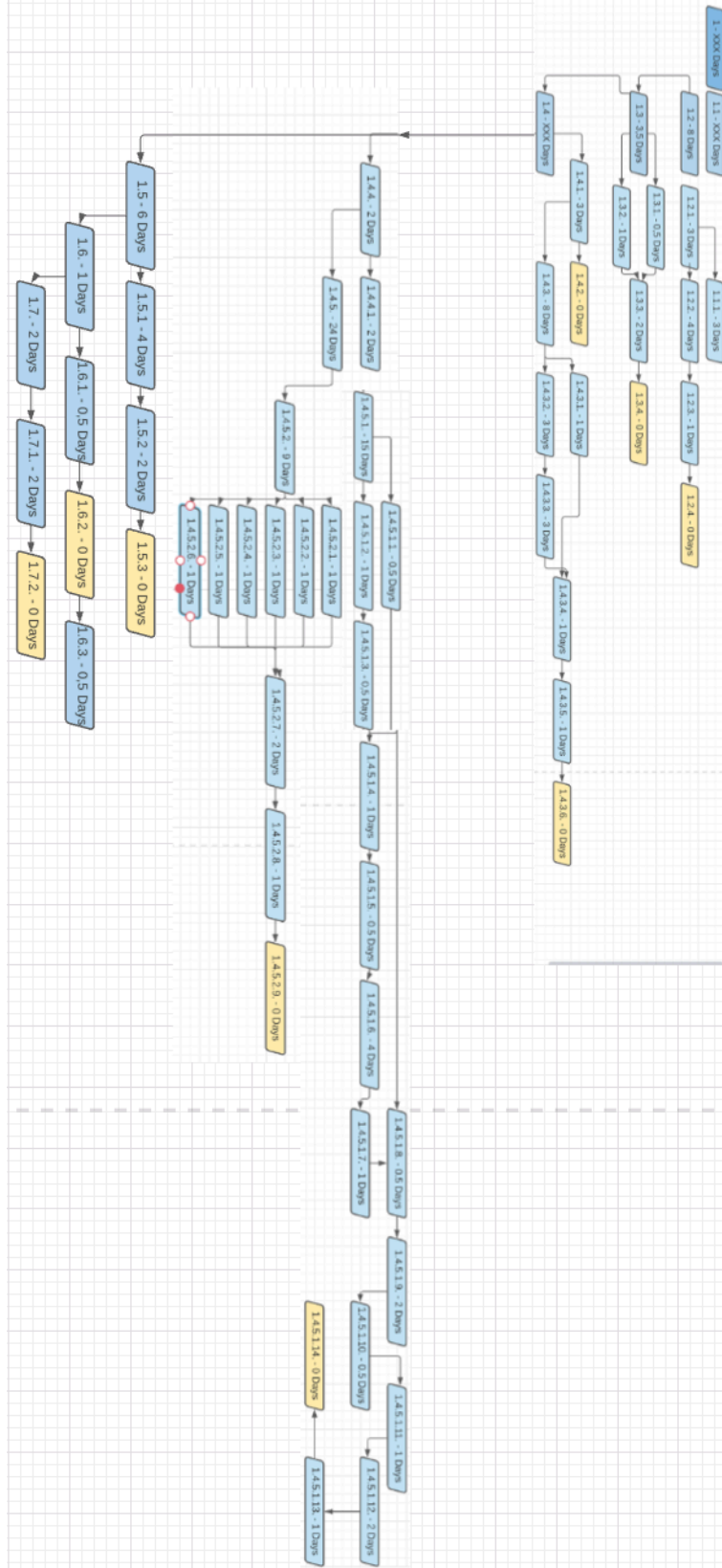


Figure 49. Diagram of precedencies



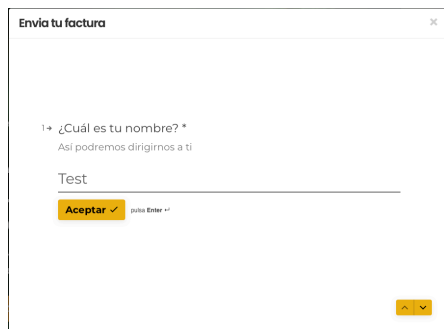
6. Annex B

Modo de	EDT	Nombre de tarea	Trabajo	Duración	Comienzo	Fin	Nombres de los recursos	Predecesoras
1	1	Project "First Phase of Onerz Development"	124 días	50 días	lun 20/04/20	mié 01/07/20		
2	1.1	Project Management	49 días	48 días	lun 20/04/20	lun 29/06/20		
3	1.1.1	Supervision & management of the project	48 días	48 días	lun 20/04/20	lun 29/06/20	Project Manager	59FF
4	1.2	Market analysis	8 días	8 días	lun 20/04/20	jue 30/04/20		
5	1.2.1	Identification of the needs of our clients	3 días	3 días	lun 20/04/20	mié 22/04/20	Development Manager	
6	1.2.2	Analysis of technological solutions	4 días	4 días	jue 23/04/20	mar 28/04/20	Development Manager	5
7	1.2.3	Research and analysis of similar cases and	1 día	1 día	jue 30/04/20	jue 30/04/20	Development Manager	6
8	1.2.4	Milestone 1	0 días	0 días	jue 30/04/20	jue 30/04/20		5FF;6FF;7FF
9	1.3	Acquisitions	3,5 días	3,5 días	lun 04/05/20	jue 07/05/20		4
10	1.3.1	Analysis of material needs	0,5 días	0,5 días	lun 04/05/20	lun 04/05/20	Development Manager	
11	1.3.2	Analysis of required licenses and software	1 día	1 día	lun 04/05/20	mar 05/05/20	Development Manager	10
12	1.3.3	Acquisition of needed resources	2 días	2 días	mar 05/05/20	jue 07/05/20	Development Manager	11
13	1.3.4	Milestone 3	0 días	0 días	jue 07/05/20	jue 07/05/20		12FF
14	1.4	Software development	48 días	29 días	jue 07/05/20	mié 17/06/20		9
15	1.4.1	Analysis of functional needs	3 días	1,5 días	jue 07/05/20	vie 08/05/20	Developer 1;Development Manager	
16	1.4.2	Milestone 2	0 días	0 días	vie 08/05/20	vie 08/05/20		15FF
17	1.4.3	Development of the CRM	16 días	8 días	lun 11/05/20	mié 20/05/20		15
18	1.4.3.1	Creation of tables	1 día	1 día	lun 11/05/20	lun 11/05/20	Developer 1	
19	1.4.3.2	Definition of functions	6 días	3 días	lun 11/05/20	mié 13/05/20	Developer 2;Development Manager	
20	1.4.3.3	Coding of functions	6 días	3 días	jue 14/05/20	lun 18/05/20	Developer 1;Developer 2	18;19
21	1.4.3.4	Tests	2 días	1 día	mar 19/05/20	mar 19/05/20	Developer 1;Development Manager	20
22	1.4.3.5	Connection with sources of information	1 día	1 día	mié 20/05/20	mié 20/05/20	Developer 1	21
23	1.4.3.6	Milestone 4	0 días	0 días	mié 20/05/20	mié 20/05/20		22
24	1.4.4	Web app design	4 días	2 días	jue 21/05/20	vie 22/05/20		17
25	1.4.4.1	Web app sketch (front-end)	4 días	2 días	jue 21/05/20	vie 22/05/20	Developer 3;Development Manager	
26	1.4.5	Web app development	25 días	17,5 días	lun 25/05/20	mié 17/06/20		17
27	1.4.5.1	Back-end development	16 días	15,5 días	lun 25/05/20	lun 15/06/20		17
28	1.4.5.1.1	Creation of Docker containers in a se	0,5 días	0,5 días	lun 25/05/20	lun 25/05/20	Developer 2	
29	1.4.5.1.2	Definition of models and entities	1 día	1 día	lun 25/05/20	lun 25/05/20	Developer 1	
30	1.4.5.1.3	Definition of databases context	0,5 días	0,5 días	mar 26/05/20	mar 26/05/20	Developer 1	29
31	1.4.5.1.4	Migration to databases in a server	1 día	1 día	mar 26/05/20	mié 27/05/20	Developer 1	28;30
32	1.4.5.1.5	Population of databases with test da	1 día	1 día	mié 27/05/20	jue 28/05/20	Developer 1	31
33	1.4.5.1.6	Definition of services	4 días	4 días	jue 28/05/20	mié 03/06/20	Developer 1	32
34	1.4.5.1.7	Definition of API controllers	1 día	1 día	mié 03/06/20	jue 04/06/20	Developer 1	33
35	1.4.5.1.8	Upload API image into development	0,5 días	0,5 días	jue 04/06/20	jue 04/06/20	Developer 2	28;34
36	1.4.5.1.9	Tests with server in development	2 días	2 días	vie 05/06/20	lun 08/06/20	Developer 1	35
37	1.4.5.1.10	Upload API image into production co	0,5 días	0,5 días	mar 09/06/20	mar 09/06/20	Developer 2	36
38	1.4.5.1.11	Tests with server in production	1 día	1 día	mar 09/06/20	mié 10/06/20	Developer 1	37
39	1.4.5.1.12	Population of databases with real da	2 días	2 días	mié 10/06/20	vie 12/06/20	Developer 1	38
40	1.4.5.1.13	Elaboration of a manual	1 día	1 día	vie 12/06/20	lun 15/06/20	Developer 1	39
41	1.4.5.1.14	Milestone 5	0 días	0 días	lun 15/06/20	lun 15/06/20		40FF
42	1.4.5.2	Front-end development	9 días	17,5 días	lun 25/05/20	mié 17/06/20		24
43	1.4.5.2.1	Authentication page	1 día	1 día	lun 25/05/20	lun 25/05/20	Developer 3	
44	1.4.5.2.2	Index page	1 día	1 día	mar 26/05/20	mar 26/05/20	Developer 3	43
45	1.4.5.2.3	Emissions page	1 día	1 día	mié 27/05/20	mié 27/05/20	Developer 3	44
46	1.4.5.2.4	Expenses page	1 día	1 día	jue 28/05/20	jue 28/05/20	Developer 3	45
47	1.4.5.2.5	Consumption page	1 día	1 día	vie 29/05/20	vie 29/05/20	Developer 3	46
48	1.4.5.2.6	Savings page	1 día	1 día	lun 01/06/20	lun 01/06/20	Developer 3	47
49	1.4.5.2.7	Interconnection back-front end	2 días	1 día	lun 15/06/20	mar 16/06/20	Developer 1;Developer 3	48;40
50	1.4.5.2.8	Upload to development	1 día	1 día	mar 16/06/20	mié 17/06/20	Developer 3	49
51	1.4.5.2.9	Milestone 6	0 días	0 días	mié 17/06/20	mié 17/06/20		50FF
52	1.5	Final tests	12 días	6 días	mié 17/06/20	vie 26/06/20		27;42
53	1.5.1	Testing in development	8 días	4 días	mié 17/06/20	mar 23/06/20	Developer 3;Developer 1	
54	1.5.2	Testing in production	4 días	2 días	mar 23/06/20	vie 26/06/20	Developer 3;Developer 1	53
55	1.6	Project publication	1,5 días	1,5 días	vie 26/06/20	lun 29/06/20		52
56	1.6.1	Publication of the final web application	1 día	1 día	vie 26/06/20	lun 29/06/20	Developer 3	
57	1.6.2	Milestone 7	0 días	0 días	lun 29/06/20	lun 29/06/20		56FF
58	1.6.3	Communication to clients	0,5 días	0,5 días	lun 29/06/20	lun 29/06/20	Development Manager	56
59	1.7	End of project	2 días	2 días	mar 30/06/20	mié 01/07/20		58
60	1.7.1	Closing document	2 días	2 días	mar 30/06/20	mié 01/07/20	Project Manager	
61	1.7.2	Milestone 8	0 días	0 días	mié 01/07/20	mié 01/07/20		60

Table 10. Table with the project planning

7. Annex C

In this annex, we can see an example of data filling of the used form, in order to retrieve information from clients. Due to the company works in Spain, the form is written in Spanish.



Envía tu factura

1+ ¿Cuál es tu nombre? *

Así podremos dirigirnos a ti

Test

Aceptar ✓ Salvo Enter ↵

Figure 50. Question 1



Envía tu factura

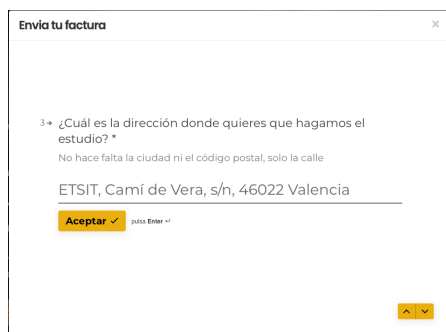
2+ ¿y tu email? *

A este email te enviaremos el estudio de cuánto puedes ahorrar

diesabel@ade.upv.es

Aceptar ✓ Salvo Enter ↵

Figure 51. Question 2



Envía tu factura

3+ ¿Cuál es la dirección donde quieres que hagamos el estudio? *

No hace falta la ciudad ni el código postal, solo la calle

ETSIT, Camí de Vera, s/n, 46022 Valencia

Aceptar ✓ Salvo Enter ↵

Figure 52. Question 3



Envía tu factura

4+ Haz una foto de la parte delantera *

O sube la factura completa en .pdf

dummy.pdf

Aceptar ✓ Salvo Enter ↵

Figure 53. Question 4

Envía tu factura

5+ Ahora haz una foto de la parte trasera
O si ya has adjuntado la factura entera en .pdf puedes saltar esta pregunta (apretando en el símbolo "v" naranja de abajo)

Elige el archivo o arrastra aquí
Límite de tamaño: 30MB

Acceptar

Figure 54. Question 5

Envía tu factura

6+ ¿Cuál es tu número de teléfono?
Solo contactaremos contigo si tenemos alguna duda con respecto a tu factura, ¡prometido!

612 34 56 78

Acceptar

Figure 55. Question 6

Envía tu factura

7+ ¿Aceptas nuestra política de privacidad y protección de datos? *

Solo utilizaremos tus datos con el fin de enviarte un estudio personalizado, ¡prometido!

Acepto



No acepto

Figure 56. Question 7



8. Annex D

IBERDROLA CLIENTES, S.A.U.
CIF A-95758389

Página 1 / 3

FACTURA DE ELECTRICIDAD PLAN ESTABLE

Remite: IBERDROLA CLIENTES, S.A.U. Apartado de Correos 61175 28080 Madrid

Test One ETSIT, Camí de Vera s/n 46022, Valencia

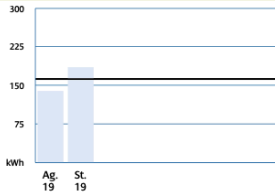
Dirección de suministro: **ETSIT, Camí de Vera s/n**

DATOS DE FACTURA

Periodo de facturación 04/08/2019 – 02/09/2019
 Número de factura [REDACTED]
 Fecha de emisión de factura 4 de septiembre de 2019
 Fecha prevista de cargo 12/09/2019
 Factura con lectura real
 Titular Test One
 NIF: 00000000X
 Referencia contrato suministro

TOTAL IMPORTE FACTURA: 50,97 €

EVOLUCIÓN DE CONSUMO



Este gráfico muestra la evolución de su consumo.
 Su consumo medio diario en este último periodo facturado ha sido: 1,75 €
 Su consumo medio diario en los últimos 2 meses ha sido: 1,43 €
 1 kilovatio-hora (kWh) equivale al consumo de una lámpara de 100 vatios funcionando durante 10 horas.

RESUMEN DE FACTURACIÓN

ENERGÍA	38,56 €
SERVICIOS Y OTROS CONCEPTOS	3,56 €
IVA 21% s/42,12 €	8,85 €
TOTAL A PAGAR	50,97 €

> ver detalle de facturación y consumo en el reverso

i Según indica el Real Decreto 1718/2012 le informamos que el importe correspondiente a las tarifas de acceso a redes en esta factura, sin impuestos, ha ascendido a 17,23 €, distribuidos del siguiente modo:
 Término de potencia: 10,43 € Término de energía: 6,03 € Alquiler equipos medida: 0,77 €
 A estos importes les son aplicables los impuestos correspondientes sobre el total (Impuesto Eléctrico incluido).
 Estos valores son informativos y no representan ningún incremento de coste para Vd. ya que están englobados en su factura de energía.
 Le informamos que el 100% de los kWh consumidos han sido producidos por fuentes de energía recogidas por la Certificación de Garantía de Origen.

i **Atención al Cliente: Consultas, gestiones y reclamaciones 24 horas en el 900 225 235**

Atención Averías de Red: 900171171

Servicio Asistencia Técnica: 900 22 45 22

Puntos de atención
www.iberdrola.es/puntosdeatencion

www.iberdrola.es

@Tuiberdrola










Figure 57. Page 1 of a test bill

DATOS RELACIONADOS CON SU SUMINISTRO

Nº contador: **1-1-1-1**
Referencia contrato suministro: **1-1-1-1**
Empresa distribuidora: I-DE, **Redes Eléctricas Inteligentes, S.A.U.**
Número de contrato de acceso: **1-1-1-1**
Identificación punto de suministro (CUPS): ES 0000 0000 0000 0000 AA
Forma de pago: DOMICILIACION BANCARIA
Entidad:
IBAN:
BIC:
Código de mandato:
**** Ocultos para su seguridad

Potencia contratada: **3,45 kW**
Peaje de acceso a la red (ATR): **2.0DHA**
Precios de peajes de acceso: **B.O.E. del 27/12/2017**
Duración de contrato hasta: **18/07/2020**
Dirección fiscal: **ETSIT, Camí de Vera s/n**
Con contador inteligente efectivamente integrado en el sistema de telegestión.
Portal de medidas:
www.iberdroladistribucionelctrica.com/consumidor

CONOZCA AL DETALLE SU FACTURACIÓN Y CONSUMOS

ENERGÍA			
Potencia facturada	3,45 kW x 29 días x 0,123288 €/kW día		12,33 €
Energía facturada	185 kWh x 0,154877 €/kWh		28,65 €
Descuento sobre consumo 15%	15% s/28,65 €		-4,30 €
Impuesto sobre electricidad	5,11269632% s/36,68 €		1,88 €
TOTAL ENERGÍA			38,56 €
SERVICIOS Y OTROS CONCEPTOS			
Alquiler equipos medida	29 días x 0,02663 €/día		0,77 €
Protección Eléctrica Hogar	0,94 mes x 5,95 €/mes		5,59 €
Descuento Protección Eléctrica Hogar	50% s/5,59 €		-2,80 €
TOTAL SERVICIOS Y OTROS CONCEPTOS			3,56 €
IMPORTE TOTAL			42,12 €
IVA	21% s/42,12 €		8,85 €
TOTAL IMPORTE FACTURA			50,97 €

CONSUMOS

Periodo horario	Desde	Lectura	Hasta	Lectura	Consumo/Potencia
PUNTA	04/08/2019	010753	02/09/2019	010847	94 kWh
VALLE	04/08/2019	010583	02/09/2019	010674	91 kWh

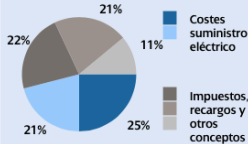
Última lectura: real

La **lectura real** es el valor leído por su distribuidor en su contador en la fecha indicada.

La **lectura estimada** es un valor que su distribuidor calcula tomando como base los consumos históricos y según una fórmula reglamentada por el Ministerio de Industria.

EL 54% DE SU FACTURA

ESTÁ DESTINADO A IMPUESTOS Y OTROS RECARGOS



Costes suministro eléctrico 21,75 €

Coste de producción de electricidad 11,66 €

Coste de redes de transporte y distribución 10,09 €

Impuestos, recargos y otros conceptos 25,66 €

Impuestos aplicados 10,73 €

Incentivos a las energías renovables, cogeneración y residuos 9,87 €

Otros costes regulados 5,06 €

TOTAL IMPORTE FACTURA 50,97 €

A los importes debe añadirse el alquiler de los equipos de medida y otros servicios, en caso de tenerlos contratados.

Conozca el detalle en www.iberdrola.es

INFORMACIÓN DE UTILIDAD

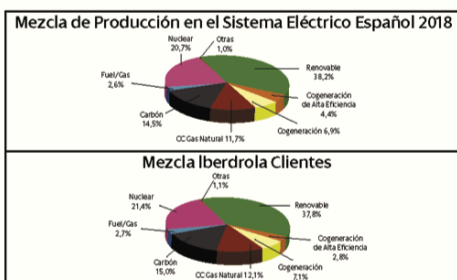
- Para reclamaciones relacionadas con el contrato de suministro o la facturación puede contactar con nosotros en el teléfono gratuito 900 225 235, en clientes@iberdrola.es o en el Apartado de Correos 61090, 28080 MADRID. También, puede dirigirse a la Junta Arbitral de Consumo de su comunidad autónoma (www.iberdrola.es) o a los órganos competentes en materia de Consumo o de Energía de dicha comunidad.

INFORMACIÓN SOBRE SU ELECTRICIDAD

Origen de la electricidad

Si bien la energía eléctrica que llega a nuestros hogares es indistinguible de la que consumen nuestros vecinos u otros consumidores conectados al mismo sistema eléctrico, es posible conocer el origen de la producción de energía eléctrica equivalente a la que usted consume.

A estos efectos, se proporciona el desglose de la mezcla de tecnologías de producción nacional para así comparar los porcentajes del promedio nacional con los correspondientes a la energía vendida por su Compañía Comercializadora.



Origen	Mezcla de Producción en el Sistema Eléctrico Español	Mezcla Iberdrola Clientes
Renovable	38,2%	37,8%
Cogeneración de Alta Eficiencia	4,4%	2,8%
Cogeneración	6,9%	7,1%
CC Gas Natural	11,7%	12,1%
Carbón	14,5%	15,0%
Fuel/Gas	2,6%	2,7%
Nuclear	20,7%	21,4%
Otras	1,0%	1,1%

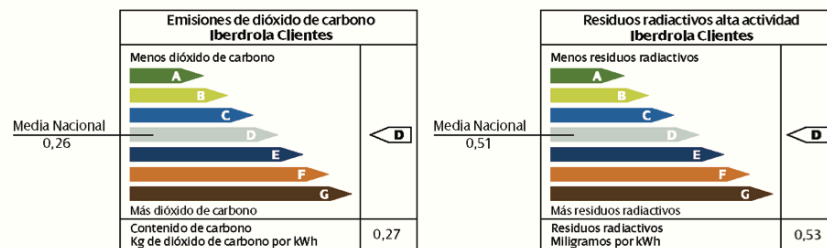
El sistema eléctrico nacional ha importado un 3.5% de producción neta total.

Impacto medioambiental

El impacto ambiental de su electricidad depende de las fuentes energéticas utilizadas para su generación.

En una escala de A a G donde A indica el mínimo impacto ambiental y G el máximo, y que el valor medio nacional corresponde al nivel D, la energía comercializada por Iberdrola tiene los siguientes valores.

De acuerdo con el sistema de Garantía de Origen e Información al Consumidor, implantado por la Comisión Nacional de la Energía, Iberdrola informa que toda la electricidad comercializada en 2018 ha sido etiquetada en la categoría D que indica un impacto ambiental en línea con la media nacional.



LA ELECTRICIDAD DE IBERDROLA PROCEDE EN SU MAYOR PARTE DE FUENTES RENOVABLES Y ES RESPETUOSA CON EL MEDIO AMBIENTE.

TUS GESTOS AHORRAN ENERGÍA Y CUIDAN DEL PLANETA

Cada vez consumimos más energía y los recursos para generarla no son infinitos. Por eso, pequeños gestos pueden disminuir tu consumo manteniendo el nivel de confort. Así, conseguirás un ahorro y estarás cuidando del planeta que nos proporciona esa energía.

- Las **lámparas LED**, además de proporcionar una luz de calidad, proporcionan una gran cantidad de luz con un consumo reducido. Son la tecnología actual más eficiente del mercado, y además las que tienen una mayor duración (siempre y cuando sean de calidad).
- El **frigorífico** es normalmente el electrodoméstico que más energía consume a lo largo del año en un hogar. Ajusta la temperatura para que no enfríe de forma excesiva, y cuando vayas a meter o sacar algo, intenta que la puerta esté abierta el menor tiempo posible.
- Siempre que puedas, **utiliza el lavavajillas y la lavadora al máximo de su capacidad**, seleccionando la menor temperatura de lavado que sea posible. De esta forma, ahorras energía, agua y detergente, y alargas la vida del electrodoméstico.
- **Tapa los recipientes mientras cocinas**. Así consumirás menos energía, al lograr un calentamiento más rápido.
- El **aire acondicionado** gasta entre un 4% y un 6% más de energía por cada grado que se baja la temperatura. Úsalo con moderación.

Puedes encontrar más consejos en www.iberdrola.es/consejosdeahorro





9. Annex E

Block	Column	Value
Client	Name	Input from Typeform
	Surname	Input from Typeform
	Address	Input from Typeform
	Email	Input from Typeform
	Typeform Document 1	Input from Typeform
	Typeform Document 2	Input from Typeform
	Google Drive Document	Input from Typeform
Docsumo	Docsumo ID	Input from Docsumo
	ZIP Code	Input from Docsumo
	City	Input from Docsumo
	Province	Input from Docsumo
	Owner	Input from Docsumo
	NID	Input from Docsumo
	CLUPS	Input from Docsumo
Current	Result	OK -> acceptable data REVIEW -> needed a revision
	Period	365 days (period of time for the study)
	Current Toll	Read Toll
	Current Power 1	Read Power 1
	Current Power 2	Read Power 2
	Current Power 3	Read Power 3
	Current Company	Read Company
Docsumo	Read Toll	Input from Docsumo
	Read Power 1	Input from Docsumo
	Read Power 2	Input from Docsumo
	Read Power 3	Input from Docsumo
	Read Company	Input from Docsumo
	PVPC?	Input from Docsumo
Current	Peak Consumption	IF(Valley Consumption = 0 & PVPC = false) -> Period/(final Date - start Date)*0,45*Read Peak Consumption IF(Valley Consumption != 0 & PVPC = false) -> Period/(final Date - start Date)*Read Peak Consumption
	Valley Consumption	IF(Valley Consumption = 0 & PVPC = false) -> Period/(final Date - start Date)*0,55*Read Peak Consumption IF(Valley Consumption != 0 & PVPC = false) -> Period/(final Date - start Date)*Read Valley Consumption
	Supervalley Consumption	12*Read Supervalley Consumption
	Read Peak Consumption	Input from Docsumo
Docsumo	Read Valley Consumption	Input from Docsumo
	Read Supervalley Consumption	Input from Docsumo
	Price Power 1	IF(PVPC) -> 0,104229 ELSE Read Power Price 1
Current	Discount Power 1	Read Discount Power 1
	Price Power 2	Read Price Power 2
	Price Power 3	Read Price Power 3
	Peak Price	IF(PVPC) -> (IF(Toll = 2.0A) -> 0,108786 ELSE 0,128467) ELSE Read Peak Price
	Valley Price	IF(PVPC) -> (IF(Toll = 2.0DHA) -> 0,058578 ELSE 0,108786) ELSE IF Read Valley Price != 0 -> Read Valley Price ELSE 0
	Supervalley Price	Read Supervalley Price
	Consumption Discount	Read Consumption Discount
	Services	Read Services
	Read Power Price 1	Input from Docsumo
Docsumo	Read Discount Power 1	Input from Docsumo
	Read Price Power 2	Input from Docsumo
	Read Price Power 3	Input from Docsumo
	Read Peak Price	Input from Docsumo
	Read Valley Price	Input from Docsumo
	Read Supervalley Price	Input from Docsumo
	Read Consumption Discount	Input from Docsumo
	Read Services	Input from Docsumo
	Start date	
Total	Year Price	(Power 1*Price Power 1*(1-Discount Power 1)*Period+Power 2*Price Power 2*Period+Power 3*Price Power 3*Period+ +PeakConsumption*PeakPrice*(1-Consumption Discount)+ValleyConsumption*ValleyPrice*(1-ConsumptionDiscount)+ +SupervalleyConsumption*SupervalleyPrice*(1-ConsumptionDiscount))*(1+ElectricityTax (DDBB))* *(CounterRenting(DDBB)*Period*(1+IVA(DDBB))+(Services/30)*Period
	CO2	(PeakConsumption+ValleyConsumption+SupervalleyConsumption)*CompanyCO2(DDBB)
	Wastes	(PeakConsumption+ValleyConsumption+SupervalleyConsumption)*CompanyWastes(DDBB)
Proposal	Proposed Toll	2.0 DHA
	Proposed Power 1	Current Power 1
	Proposed Power 2	0
	Proposed Power 3	0
	Proposed Company	Gans Energía OR Indexada (PVPC, Manually)
	PricePower 1	Search Price in DDBB
	PricePower 2	Search Price in DDBB
	PricePower 3	Search Price in DDBB
	Price Peak	Search Price in DDBB
	PriceValley	Search Price in DDBB
	PriceSupervalley	Search Price in DDBB
	PriceServices	IF(Indexada) -> 4,5 ELSE 0
	Annual Proposal	(Power 1*Price Power 1*(1-Discount Power 1)*Period+Power 2*Price Power 2*Period+Power 3*Price Power 3*Period+ +PeakConsumption*PeakPrice*(1-Consumption Discount)+ValleyConsumption*ValleyPrice*(1-ConsumptionDiscount)+ +SupervalleyConsumption*SupervalleyPrice*(1-ConsumptionDiscount))*(1+ElectricityTax (DDBB))* *(CounterRenting(DDBB)*Period*(1+IVA(DDBB))+(Services/30)*Period
	CO2 Proposal	(PeakConsumption+ValleyConsumption+SupervalleyConsumption)*CompanyCO2(DDBB)
Wastes Proposal	(PeakConsumption+ValleyConsumption+SupervalleyConsumption)*CompanyWastes(DDBB)	
Savings	Power 1 Savings	1-ProposedPower1Price/CurrentPower1Price
	Power 2 Savings	1-ProposedPower2Price/CurrentPower2Price
	Power 3 Savings	1-ProposedPower3Price/CurrentPower3Price
	Peak Savings	1-ProposedPeakPrice/CurrentPeakPrice
	Valley Savings	1-ProposedValleyPrice/CurrentValleyPrice
	Supervalley Savings	1-ProposedSupervalleyPrice/CurrentSupervalleyPrice
Total Savings	Annual Savings	Year Price - Annual Proposal
	CO2 Savings	CO2 - CO2 proposal
	Wastes Savings	Wastes - Wastes proposal

Table 11. Formulas of the calculator spreadsheet



Block	Column	Value
Control Code	Control Code	0 -> Only Registration Date Exists
		1 -> Rest of Automatic cases
		2 -> Subscription Date Doesn't Exist BUT Proposal Date Exists
		3 -> Subscription Date Exists
		4 -> Manual Introduction
		6 -> Manual Introduction
Client	Name	From Calculator Spreadsheet
	Surname	From Calculator Spreadsheet
	DNI	From Calculator Spreadsheet
Contact Data	Email	From Calculator Spreadsheet
Intranet	Password	User Password
Address	Road	From Calculator Spreadsheet
	ZIP Code	From Calculator Spreadsheet
	City	From Calculator Spreadsheet
	Province	From Calculator Spreadsheet
Owner	Name	From Calculator Spreadsheet
	DNI	From Calculator Spreadsheet
Electricity	CUPS	From Calculator Spreadsheet
	Proposed Toll	From Calculator Spreadsheet
Bank Account	CUPS	Written Manually by coworker
	IBAN	Written Manually by coworker
State	Registration Date	Automatic, written from Integromat
	Bill Date	Automatic, written from Integromat
	Proposal Date	Written Manually by coworker
	Second Proposal Date	Written Manually by coworker
	Calling Date	Written Manually by coworker
	Subscription Date	Written Manually by coworker
	Intranet Subscription Date	Automatic, written from Integromat
Removed Date	Written Manually by coworker	
Marketing	Source	Written Manually by coworker

Table 12. Formulas of the client's spreadsheet

Block	Column	Value
Control Code	Control Code	Defined in client's page.
Client	Name	Data from Calculator spreadsheet
	Surname	Data from Calculator spreadsheet
	Address	Data from Calculator spreadsheet
Company	Proposed Company	Data from Calculator spreadsheet
Comission	Anual comission	Mensual Comission *12
	Mensual comission	IF(Proposed Company = PVPC) -> 0 IF(Proposed Company != PVPC) -> 2
Current	Anual cost	Data from Calculator spreadsheet
	Mensual cost	Anual cost / 12
Proposal	Anual proposal	Data from Calculator spreadsheet
	Mensual proposal	(Anual Comission + Annual cost)/12
Savings	Anual savings	Anual cost - Annual comission - Annual proposal
	Mensual savings	Mensual cost - Mensual comission - Mensual proposal
Co2	Co2	Data from Calculator spreadsheet

Table 13. Formulas of the proposal's spreadsheet



10. Annex F

```
{
  "data": {
    "Section 1": {
      "CUPS": {
        "model": "",
        "orig_value": "",
        "position": [ 535, 2664, 778, 2693],
        "value": "ES 0000 0000 0000 0000 AA"
      },
      "Compañia": {
        "model": "",
        "orig_value": "",
        "position": [],
        "value": "Iberdrola"
      },
      "Consumo pico": {
        "model": "",
        "orig_value": "",
        "position": [990, 3598, 1064, 3617],
        "value": 94.0
      },
      "Consumo supervalle": {
        "model": "",
        "orig_value": "",
        "position": [],
        "value": 0.0
      },
      "Consumo valle": {
        "model": "",
        "orig_value": "",
        "position": [989, 3622, 1063, 3642],
        "value": 91.0
      },
      "Descuento consumo": {
        "model": "",
        "orig_value": "",
        "position": [546, 3079, 673, 3110],
        "value": "15%"
      },
      "Descuento potencia": {
        "model": "",
        "orig_value": "",
        "position": [],
        "value": ""
      },
      "Dirección de suministro": {
        "model": "",
        "orig_value": "",
        "position": [1169, 763, 1584, 811],
        "value": "ETSIT, Camí de Vera s/n"
      },
      "Fecha final": {
        "model": "",
        "orig_value": "",
        "position": [638, 3596, 746, 3617],
        "value": "02/09/2019"
      },
      "Fecha inicio": {
        "model": "",

```




```
"orig_value": "",
"position": [330, 3597, 447, 3618],
"value": "04/08/2019"
},
"Importe factura": {
  "model": "",
  "orig_value": "",
  "position": [604, 758, 713, 801 ],
  "value": 50.97
},
"Localidad": {
  "model": "",
  "orig_value": "",
  "position": [1062.7, 555.9, 1216.1, 595.3],
  "value": "Valencia"
},
"NIF": {
  "model": "",
  "orig_value": "",
  "position": [164, 663, 259, 694],
  "value": "00000000x"
},
"PVPC": {
  "model": "",
  "orig_value": "",
  "position": [],
  "value": "false"
},
"Peaje": {
  "model": "",
  "orig_value": "",
  "position": [1166, 2579, 1240, 2609 ],
  "value": "2.ODHA"
},
"Potencia 1": {
  "model": "",
  "orig_value": "",
  "position": [1080, 2552, 1159, 2582],
  "value": "3,45 kW"
},
"Potencia 2": {
  "model": "",
  "orig_value": "",
  "position": [],
  "value": 0.0
},
"Potencia 3": {
  "model": "",
  "orig_value": "",
  "position": [],
  "value": 0.0
},
"Precio pico": {
  "model": "",
  "orig_value": "",
  "position": [
    645.8,
    3049.8,
    747.4,
    3072.7
  ],
  "value": "0,154877"
```



```
    },
    "Precio potencia": {
      "model": "",
      "orig_value": "",
      "position": [739.1, 3020.5, 838.7, 3039.1],
      "value": "0,123288"
    },
    "Precio supervalle": {
      "model": "",
      "orig_value": "",
      "position": [],
      "value": "0"
    },
    "Precio valle": {
      "model": "",
      "orig_value": "",
      "position": [],
      "value": "0"
    },
    "Provincia": {
      "model": "",
      "orig_value": "",
      "position": [1058.5, 553.9, 1228.6, 601.6 ],
      "value": "Valencia"
    },
    "Servicios": {
      "model": "",
      "orig_value": "",
      "position": [1031, 3311, 1085, 3334 ],
      "value": "3,56 €"
    },
    "Titular": {
      "model": "",
      "orig_value": "",
      "position": [928, 450, 1084, 490],
      "value": "Test One"
    },
    "ZIP Code": {
      "model": "",
      "orig_value": "",
      "position": [932, 553.9, 1052.3, 597.4],
      "value": "46022,"
    }
  }
}
},
"error": "",
"error_code": "",
"message": "",
"meta_data": {
  "doc_meta_data": "",
  "status": "processed",
  "title": "Factura Electricidad TFM.pdf",
  "user_doc_id": null,
  "user_id": "5e68cb567475e7ee50ab5f34"
},
"status": "success",
"status_code": 200
}
```



11. Annex G

```
#!/bin/bash
```

```
#CONFIG
```

```
dockerServer= SERVERNAME
```

```
dockerServerNamespace= SERVERNAMESPACE
```

```
#To use docker push, previous login is required
```

```
#docker login SERVERNAME/SERVERNAMESPACE -u nologin -p PASSWORD
```

```
if [ -z "$1" ] || [ -z "$2" ]
```

```
then
```

```
    echo "Missing arguments. Usage: ./docker-upload.sh [image name] [tag]"
```

```
    exit -1
```

```
fi
```

```
docker build -t $dockerServerNamespace/$1:$2 -f Dockerfile .
```

```
docker tag $dockerServerNamespace/$1:$2 $dockerServer/$dockerServerNamespace/$1:$2
```

```
docker push $dockerServer/$dockerServerNamespace/$1:$2
```

12. Annex H

The bill registry is a Google Sheet's spreadsheet created as a helpful tool for processing data during the bill's uploading process in the database tables, consisting on two pages, one for Bill Details, and another one for Consumptions.

This file helps with the following tasks:

- Provides a proper timing to the process, using flags, useful when checking in which part of the process we are involved.
- Processes information, adapting it to the database format.
- Generates values automatically, like the electricity provider or title of the bill.

The following columns are included in the registry.

	Electricity Bill		Consumption	
	Column	Value	Column	Value
A	Bill Id	Upper cell + 1	Consumption ID	Upper cell + 1
B	Bill Name	From Integromat	Used Consumption?	From Integromat
C	Shareable link	Shared link	Start date	Processed U
D	OCR ID	From Integromat	End date	Processed V
E	Username	From OCR	Days	D-C
F	CUPS	Processed V	Peak consumption	Processed O
G	Company	Processed O	Valley consumption	Processed P
H	House ID	From Integromat	Supervalley consumption	Processed Q
I	Consumption ID	Consumption A	Total consumption	F+G+H
J	Title	Parametrized	Co2 Waste	Processed V
K	Start date	Processed P	Radio Waste	Processed W
L	End date	Processed Q	Power 1	Processed R
M	Amount	Processed R	Power 2	Processed S
N	Service	Processed S	Power 3	Processed T
O	Company	From OCR	Peak consumption	From OCR
P	Start date	From OCR	Valley consumption	From OCR
Q	End date	From OCR	Supervalley consumption	From OCR
R	Amount	From OCR	Power 1	From OCR
S	Service	From OCR	Power 2	From OCR
T	Correctly Processed	FLAG	Power 3	From OCR
U	Onerz?	True -> our company origin	Start date	From OCR
V	CUPS	From OCR	End date	From OCR
W	Dropbox	Storage route in Dropbox	Co2 wASTE	0,26*I
X			Radio Waste	0,51*I
Y			Flag	From Integromat
Z			Processed correctly	From Integromat

Table 14. Formulas of the proposal's spreadsheet