DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

CO2N: Let's convert!

CO2N

2020 / 2021

1200263 - Juan Vallejo Domínguez

Instituto Superior de Engenharia do Porto

CO2N: Let’s convert!

CO2N

2020 / 2021

1200263 – Juan Vallejo Domínguez

DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

Licenciatura em Engenharia Informática

July 2021

Orientator ISEP: **Ricardo Almeida**

External Supervisor: **Rigo Belpaire**

To my family and friends

for making the world a better place

Acknowledgements

I would like to thank the CO2N team, with whom I have been working and developing this project for four months and have done an amazing job.

I would also like to thank Ricardo Almeida, Nuno Escudeiro and everyone involved in the BlendED AIM project for allowing me to live this great experience and meet so many amazing people.

Thank you to my family and friends who have always stayed by my side and supported me.

Summary

The world is in danger. Carbon dioxide emissions have increased exponentially since the industrial revolution [0.0] affecting the environment and causing unprecedent temperature records and natural disasters. CO2N is focused on reducing carbon dioxide emissions and achieving a CO2 neutral way of working and living by 2050.

In this project a web application that helps individuals, families, organizations, and companies reduce and offset their CO2 emissions has been developed. A multidisciplinary team composed by seven international students has joined under the context of the BlendEd AIM project to develop a web application, marketing plan and design book for the startup CO2N (CO2 Neutral).

The main approach to achieve these goals consists of five steps. Calculating the users carbon footprint. Offsetting the footprint by contributing with environmentally friendly projects. Reducing the carbon footprint by making changes in daily activities. Receiving feedback on how these changes are being effective and how the user's contribution is really helping. Sharing the progress that the user has made with its environment. Additionally, creating a carbon neutral and energy efficient website and a great community that supports this transition.

The developed system allows humanity achieve sustainability by using technology as the main tool. Nowadays there are massive amounts of tools and technologies that could help to achieve a better future for mankind, all that is required is for companies and individuals to work together and develop solutions that bring closer a better future.

Key words: Carbon Dioxide, Sustainability, Environmental Activism, Carbon Footprint, Carbon Neutrality, Web Application Development, MEAN stack, Interdisciplinary Teamwork, Agile Software Development.

Index

Index of figures

Index of tables

Notation and Glossary

API	Application Programming Interface. Combination of methods and functions that provide a library for other software to use offering a layer of abstraction.
CO2	Carbon Dioxide. Gas molecule.
CO2N	CO2 Neutrality. Name of the company.
CPU	Processor. Central processing unit. Electronic circuitry that executes instructions comprising a computer program.
DTO	Data Transfer Object. Object that transports data between processes, normally through a shared network like the internet.
GUI	Graphic User Interface. User interface that allows users to interact with electronic devices through graphical icons and figures.
IT	Information Technology. The use of computers to create, process, store and exchange all kinds of electronic data and information.
JWT	Json Web Token. Implementation of the industry standard RFC 7519 that defines a compact and self-contained way for securely transmitting information between parties as a JSON object. In this project they are used to authorize a user throughout all the requests he makes while signed in.
OID	Object Identifier. Unique character string that identifies an object in an environment. Usually used to identify objects in a database but can be used in various contexts.
OS	Operating System. System software used to manage computer hardware, software resources and provides common services for computer programs.
REST	Representational State Transfer. A RESTful API is an API that work as a server and responds to client requests, it is stateless, cacheable, uniform and is a layered system [0].
UN	United Nations. Intergovernmental organization that aims to maintain international peace and security. Created in 1945 composed by 193 states.

URI	Uniform Resource Identifier. Unique string that identifies a resource inside a network. They can be URLs (uniform resource locator), URNs (uniform resource name) or both.
WS	Web service. Server running on a computer device, listening for requests at a particular port over a network, serving web documents.

1 Introduction

1.1 Frame/Context

Nowadays everyone is aware or has heard about the climate crisis in which we are immerse. CO2 levels are still at record highs. Billions of tons of CO2 are released into the atmosphere every year as a result of coal, oil and gas burning. Human activity is producing greenhouse gas emissions at a record high, with no signs of slowing down [0.0,2].

As consequence, the last four years were the hottest on record, glaciers and ice sheets in polar and mountain regions are already melting faster than ever, causing sea levels to rise. If no action is taken, entire districts like New York, Shanghai, Abu Dhabi, Osaka, Rio de Janeiro and many other cities could find themselves underwater within our lifetimes, displacing millions of people and causing one of the biggest humanitarian crisis the world has yet to see [1].

Under this context hundreds of organizations all over the world have appeared to fight climate change and establish a new way of living, working, and producing.

The United Nations has called it "the world's most urgent mission" and set a deadline for a carbon neutral economy by 2050 (UN, 2020) [1].

Under this context, CO2N is born. It is a startup that focuses on supporting the transition of individuals, organizations, and companies towards a CO2 neutral way of living and working. This is a pressing issue as to address climate change concerns and contributes to the European Union’s decision to have a CO2 neutral economy by 2050.

Given this situation, the author of the document decided to contribute to the cause by developing a web application that would help achieving CO2N’s goals and by doing so, helping to build a better future.

1.2 Description of the problem

Carbon dioxide emissions continue to pollute the atmosphere, making climate change the world’s most urgent threat in the last centuries. Different initiatives and movements have appeared with the goal of diminishing these emissions and offsetting the remaining ones but not many of them have a transparent way of working and acting. It is impossible for an individual to really know what has been done with the money that has been used to offset if there is no feedback of this investment.

1.2.1 Goals

CO2N is a non-profitable company that seeks diminishing the CO2 emissions around the world to reach a carbon neutral way of living and working by 2050, as intended by the United Nations [1].

Its goal is to reach as many companies and individuals as possible so they can all work together and have a greater impact on the climate crisis.

The goal of CO2N is to help individuals, organizations and companies reduce and offset their CO2 emissions.

In order to reach this carbon neutrality goal, 5 steps will be necessary:

1. Calculate: Calculating a user’s carbon footprint of the previous year.
2. Compensate: Offset (fully or partially) your carbon footprint by investing in sustainable projects like tree planting, installing solar panels or cleaning the ocean.
3. Reduce: Giving users personalized tips, advice, and courses so they can learn how to diminish their carbon footprint for next year’s calculation.
4. Progress: Give users feedback on the CO2 you have reduced along the years and how you have reduced it given your carbon footprints. Also give clear and real time feedback on how the compensation investment is going and how much CO2 emissions has been reduced or avoided.
5. Share: Sharing what the user has accomplished with friends, family, or customers through the CO2N label. In case of companies and organizations, sponsor their CO2 neutral products.

Another extra goals would to be reach as many companies, organizations, and individuals as possible and to create a community that feels like it is really helping to achieve this transition.

The web application will, indeed provide a platform that supports the realization of these five steps.

1.2.2 Approach

The project will be developed by using an Agile methodology: SCRUM. Following the organization imparted by IMEC in the SCRUM workshop during the kickoff week [3].

There will be three teams that will work concurrently: Marketing team (one member), Design team (two members) and IT team (four members). From here onward the report will refer to the IT team, it’s progress and organization.

Sprints will have the length of one or two weeks, depending on the complexity of the tasks to be done. All the teams will meet at the begging of every week to show the progress they have made in the previous sprint and plan the following sprint. Progress will periodically be discussed with the company and the teachers involved in the project.

The first sprints are dedicated to discussing the requirements and plan the way of working within the team, as well as to prepare the workspaces and communication channels.

The following sprints will be used to make the analysis of the given requirements and decide upon the technologies and tools to use, as well as to design the first data models that the website will require to have and the systems architecture.

From here onward, every sprint will have its own design, implementation, and test phase for each feature that will be developed.

On the final weeks of the project the team will prepare the product to deliver to the company and make the final testing and adjustments.

1.2.3 Contribution

CO2N gives another insight to climate action and CO2 reducing. Anyone with access to the internet will be able to calculate their carbon footprint and offset the CO2 emitted with only a couple of steps.

It allows to unite companies and individuals into working together, using the same platform to achieve a better world. Most companies that are already developed focus on a single target group such as companies or individuals. In the case of CO2N the aim is to unite all the society into achieving one complicated but necessary goal, achieving total carbon neutrality.

Another issue is that nowadays most initiatives that have been developed with the same or similar goals do not give users clear feedback about where their investments are going. For example: A user can pay to offset the CO2 that their flight will emit into the atmosphere, but they do not know where that money is really heading.

This supposes a problem to the user and makes it harder for them to trust a website, company, or organization. The lack of transparency is the main problem and this may lead a user to fall into the scam of “Greenwashing”. It’s a marketing spin used by companies to deceive their clients making them believe that the company and/or its products are environmentally friendly [4].

CO2N is based on transparency, so it provides clear feedback to the users with real life images and/or videos and with concrete values of how much CO2 has been reduced/avoided and what percentage is thanks to their contribution.

Society will be more sustainable and environments worldwide will be healthier, leaving a better future for generations to come.

1.2.4 Planning of the project

The development of the project follows the dates provided by the BlendED AIM program in which it’s included. These dates are:

- **Kickoff Week: 22nd to 26th of February 2021.**
  The purpose of the week is to build the team and introduce the project to the members.
  In this week, the team will meet the rest of the members and the founders of the company. They will also take part in cultural awareness and team dynamics sessions and attended a SCRUM workshop.
  The project will be explained by the company and the first ideas will be exposed and discussed.
- **Project development stage 1: 27th of February to 18th of April 2021.**
  In this phase the team will start developing the project and organizing themselves autonomously.
- **Midterm Week: 19th to 23rd of April 2021.**
  The purpose of the week is to show the company the progress until the date.
  A workshop dedicated to team building and group dynamics will also be imparted following the Merging Minds methodology.
- **Project development stage 2: 24th of April to 20th of June 2021.**
  The team will continue to work on developing the project, continuing with the self-organization.
- **Final Week: 21st to 23rd of June 2021.**
  The purpose of the last week is to present the results to the company. The team will build the final pitch and the company/tutors will give feedback and grade the team.
- **Submission deadline: 12 September 2021.**
  From the final week onward, the main goal is preparing the documentation and presentation for the defense at ISEP in front of a jury.

Relative to the project features development, the steps and dates will be the following. They can be resumed into the following phases:

1. **Requirements: February to March**

   The website's features will be discussed within the team and the company, defining realistic objectives. Functional and non-functional requirements will be established.
2. **Analysis: March**

   Given the requirements, a first approach of the solution and the technologies to use will be designed. Analyzing the data the website will have to manage and the way it would do so.
3. **Sprint 1: First week of April**

   Skeleton of the web applications development, repositories creation and initializing, layers integration.

   The different workspaces will be created and initialized in order that any member of the team could work using the same tools over the same platform and having the latest version of the project. The structure of the app will be designed, implemented, and tested.
4. **Sprint 2: Second week of April**

   Designing, implementing, and testing features related to users' account.
5. **Sprint 3: Third and Fourth week of April**

   Designing, implementing, and testing features related to the carbon footprint calculator.
6. **Sprint 4: First and Second week of May**

   Designing, implementing, and testing features related to the compensation step and projects investment.
7. **Sprint 5: Third and Fourth week of May**

   Designing, implementing, and testing features related to the reduction step like tips and advice and marketplace.
8. **Sprint 6: First week of June**

   Designing, implementing, and testing features related to the feedback step such as chart creation and compensation previewing.
9. **Sprint 7: Second and Third week of June**

   Final integration, adapting frontend to final design, testing and final pitch preparation.

1.3 Structure of the report

The structure of the report will follow the software engineering process that is applied.

On the second chapter the state of art is described, and related projects have been compared and analyzed.

On the third chapter the analysis and design phases of the engineering process are explained. The requirements are listed as well as the analysis made, and decisions taken to choose the different technologies. The architectural design and most complex components are explained. The data models and use cases are also explained and presented in this chapter.

On the fourth chapter the implementation of the solution is described. The important decisions that the team took and the different ways of collaborating and working together are explained. Testing and validation of the solution are also explained in this chapter.

2 State of art

2.1 Related projects

To the date there are several companies that work on the same goals as CO2N. A lot of them allow calculating a user's carbon footprint using a survey. A lot of them allow users to offset their emissions but practically none of them can prove what the money is used for.

CO2N will allow users to choose where the money they invest will be used, by selecting between a list of projects. Also, one can verify that this money has been used to really offset the carbon emissions since project related feedback will be handed over to every user.

The different companies working towards the same goals have been analyzed and compared on the table 1 below.

Table 1 Related projects comparison

Company	Calculator	Offset	Projects	Guidance	Feedback
CO2 Logic	YES	YES	YES	NO	NO
Greenfish	NO	NO	NO	NO	NO
Climate Neutral Group	YES	YES	YES	NO	NO
Encon	YES	YES	YES	YES	NO
Milieu Centraal	NO	NO	NO	YES	NO
ClimateCare	YES	YES	YES	NO	NO
Eliza was here	NO	NO	NO	YES	NO
Be Climate	YES	YES	YES	NO	NO
Climate Neutral	YES	YES	NO	NO	NO
BP Target Neutral	YES	YES	NO	NO	NO
Global Footprint Network	YES	NO	NO	NO	NO

Guidance refers to if the company gives tips and advice to users to reduce their footprint.

All of them work towards the same goals and have different approaches. For example, *"Eliza Was Here"* is centered on vacation homes and hotels and it calculates the CO2 emissions of your vacations [20]. This is a very specific approach that is very useful when you plan your vacations but seems insufficient during the rest of the year. "*Be Climate"* on the other hand focuses on climate neutral fruits and vegetables and cultivates, transports, packages and delivers them, controlling every process in the chain and achieving climate neutrality with their products [21]. This is also a very specific approach but could be more widespread across the fooding industry. Even though these are interesting and useful ideas, they still won't be enough to reach CO2 neutrality in the remaining aspects of a person's daily life.

Other companies already have successful and popular websites that follow the same, or very similar steps as CO2N like "*Climate Neutral"* which measures, offsets, reduces and certifies other companies. It also helps visitors to find climate neutral certified companies [22]. This is very similar to the company-oriented features of CO2N which enhances companies to act. Getting certified as carbon neutral can be a great plus for a company and appearing in a website that focuses on climate neutrality will attract a lot of conscious clients. Nevertheless, this website provides no feedback nor allows individuals and families to participate, making it less reliable and trustworthy and reducing significantly its target group.

Other multiple companies share the same methodology as *"Global Footprint Network"* [23] or *"Climate Care"* [24], they have an integrated calculator in their websites and after calculating, allow to offset investing in projects (Climate Care) or donating to the ONG (Global Footprint Network). Both these ideas are integrated in CO2N and are two of the five main steps.

Many of these websites have a section dedicated to environmental news and/or links to blogs of the company. This is used to create a community and raise users' awareness.

CO2N has also planned on creating a blog and social media accounts where this type of news will be posted and users can interact between themselves using their personal accounts. The marketing team will oversee this task.

The one thing all these companies have in common is that after offsetting, users lose track of where their money goes since there is no evidence that the money has reached the project they selected (in case of investing in a project).

CO2N provides a solution to this problem as well as adopting most of the features previously discussed.

A problem for CO2N is that it's a new company and in the beginning it will take time to gain peoples trust. Also, depending on the projects, CO2N will give feedback periodically. For example, in tree planting you cannot show feedback until the trees are planted and users should have to wait until the season reaches.

2.2 Existing technologies

The first technology to be chosen is the database in which the data was going to be stored. There is a need for a database that could be easily scalable since one of the goals is to reach the greatest number of individuals and companies as possible. The database must include flexible data structures since it will work with companies that operate in different sectors, thus a NOSQL database is required. It must also be easy to handle and manipulate data since the startup has, up to the date, no IT professionals capable of handling a non-user-friendly database. It is also important for CO2N that the database is well known and has big support and community. It is also interesting yet not necessary for the database to be free and open source since CO2N is a non-profit and community-oriented company.

Given these conditions, the most reliable and popular technologies were contemplated:

Table 2 Database comparison

Database	Scalability	Relational	User-Friendly	Open-Source
MySQL	Bad	Yes	No	Yes
PostgreSQL	Bad	Yes	No	Yes
MongoDB	Good	No	Yes	Yes
Oracle DB	Bad	Mixed*	No	No

*Oracle DB cloud services supports multi-model databases

The final decision was to use MongoDB since it satisfied all the requirements. It offered a clear and simple GUI in both local and cloud storage alternatives. It is flexible, scalable, distributed and has cloud storage systems in Belgium, city where the startup has been founded. This cloud storage system or MongoDB Atlas is fully automated and elastic and provides fault tolerance thanks to its self-healing clusters. This helps contribute to diminishing network costs and provides local storage and backup in the same country.[25]

MongoDB is a JSON oriented database where objects are stored in a key-value format. The GUI allows a standard user to search through the database thanks to a search bar and manually manipulate the content without further complexity. This is ideal in the current situation of the company since it requires no IT professional to manage the database.

It is an open-source technology that has official documentation and a great community. It has drivers for more than 10 programming languages.

As consequence, MongoDB was selected as the database to be used.

After selecting MongoDB, compatible technologies had to be chosen. For the application tier, Express and NodeJS was chosen and for the client tier or frontend, AngularJS was selected. All these together make the MEAN stack. A well-known and efficient stack of technologies that work very well together.

The main reasons for selecting these technologies were:

- Vast documentation, tutorials and community support the MEAN stack.
- Integration between them is very efficient and simple since they are all based on the same language, JavaScript.
- It is based on the Client-Server architecture, Node.js at the server’s side and Angular.js at the clients.
- Only JavaScript specialists are required once the project is handed out to the company. There is no need for CO2N to hire different specialist for the same project which makes it more cost-effective.
- Node.js is a very portable language since it runs in Linux, Windows and OS X operating systems. It uses non-blocking I/O calls to handle incoming requests which allows handling a great number of concurrent requests efficiently. Node.js is completely scalable.
- The use of JSON (JavaScript Object Notation) is spread across the whole stack, meaning that data interchanged through all layers uses JSON documents. This simplifies hugely interaction between the different tiers to develop.
- Angular provides real-time changes on the run and makes use of SPAs (Single-Page Applications).

Nevertheless, the MEAN stack has disadvantages as well. To fully take advantage of all the features JavaScript provides, great knowledge of this programing language is required. Most of the team will have to put a lot of effort to learn these technologies to contribute to the project. Angular also has a great learning curve so the team will have to struggle to learn this tool as well.

Other programing languages would be easier to learn and/or develop and other stacks would be required, nonetheless, the MEAN stack provided the most flexible, scalable, and non-structured community-oriented solution that the team could find.

Several ideas written in this section were extracted from the article in reference [26].

3 Analysis and Design of the solution

3.1 Domain of the Problem

The domain of the problem is climate change, emission of greenhouse gases, sustainability, web application development and team organization via SCRUM.

3.2 Functional and Non-Functional System Requirements

The functional requirements of the system are defined in the table 3.

Table 3 Functional Requirements

Code	Description
FR0	A user can create an account in the system
FR1	A user can delete his account from the system
FR2	A user can login to the system
FR3	A user and visitor can view information about our company and related topics
FR4	A visitor, individual and family can calculate his/her/their current CO2 footprint at any moment
FR5	An individual and family can update their current CO2 footprint once a year (by calculating or selecting average)
FR6	A company and other organizations can contact an employee to calculate their current CO2 footprint using their last years accountancy data
FR7	An individual, family, company and other organizations can receive general tips on how to diminish their CO2 emissions
FR8	An individual, family, company and other organizations can receive personal guidelines on how to diminish their CO2 emissions
FR9	An individual, family, company and other organizations can subscribe to monthly payment to a project that helps reduce CO2 emissions/concentration in the atmosphere to compensate their current carbon footprint.

FR10	An individual, family, company and other organizations can cancel their subscription(s) to a project at any moment
FR11	An individual, family, company and other organizations can donate once to a project that helps reduce CO2 emissions/concentration in the atmosphere
FR12	An individual, family, company and other organizations can get feedback about the projects they have invested in
FR13	An individual and family can view their progress in CO2 reduction with a simple dashboard
FR14	A company and other organizations can see their progress in CO2 reduction with a complex dashboard
FR15	A company and other organizations can share specific advice to lower CO2 emissions with their employees
FR16	A company and other organizations can share a pdf/webpage/report with information about their inversions and progress reducing their CO2 emissions
FR17	An individual, family. company and other organization can enroll in a virtual training course
FR18	Any user can view information about the projects
FR19	Any user can view information about the courses
FR20	Any user can view CO2N certified products in the marketplace
FR21	A company and other organizations can sponsor their CO2N certified products in the marketplace
FR22	Any user can view a list of CO2N certified companies
FR23	An individual and family can participate in a game related to climate change
FR24	An individual and family can share their progress on the game via social media

The non-functional requirements of the system are defined in the table 4.

Table 4 Non-Functional Requirements

Code	Description
NFR0	The website will be developed using open-source tools
NFR1	The sharable report must be visually attractive and graphical
NFR2	The specific advice for employees will be composed by a list of specific guidelines for the company
NFR3	The tips and advices shall be notified periodically to users
NFR4	The system will automatically suggest courses to users depending on their CO2 emissions
NFR5	The system will notify the clients via email when they have new feedback
NFR6	The system will be implemented using well known tools and technologies
NFR7	The website will be responsive for pc, laptop, and mobile devices
NFR8	The system will be able to validate a user by an email and a password
NFR9	The system will have documentation that explains how it was developed
NFR10	The system will be graphically attractive and appeal to the users to take part
NFR11	The system will calculate a user’s carbon footprint with a margin of error of ± 1 ton of CO2e
NFR12	The system will be light weight as to browser CPU consumption, leaving operations with major cost to the application’s backend server
NFR13	The system will work correctly with Mozilla Firefox and Google Chrome
NFR14	The system will be able to recover no matter the downtime
NFR15	Communication between the distinct elements that form the system will be made through HTTPS to assure privacy

All the requirements have been discussed within the team and with the company, reaching an agreement on the main features that the system would have.

3.3 Use cases

The use cases from which the requirements in the table 3 have been obtained are described in the following use case diagram in the figure 1

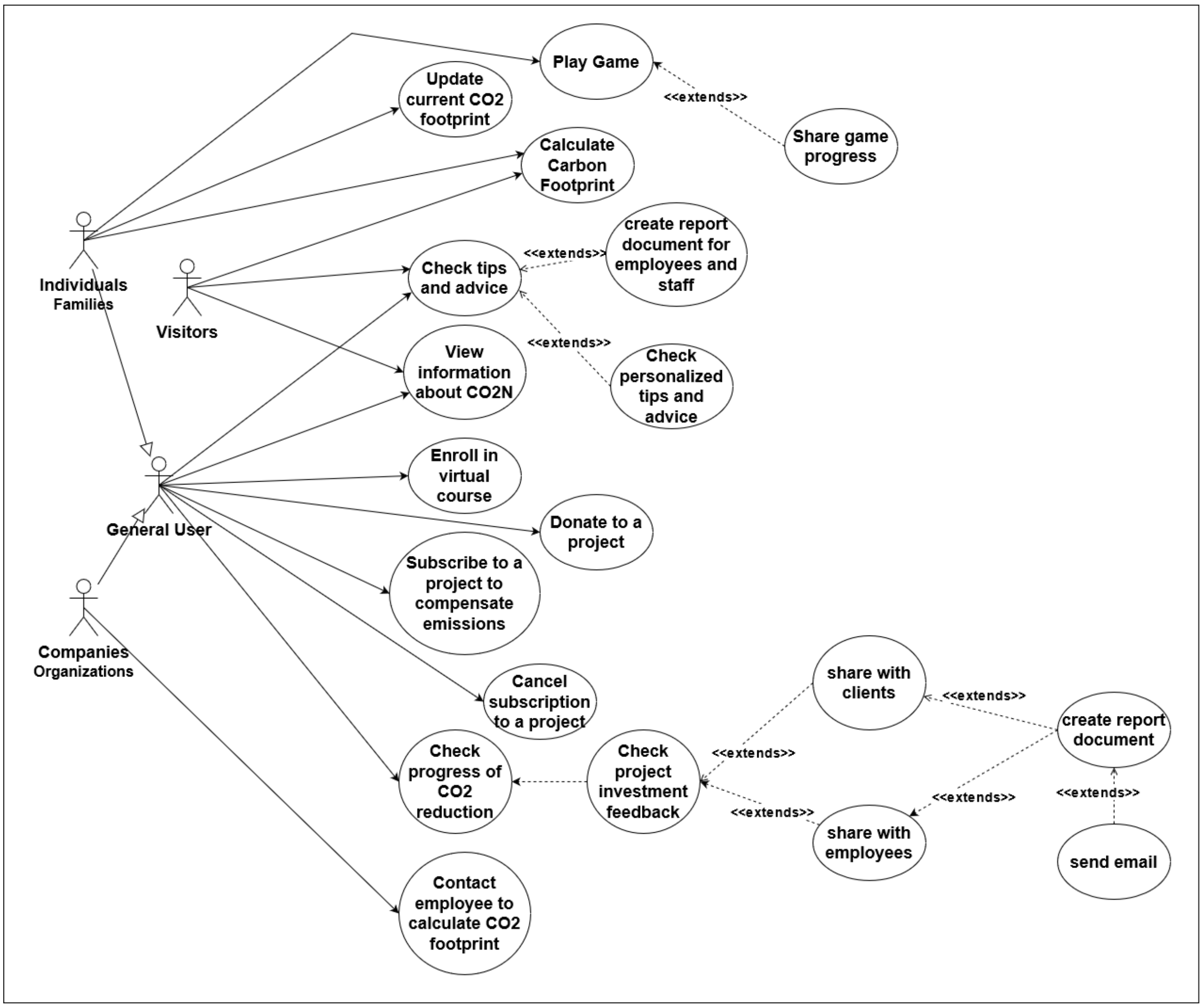

Play Game
<<extends>>
Share game progress
Update current CO2 footprint
Calculate Carbon Footprint
Individuals Families
Visitors
Check tips and advice
<<extends>>
create report document for employees and staff
<<extends>>
Check personalized tips and advice
View information about CO2N
General User
Enroll in virtual course
Donate to a project
Subscribe to a project to compensate emissions
Cancel subscription to a project
Companies Organizations
Check progress of CO2 reduction
Check project investment feedback
<<extends>>
share with clients
<<extends>>
share with employees
<<extends>>
<<extends>>
create report document
<<extends>>
send email
Contact employee to calculate CO2 footprint

Figure 1 Use case diagram (Juan 2021)

3.4 Domain model

The domain model has been designed in function of the necessary information that the system will handle. The main classes and methods are defined in the figure 2.

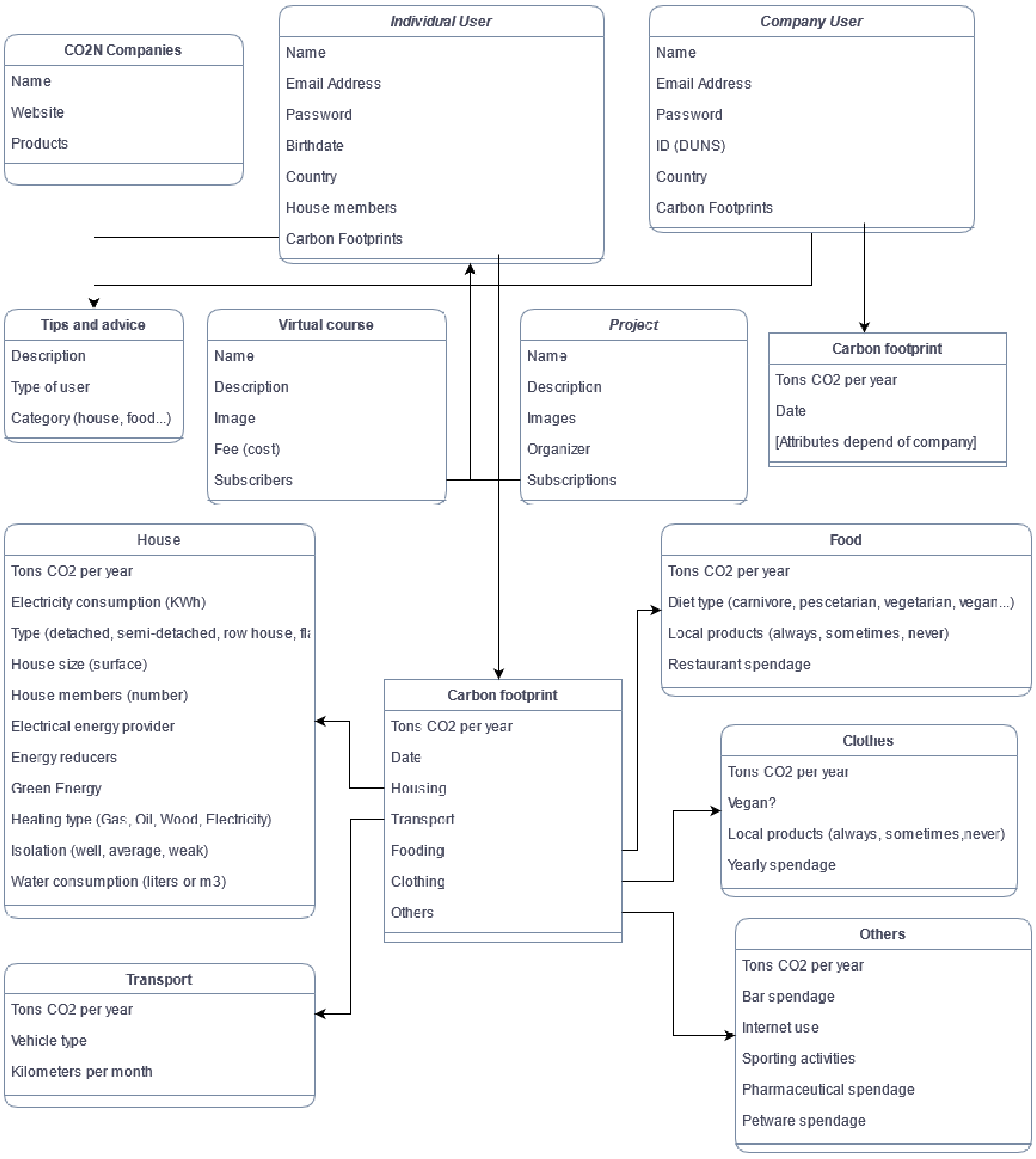


Figure 2 Domain model (Juan 2021)

There are two different types of users, individuals, and companies. Both will have a list of carbon footprints calculated over time. A company’s carbon footprint will depend on the activity it develops and the sector in which it is included. An individual’s carbon footprint will be composed by five main categories:

- Housing: It will store information about the house where the user lives in and its consumption.
- Transport: It will store information about the user’s way of transport, including personal vehicles, public transport and long trip vehicles like plains or trains.
- Fooding: It will store information about the user’s diet and meat consumption.
- Clothing: It will store information about the user’s way of dressing and shopping clothes.
- Others: It will store information about different aspects of a user’s life such as activities it participates in or pharmaceutical usage.

These five categories englobe human daily activity and try to quantify the CO2 they emit. The decision of dividing into these categories was the result of investigating, checking already existing calculators such as the one created by the Global Footprint Network which the team had as inspiration [30], and discussing within the team and company.

There are carbon neutral companies that will list they’re CO2N products on the marketplace.

Every user will have tips and advice depending on the carbon footprint they have. That’s why the category is important in the domain object.

Users will also be able to enroll in virtual courses, which will be imparted online by an organizer that can be another organization, individual or company.

Users will also be able to subscribe and/or donate to projects to offset the carbon footprint they calculated.

This domain model gathers all the main domain classes that the web service will need to give the desired service.

3.5 Design

The systems architecture consists of a three-tier web application developed using the MEAN stack technologies. The three tiers consist of the database tier, application tier (Backend) and client tier or GUI (Frontend). The three tiers are described graphically in figure 3.

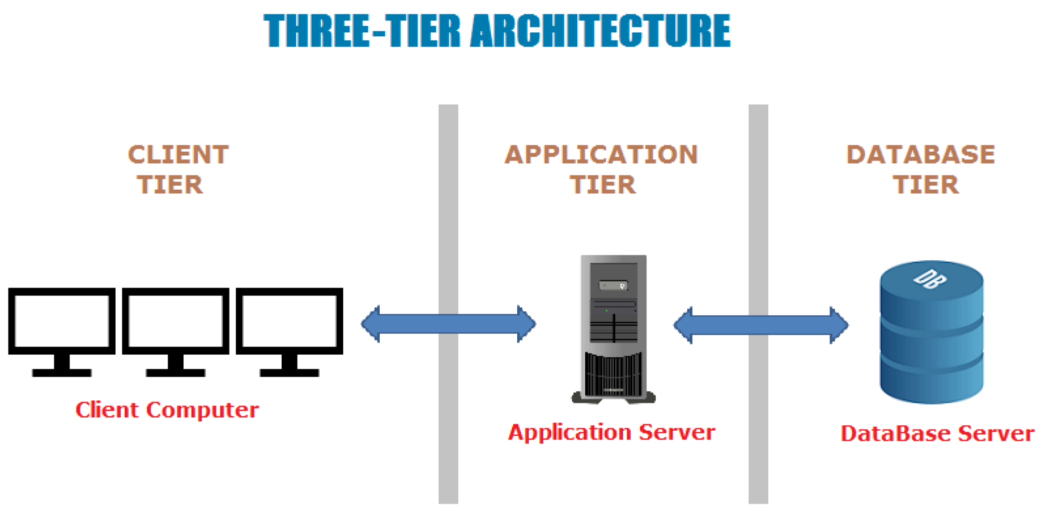


Figure 3 Three-tier architecture (softwaretestingmaterial.com 2016)

The layers follow a client-server architecture. The client tier is a client to the application tier, which would behave as a server. At the same time, the application tier works as a client to the database tier, which behaves as a server. The three layers communicate with each other using an API. These API's respond to HTTP requests over the internet. They are RESTful API's and follow its principles. Every necessary resource has been designed by the IT team and is documented in the Annex A.

There is one API that will serve the Backends' business operations allocated inside the application tier. Another API will be developed to access the database, abstracting the operations, and giving it an extra layer of security. This way, modifications to the database will only be possible through the API that will be created.

The APIs provide the CRUD (Create, Read, Update, Delete) operations for each domain model (object) available in the database which are necessary for the functionality of the application.

Thanks to this architecture the system can be easily scalable and portable and can be hosted in cloud hosting services. This allows the community to grow without greater hosting costs thanks to the great number of flexible cloud solutions that currently exist in the market.

The component diagram of the architecture is described in the figure 4. It describes the main components of the three tiers that compose the system and their interfaces.

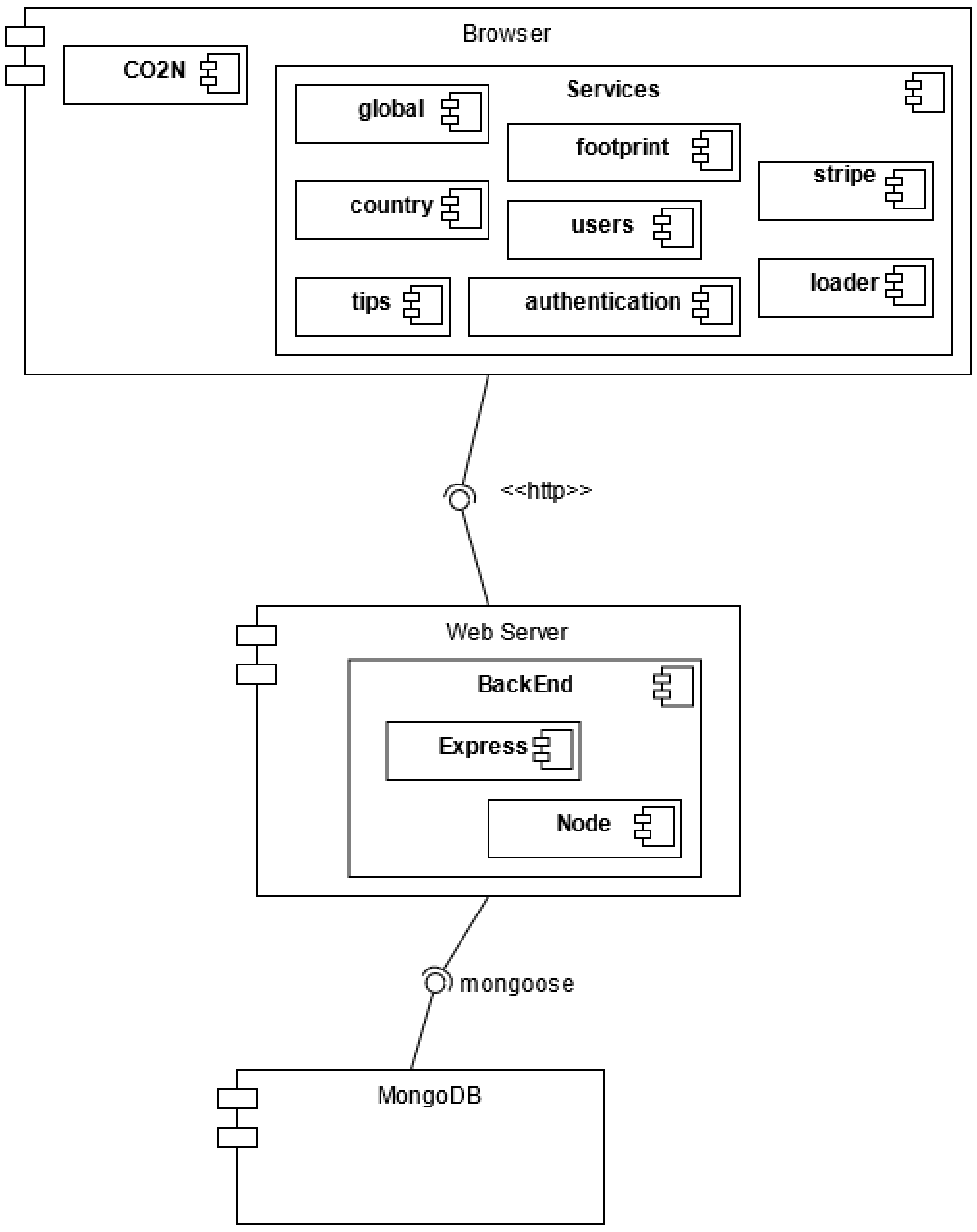


Figure 4 Component Diagram (Juan 2021)

Every tier will work following the MVC (Model-View-Controller) pattern. This design pattern is described in the figure 5.

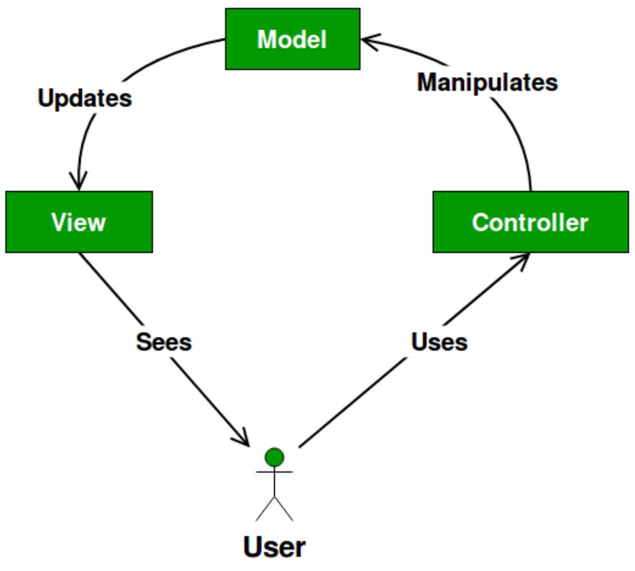


Figure 5 Model-View-Controller Design Pattern (geeksforgeeks.org, 2018)

It's based on three classes; the model is where the data is stored. Models contain the important information, they are data objects. The user interacts with the controller, which modifies and manipulates the model. A change in the model forces a change in the view, altering the result and what the user sees.

In the case of the second tier, the backend, there will be no view since it does not have a graphical interface. Instead, it will return a response such as a DTO (Data Transmission Object) when necessary or simply an HTTP response carrying a message and response code. It follows

A common interaction with the system is described in the figure 6.

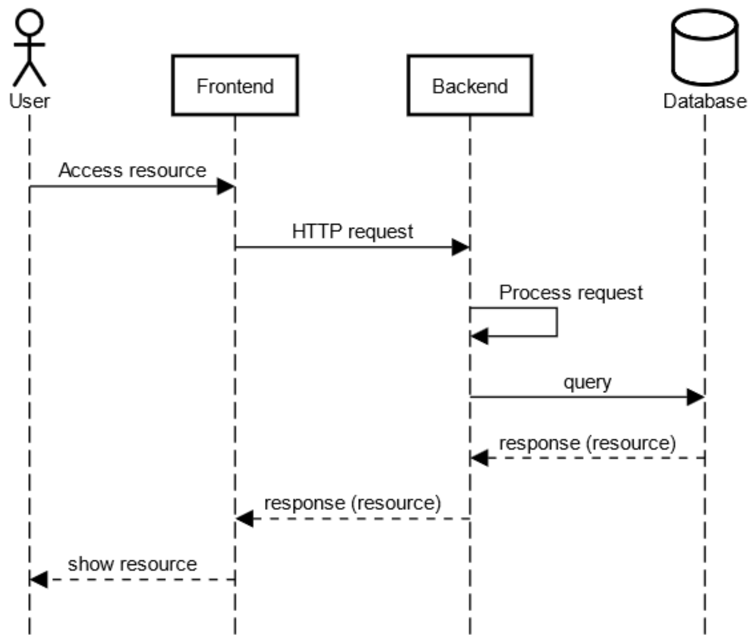


Figure 6 General Component Interaction (Juan 2021)

It can be seen in the figure that the user accesses a resource using its browser. This is done through the Frontend, which contains the GUI of the website. This generates an HTTP request to the backend, which will process the request and if it needs to retrieve something from the database, it will call the database using its API. The database will return a response to the backend that will be sent using a DTO onto the frontend that will show the desired resource to the user. This is assuming that everything works correctly throughout the stack.

A description of the frontend’s general way of working is described in the figure 7.

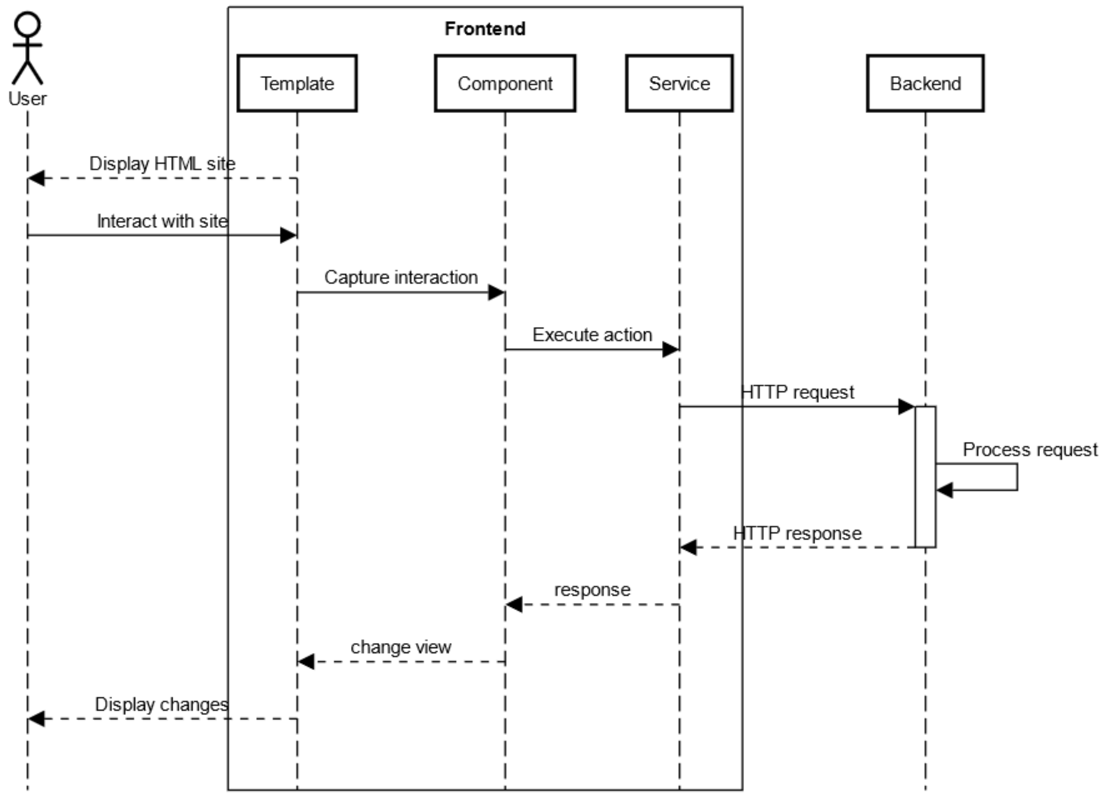


Figure 7 Frontend Component Interaction (Juan 2021)

We can see here that the service does not block itself when it receives an instruction to execute a method. This is because of Angular’s dependency injection that instantiates a service for the component instead of using a common shared one.

It calls the method and captures the response in a callback function that processes the response and modifies the components data, which changes the view.

This means that a user can make several calls to a service without blocking it, allowing the website to look and work fluently.

In the figure 8 we can see the internal interaction used in the backend tier.

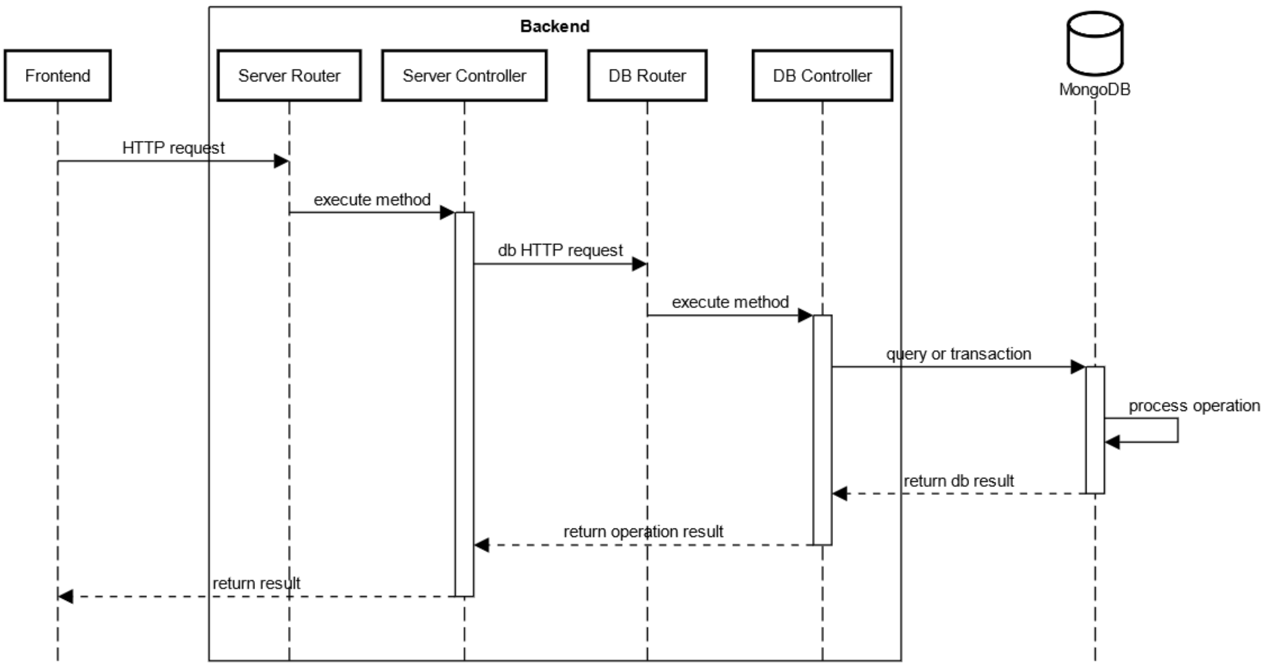


Figure 8 Backend Component Interaction (Juan 2021)

Here can be seen that the backend receives an http request from the frontend, if it matches with any of the defined routes declared in its API, it calls the method in the controller that makes the operations and processes the request. If it is necessary, it will create and call the database’s API which will do the same and redirect the new http request to its controller. This database controller will execute the necessary operations in the database fulfilling the request and returning the result to the controllers, which will then return the result to the backend, constructing a DTO if necessary.

The final data model follows the class diagram defined in the figure 9. It extends the domain model from the previous section of this report to make it compatible with MongoDB without losing any features.

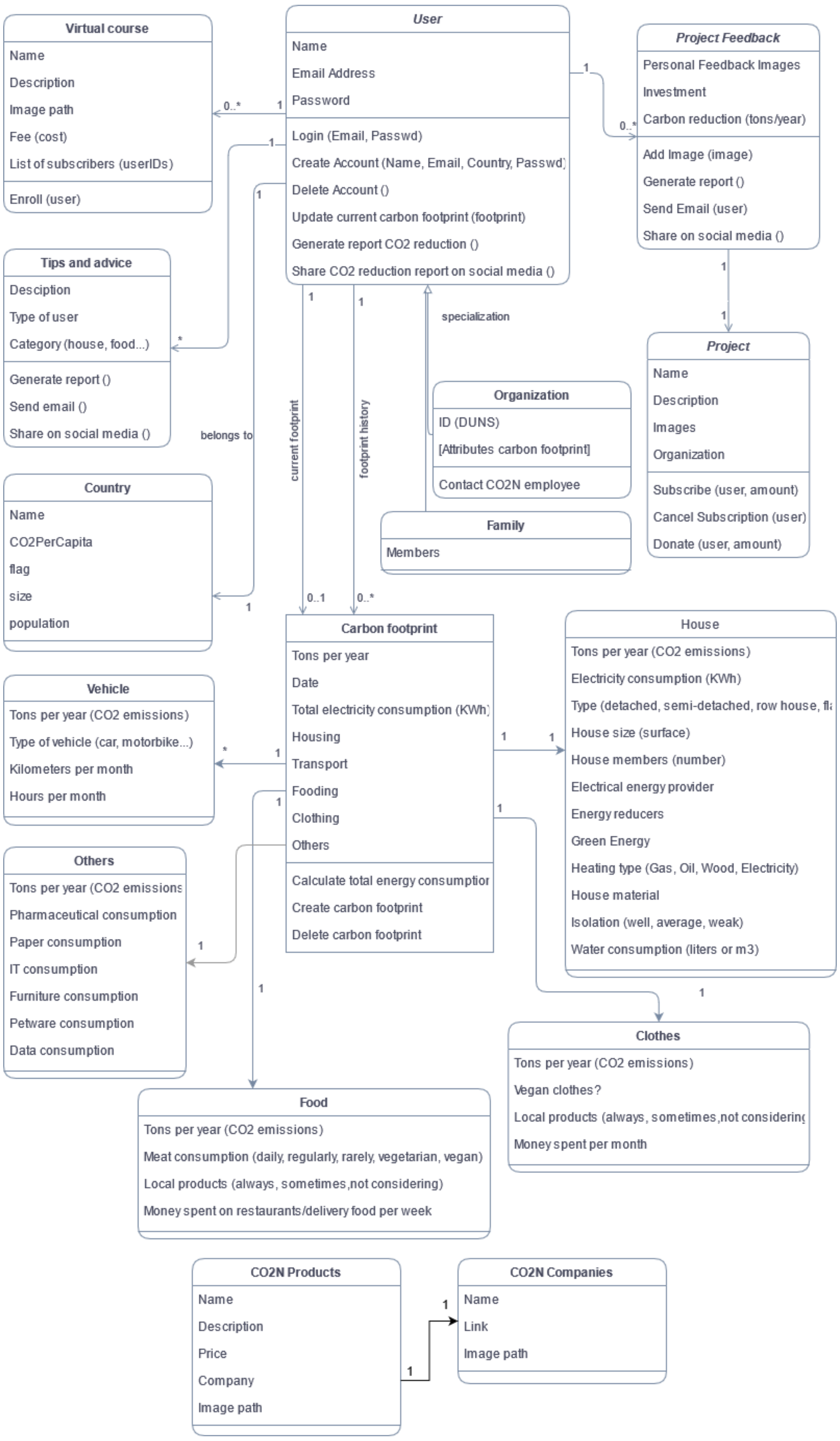


Figure 9 Class Diagram (Juan 2021)

A user would have different states depending on what step of the process he is. The different states are represented in figure 10.

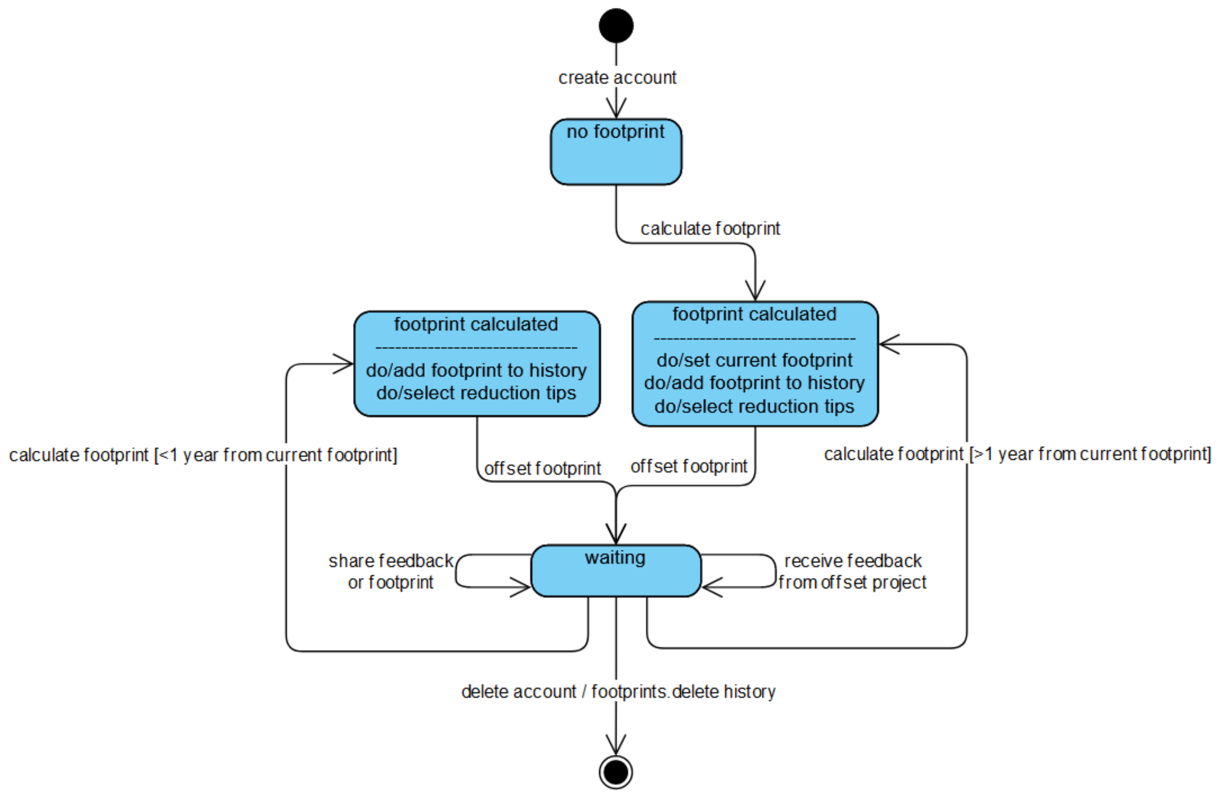

create account
no footprint
calculate footprint
footprint calculated
do/add footprint to history
do/select reduction tips
footprint calculated
do/set current footprint
do/add footprint to history
do/select reduction tips
calculate footprint [<1 year from current footprint]
offset footprint
offset footprint
calculate footprint [>1 year from current footprint]
share feedback or footprint
waiting
receive feedback from offset project
delete account / footprints.delete history

Figure 10 User State Diagram (Juan 2021)

Interaction between project subscriptions and projects is represented in the state diagram in figure 11.

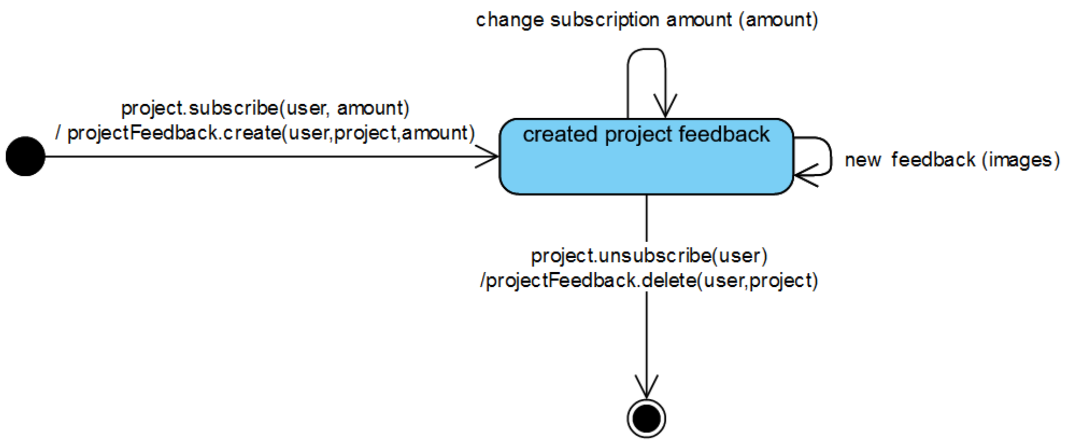

change subscription amount (amount)
project.subscribe(user, amount) / projectFeedback.create(user,project,amount)
created project feedback
new feedback (images)
project.unsubscribe(user) /projectFeedback.delete(user,project)

Figure 11 Project and Subscription state diagram (Juan 2021)

Another class that interacts with other classes is the virtual course class, which's functioning is described in the state diagram of figure 12.

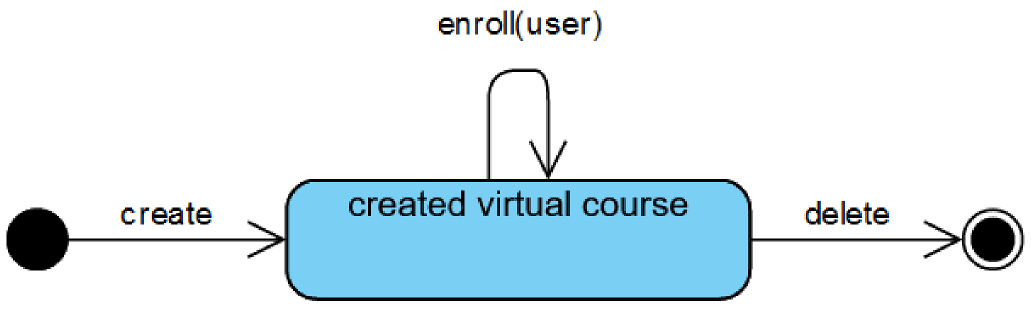


Figure 12 Virtual Course State Diagram (Juan 2021)

The remaining classes have a very simple way of operating and will be created and deleted depending on the users or administrators will.

An alternative design approach would be making the application a two-tier system. Being the first tier the client tier and the second the server or repository tier. This would centralize all operations in the second tier and the first one would only serve to display the graphical elements and send the interactions to the server. This architecture would have problems when it comes to scaling and becomes a more rigid system. The system requires a central database that stores sensitive information so other architectural models such as peer-to-peer aren't fit for this project.

The design of the Frontend has been delegated to the design team and will be developed by it over the three months that the project lasts. This approach leaves the IT team very little time to implement the final design as design is one of the first phases in the software engineering process but here both teams work concurrently. All the help that the design team will need will be given during the process and the designs will be periodically shared and discussed with the IT team and the company.

Concurrently the IT team will work to develop solutions that hopefully please the design team. Four different wireframes containing different ideas will be elaborated and discussed in a SCRUM meeting.

The design team will also focus on making a CO2 neutral design by minimizing images and heavy resources or colors that require more electricity. CSS animations have been designed and low weight images as well. This makes the frontends design more CO2 neutral and provides the company and website an environmentally friendly look.

The design teams work can be accessed through the brand book and documentation [31].

While the design team works on the design, the IT team will design several aspects of the frontend such as the MVC and client-server architecture already explained and the module-based architecture that allows lazy loading.

This modular approach will consist of dividing the frontends GUI into several different modules that work together but independently. In this case, a module will contain all the information necessary to display several views. This modular design is explained thoroughly in the implementation section of this same document.

4 Implementation of the solution

4.1 The process and environment

The implementation of the project has been developed between the 7 members of the team, working remotely and concurrently, using English as the communication language. As explained in the previous section of this report, the teams are divided into marketing with one member, design with two members and IT with four members. They all worked concurrently but not independently since interaction was necessary between the three teams to reach better solutions. Every week the team members met and a groupal team call with members from all the teams took place to explain their progress and difficulties. If it were necessary, help was asked for and given.

Communication tools used were the following:

- Discord: Platform where the team members would meet regularly for the SCRUM meetings and other doubts. Documentation and resources were also shared through different text channels created in the team’s server.
- Microsoft Teams: Platform used by the teachers to meet with the students and rest of teams for the programed activities.
- Google Meet: Platform used by the company to meet with the team.
- WhatsApp: Quick contact with anyone involved in the project.

The use of so many communication tools sometimes became a problem and created confusion over where to find certain elements. The communication should be centralized in one single platform to avoid these issues.

Other tools and platforms that were used to share resources are:

- Google Drive: All the documentation and important documents have been submitted to this platform. It’s the platform where the team has worked together to develop pitches and share resources as well.
- Trello: Team organization platform. In this platform the team planned the work to be done and prepared important meetings and dates, setting the backlog and updating with each sprint.

Trello has been the tool that allowed planning the three teams in a general way, nevertheless, GitHub has been the main tool for the IT team organization since it provides very similar features to Trello.

Programing tools and platforms used were:

- GitHub: Main source control tool and development platform. The repositories are hosted in GitHub and it was the IT team’s main organization platform. Sprints have been planned and followed up with this tool. The team has worked collaboratively creating different branches when necessary.
- Visual Studio Code: Personal code editor used for coding, implementing, and testing the project. Also allows live coding with several members.

In the case of the IT team, it was split into two sub teams, one would develop Frontend and another one would develop Backend. Every member would develop using their personal computers. In the case of the author of this report, a MSI PS48 with 8GB of RAM and 512GB of internal memory, powered by an Intel Core i7-8550U.

4.2 Description of the implementation

The implementation starts when creating the structure of both repositories on GitHub. Previously creating an organization for all team members. The repositories Backend (https://github.com/CO2N-team/BackEnd) and Frontend (https://github.com/CO2N-team/FrontEnd) were created and initialized. The backend repository was created using node’s initialization tool and the frontend using Angular’s.

With the skeleton of the app created, the code was pushed to the repository and the team members were notified that the project was available for everyone.

The first step taken was to analyze and discuss the requirements with the company and rest of team members. Using the above-mentioned communication tools, the team managed to reach a first glimpse of the requirements and created the use case diagram displayed on section 3.3 of this same report.

The functional and non-functional requirements were put into discussion and presented to the company. After a couple of weeks of debate the company agreed on the requirements and the team could go on with the design and implementation. The use case diagram was also presented and agreed on.

Since the project would develop following an AGILE methodology based in sprints, the priority of the requirements was the next subject to discuss with the company. The priorities followed the next order:

1. User related features such as creating an account, logging in, updating users’ information, and deleting a user’s profile. Creating the frontends shared elements and skeleton for the following designs.
2. Calculation related features: the carbon footprint calculator.
3. Compensation related features: the list of projects and the subscription/donation features.
4. Reduction related features: the tips and advices.
5. Progress and feedback related features: dashboards and investment feedback.
6. Sharing and marketplace features: marketplace with CO2N certified products and companies. Report and shareable pages generation.
7. Gamification features: playable game related to climate change and CO2.

Each sprint had one week of duration. Every Monday the team met to show their progress to the rest and prepare the upcoming sprint. Each team member was assigned tasks and every sprint began with the design of the features to be develop. Most of the features follow the same architecture and are designed identically from one another so the most important designs were made during the first sprints. After the design, the implementation phase came and after implementing, testing.

Here will be explained how the implementation phases went throughout the development, the main difficulties the team encountered and in case of solving them, their solution.

4.2.1 Database implementation

The database stores all the data models that are needed to represent the information of the system. Some of these data models were updated while the team progressed through the requirements due to minor lacks. The final databases data model is displayed in the figure 13.

A footprint is composed by a clothes, a list of vehicles or transport, a house, a diet and an others object. They are stored in the footprint object as references. References are declared in MongoDB as an attribute with the OID (objects identifier) of the referred object. References allow retrieving the child objects when retrieving the parent in the same query.

Every user has a current carbon footprint (except if the user still has no footprints) and a list of all the footprints that have been calculated over time (carbonFootprints). All of these are references. A user has a list of references to the feedback of the projects he has invested in. These feedback objects have as well the id for its relative project and user. Every virtual course has as well a list of all the users that have enrolled in it. The remaining classes have no greater complexity since they have no relations with the rest of the data models.

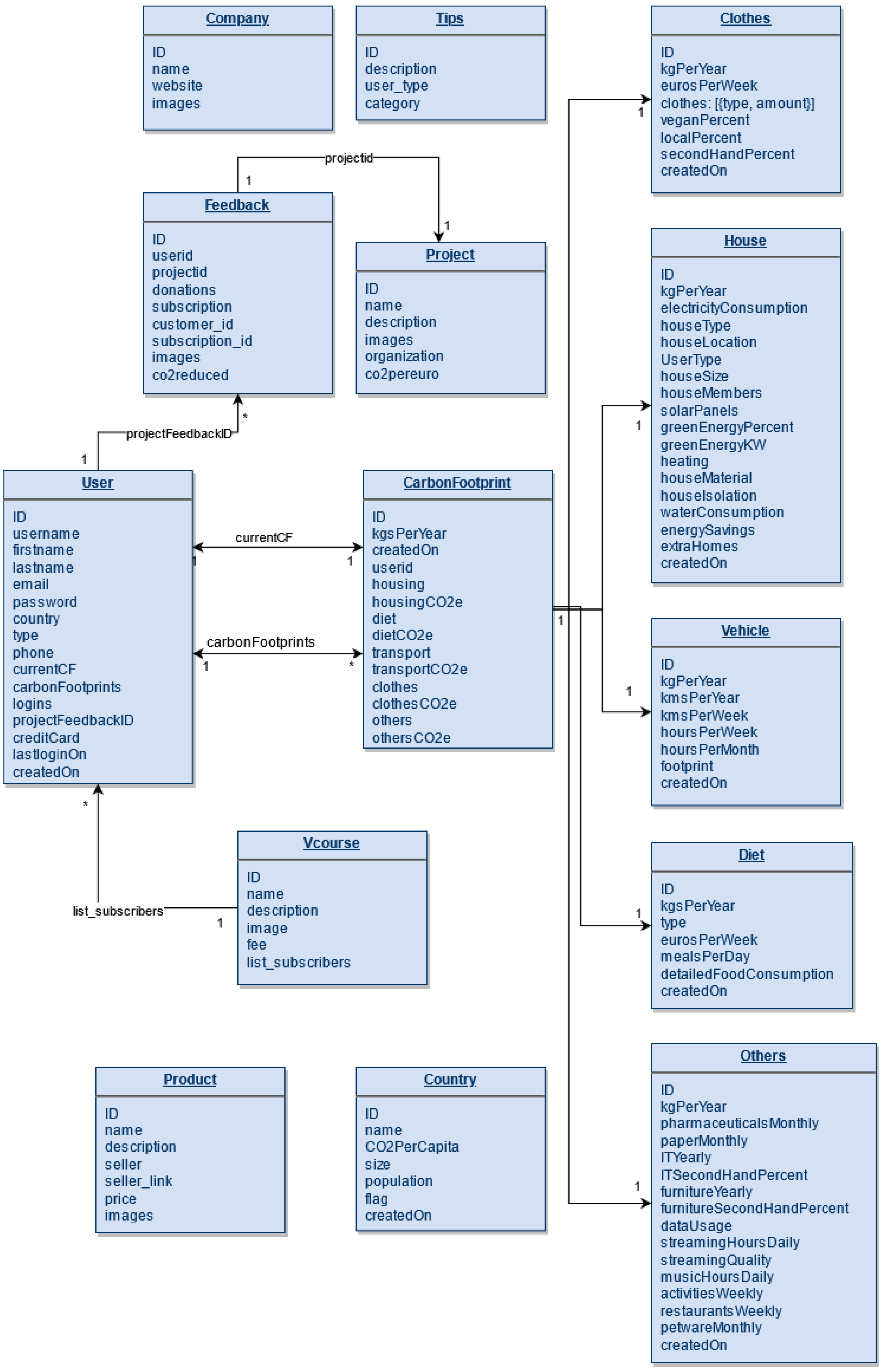


Figure 13 Final Data Models (Juan 2021)

4.2.2 Backend implementation

The implementation of the backend was made progressive as well. Every time a new object was created in the database, an API with the CRUD operations was developed in both APIs.

The Model-View-Controller design patter was adopted. The HTTP requests that reach the API call their respective controller, which manipulated the model, or object in the database. The use of two different APIs and fetching resources from one to another implements the client-server approach. At the same time the whole system would serve as a server itself for the frontend, which would be the client.

The Backend is divided into two different modules. One module serves an API for the frontend to use and the other module offers an API for the database to be manipulated. From now onward they will be referred to as the server’s API and the databases API. These modules allow to encapsulate both the server’s operations and the databases, providing two different layers of abstraction inside the backend. This facilitates the tier division and portability. Different MongoDB databases can be used by changing only the connection string or URI.

The databases API contains four different modules:

- Shared data such as constants used for calculating carbon footprints. General database configuration and authentication strategies.
- The data models stored in the database: Using Mongoose we declare every object that will be stored in the database and its attributes, with their respective types. MongoDB allows embedding objects inside other objects and allows referencing them as well. Array types are also permitted in MongoDB.
- Controllers: They are the functions that will manipulate the models declared in the database and run all the business logic. There is one for each data model and one for database configuration.
- Routers: They are the entry points of the API. They declare what controller should be executed depending on the request. There is one router for each controller and more than one entry point for each router.

Both the controllers and routers implement the CRUD operations for each data model. The module diagram of this databases API is represented in the figure 14.

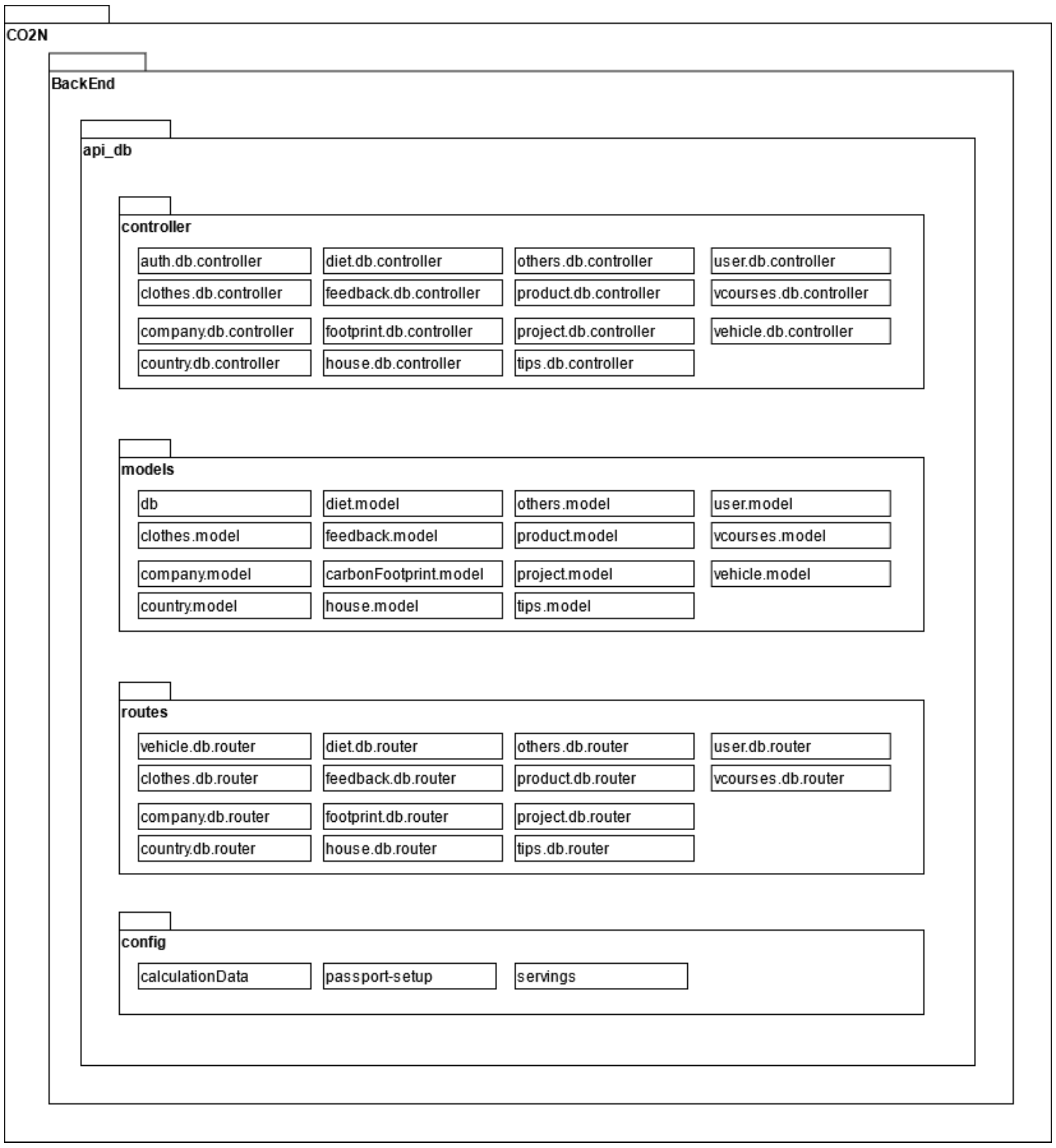


Figure 14 Database API Module Diagram (Juan 2021)

The server’s API on the other hand only has two modules:

- Controllers: They receive requests and create a new one with the necessary information for the databases API. They forward the request to the databases API and return the response. Two controllers have functionality beyond the CRUD operations. These two controllers are:
    - User controller: It implements the login function as well. If the database API returns a success on login, this controller will create and sign the JWT that allows authorizing a user’s requests. The whole user is contained in the signed JWT furthermore, every http request checks that the user that sent the request is the same as the user contained in the JWT. There is no need for

implementing a logout function in this module since removing this JWT from the browser's local storage would be enough.

- Footprint controller: The create operation creates not only the footprint but all of its sub elements. A footprint needs to create several vehicle objects, a house object, a diet object, a clothes object, and an others object. To save the references of these objects inside the new footprint, these objects need to be created first. To implement this, the footprint controller will create http request for each object for the database API. In case all of them succeed on creating the objects, the footprint will be created with their database references (OIDs, Object Identifiers). In case one of them fail to create the object, the new footprint won't be created. A graphical description of this sequence is displayed in the annex A.

- Routes: They enable the endpoints of the API that the frontend must use. They run a method from a controller depending on the request they receive. There is one endpoint for every necessary resource that the frontend needs. A list of the endpoints, what they expect, and the response are listed in the annex A.

The module diagram for the server's API is displayed in figure 15.

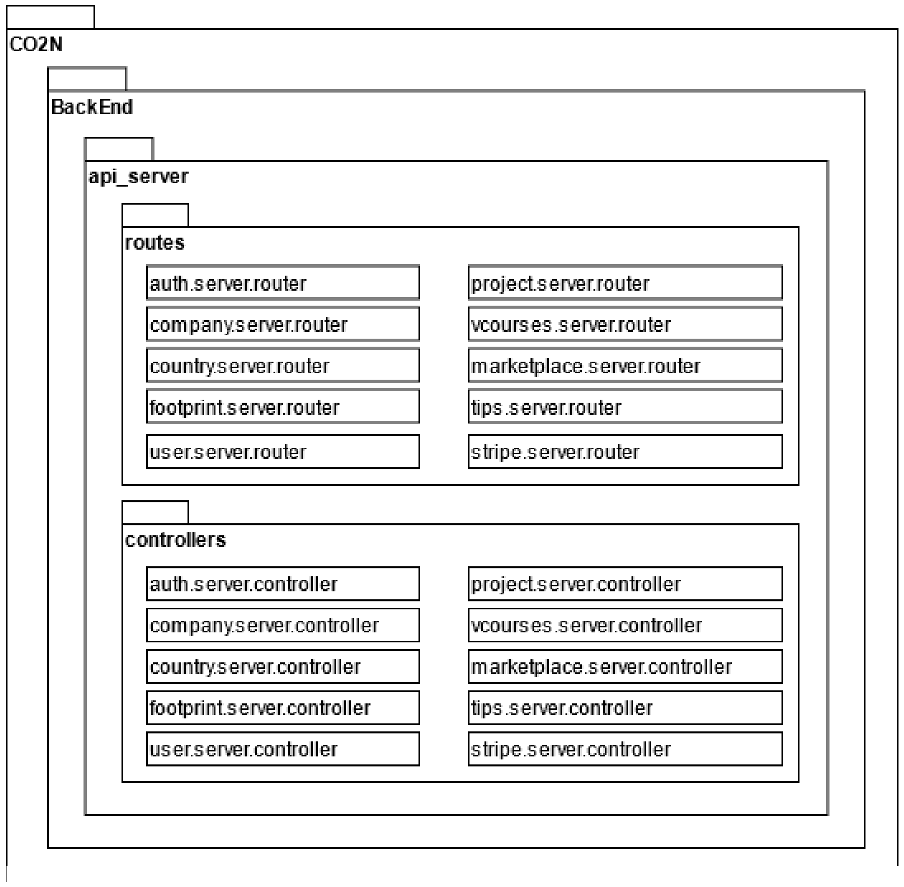


Figure 15 Server API Module Diagram (Juan 2021)

All data needed to calculate a user’s carbon footprint was provided by the company or extracted from the internet. All the sources are documented in the code. It’s important to highlight the use of the open-source tool for carbon footprint calculation from NMF-earth [39]. It was integrated into the project and many constants were used to calculate the carbon emissions of several components.

To calculate a carbon footprint, all its sub elements need to be created as well. The way they are created is described below:

- Clothes: There is two ways of calculating it’s CO2 emissions. The first one is by multiplying the amount of money spent per week times the kilograms of CO2 per euro that clothes emit times the weeks per year. The other way is if the user has specified the amount of different clothes (such as t-shirts, pants, hoodies, etc.) per season, multiply each element per the amount of CO2 manufacturing it emits and multiplying it by the number of seasons per year.
- Diet: The way of calculating it’s CO2 emissions is multiplying the type of meal the user has entered (vegetarian, pescatarian, vegan, carnivore…) times the number of meals the user has per day times the amount of CO2 that meal emits in a day times the number of days in a year.
- Others: For each question, there is a constant of CO2 that the answer emits. It is all multiplied by it’s corresponding constants and summed up.
- Vehicles: For each type of vehicle there is a constant of CO2 that a KM travelling with the vehicle emits. The answers are multiplied by the constant and all the user’s vehicle emissions are added.
- House: The way of calculating it’s CO2 emissions is more complex than the rest. If the user hasn’t entered its energy consumption, it is approximated by using the next formula:

```
approximateConsumption = UserTypeBase + (ADDITIONALRESIDENT * (houseMembers - 1)) + houseSize * (SQRTMETERS + (ADDITIONALRESIDENTSQRT * (houseMembers - 1)))
```

The value UserTypeBase corresponds to low consumption, average consumption, and high consumption. They are constants for KWh consumption.
After having the electricity consumption, manually entered by the user, or approximated like above, it is multiplied by the CO2 that generating a KWh emits in the user’s country. After that, we sum up the heating’s CO2 emissions (if the user uses Gas, Oil or Wood) and the water consumptions emissions.

The carbon footprints total amount of CO2 emitted is the sum of all the sub elements CO2 emissions.

All the questions that the user is asked to answer (or skip if desired) are in Annex A.

The Stripe service is a tool that allows online payment and charges. It offers an API that allows payments on the website. Thanks to Stripe the team abstracts difficult economical concepts and using its services it allows users to subscribe and donate to the projects. Its implementation is not fully complete since a premium account is required to make real payments. The created APIs have been tested with the test account that Stripe presents, and the result was positive.

4.2.3 Frontend implementation

The frontend was developed the same way as the backend. Incrementally and with each sprint, new features were incorporated. The frontend’s design was developed by the design team during the three months that the BlendEd AIM program perdured, so the IT team had no final design until practically the last week of the program.

To remedy this problem, the IT started its own design and proposed different wireframes with different ideas. Finally, the team agreed on dividing the website into five sections, one for each step. With this division the process is linear and user friendly.

Other ideas that the team agreed on were to incorporate a marketplace with CO2N verified products and companies. To use a top menu bar with sections and subsections and to have a footer with contact information and social media accounts.

The selected tool for this tier is Angular, the frontend and client side of the MEAN stack.

Angular’s main building block is a component which are organized into modules. Each component consists of an HTML template, a Typescript class that defines behavior and a CSS that defines the style of the component. A component englobes both style and behavior for a view (webpage). Components use services, which provide functionality and are injected into these as dependencies. This makes Angular’s code modular, reusable, and efficient.

Angular’s modules configure the injector and the compiler and help organize related things together. Each Angular module describes how to compile a component's template and how to create an injector at runtime. It identifies the module's own components, directives, and pipes, making some of them public, through the export’s property, so that external components can use them. They can also add service providers to the application dependency injectors [40].

Angular Material was chosen as a user interface component library. It provides simple yet attractive components that give the design a better look and feeling.

The five divisions of the website are implemented using modules. This allows lazy loading, so the client’s browser does not have to load the full website on entering it. Every time they access one section, this full module will load all its components and pages, making the users experience smoother and time efficient.

Every module has at least two components and a router, except for the module where the footer and header are stored since these components will always be embedded on the website and no routing is necessary.

Every view or page that the user sees is composed by three components: the header component, the footer component, and the main pages component.

Angular implements dependency injection and allows components to instantiate services.

The main modules and implemented components are explained below:

- **Authentication module**: It’s in charge of all user authentication and account management related features.
    - **Components**: It’s composed by the forgot password, login, register and reset password components.
    - **Backend interaction**: It calls the backends API using the injected user service. It fetches user related data and sends credentials to the backend.
- **Calculate module**: It’s in charge of the carbon footprint calculator related features. The calculator is implemented so that a user doesn’t have to answer all the questions since there are default values that simplify the user’s input. A user can also select the average carbon footprint per capita from his country and save it as his footprint. This would simplify even further the user’s interactions. Every time that a question is answered the modifications are changed in this service. While advancing through the questionary, the user skips from one category to the next one (or back) and can change answers before submitting the result. Once the result has been submitted, the results are calculated and a view with a chart indicating what percentage of the user’s footprint belongs to each category. Next to this chart, the option to compensate is presented and on selecting the amount and clicking on the compensate button, the view changes to the project selection and offsetting view.
    - **Components**: There is a component for each question category (house, food, transport, clothes, and others). A question card component has been created to simplify the questions component. This way, the frontend only builds one

question at a time and not the whole questionary. There are components for the results, the average carbon footprint selection and the view to choose between these two types of calculations.
    - **Backend interaction**: The calculator’s questions are stored in the footprint service that is injected. On submitting the questionary, the footprint is sent to the backend, which calculates the total amount of CO2 and the amounts per category. This returns the result that is later displayed as a view to users.

- **Compensate module**: The compensate module oversees the project and compensation features. After calculating a user’s footprint offsetting it is up next. The users can select between a list of projects that the frontend requests to the backend on loading and then can choose to donate once or subscribe monthly to a project. The team also worked on creating a preview tool of the area that the user’s compensation would associate to.
    - **Components**: There is a component for the project list, a component for the projects and a component for the investment popup. The view of different projects is created using the same component. This way code is reused.
    - **Backend interaction**: The compensation module requests the list of projects from the backend. On subscribing or donating to a project, it uses the backends’ stripe API as well to make the payments and subscriptions.
- **Contact module**: It oversees general content such as the contact page or frequent answers and questions page.
    - **Components**: One for the contact page, the frequent questions and answers page, the privacy policy page and the terms and conditions page.
    - **Backend interaction**: It requires no information from the backend.
- **Feedback module**: It’s in charge of feedback for users. It is divided into two different pages. One is for the user to see his personal progress and the other one is for everyone to see the progress that the whole community has made. Giving feedback for each project globally. In the personal progress views, there will be for each project images that correspond to his investment and percentages of how his contribution has helped to the project.
    - **Components**: it contains components for the general and personal progress pages.
    - **Backend interaction**: The general progress page requests the list of projects from the backend to show the achievements that each project has made

thanks to all the user’s contribution. The personal progress page requests the user’s footprint history from the backend and displays dashboards about the user’s footprint evolution over time, the current footprint, and the number of tons of CO2 for each category.

- **Home module**: it contains the main page and about page components, none require information from the backend.
- **Reduce module**: it oversees the tips and advice and virtual courses features.
  - **Components**: It contains the advice and virtual courses components. The advice contains a list of tips by categories about how to diminish the users CO2 emissions in that specific field. Depending on the user’s footprint, they will be listed in one order or in another one. They idea of hiding tips of the category that the user has fewer emissions was contemplated but it was discarded since the tips could help the user reduce in that field even more.
  - **Backend interaction**: It fetches from the backend the list of tips, the users current carbon footprint and the list of virtual courses. It will use the backends’ Stripe API as well when the user enrolls in a virtual course.
- **Share module**: it oversees the sharing and generating reports features as well as the marketplace. The generating reports and gifts have the interface implemented but do not have the associated functionality due to lack of time and design. The marketplace shows products and companies that are CO2N verified.
  - **Components**: It has components for generating reports, gifts, marketplace, and the intermediate screens.
  - **Backend interaction**: This module will fetch the list of products and companies from the backend to display them in the marketplace. It will also need to fetch the user’s information as well as his footprint history and current carbon footprint to generate the reports and gift cards.
- **User profile module**: It oversees the features relative to the users account and data display. It allows modifying its data and deleting the account.
  - **Components**: It has one component which is the profile. It displays the user’s data and allows them to modify their values or delete their account.
  - **Backend interaction**: This component uses the user’s service which interacts with the backend API to modify and delete user information.

All these modules are graphically represented in the module diagram of figure 16.

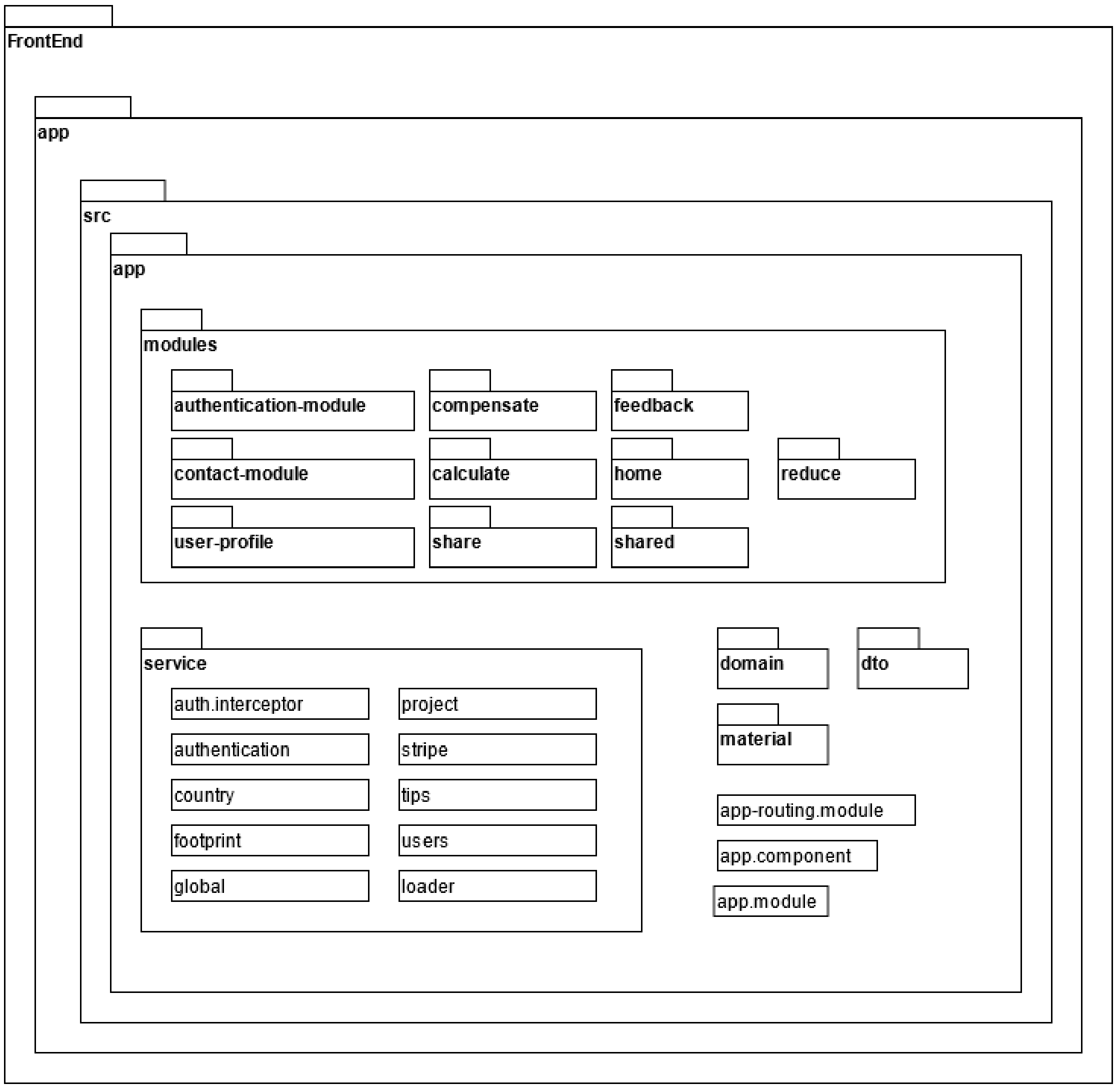


Figure 16 Frontend Module Diagram (Juan 2021)

Global styling has also been made in order to give the website a clear and uniform look. This also helps designer to make future changes in the style since it is centralized. Nonetheless, components that gave styling issues were modified in their own .scss file.

The font suggested by the design team is Garnett and was downloaded and incorporated into the website by creating the font’s interface using scss and woff files.

The global router routes each submodule that when accessed, loads its own router.

4.2.4 General implementation aspects

The final structure of the system would look like the module diagram represented in the figure 17.

We can see both modules, Backend and Frontend in the same application, CO2N.

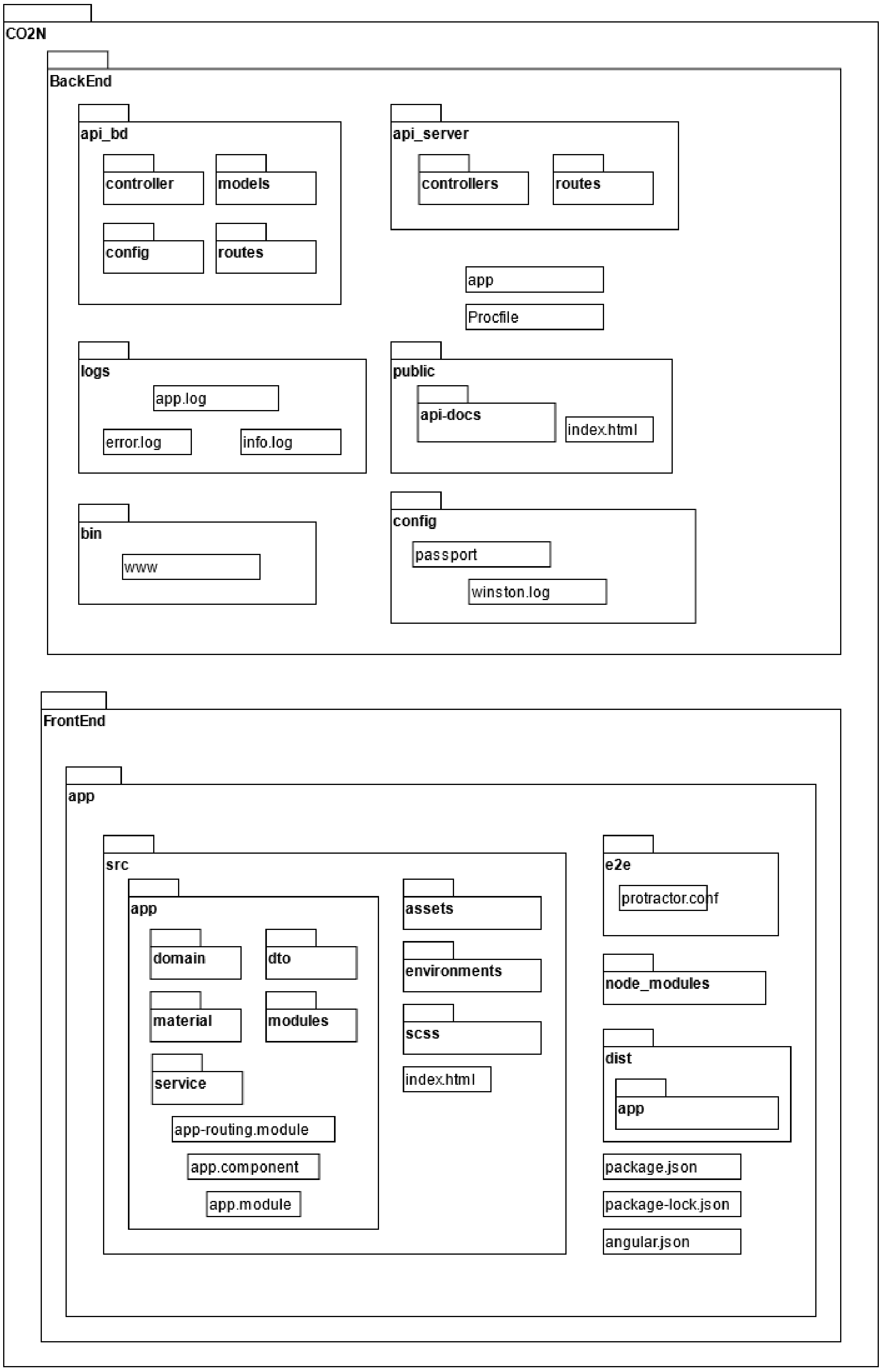


Figure 17 Module Diagram (Juan 2021)

4.3 Tests

Testing has been made in every sprint during the whole process. For every new feature or change in the code, tests have been made to assure the new version of the software was secure and usable.

For the backend a bottom-up approach was adopted.

Unitary tests have been made for both APIs in the backend. The tests consist of inputting different values, both correct and incorrect to look for possible bugs in the code. They have been implemented using the tool Postman[43]. Postman is a software used for testing APIs. It allows shared development by creating a team and testing with scripts and variables.

As a new method, function or endpoint was created in the databases API, a corresponding test in Postman was implemented. Thanks to this method, all the endpoints have been tested.

On the implementation of endpoints and functions in the server's API, unitary and integration tests were created as well. These tests verified the code and the communication with the lower-level API, validating integration between the database and server API.

As for the frontend, all the tests have been made manually. On the implementation of new features, the frontend section of the IT team checked rigorously the different flows that a user could make. This also helped discovering errors and helped with minor adjustments in every sprint of the process.

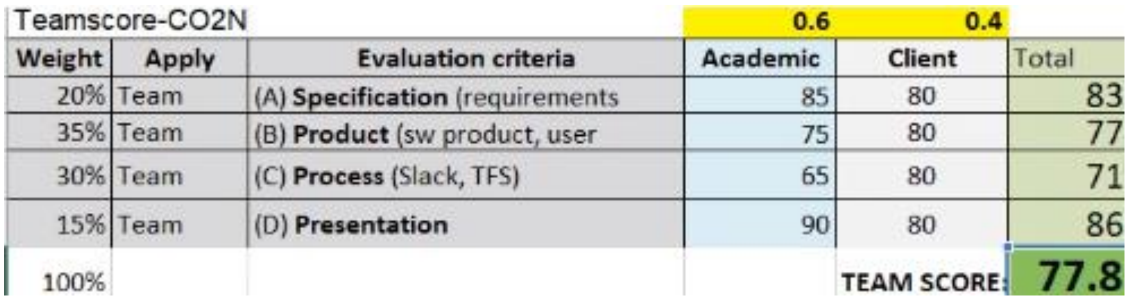
As the product was not deployed, performance and security testing by online tools have not been made.

4.4 Validation of the solution

The solution was validated according to the BlendEd AIM programation. During the final week of the project, the team made the final adjustments to the product and prepared the final pitch that would be evaluated by the company and the teachers involved.

The final pitch consisted of a description of the different tasks that the team had done, how it was constructed, how teamwork had been and finally, a demonstration of the reached solution.

The evaluation depended on both the teachers and client's grades over four different categories: Specification (requirements), Product (sw product, user satisfaction), Process (teamwork, TFS) and Presentation. The final evaluation is presented in the figure 18.



Teamscore-CO2N			0.6	0.4	
Weight	Apply	Evaluation criteria	Academic	Client	Total
20%	Team	(A) **Specification** (requirements	85	80	83
35%	Team	(B) **Product** (sw product, user	75	80	77
30%	Team	(C) **Process** (Slack, TFS)	65	80	71
15%	Team	(D) **Presentation**	90	80	86
100%				TEAM SCORE:	77.8

Figure 18 Final team evaluation (BlendEd Mobility, 2021)

There is documentation both in the code and in Google Drive, where the team has been working throughout the year. The product is usable, a user can navigation through all the sections of the website.

Around a 20% of the website is responsive to different devices.

12 out of 23 functional requirements are fully implemented.

6 out of 23 functional requirements are partially implemented.

The remaining 5 functional requirements haven't been implemented.

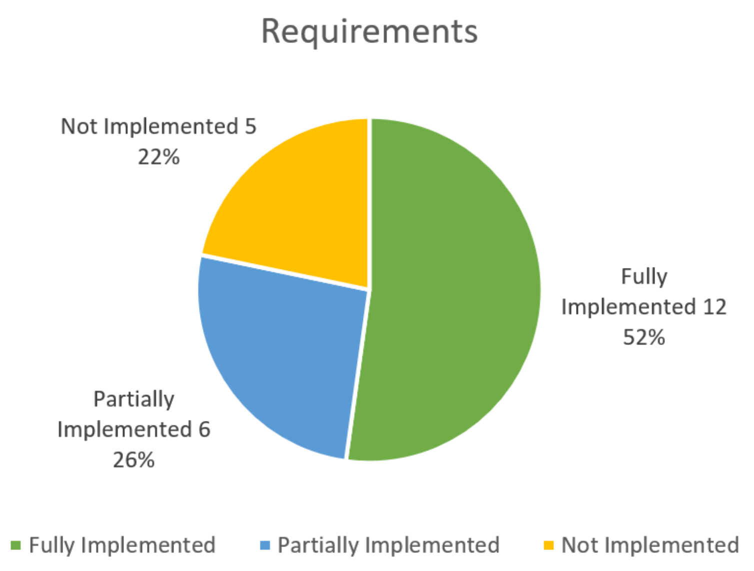


Figure 19 Fulfilled Requirements (Juan 2021)

No further validation has been possible since the system hasn't been deployed to a real time production service.

5 Conclusions

5.1 Reached goals

The goals presented at the beginning of this report have been listed and analyzed below:

- Reach as many companies and individuals as possible: The product is not yet in production. This goal must be achieved over time with a strong marketing plan, developed by the marketing team [50].
- Help individuals, organizations and companies reduce and offset their CO2 emissions. This goal is partially implemented since several steps are required. Around a 75% has been achieved:
    - "*Calculating a user's carbon footprint*": Fully implemented. Any user can calculate his or her carbon footprint using the website. Companies can also get in touch with someone from the company to have their carbon footprint calculated as well.
    - "*Offset (fully or partially) your carbon footprint by investing in sustainable projects*": Partially implemented. Most of the feature's work but depend on an extra element to be fully working. Stripe is missing a premium account to make subscriptions and donations effective. The frontend is implemented but needs to be improved. Information about real projects is left to be added by the company. The 85% of this goal is implemented.
    - *"Give users personalized tips, advice and courses so they can learn how to diminish their carbon footprint for next year's calculation":* Partially implemented. Users can navigate through a list of tips on how to reduce their emissions by changing daily habits. Real virtual courses are yet to be added by the company and enrollment features are missing in the frontend. The missing code is practically the same as the project subscription feature already implemented. The 85% of this goal is implemented.
    - *"Give users feedback on the CO2 they have reduced along the years and how you have reduced it given your carbon footprints. Give clear and real time feedback on how the compensation investment is going and how much CO2 emissions has been reduced or avoided":* Partially implemented. Project related feedback is missing due to the lack of real projects. Feedback of CO2 reduced over time and dashboards for current and previous carbon footprints is fully implemented. The 85% of this goal is implemented

    - "*Sharing what the user has accomplished with friends, family or customers through the CO2N label. In case of companies and organizations, sponsor their CO2 neutral products"*: Partially implemented. The marketplace is fully implemented with CO2N products and organizations, real data is still missing . Frontend for sharing and report generation is implemented but has no functionality. A real sharable report generating algorithm is missing. The 70% of this goal is implemented.
- Create a community that feels like it is really helping to achieve this transition: This goal will be achieved over time thanks to the marketing plan [50] that involves generating social media accounts and a blog for CO2N.

The developed web application is in line with the sustainable development goals proposed by the United Nations that conform 17 global goals that all communities worldwide should try to reach.

The developed system serves as a backbone for CO2N to develop its final product and contribute to the world and environment, focusing on individual actions like changing a company's production model or offsetting by investing in planting trees.

If every individual and company lived and worked by CO2Ns principles, climate change could be slowed down drastically, and the world would aspire to a better future.

5.2 Limitations and future work

The project has encountered several limitations and impediments to fully develop the project. Regarding the team and it's integrates, half of the IT team, two out of four, didn't have programming knowledge since they were electrical engineers and not IT specialists. This caused a great delay in several aspects. In first place, learning and explaining new concepts, tools and technologies cost the team several weeks that caused a great delay. In the other hand, almost all the code had to be double checked by the team members that did have IT knowledge and skills, which caused great delays as well.

This could have been fixed with a better selection of the participating members by the BlendEd AIM organization.

In the other hand, each team member had a different amount of ECTS assigned to participating in the project, which made complicated the division of tasks. One of the IT members had 0 ECTS assigned so this team members participation was significantly smaller.

This would have also been fixed with a better selection of the members that were going to engage in this project and assigning a similar amount of ECTS to each one.

Another limitation for the frontend team was that the design team had 3 months to fulfill the final design while the IT team had the same time to fulfill the entire system. In a software engineering process, the design is the third step, after the requirements definition and analysis phases. This caused major confusion and delay in the frontend team since the final design was handed over by the design team a week before the final presentation. This significantly affected the final product since there were a lot of features that couldn’t be designed and tested in such few time.

This could be fixed by the BlendEd AIM organization by selecting projects that have already thought of some design aspects, or by making the process more iterative and having the design team hand over the work before the IT team starts with the implementation phase.

Another limitation has been the need of discussing every aspect with the company. This is a normal problem in real projects and could have been handled better by the team, putting more work into the meetings with the company. Another limitation factor is that nobody in the company had IT knowledge and every idea and step of the process required more explanation in a more user-friendly language. This was handled decently by the team trying to avoid technical language and explanations.

The organization has also been a factor to remark. The team was self-organized, and every team member had a role. Very few team members had real experience in AGILE methodologies such as SCRUM and no one was an expert in team organization. This took the team a lot of work and complicated the project and its development. Still the team managed to meet every week and work persistently to achieve a good final product thanks to putting in hard work and extra effort.

Future work implies finishing the remaining requirements. Having better organization routines and a more consistent team. The fact that the team did not get to known each other in person also made it more complicated and working virtually without a physical office also influenced the development of the project.

In the future there will be better communication, better organization between and within the teams and hopefully more experts will work together with less experienced members.

5.3 Final evaluation

The general feeling of the project is positive. A lot of work has been put into it from all sides. From teachers to students from companies to universities, all have worked together to organize a very enriching experience. A lot has been learned and not only about IT, software development and team organization. Culturally and personally this experience has helped Juan grow as a person and team member. It is gratifying to see that people from seven different countries and cultures that all speak different native languages have come together and organized themselves to work on the same project and reach a solution.

It is an example that when individuals and organizations work towards a common goal, marvelous things can be achieved. The technologies exist and they should be taken advantage from to speed up the slowing down of climate change and switch to a sustainable way of living.

The project itself has been personally satisfying. The goal is to help the world reach a more sustainable way of living and producing to prevent the disasters of climate change and the non-stopping growth in which companies live in. This reality has no place in the planet in which we live since resources are limited and continuing with resource exploitation will bring the worst-case scenario for use, our future generations, and our planet.

Collaborating with a startup that has the same perspective has helped transform the huge amount of work into a sensation of helping the world to become a better place.

References

[0.0] Our World in Data (2021) – “Annual CO2 emissions” (https://ourworldindata.org/co2-emissions, 08/09/2021)

[0.1] Kurzgesagt (2020) – “Who is responsible for climate change? Who needs to fix it?” (https://youtu.be/ipVxxxqwBQw, 08/09/2021)

[0] Guiding Principles of REST - (https://restfulapi.net/, 07/09/2021)

[1] António Guterres, UN (2020) – “Carbon neutrality by 2050: the world’s most urgent mission”. Published online at UN.org. Retrieved from: ‘https://www.un.org/sg/en/content/sg/articles/2020-12-11/carbon-neutrality-2050-the-world%E2%80%99s-most-urgent-mission’ [Online Resource]

[2] Hannah Ritchie and Max Roser (2020) - "CO_2 and Greenhouse Gas Emissions". Published online at OurWorldInData.org. Retrieved from: “https://ourworldindata.org/co2-and-other-greenhouse-gas-emissions” [Online Resource]

[3] Wouter Van Den Bosch, IMEC (2021) – “BlendEd - SCRUM workshop”. *SCRUM workshop*. Microsoft Teams, Belgium, February [Online Conference]. Retrieved from: https://docs.google.com/presentation/d/1k_v-MJM2B7YjsTRdJ48GDva7ZKf84lik/edit?usp=sharing&ouid=114379122710603649404&rtpof=true&sd=true

[4] Greenwashing. (2021, August 13). In Wikipedia. https://en.wikipedia.org/wiki/Greenwashing

[20] Eliza Was Here (2021) – “Eliza was here”. Retrieved from: “https://www.elizawashere.nl/“ [Online Resource], 30/7/2021.

[21] Be Climate (2021) – “BE CLIMATE | BUY CO2 NEUTRAL”. Retrieved from: https://www.beclimate.com/en/, 30/8/2021.

[22] Climate Neutral (2020) – “Climate Neutral Certified | About Climate Neutral”. (https://www.climateneutral.org/about, 30/8/2021)

[23] Global Footprint Network (2021) – “Global Footprint Network”. (https://www.footprintnetwork.org/, 30/8/2021)

[24] Climate Care (2021) – “Climate Action Starts Here | ClimateCare”. (https://www.climatecare.org/, 30/8/2021)

[25] MongoDB (2021) (https://www.mongodb.com/es, 5/9/2021)

[26] Prat, Codecondo (2015) – "7 Good Reasons to use MEAN Stack in your next web project". (https://codecondo.com/7-good-reasons-to-use-mean-stack-in-your-next-web-project/, 4/9/2021)

[30] Global Footprint Network (2021) – "Ecological Footprint Calculator". (https://www.footprintcalculator.org/home/en, 5/9/2021)

[31] Lisa Van Langenhove (2021) – "CO2N" – Graphical Design degree thesis – Luca | School of Arts, Belgium.

[39] NMF-earth/carbon-footprint (2021) (https://github.com/NMF-earth/carbon-footprint, 06/09/2021)

[40] Angular (2021) – "Angular - – NgModules". (https://angular.io/guide/ngmodules, 6/9/2021)

[43] Postman (2021) (https://www.postman.com/, 6/9/2021)

[50] Hugo Cunha (2021) – "Marketing Plan" – Marketing degree thesis – Instituto Universitário Da Maia, ISMAI, Porto, Portugal

Anexo A Content

A.4.2.2 Backend implementation

The list of questions that are asked to the users are the following:

Calculator questions:

20 question pages, sometimes several questions on 1 page because they belong together

Categories: Transportation - Diet - Clothing - Housing - Other consumptions

Housing

<u>Do you know your electricity consumption?</u>

Yes — No

Yes:

<u>How many kwH do you consume each year?</u>

0 — 1000 kwH (slider)

No:

<u>What's the surface of your home?</u>

0 — 1000 m2

<u>How many people are living in this home?</u>

0 — 20 people (slider)

<u>How conscious are you of your energy waste?</u>

Low maintenance — average — high maintenance

<u>Do you have a green electricity contract? (info: energy comes from renewable energy sources or CO2 offset is compensated)</u>

No; less than 100%; 100%

<u>How do you heat your home?</u>

Gas — Oil — Wood — Electricity

<u>Do you know your annual consumption?</u>

yes / no

yes

<u>No: How warm do you heat your home on average in winter</u>

below 14C; 14-17C;18-21C; over 21C

<u>What is your average monthly water consumption?</u>

0 — 10.000 liters

<u>Which of the following energy savings have you made in your home?</u>

energy saving light bulbs

condensing boiler

solar panels

solar water heater

double glazing

wall insulation

loft insulation

low flow fittings to taps and showers

regular maintenance

Transportation

<u>Which type of car(s) do you use?</u>

None — Electric car — plug-in hybrid car - Hybrid car — small/medium/large petrol or diesel car

(options from less polluting to very polluting)

(possibility to add another car)

<u>Which type of motorcycle(s) do you use?</u>

None — Electric motorcycle — Fossil motorcycle

(options from less polluting to very polluting)

(possibility to add another motorcycle)

<u>How far do you travel with these vehicles each week?</u>

Car 0 km - 3000 km

Motorcycle 0 km - 3000 km

I don’t know => <u>how many hours per week</u>

(sliders from less polluting to very polluting)

<u>How many hours per week do you use public transport in your daily life?</u>

Bus 0 hrs - 30 hrs

Tram 0 hrs - 30 hrs

Train 0 hrs - 30 hrs

Metro 0 hrs - 30 hrs

(sliders from less polluting to very polluting)

<u>In the last year, how many hours did you fly</u>

0 hrs - 200 hrs

(sliders from less polluting to very polluting)

Food

<u>How much do you spend on average on food per week?</u>

(slider)

<u>Which diet describes your meals the best?</u>

Vegan — Vegetarian — Pescetarian — meat almost never - meat in some meals — meat in all meals

<u>How many meals do you eat each day? (info: what is a meal?)</u>

1 — 10 meals (slider)

<u>How often do you eat the following products?</u>

Fruits & vegetables / 0 times a week — 25 times a week

Beans & nuts / 0 times a week — 25 times a week

Carbs (info: potatoes, rice, pasta) / 0 times a week — 25 times a week

Dairy (info: milk, cheese, eggs) / 0 times a week — 25 times a week

Soy & tofu / 0 times a week — 25 times a week

Meat (info: beef, lamb, pork, …) / 0 times a week — 25 times a week

Poultry (info: chicken, turkey, …) / 0 times a week — 25 times a week

Fish & seafood / 0 times a week — 25 times a week

Coffee & chocolate / 0 times a week — 25 times a week

Sodas / 0 times a week — 25 times a week

Fastfood & snacks / 0 times a week — 25 times a week

Clothing

<u>How many units of these clothes do you buy each season?</u>

Coats / 0 — 20 units

Sweaters / 0 — 20 units

Shirts & t-shirts / 0 — 20 units

Pants, skirts & shorts / 0 — 20 units

Dresses / 0 — 20 units

Shoes / 0 — 20 units

Underware / 0 — 20 units

Sportswear / 0 — 20 units

Bags & accessoires / 0 — 20 units

How many of these clothes did you buy local? (info: what is local? H&M isn’t local just because it’s located in your city - explaining this)

None — All (slider)

How many of these clothes are vegan? (info: what is vegan?)

None — All (slider)

How many of these clothes did you buy second hand? (info: what is second hand?)

None — All (slider)

Other consumptions

How much do you spend monthly on average on pharmaceuticals?

How much do you spend monthly on average on paper books, magazines or newspapers?

Did you buy this year any of the following?

computer/laptop/tablet/mobile?

Is it second hand or refurbished?

Yes — No

furniture or other manufactured goods?

yes - amount range

What is your internet usage each month? (info: mobile data & wifi)

0 — 100 GB

Don’t know

How much do you spend weekly on restaurants, pubs and takeaways?

0 ; 0-20; 20 - 50; 50-100; more than 100

<u>How much do you spend weekly in recreational, cultural and sporting activities?</u>

0 ; 0-20; 20 - 50; more than 50

<u>How much do you spend on your pet and pet ware monthly?</u>

The sequence diagram that explains the creation of a carbon footprint is shown in the figure A.4.22.

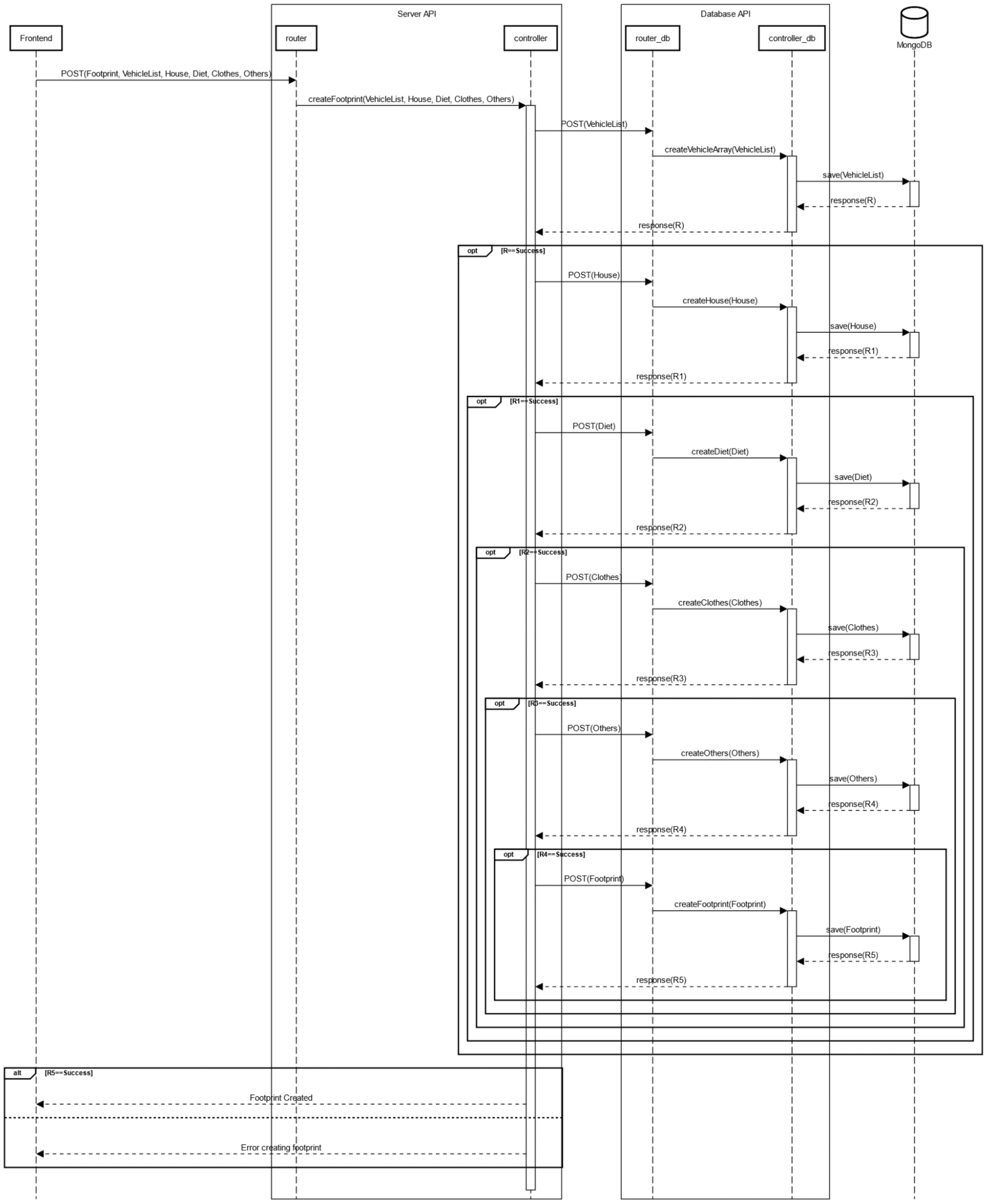


Figure A.4.22: Carbon Footprint creation – Backend communication (Juan 2021)

The list of endpoints the server API has, an example of use and the expected answer is listed below:

Users:

Create user - registration

POST	http://localhost:3000/user/

BODY (JSON)	{
	   "username":"lisa",
	   "country":"Belgium",
	   "password":"12345678",
	   "email":"lisa@mail.com",
	   "type":"individual"
	}

Returns:

Success	`Created`
Error	`Error creating user`

Login user - logging in

POST	http://localhost:3000/user/login

BODY (JSON)	{
	   "password":"12345678",
	   "email":"lisa@mail.com"
	}

Returns

Success	{
	   "token": JWT,
	   "type": USERTYPE,
	   "id": USERID,

	"username": USERNAME, }
Error	Error

Delete user - removing user from db

DELETE	http://localhost:3000/user/{{user_id}}
TOKEN	JWT

Returns

Success	Deleted
Error	Error deleting

Update user - modifies records in db.

POST	http://localhost:3000/user/{{user_id}}

BODY (JSON)	{
	"username":"lisa",
	"country":"Belgium",
	"email":"lisa@mail.com",
	"type":"individual"
	}

Returns

Success	Deleted
Error	Error deleting

Footprint:

Create footprint - creating carbon footprint in db

POST	http://localhost:3000/footprint/{{user_id}}
BODY (JSON)	{

```
  "clothing": {
    "clothes" : [
      { "type" : "shirt" , "amount" : "{{shirt}}"},
      { "type" : "tshirt" , "amount" : "{{tshirt}}"},
      { "type" : "jeans" , "amount" : "{{jeans}}"},
      { "type" : "sweater" , "amount" : "{{sweater}}"},
      { "type" : "dress" , "amount" : "{{dress}}"},
      { "type" : "shoes" , "amount" : "{{shoes}}"},
      { "type" : "coat", "amount" : "{{coat}}"}
    ],
    "veganClothesPercent" : "{{veganClothesPercent}}" ,
    "localProducts": "{{localProducts}}"
  },

  "diet": {
    "type" : "{{meatType}}",
    "mealsPerDay" : {{mealsPerDay}},
    "detailedFoodConsummption" : [ { "food" : "{{food1}}" ,    "servingsPerWeek" : "{{food1_meals}}"} ],
    "buyLocal" : "{{buyLocal}}",
    "waste" : "{{waste}}",
    "recycling" : { "food" : "{{recycling_food}}" , "tin" : "{{recycling_tin}}", "plastic" : "{{recycling_plastic}}" , "glass" : "{{recycling_plastic}}" , "paper" :"{{recycling_plastic}}"}
  },

  "housing": {
    "electricityConsuption" : "0",
    "houseType" : "flat",
     "houseSize" : "20",
     "houseMembers" : "1",
    "userType" : "LOW",
     "country" : "belgium",
        "solarPanels" : "no",
        "greenEnergyPercent" : "10",
        "greenEnergyKW" : "0",
        "heating" : [{ "type": "OIL", "amount" : "2", "kWh" : "0", "time" : "0"}],
       "houseMaterail":"Brick",
        "houseIsolation" : "weak",
        "waterConsumption":"10"
    },

   "transport": [
       {   "transportType" : "bus",
          "hoursPerWeek": "20",
          "kmsPerWeek": "30"
       },
       {   "transportType" : "plane",
          "hoursPerWeek": "20",
          "kmsPerWeek": "30"}
       ]
}
```

Returns:

Success	`{ "msg": "Created footprint x" }`
Error	Error creating clothes/diet/house/vehicle

Get footprint - fetches footprint record from db

GET	localhost:3000/footprint/{{footprint_id1}}
RESPONSE BODY (JSON)	`{ "housing": [ "60b42af66d3ef731a42bbecc"`

	],
	    "transport": [
	        "60b42af66d3ef731a42bbece",
	        "60b42af66d3ef731a42bbecf"
	    ],
	    "_id": "60b42af66d3ef731a42bbed0",
	    "userid": "60b4281a6d3ef731a42bbeb5",
	    "tonsPerYear": 232307.04444583735,
	    "housingCO2e": 4748.400277837333,
	    "transportCO2e": 225739.51416800002,
	    "diet": "60b42af66d3ef731a42bbecb",
	    "dietCO2e": 1022.73,
	    "clothes": "60b42af66d3ef731a42bbec3",
	    "clothesCO2e": 796.4000000000001,
	    "createdOn": "2021-05-31T00:16:54.276Z",
	    "__v": 0
	}

Get users footprints - fetches all users footprints history

GET	localhost:3000/footprint/all

RESPONSE
BODY (JSON)
	{
	[FOOTPRINTLIST]
	}

Project

Create projects

POST	localhost:3000/project
BODY	{

(JSON)	"name":"project1", "description": "description", "organization": "org", "co2pereuro" : "5" }

Returns:

Success	{
	"msg": "Project created",
	"_id": "60b4d65dcabf272ac8a26feb",
	"name": "project1"
	}

Error	Error creating project

Get all projects

GET	http://localhost:3000/project/
RESPONSE BODY (JSON)	[{ "images": [], "_id": "60ab91ddc9844112249d8bdb", "name": "project1", "description": "description", "organization": "org", "co2pereuro": 5, "__v": 0 }, { "images": [], "_id": "60ab929142c5ca3584770815", "name": "project1",

	"description": "description", "organization": "org", "co2pereuro": 5, "__v": 0 },]

Get a project

GET	http://localhost:3000/project/:project_id

RESPONSE BODY (JSON)	{
	"images": [],
	"_id": "60b4d65dcabf272ac8a26feb",
	"name": "project1",
	"description": "description",
	"organization": "org",
	"co2pereuro": 5,
	"__v": 0
	}

<u>ProjectFeedback:</u>

Create feedback

POST	localhost:3000/project/feedback/{{userid}}

BODY (JSON)	{
	
	"projectid" : "*{{project_id1}}*",
	"donations" : "1",
	"subscription" : "1",
	"images": "none",
	"co2reduced": "25"
	}

Returns:

Success	`{`
	`    "msg": "Feedback created",`
	`    "_id": "60b4bd9a73c76d0de88499c8",`
	`    "userid": "60b4281a6d3ef731a42bbeb5"`
	`}`

Error	Error creating feedback

Get all users feedback

GET	localhost:3000/project/feedback/{{userid}}

RESPONSE BODY (JSON)	`[`
	`    {`
	`        "donations": [`
	`            1`
	`        ],`
	`        "images": [`
	`            "none"`
	`        ],`
	`        "_id": "60b4b0416d805441004aa524",`
	`        "userid": "60b4281a6d3ef731a42bbeb5",`
	`        "projectid": "60ab91ddc9844112249d8bdb",`
	`        "subscription": 1,`
	`        "co2reduced": "25",`
	`        "__v": 0`
	`    },`
	`    {`
	`        "donations": [`
	`            1`
	`        ],`
	`        "images": [`

	"none"], "_id": "60b4b0606d805441004aa525", "userid": "60b4281a6d3ef731a42bbeb5", "projectid": "60ab91ddc9844112249d8bdb", "subscription": 1, "co2reduced": "25", "__v": 0 },]

Get specific feedback

GET	localhost:3000/project/feedback/{{feedback_id1}}

RESPONSE
BODY (JSON)
	{
	"donations": [
	1
	],
	"images": [
	"none"
	],
	"_id": "60b4bd9a73c76d0de88499c8",
	"userid": "60b4281a6d3ef731a42bbeb5",
	"projectid": "60ab91ddc9844112249d8bdb",
	"subscription": 1,
	"co2reduced": "25",
	"__v": 0
	}

Images - how to view images.

LINK URL	http://localhost:3000/images/smile.jpg

RETURNS	[IMAGE]

TIPS

Tips add

POST	localhost:3000/tips

BODY (JSON)	{
	    "description" : "lalala",
	    "user_type" : "high user",
	    "category" : "diet"
	}

Returns:

Success	{
	    "msg": "Tips added",
	    "_id": "60ccbb3f00236630fc0aabb5"
	}

Error	Error creating

Tips get

GET	localhost:3000/tips/60ccbb3f00236630fc0aabb5

RESPONSE BODY (JSON)	{
	    "_id": "60ccbb3f00236630fc0aabb5",
	    "description": "lalala",
	    "user_type": "high user",
	    "category": "diet",

	"__v": 0 }

Tips get all

GET	localhost:3000/tips

RESPONSE
BODY (JSON)
	[
	{
	"_id": "60ccbb3f00236630fc0aabb5",
	"description": "lalala",
	"user_type": "high user",
	"category": "diet",
	"__v": 0
	}
	]

Tips remove

DELETE	localhost:3000/tips/{id}

Returns

Success	
Error	Error deleting

COMPANY

Company add

POST	localhost:3000/company

BODY (JSON)	{ "name" : "company", "website": "dod.com", "images": "link" }

Returns:

Success	{ "msg": "Company added", "_id": "60cd431d00236630fc0aabb6", "name": "company" }
Error	Error creating

Company get

GET	localhost:3000/company/60ccbb3f00236630fc0aabb5
RESPONSE BODY (JSON)	{ "_id": "60cd431d00236630fc0aabb6", "name": "company", "website": "dod.com", "images": "link", "__v": 0 }

Company get all

GET	localhost:3000/company
RESPONSE BODY (JSON)	[{ "_id": "60cd431d00236630fc0aabb6", "name": "company", "website": "dod.com",

	"images": "link", "__v": 0 }]

Company remove

DELETE	localhost:3000/company/{id}

Returns

Success	
Error	Error deleting

MARKETPLACE

Product add

POST	localhost:3000/marketplace

BODY (JSON)	{
	"name": "apple",
	"description": "fruit",
	"images": "apple.jpg",
	"seller": "Poland.inc",
	"seller_link": "apple.com",
	"price": "20"
	}

Returns:

Success	{
	"msg": "Product added",
	"_id": "60cd49620a3a532bb49f67eb",
	"name": "apple"
	}

Error	Error creating

Product get

GET	localhost:3000/marketplace/60ccbb3f00236630fc0aabb5

RESPONSE BODY (JSON)	`{`
	`    "_id": "60cd49620a3a532bb49f67eb",`
	`    "name": "apple",`
	`    "description": "fruit",`
	`    "images": "apple.jpg",`
	`    "seller": "Poland.inc",`
	`    "seller_link": "apple.com",`
	`    "price": 20,`
	`    "__v": 0`
	`  }`

Product get all

GET	localhost:3000/marketplace

RESPONSE BODY (JSON)	`[`
	`  {`
	`    "_id": "60cd49620a3a532bb49f67eb",`
	`    "name": "apple",`
	`    "description": "fruit",`
	`    "images": "apple.jpg",`
	`    "seller": "Poland.inc",`
	`    "seller_link": "apple.com",`
	`    "price": 20,`
	`    "__v": 0`
	`  }`
	`]`

Marketplace remove

DELETE	localhost:3000/marketplace/{id}

Returns

Success	
Error	Error deleting

VIRTUAL COURSES

Course add

POST	localhost:3000/marketplace

BODY (JSON)	{
	"name": "food v2",
	"description": "reduce food waste",
	"image" : "food.png",
	"fee" : "0",
	
	}

Returns:

Success	
Error	Error creating

Course get

GET	localhost:3000/vcourse/60ccbb3f00236630fc0aabb5
RESPONSE BODY (JSON)	

Course get all

GET	localhost:3000/vcourse

RESPONSE BODY (JSON)	

Course remove

DELETE	localhost:3000/vcourse/{id}

Returns

Success	
Error	Error deleting

Stripe

Create customer - creates stripe customer to process transaction

POST	localhost:3000/stripe/customer
BODY (JSON)	`{ "name": "Xenos", "email": "xenos@gmail.com", "stripeToken" : "default", "card": { "number": "4242424242424242", "exp_month": 6, "exp_year": 2022, "cvc": "314" } }`

Returns:

Success	`{ "msg": "Suscesefull", "customer_id": "cus_JhVCvmdHHcJXIl", "paymentMethod_id": "pm_1J46C5HqSfcBXebdavqZOjYZ"`

	}
Error	Error

Donate - charges certain amount from account.

POST	localhost:3000/stripe/charge

BODY (JSON)	{
	    "name": "Deniel",
	    "email": "deniel840@gmail.com",
	    "feedbackid": "60ab9421eca9f320d0f78c92",
	    "customerid": "cus_JhUdwtx1OyuVwR",
	    "stripeToken": "default",
	    "amount": "10"
	}

Returns:

Success	Susceseful l, customer id: cus_JhVeZJ3d32sbMS donated 10
Error	Error 404

Subscribe - crates stripe subscription with int

POST	localhost:3000/stripe/subscribe

BODY (JSON)	{
	    "project_name" : "plant trees",
	    "feedbackid": "60ab9421eca9f320d0f78c92",
	    "customer_id": "cus_JhVCvmdHHcJXIl",
	    "paymentMethod_id" : "pm_1J46AKHqSfcBXebdCqlvGhY6",
	    "amount": "10",
	    "stripeToken": "default",
	    "interval" : "month"
	}

Returns:

Success	{
	    "id": "sub_JhHFh7pYabLnz5",
	    "object": "subscription",
	    "application_fee_percent": null,
	    "automatic_tax": {
	        "enabled": false
	    }, // too long to list.
	.....
	}

Error	Error