GO GREEN FINAL REPORT

If you want to change the world, start by changing yourself

Project GREEN

Group 23
Computer Science and Engineering
Jahson O'Dwyer Wha Binda - 4772288
Julius de Jeu - 4781775
Vlad Florea - 4791487
Paul Stepanov - 4849736
Kamron Geijsen - 4892836
Pablo Rodrigo Valero - 4902955

Table of Contents

Introduction

The project we have been developing along these nine weeks goes further from being able to design, program and presenting a Java application. It is indeed, an example of the power that cooperation has and the ambitious goals it can achieve. Furthermore, the topic of this project has allowed us to enlarge its nobel mission into something more than an application but a tool to raise awareness about the environmental challenge, we as a society, are currently facing. In order to achieve this, our personal motivation is to inculcate the users that in order to obtain bigger goals smaller changes in our daily habits have to be performed.

As a group, we have faced many decisive moments in which a correct direction for the project was not ensure, nevertheless we were always able to make a step-back, rethink the problems that were decelerating the development of the project and find alternative solution to overcome these difficulties. We dedicated some time to the organizational issues and this ended up being key to the development of our project.

This period of time, often underestimated, we believe it has been key to prevent unresolvable problems or conflicts that could have delayed or worsen the development of this project. From this, we have obtain an important take-away, planification is a necessary step before starting developing the actual product and is extremely useful to prevent unforced errors and multi-disciplinary conflicts.

Nevertheless, not everything was straightforward. During the learning process we also had some stages in which we had some difficulties, for example learning how to properly use git was a challenge for most of us. However, with practice, mutual help and perseverance we managed to make an adequate use of git.

Along this path, we have had to decide on technical choices, in which several factors have been taken into account, but some of the most common influencers facts are the visual impact, the unnecessary complexity for the programmer and the accessibility for the widest range possible of the society. In particular, the last influencer mentioned we have tried to make our application the most inclusive as possible. One example of this, we have implemented functionalities with the aim of making our application more accessible to the color blind minority.

We have tried to face this project not only as a technical challenge but also as an opportunity to learn from each other, and the many things that involve a project which can not be taught by a book.

This report has the aim to exhibit not only our final result, but also our perspective along the path that took us to achieve this result as well as, the causes of some both organizational and technical design choices and its consequences.

Product

Design choices

From the start we attempted to lay base rules of design choices. Are we going to decorate the program with animations and transitions or focus more on functionality and efficiency? Are we going to use the Swing or JavaFX as graphical user interface framework? We quickly converged to the same answer to both: we all wanted a program with plentiful animations and transitions, and decided to use the JavaFX framework because it was more up-to-date and makes use of Lambda functions. During that first meeting, we made a list of design protocols, which included:

Intuitive design

Having a universally understandable design, for beginners and for experienced users: Easy to understand and easy to use. As an example, the leader board graph shows the progress of a user, but experienced users can also see the legitimacy of their activities (whether they have sudden unrealistic spikes of points)

Coherent design

We designed a systematic page layout, to make the transition between pages more natural. We used the same button design, colour pallets and icons for the same ends.

Functional design

To group features together into one, to minimize button cluttering and optimize feature count. As an example, our Ring has the function of showing the user progress, but also in which category the user is doing well and other categories where the user can do a little extra effort.

Feature choices

Our features are distributed over four pages, the Main screen, Event page, Leaderboard page, and Profile page. Each page has its core feature, deducible from their name.

Ring

The main component of showing the user their progress is the Rings. These contain a detailed description of the users' green activities, while having an intuitive overview of their overall daily progress. This progress is measured in Green Points, which are linearly dependant on your estimated real world CO2 savings.

Green activities and points

To increase this score, the user can record their Green activities such as 'eating a vegetarian meal' or 'Installing solar panels'. These will be stored on the server and will be fed into the CoolClimate Web API to return your estimated CO2 savings. Note that we only record positive Green activities since it would be less satisfying to input your negative Green activities, and be more like a chore. That is not the aim of our project.

Followers

We designed our following schema to be one-sided, such as in the popular social networking service Instagram. You can follow a user and they will be notified that you are now following them, and they get the chance to follow you back. Following a person implies that they will show up in your leaderboard, so you can compare your scores with them. Followers do not need to be mutual, so everyone is allowed to follow you but you can choose not to follow them back.

Profiles

Each person has their own profile, which can be used to display their recent Green activities and achievements. Each person has their own unique ways of contributing to our environment, which will be expressed on the Profile page.

Events

Users can organize events to meet other people and realize environmental-friendly activities such as beach cleaning, water reuse...

Technical choices

For the backend, which includes client-side HTTP communications, and the entire server-side, we early on decided that we should use Spring framework as server-client communication, and relational model PostgresSQL for the database functionalities.

Spring framework

We did this mostly because it can do everything we need to do, and since it has a large library of sources that we could use for examples. Spring has a lot of functions that we can use to make our life easier, such as a default json output. It is however very heavy weight, and looking at the features we actually used we perhaps should have used some alternative library.
Following that, some alternatives for Spring are for instance SparkJava, which is smaller and thus more lightweight, but harder to configure and Javalin, which looks a lot like SparkJava but is easier to configure, but it is documented a bit worse

Server-side logic-based

We attempted to do all the retrieval queries on the server-side, to minimize both the communication bandwidth and the amount of client-side calculations. The client should run as smoothly as possible for the best user experience.
The server can theoretically handle unlimited clients under normal load, as it is runs thread-safe, given that there will be some lag. Since the database and the server are hosted on separate hosts they don’t experience any delays amongst each other. It is however important that there is an internet connection available to both, since they need to communicate.

Server - client communication

The communication can probably be improved, since we use the same query with some different parameters for instance when requesting the followers or the people a user is following, all that changes is the url, even the method is the same.

Process

On paper our group had a clear separation between client-side programmers, and server-side programmers (and one in the middle connecting the two sides). Though in reality we all collectively came up with ideas and improvements for each side.

Communication

Most of our communication was done using chat messages, meetings and GitLab issues and replies. Important matters such as new features or code organisation were discussed during the meetings. During these meetings we would create a to-do list, and convert it into Issues on the Scrum board in GitLab. We would use the chat when we had quick updates or questions while working on the issues.

Meetings

We had meetings two times each week, one mandatory meeting with the TA and for the other we would reserve conference rooms (mainly in the library). While waiting for everyone to be present, we would do some general chit-chatting, this resulted in some pretty good ideas. When everyone had arrived, we would share our work from the past week (without any specific organisation) and note the bugs and unfinished features. This was mostly done to tell all of us how far we got, and as a preparation for the sprint reflection. Then we would all converge to the decision of starting the formal meeting, where we would compare our developments with the past sprint planning. The chairman would then make the sprint retrospective, and immediately after the new sprint planning. At this point we were ready for the meeting summary with our TA. The TA would give feedback and additional information about recent and future points of actions or deadlines. At this point the meeting was mostly over, but we usually stayed to discuss the features in more detail and occasionally we would implement the features during this period.

Responsibility and task division

During each meeting we discussed issues that we wanted to work on. We were always able to debate using arguments or find a middle way.
After discussing the issues we quickly assigned the issues to the person or people whose tasks would be affected by the issue. Each person has their own assigned task cluster which is carefully chosen to be closely related to each other, with as little as possible overlap to prevent merge conflicts and to optimise ‘parallel’ working.

Code and Git Project

We worked using Git for version control, as instructed, and used branch/pull based development for our application. We divided our work and worked on our own branch most of the time, this was to prevent accidentally editing others code that was not yet pushed. Sometimes we did work together on a branch, but this was mostly when two people were working in pair programming, since then you could sit next to each other to make sure you didn’t edit the same file. Our branches are a bit of a mess because we didn’t agree on a format of our branches.
Our project was published and checked on the EWI GitLab, which was also used for our CI services. We set up a custom CI in the first week since instead of using Maven we opted to use Gradle. This was mostly because some of us had more experience with it and because it is used more and more compared to Maven.

Reflection

Regarding the process, we believe that the design choices that we took led to a consistent user interface and a database with complete functionality. However, we did run into a few issues.

In the very beginning of the project, we encountered issues with communication, specifically regarding the features that were to be implemented. Since our concept for the app was not yet well-defined, some of us had different visions for the way everything should work and we did not fully make use of GitLab and Sprint planning in order to arrive at a singular plan for the future. This was remedied in later weeks, but it did slow down the connection of features with one another.

Also, we did not use optimal coding practices with regards to branches, and had multiple people in the client team working on the same branch for portions of the project. Despite this not causing any major problems, if communication between the team had been worse, this could have led to confusion and discord between the members.

Finally, we went through multiple iterations of the profile page and leaderboard due to not being able to settle on a singular style for the first few weeks. This led to the GUI team taking longer to implement certain features than the Server and Database team, and caused the linking of the two sides to take slightly longer than expected.

What we can take from this for future projects, is to make proper use of Git and to create GitLab issues in the beginning of the project. Due to the underuse of issues in this period, some of the members had slightly unclear ideas with regards to the features that had the highest priority, and subsequently caused delays in the addition of said features. In spite of this delay, we did achieve a cohesive look for the entire program, and all the functionality that we planned on adding.

Individual Reflection

Julius

I have learnt a lot during the project. Mostly to not judge a book by its cover and that leading a project group is harder than it looks. It was a learning experience for me to work on a programming related project with this many enthusiastic people. I have learnt a lot about client/server communication, since I wrote a lot of smaller applications, but never one with such a clear target as this one, it has also taught me about the importance of communicating clearly with my groupmates, since I had to redo both sides multiple times until both sides got what they wanted. I have also learnt quite a lot about databases, particularly postgresql, since I did the initial setup for it and had to edit some queries for some requests to work. This project has also taught me to work together with people with different code styles, since checkstyle enforces some but not all things that I do, so I had to change some things about how I coded to make sure the others understood how I worked. The project has also taught me that I need to add comments to my methods, and maybe explain what some lines do since it confused me also some times.

Paul

This project has been one of the subjects that helped me the most since I started my study because I gained a lot of experience. First of all, it helped me realize I enjoy working with database and this may contribute to some future decisions I will have to make (for example when I will have to choose electives in the future years, or maybe even master and career). Furthermore, it increased my programming level, making me a better programmer, because I had to search a lot of new things in order to complete the tasks I was assigned to). The fact that we had deadlines to meet and a “customer” to satisfy made all the experience look as real as possible for this level of knowledge that we currently have. Last but not least, I learned a lot about how a real project should take place and how you have to communicate with the other member, how to plan meetings, how to write reports, make decisions and helped me familiarize with widely used softwares like gitlab, gitkraken, etc. and with the “techniques” they use, as pushing and pulling or writing issues for yourself or your colleagues. I was also lucky that I could work with people who gave their interest in putting work of good quality and with whom I could communicate easily. In conclusion, I personally feel like this project has been a really big step forward towards my personal development.

Vlad

I feel like this project has been not only a fun learning experience, but also one that helped me get a much better grasp on how I should work as part of a team and coordinate with fellow programmers. Working on the graphical user interface helped me get to familiarise myself with working in Java and learn some of the intricacies of the language, and I feel more comfortable working with Git as well. Checkstyle and working with a team also helped me get rid of some bad programming practices (for the most part), such as leaving uncommented code, inconsistent variable naming and copying code from one section of the program to another. Along with Git and Java, I have also learned to use JavaFX Scene Builder effectively and combine HTML with CSS code in order to create a cohesive look for the program. I am lucky to have been matched with a friendly and communicative team, most of whom are above my skill level and who put their interest into delivering quality work, which helped me stay motivated to work on the project and made for an enjoyable and streamlined learning experience. Finally, I feel that this project has been a good stepping stone towards learning how to write code as part of a team, and has helped me overcome some of the hurdles on the road to becoming a better programmer.

Pablo

This project is much more than a simple block of code put together which hopefully works and is graded upon the final result. In my case, it has also helped me to have a better understanding of what engineering since I have had the opportunity to see the application of the knowledge absorbed in the first semester of our degree in a real-world scenario like this one. I would also like to remark the nice atmosphere of my group which made this process even more enjoyable.
They all give their best and it has been an honour to be accompanied by this group along this path. To sum up, within this project we have learned some basic management skills also related to engineering such us code organization, the use of git, and a key concept such us *pair programming*.
Respecting the TA, I think he has done a great job, since he balanced successfully his duties as a customer demanding some functionalities or spotting bugs and its support as a TA in which he was always available to clarify any doubt we have had. To conclude, I believed that this course is one of the courses I have learnt the most since I admire the methodology learning-by-doing since for me, the best way to learn something is to be obliged to use it and researching until you understand it and you are able to implement it.

Jahson

To be honest when I first started this project I wondered how I would be able to finish it. I thought my knowledge in Java would not be enough and that I may hold back my team. To my relief it seems like my worries were for nothing. Not only was I able to stand on equal footing with the rest of my team, but also discovered that I may know and am capable of more than I thought.
Of course with the knowledge and skills I had at the start of the quarter was not enough to be able to produce what was expected of me. But I have learnt a useful skill, which is to learn to fill gaps in my knowledge rather than learn how to do something because I would be asked it in an exam. To think in a way where I wanted something to be done and learn what steps are needed to make it a reality.
Not only in my Java skills but as well as working in a team, I have never worked with such a large group of people before and practicing using environments like Git as well as meeting communicating with a group of people in a structured format feels like something I can take with me for the foreseeable future.

Kamron

I am very glad about how this project has turned out. Not only happy with the final product, but mostly with the attitude and coherence of our group. We each had our pro's and con's which filled each other in. When the one struggles with database, the other gladly takes the task. From the start we realized that communication between members was going to be key in this project, which turned out a great success. During out meetings we shared our recent progress and discussed future improvements, with and without the use of the sprint planning.
The first two weeks of the project really got me worried, as I for the first real time I felt like I was in unknown territory, using JavaFX instead of Swing, using GitLab instead of standard messaging of classes. The list goes on. I had stumbled upon a huge speedbump at the beginning of the project, more than half of the time not knowing what to do. Though I wasn't alone, since my teammates also had little experience with those subjects, I could learn together with them. As the time progressed, and we all started to get a hang of the project and each other. The sphere in the meetings was always good, everyone was always present (Except for one time when I was sick). I enjoyed discussing new features and their organisation, and I genuinely enjoyed my role in client-side.

Value Sensitive Design

Ethics is all around us, and the field of Engineering is not an exception. However, the implications of ethics and values can sometimes be not obvious. During this project, we have also encountered with decision that had ethical implications and that influence our outcome of this decisions. Some of implications are regarding privacy, information and safety. Nevertheless, ethics goes much further arising fundamental questions such as which are the stakeholders taken into account when designing the product, are minorities respected when designing the applications and is the application accessible for all collectives.

The stakeholders of our application are very diverse, the most obvious ones are the users that are going to make use of the application. Nevertheless, an import sector of the stakeholders are the industries affected by the main activities that contribute to the C02 consumption. For instance, the energy sector is directly related with the behaviour of the user with our application since deciding to use green energy will have a direct impact on non-renewable energy industries. Nevertheless, our application was design to encourage users to reduce C02 so we decided not to add features that are not environmentally clean. However, if we would have done it, we would degrade the image of these industries.

Therefore, we decided to focus our design on the stakeholders that indeed contribute to a cleaner usage of resources, green energy sectors, usage of public transport, carpool, and usage of LED lights. The reason for this, is to motivate the user to change his habits in order to become a world-saver.

In our application, we have tried to include as many values as possible but we have mainly focused on three: inclusion, ecological respect and perseverance. Inclusions has been a must for us since we believe that all collectives and minorities have equal rights and should be taken into account when designing any product to ensure its accessibility. Ecological respect is implicit in the nobel mission of this project, since its aim is to encourage users to change its habits in order to improve the world in which we live. And the latter, we believe changing your daily habits is not easy and therefore, we have design the application with the goal of rewarding the user for his effort in contributing to a cleaner planet.

Furthermore, our goal has always been to make our application the more accessible as possible to all minorities, and we decided to focus on helping a often ignored minority, the color blind. After analyzing their difficulties, we conclude this minority would struggle the most detecting the different colours of the ring located at the user's main page which divides the activities that contribute to the C02 reduction into different categories, distinguishable by colour. Hence, we decided to add extra information about the categories when the user hovers over the different categories.

Moreover, privacy is undoubtedly one of the biggest concerns for users, and therefore we had the moral obligation to take this preoccupation into account when managing user's personal data. In order to deal with this issue, we hash the passwords ensuring if these are intercept it they can not be used without the allowance of the user.

We can conclude that ethics is a determinant aspect that has clear influence in many of the decisions along the process of developing a product. We have tried to take this considerations into account when brainstorming and designing our project and we can affirm we are now more aware of the importance of ethics in engineering.

Appendix 1: Pictures

Fig 1: Our log in screen

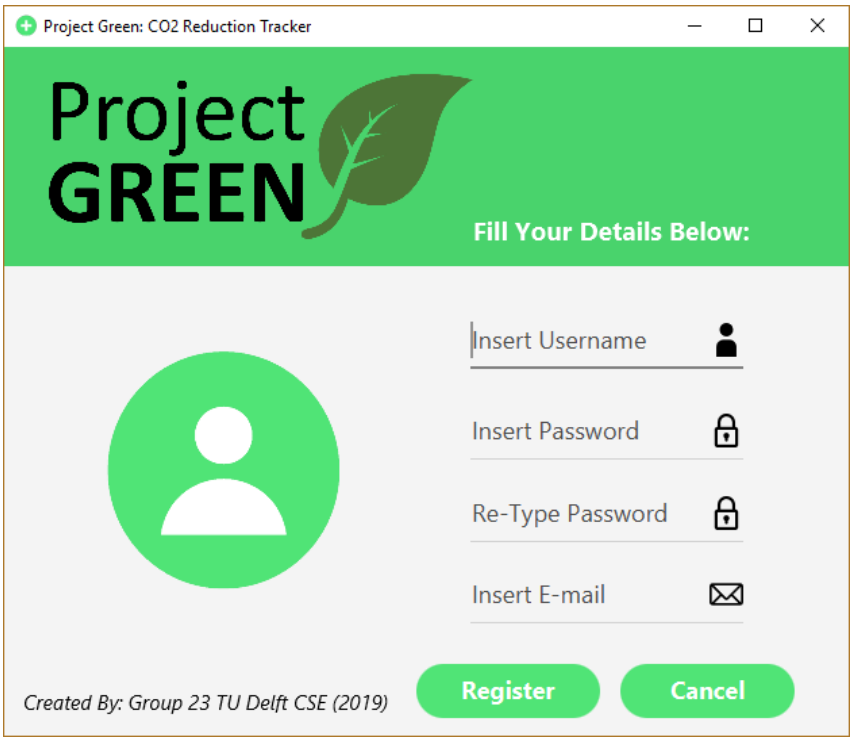

Fig 2: Our register screen

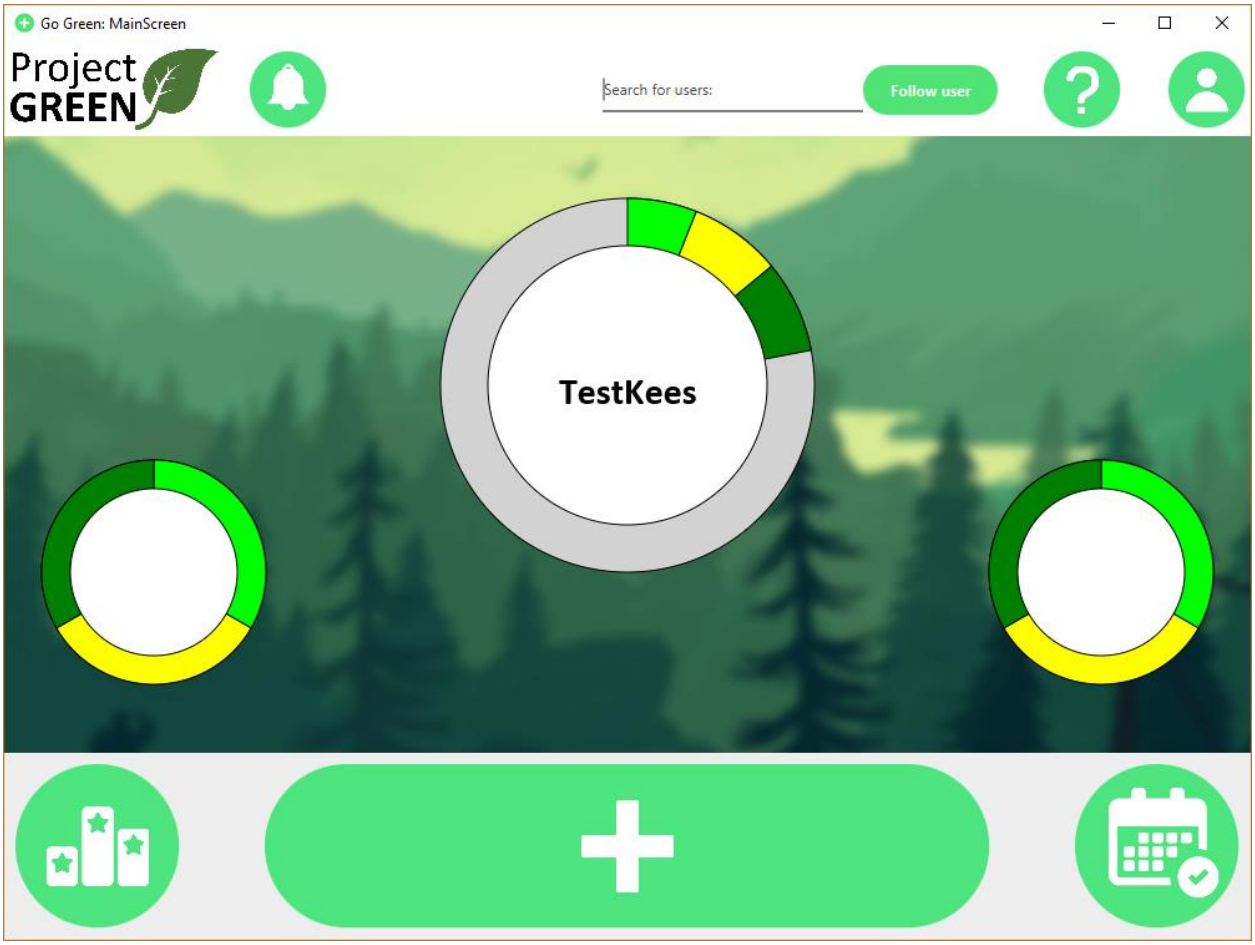

Fig 3: Our main screen, the different colours in the big circle tell you about progress in the separate categories.

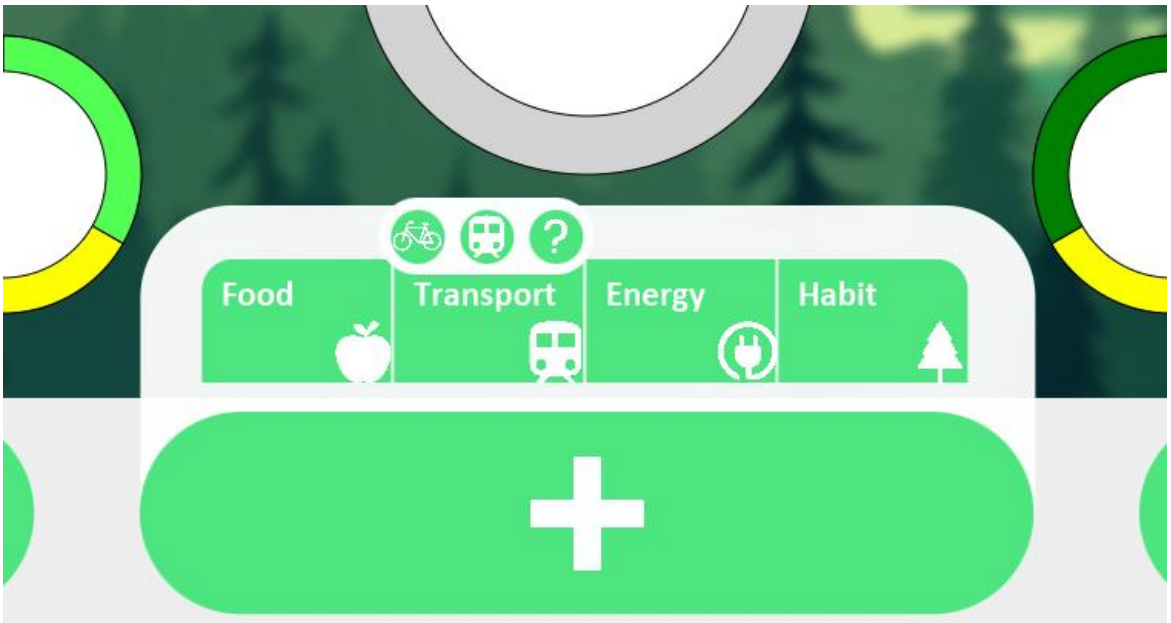

Fig 4: Adding a CO2 saving activity

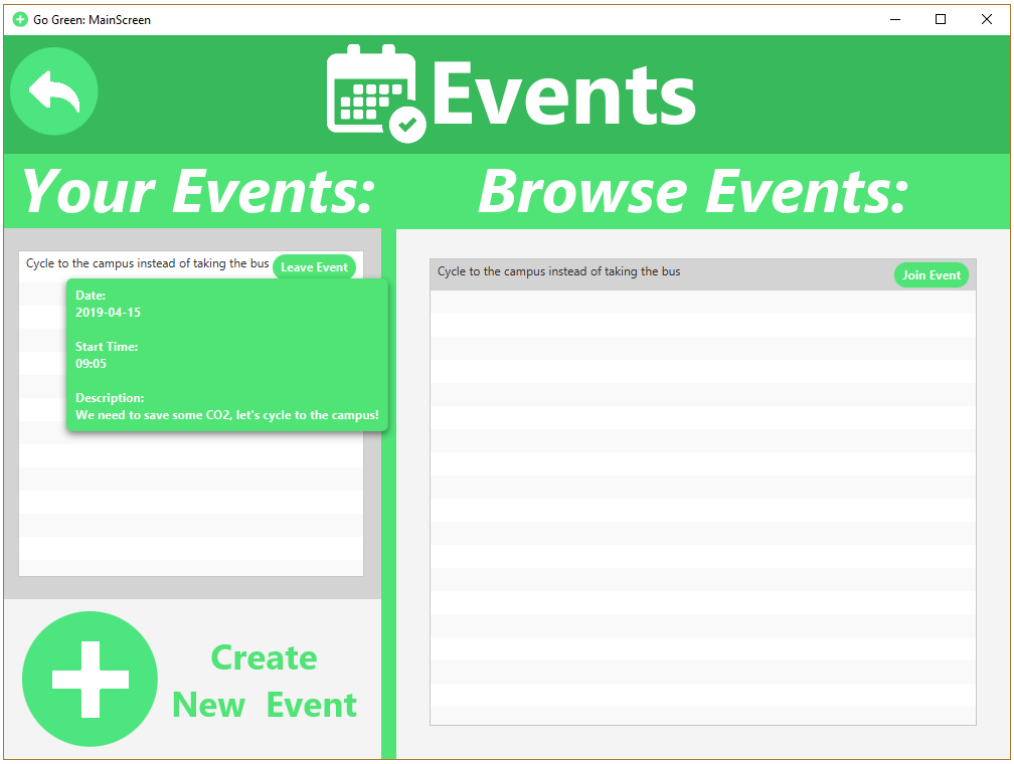

Fig 5: Our events page, with one event that the user has joined

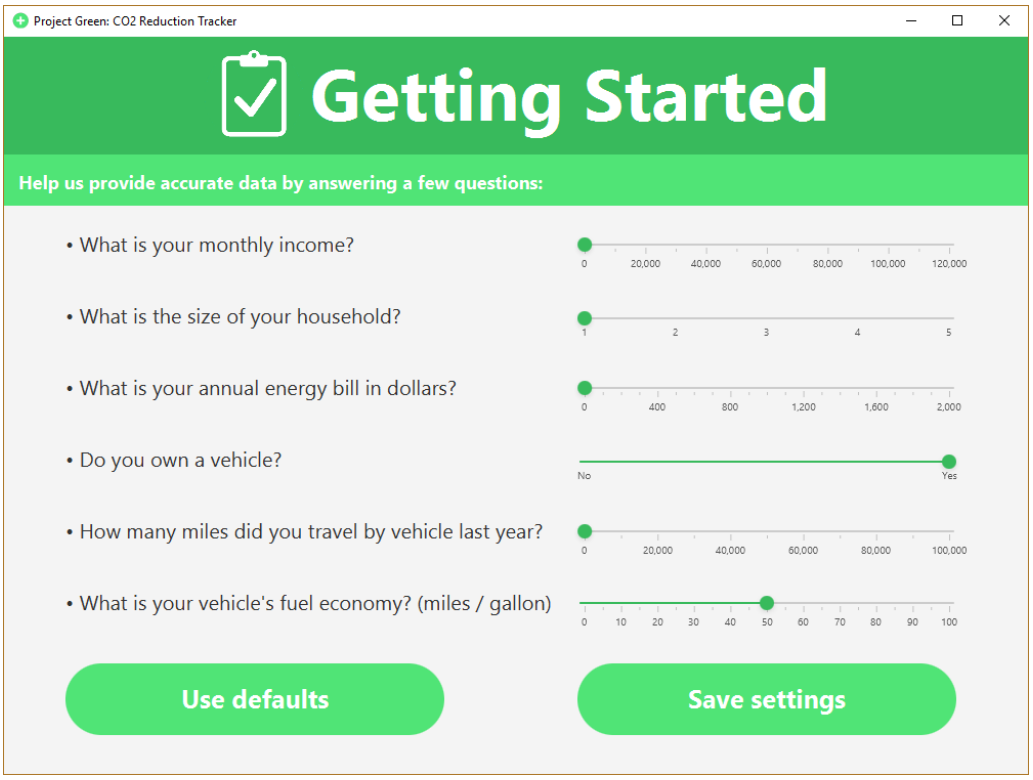

Fig 6: Our intro quiz

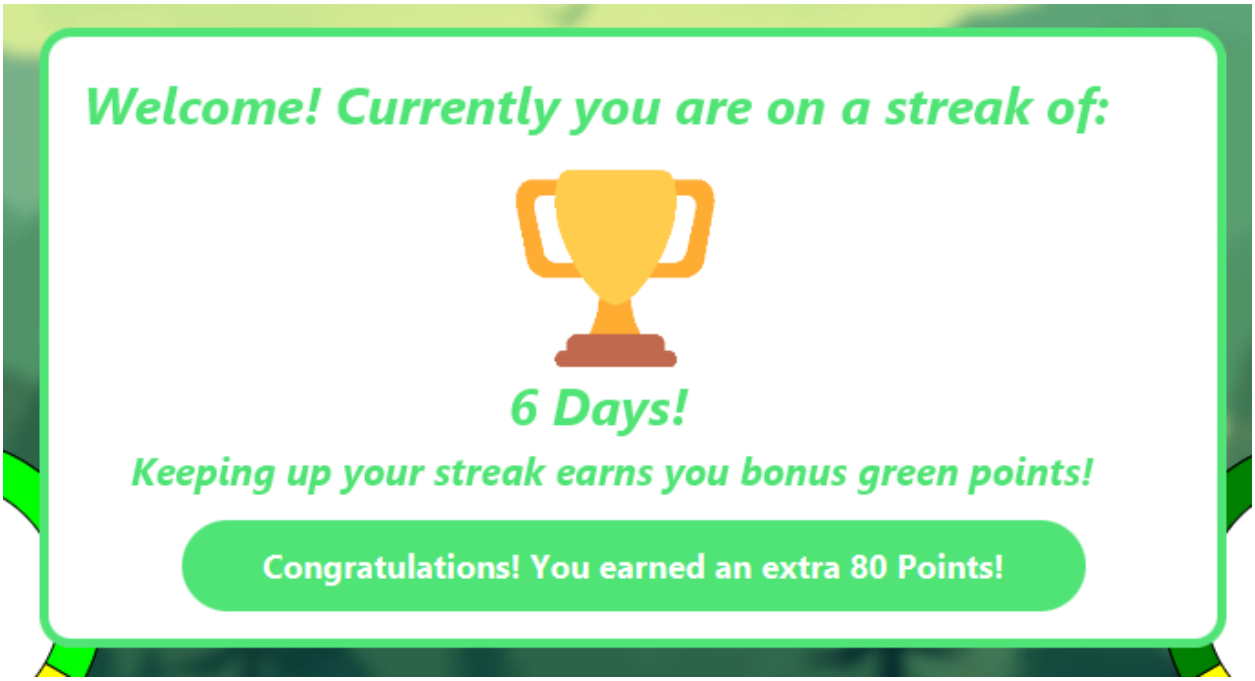

Fig 7: The popup when you have a streak going

Fig 8: The profile screen