

Final report OOP Project Group 43

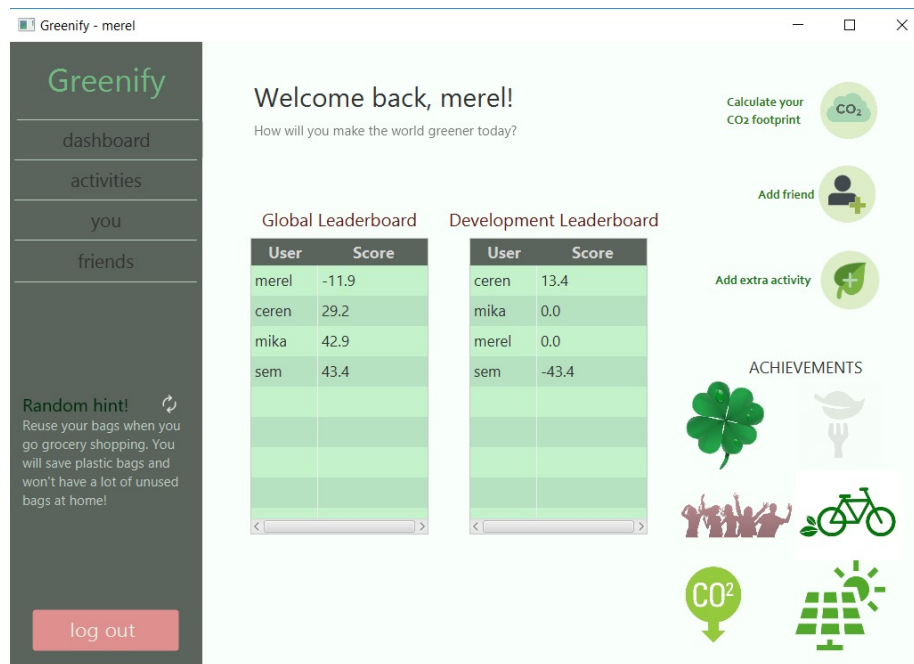
Sem van der Hoeven, svanderhoeven, 4896726

Merel Steenbergen, masteenberg, 4784871

Ceren Ugurlu, cugurlu, 4851609

Mika Wauben, mlwauben, 4834739

April 11, 2019



Welcome to Greenify!

1 General

It's always hard to start something new with people you don't know, but luckily we were all excited to start. After a week of research we were ready to begin working and coding. We started forming teams, but were a bit overenthusiastic with that. Three people were assigned to work on the GUI, while we didn't even need a GUI back then. When we realized that, we all focussed on API's and putting up a framework. After that we started planning better and using the Scrum board to our advantage. There were some minor communication errors, but we resolved them in a way we were all happy about. Next to the meeting with the TA, we had at least one meeting per week to discuss technical matters. For quick questions, we have a WhatsApp group. We learned that, in a project, everything has something to do with everything, so you cannot just start to work on something because you want to.

Our planning actually went well. Everyone did their tasks and nothing was ever done too late. We also weren't afraid to ask others for help. Maybe it has something to do with the fact that we were with only four members from week 4. We were all aware that we needed to work a little harder than the other students to make sure our app was finished on time.

We all had a very clear goal in mind. Not only do we want to pass this course, we also want to learn something from it. We wanted to get experience in working in a team and were prepared to try this as best as we could.

2 Design decisions

Technical

We decided to use Spring and Gradle very soon in the project. Spring had a few useful tutorials and the Spring framework was exactly what we needed. The Spring documentation was also very useful, so we thought this would be our best option. The Spring documentation gave the choice between Maven and Gradle, but upon doing some research, we found that Gradle's performance overall is better. It was also very clear from the beginning that JavaFX would be used for our GUI, it was recommended in the lecture and just seemed easiest. It took some time to get familiar with Gradle, Spring and JavaFX, but eventually using these frameworks was easier than doing it all ourselves. The Spring documentation is very extensive, it even has a tutorial and documentation on how to connect your application with JPA. This way the database was created.

The screenshot shows a window titled "Add extra activity - merel" with standard window controls. The main heading is "Using your public transport instead of your car". Below this, the text asks: "How many km did you travel using public transport that you would have traveled with your car?". A green input box contains "0 km". Below the input is a horizontal slider with a green leaf icon at the 0 mark and a scale from 0 to 200 in increments of 20. To the left of the slider is a vertical stack of six icons: a bowl of fruit, a shopping cart, a bicycle, a bus, a house with a lightbulb, and a solar panel. Below the slider is a green circular button with a white leaf and a plus sign, with the text "Add activity!" underneath it.

Figure 1: Add an activity

Usage

Our approach is a bit different than that of most other groups. Upon registering in our application, a calculator is opened to calculate your CO₂-footprint. The users score is based on that first calculation. From that moment on, it's possible to add certain activities and calculate the CO₂-footprint again. The score is based on the amount of CO₂ a user can save in a year with their new habits. The CoolClimate API has helped us with this. It has everything our application needed. The plan was to use the calculator to calculate the CO₂-footprint and add the activities as an extra. The extra activities would then influence the users CO₂-footprint, instead of a score that's vaguely based on averages. This way, it's also easy to implement other activities that users can do to decrease their footprint. (See figure 1)

3 Points for improvement

We have a lot of ideas that we would like to add to our application, but we didn't find the time for all of them. Of course, we could go a lot more in depth on the green actions you could take. We could've added some minigames or levels to add more gamification. Also, maybe not all our code is very efficient, but that might be the effect of working with new things like Spring and Gradle. We are very proud of our test coverage. Testing is one of the most important parts of coding, since it tells you if the code does what it needs to do. When writing an application, it's not desirable to have any hidden bugs, so testing is necessary.

There is not really any security on our code. The password the user enters in the password field is hashed and then sent to the server. If we want to make this an application open to the public, we have to encrypt the data of the users better than that.

Since our group ended up with four members, instead of seven, not all extra features could be implemented. For example: Friend requests instead of followers. Now a friend is just someone you follow. Profile pictures would make a user profile more personal. A 'remember me' function which remembers the users login credentials makes using the application even easier. There's a lot of small changes that can be made to improve the application.

On communication there has been a lot of improving. There were learning moments, but since this course is here to learn, that's okay. We all know now that a project has its ups and downs and that it is your job to make sure there are more ups than downs. The Scrum board has really helped the team with planning, dividing tasks and made it clear to see what needed to be done for our next demo. Git is very nice to make sure everyone can work on their own part of code, without interfering with others, while still working on the same project. It provides the team with the freedom to break code entirely, while still always having a working version.

We do feel that there are some points the course can improve on. Especially the start-up was rough. We had no experience with Java servers or on APIs. We know this course is about finding your own way and getting your own information, but still we would like to see a little more help in the beginning. Another thing we noticed is that the TA's weren't always up to date on the info: Some TA's said different things to their groups than ours told us. We don't think this has to do with the TA's, but that it has to do with the management of the course.

4 Individual feedback

Mika Wauben

Very early it already became clear to me that I did not have the skills required to do this project yet. While the others knew how to do the first demo within the first week, I still had struggles with this piece of code three weeks after we handed it in. By that time we already had to hand in the second demo, but still I was of no use, since I had little idea of what was going on. During this time I made myself useful by creating most of the documentation and by fixing the layout of the repository.

After the sixth week, I finally somewhat understood the code enough to actually contribute something and by now I can work together with my teammates and understand what they are doing and have been doing for the past few weeks. Since we are only with four people left, communication has been relatively easy. Therefore not many conflicts have occurred. The only time was when Merel and I worked on some piece of code all weekend, which Ceren deleted afterwards, but this was due to a misunderstanding about the design of our application. We communicated this and solved the issue the day after it happened.

Sem van der Hoeven

During this project I have definitely learned a lot. The thing I was most excited/curious about was JavaFX and making a GUI in general. I therefore am very happy that I was able to do most of the GUI coding for this project.

My strong points were that I was very passionate about my tasks. I sometimes spent more time than I'd like to admit on placing elements exactly in a particular spot, or changing the colour of an element ever so slightly to match it perfectly with the rest. As a result of this my weak points were that I sometimes spent (way) too much time on particular tasks, and then I would have to stress in the weekends to get my tasks done for the next meeting.

What I also learned a lot about which I didn't expect to learn a lot about is version control. I found out there are a lot of ways to control and document the entire lifeline of your project, and how to do commit messages and branch names properly. I even found out how some companies do their commits and branches.

Another thing I learned a lot about was working in a team. I learned a lot about how to cooperate in the best way and what to do/don't do. My weakness on this part was probably wanting to do things myself a lot. I sometimes had to remind myself that particular tasks were assigned to other people. It was unfortunate that we ended up with 4 people instead of 7, but I'm very proud of our group and what we have made.

Ceren Ugurlu

The project process was both good and rough for me. During the project, I have learned a lot and each step we took was a challenge for me. At the beginning of the project, I even did not know the names of the libraries and tools we have used. I was really excited/interested in server side, communication between server-client and creating a database. Thus, I worked on these parts during the project. This makes me really happy because I had a chance to do research on these parts and developing myself in this area. I believe that I had a good contribution to my team because I was able to do most of the coding of the parts I am responsible for.

My strong points are my determination and passion about my responsibilities. I tried to do my best and I spent a long time doing this. I studied on the parts that I didn't have any experience with. I did not give up even when I was forced. I continued to study and finished my tasks on time. After I studied on the database, server and connecting the methods between client and server, I helped my groupmate on the GUI in last weeks.

My weak point is that I was not good at group communication at the beginning, because I have not experienced this kind of project before. But I believe that I solved this problem with the help of my groupmates and we became a good team. We always tried to help each other. We shared what we had to do according to each group members abilities. We had just four people left, this has both advantages and disadvantages. The disadvantage is that, when you work with only 4 people, you have more responsibilities. But at the same time, it is a really big advantage for communication. Communicating is easier with fewer people. So the communication in our group developed over time. Therefore, I do not think that we encountered any big communication problem. We always tried to inform each other on all steps. We were all open to new ideas and gave suggestions to each other. As a result, I am very proud of our group and happy about the result.

Merel Steenbergen

One of my biggest pitfalls is wanting to lead. I find it hard to let other people do their tasks without interfering myself. During this project I tried not to be bossy, but to still let my opinion be known. I feel I did succeed in doing this, but it wasn't easy. When my team members had a different opinion than I did on the design and technical decisions of the app, I really had to remind myself that I was working in a team and couldn't make all the decisions by myself. It's just not possible to all agree on all small details of a project and I have to get used to this.

Since I did a lot of group projects in high school, I learned how to resolve problems and I feel that is really going to help me later on. Again, I learned that you should always say it if something is bothering you, otherwise you'll just be cropping up frustration and that doesn't help the project. You should keep the goal in mind and let things go once they're not relevant anymore.

I did encounter, again, that I really like structure, especially in group projects. Even though it might not seem like it, I'm a very structured person. It's just that half of my schedule is in my head. That's where the challenge is for me: Other people can't read my mind, so I have to let them know what I'm doing and when I'm going to do it. The Scrum board really has helped me in this, since every part was clearly separated, had one or multiple assignees and there was a possibility to add comments. This gave me insight in what others were doing and what I was supposed to be doing. It was really useful.

In short, I worked really hard on my weak points. I think that's also one of my strong points: Always wanting to improve and use skills I acquired earlier. This project has helped me improve myself in group projects and has given me some insight on how working in code development with a team differs from writing code by yourself.

5 Value sensitive design

This project was the first time we had to think about responsible computer science, or ethics. There are multiple sides to this. We will start with data security. We can immediately admit that our code isn't secure. Passwords are now sent from client to server in a hashed form, they're easily decrypted. It was already mentioned earlier in the report: If we want to make this application open to the public, user data needs to be encrypted better. This is not only to prevent hacking, but it's also the law. The application would also need a 'terms and conditions' to make let users know how their data is used. To adhere to the GDPR law, there's a lot of change needed.

We are aware of our lack of security. Therefore, decisions were made to save as little personal user data as possible. For example, the application doesn't have a 'forgot password' function, since the user doesn't have to enter their email-address.

Of course, the application itself does help people improve themselves. The app contributes to the awareness about CO₂-emissions. The gamification encourages users to rethink their daily life choices. If this application would become popular, it might make a difference in the attitude of the public towards the environment. If we really want to elaborate, we could add a pane that gives you information about the average CO₂-emissions of different stores and companies, to make sure the users can use this app in their daily life.

6 Extra features

Greenify is very easy to use. A lot of thought was put in making the life of the users easier and making the app more interactive. A part of the interactivity is that users can easily see the effect of their actions in the application: On every significant action an alert pops up. This happens when a new friend is added, the CO₂-footprint is calculated successfully or when a login-error occurs. (See figure 2)

The application is also very visually pleasing. The soft green background matches with all the other colours, the sliders of the calculator are leaves, the icons of the buttons match the colour scheme. All colours are matched with each other and colours are reused to create a minimal, but pretty application. (See figure 3)

To make the app feel more responsive, a lot of animations are added. This means a user can see the effect of, for example, hovering the mouse over a button. In the registration screen, the text bars slide in the screen. In the calculator, there's a small line underneath the section the user is filling in at the moment. Upon clicking a button, it becomes the colour of the background for a small amount of time and the screen fades into the new frame. (See figure 4)

To add to the gamification, 'Green hints' are displayed in the sidebar. They provide the user with arbitrary facts about the environment or extra actions they can take to become more aware of the environment. In the 'You' pane, there's a pie chart that shows the share of the extra activities like installing solar panels on your CO₂-footprint. (See figure 5)

There is one last extra feature that's implemented. It is the biggest part of our application, but since it wasn't a requirement for the project, we'll mention it as an extra feature: The CO₂-footprint calculator. It is a very elaborate calculator, where a user can fill in their usage of different services and goods. It's easy to work with, hard to break and it can easily be expanded if needed. (See figure 6)

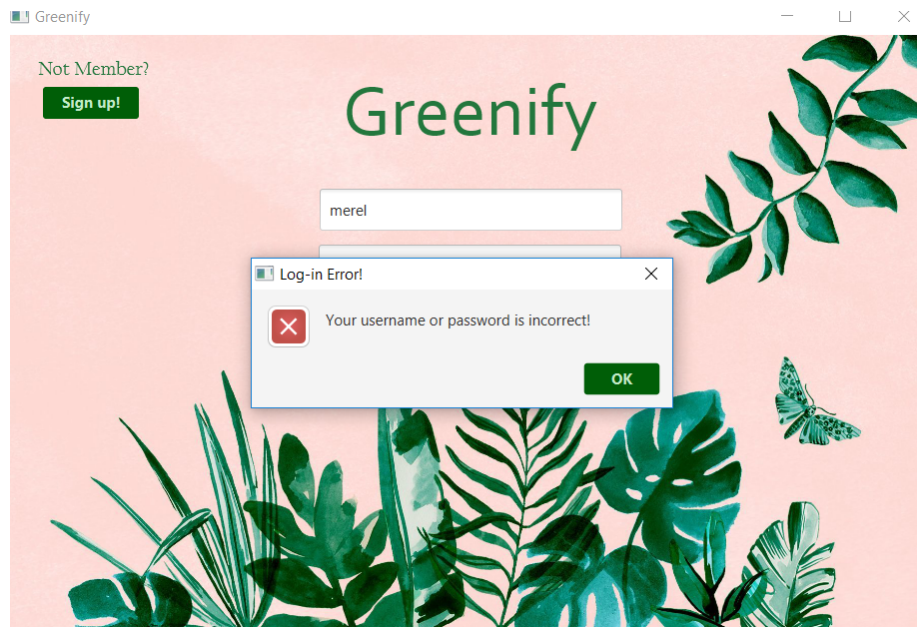


Figure 2: Alert

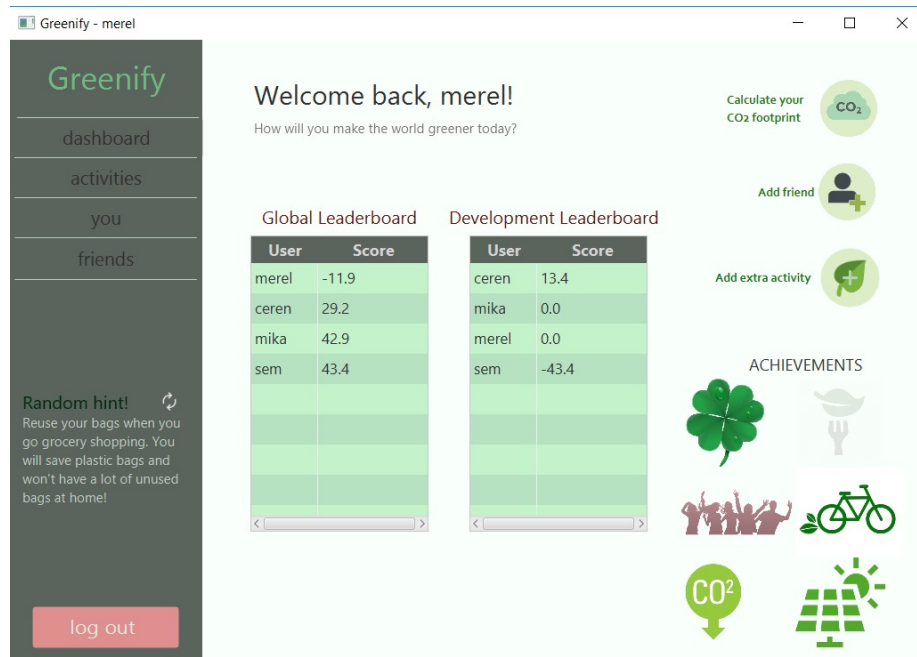


Figure 3: Dashboard

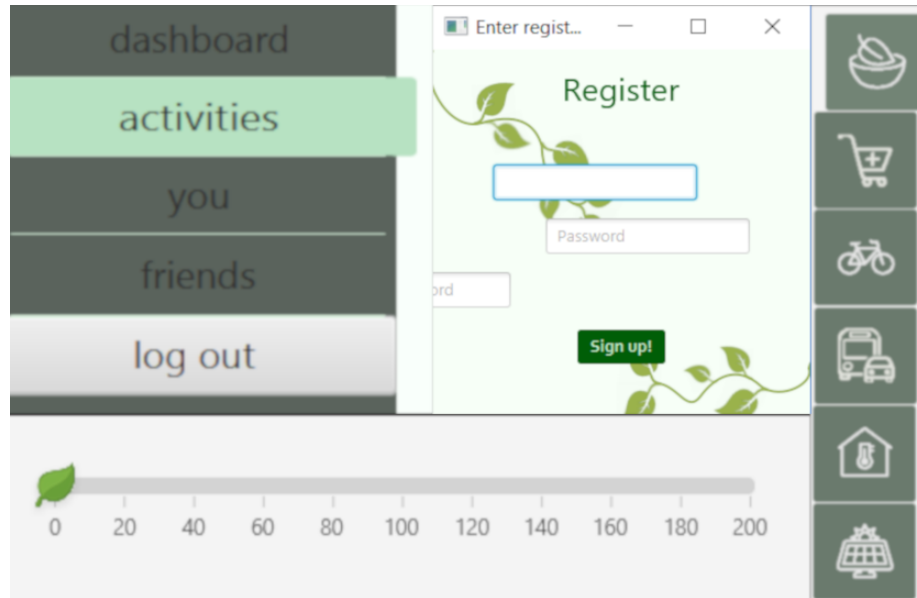


Figure 4: Animation

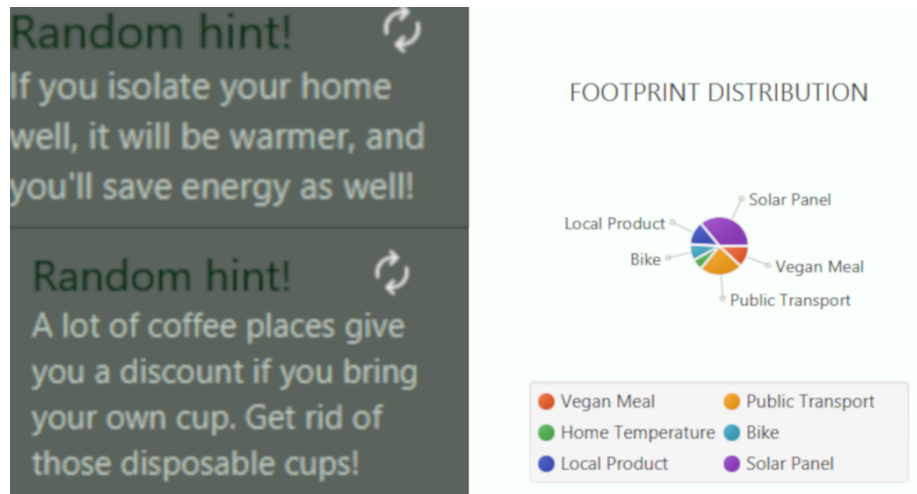







Figure 5: Hints and pie chart



Get started


Travel


Home


Food


Shopping


Extra

Save

How much do you use in your home?


Electricity

0

€/year

Percent purchased from clean energy program:

0 %



0

25

50

75

100

Natural gas

0

€/year

Heating oil & other fuels

0

€/year


Living space area

0

m²

Water usage

0% of similar households



0

100

200

300

Next

Figure 6: CO2 calculator

12