CSE 1105 (OOP Project)

Group 38

Final Report

Team members:

- Shashank Anand (4790987)
- Jean Louis Clemente (4900979)
- Vetle Grim Hjelmtvedt (4813928)
- Evaldas Latoškinas (4776798)
- Konstantinos Lyrakis (4799364)
- Alexandru Radu Moraru (4798961)
- Omar Thabet (4845463)

Product

Back End (Server)

One of the main decisions our group has made at the very beginning is to use the Spring framework library to build the server. The reason is because the Spring library is simple to use and adapt as well as providing all the necessary functionality that is simple to implement – REST controllers, database connection, definition, access & manipulation and HTTP requests.

For the database, the group has mutually decided to use MongoDB. Apart from this being a somewhat experimental choice to see how well it would work in practice, we have primarily chosen MongoDB due to it being part of the Spring data library, making it very simple to implement in the Spring server. Moreover, the JSON format of MongoDB is highly beneficial, because we did not have to worry about additional data mapping, as would be the case in other database solutions (namely SQL or Neo4J)

The Server components are as follows:

- Server Application (Server) – This is the class that starts the Spring server itself. It has no other code, apart from one method that starts the server.
- Database Service (DbService) – This is the server service that handles all the database interactions. All the database logic is contained here.
- Request Handlers (UserRequestHandler & AppRequestHandler) – The REST Controller that processes all the requests sent to the server and returns the appropriate responses. Each method has a mapping to indicate what should the URL look like.
- Repositories – The repositories (collections) that are part of the database. These need to be defined in order to store & access data in the MongoDB database. Mapping from object to the MongoDB BSON format is handled automatically by Spring.

Our Server generally follows the DAO pattern with regards to the database, which is highly beneficial, because we have successfully been able to logically separate the server code, which has been a significant benefit for code maintenance and for adding new features. All the data itself is stored in the Repositories, the database interactions are handled in the Database Service, and the requests are processed in the Request Handler. (See Figure 1)

Front End

On the front end part, the application is built using JavaFX, software used for designing desktop applications, such as GoGreen. The front end code is made up of several parts, which are split according to their individual categories.

These categories are :

1. *the Controllers* which handle most of the logic for the fxml pages
2. *the fxml pages* which are also referred to as scenes
3. *the graphical interface package* which contains several classes that handle general features that are used in multiple controllers such as :

- Confirm boxes
- Scene/Stage switching
- Event handling (Mouse clicks, Mouse hovers etc)
- Client Input Validation

4. *The requests class* which is used for sending requests to the serve, which are then handled with the RequestsHandler class.

For 1. *The Controllers,* the main strategy that was adopted was to have 1 controller that handles 1 fxml page. This strategy was used not only because it improves readability, but also because it enables quick access and debugging in case something breaks. Moreover, it reduces the number of code lines significantly than if all fxml files were to be handled by one controller.

For 2. *The fxml pages,* the tool that was mainly used for this project was Scene Builder, since it is easier than coding directly in JavaFX. However, in some instances, certain gui elements were coded directly from java, rather than using fxml. Moreover, a library that was used in a lot of the parts of the project was JFoenix, the main reason being that it contains a lot of gui elements that have a modern-looking style, JavaFX having outdated designs on the gui elements.

For 3. *The graphical interface package,* the classes are used in order to help shorten the amount of code in the controllers (All code could have been done just in one Controller, but then maintainability would have been impossible to achieve), but also to separate different features.

An important feature of the app is the MVC pattern that was adopted. In this way, it creates an easy way to separate the code into different parts, thus improving maintainability, a key feature that has been achieved, as mentioned previously above. In the case of the project, the Model consists of the User class and the Activity classes in the backend, the View consists of the fxml pages and the Controller is made up of the controller classes. For an example of the GUI, see Figure 2.

Logic

The main workflow of the application is as follows:

1. Users signs up for the first time and fill in a questionnaire regarding their current eating and transportation habits and the amount of energy they consume.
2. Users can perform different environmental friendly activities and receive feedback on how much CO_2 they save.
3. Users can always track their progress and immediately compare it with other users they have added as friends.

To calculate the CO_2 saved, after receiving the questionnaire, the application calculates the CO_2 emissions of the user for each different category (food, transportation, energy). These numbers are then divided by 365 and a user’s daily emissions are stored in the database.

It was decided to store the daily emissions, so that the amount of CO_2 a user can save each day is limited by them. This makes the application more engaging and helps users build new habits that will last.

On the technical aspect, it was decided not to use an api to calculate the CO_2 emissions, but instead create a Carbon Calculator class based on information found on environmentally-oriented websites[1][2]. This made the application faster since there is no need to make requests to an external server and more reliable since it does not depend on the availability of an API to be functional.

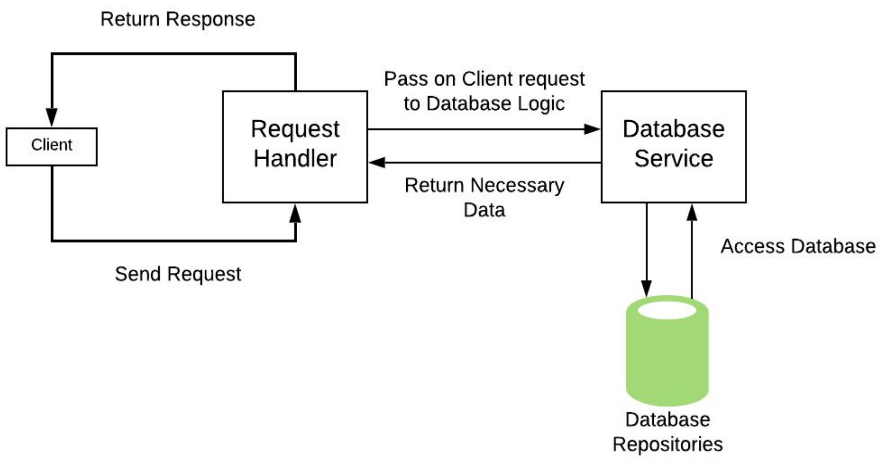
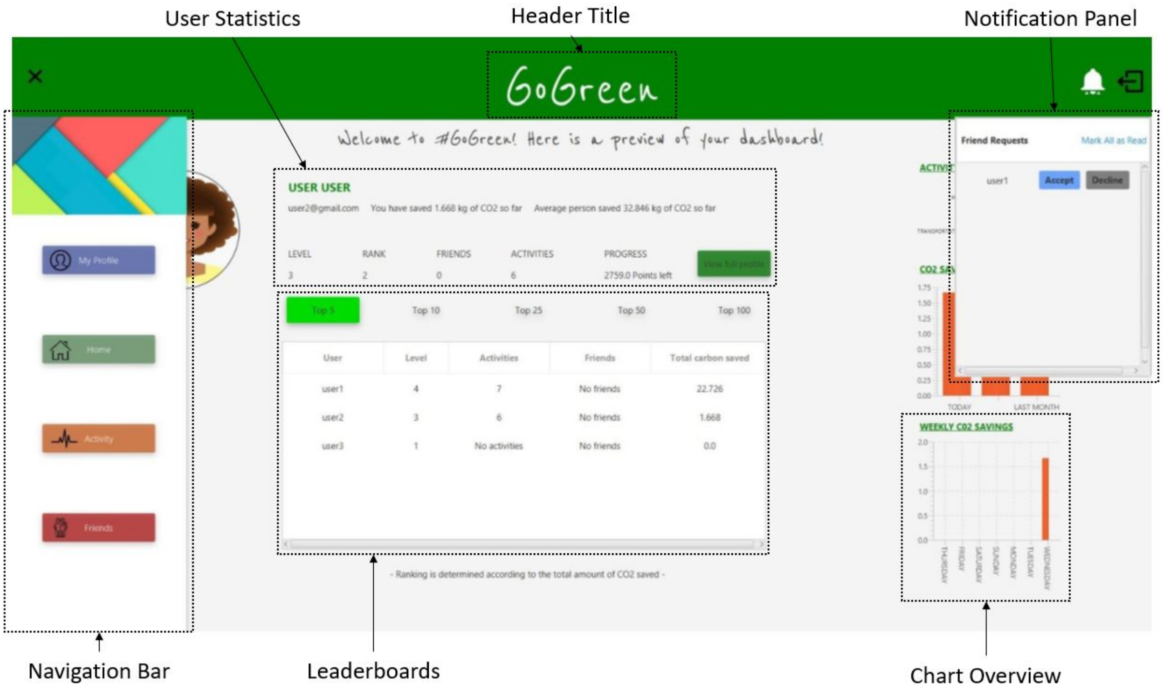


Figure 1 (Server Workflow)



USER USER

user2@gmail.com You have saved 1.668 kg of CO2 so far Average person saved 32.846 kg of CO2 so far

LEVEL	RANK	FRIENDS	ACTIVITIES	PROGRESS
3	2	0	6	2759.0 Points left

User	Level	Activities	Friends	Total carbon saved
user1	4	7	No friends	22.726
user2	3	6	No friends	1.668
user3	1	No activities	No friends	0.0

- Ranking is determined according to the total amount of CO2 saved -

user1

Figure 2 (Main Dashboard)

[1] https://www.carbonindependent.org/
[2] https://www.directenergy.com/blog/how-much-can-you-save-by-adjusting-your-thermostat/.

Process

Already in the first meeting, we divided the team into two. One would do the frontend, the other would do the backend. I believe this simplified our process significantly. When you are working in larger teams, communication overhead can obstruct the workflow and reduce the efficiency. However, by essentially dividing our project and team into two, we reduced the communication overhead at least for the first couple of weeks. Our process was also largely dictated by our SCRUM board which we updated every week with new issues, where the backend and frontend people would know what features should be added to the respective components. After a couple of weeks, we realised that the work on the backend was close to complete, and that we would have to move more people to the frontend development.

This was probably the toughest part of the process as we struggled to divide the work evenly. We had too many people working on the same issues and some were left with almost nothing to do while others had too much to do. One of the problems was that our backend people had to learn to work on the frontend, which was a challenge in the beginning.

From that point on, backend features would be more of a on-demand process. We would realise when constructing the SCRUM board for the coming week that we would need one or two smaller features in the backend and assigned one or two people to complete that. The rest of the team would work on the frontend. Seeing as more and more people were moving to the frontend development, we had to clearly split the work on the frontend as well to maintain efficiency. We quickly realised that for example having two people working on the same page (one scene in JavaFX) in the project would not be efficient at all, and so we assigned one person to each page and the remaining people for smaller features. As we reached week 7/8 we had implemented most of the required features and we started implementing bonus features. For most, the workload was smaller, maybe because we had been working well up until then.

All in all, I think every team member has become more aware of the importance of planning before doing the work. The SCRUM board and careful planning is what allowed us to create a good end product.

Reflection

The final product was satisfying for the team members and the teaching staff. However there are some points for improvement regarding the process, the final product and the course.

The application is providing accurate results based on the users’ habits. It is also really engaging and it provides stimuli to keep the users motivated to use it daily. Besides that, the user experience (UX) can be greatly improved. The application is currently counter-intuitive, so new users need a considerable amount of time to discover the provided feature and be able to take advantage of the application.

Regarding the process, as mentioned in the previous section, the work was not equally distributed in the first weeks of the project, so the team needed to redistribute it multiple times during the first month. The scrums boards were very useful to solve that problem, because at any given time every team member knew what the other members were working on, so when someone did not have something to work on he would offer to help the others with their work.

Finally, the course was really well structured and offered a lot of practical knowledge. However, the construction of the teams was not really optimal. The teaching staff divided the students as randomly as possible, but some more criteria could be applied during that process. One possible solution would be to consider students' grades on the Object Oriented Programming course. That way teams would be filled with students of different coding skills, and the whole student distribution would be more balanced.

Individual Feedback

Evaldas

Before the beginning of the project, my biggest issue was that I had absolutely no idea how to work in a team of more than 3-4 people. This project heavily assisted in providing me the experience of teamwork, as now I feel more confident and comfortable working with other people. Generally, I have improved significantly in work planning, communication, work reviewing and splitting the workload evenly due to the requirements of the project and my motivation to learn Software Engineering practices. I believe that my tendency to plan & analyze work has significantly helped me here.

Another issue that I would have in previous projects is that I would tend to do the majority of the work, including other people's work. However, during this project, I have successfully managed to maintain an even workload by sticking exclusively to my own work and allowing my teammates to do their work without my interference, as well as by encouraging equal workload splitting in every meeting. Therefore, despite my weak points in team work, I have managed to overcome that and improve significantly.

With regards to my own contributions, since I believe UI design to be my weakness, I have instead focused on my stronger parts. As such, I have worked mainly on the project's server and database, as well as some logical parts of the application, such as data filtering, conditional querying, functionality for date-related operations and server syncing with the client. Moreover, I have volunteered to do most of the project's setup work, namely Continuous Integration, dependency management, Heroku deployment, setting up a remote database for Heroku and setting up the server itself.

Overall, the project has met my expectations and goals, as I have gained valuable experience in terms of team work, have managed to get really comfortable using Git and learned some new things with regards to Java applications in general.

Shashank

Right from the beginning of the project, I had begun to concentrate on the backend as I knew that the backend was one of my stronger sides, and the front end was one of my weaknesses. It was very exciting to work with a team of six other people. Coordinating work among seven people and making sure that work was split evenly was quite a challenging task. During this project, I had also managed to learn how to efficiently coordinate with team members to solve problems in a clean way.
Regarding the technical aspect, I had worked entirely on the server side of the project. I was responsible for managing client-server communication, making sure that requests provided appropriate responses, managing the backend to handle any issues that arose during the course of development, and maintaining the server regularly and cleaning up unnecessary and unused code. Another one of my weak points was being unable to think creatively. I believe that during the course of this project, I have successfully improved my creative skills, by observing the work of my teammates.

Jean Louis

At first, I was afraid that I might not be able to contribute effectively to the project, as I had no prior coding experience apart from the oop course. This was one of the reasons I choose to focus my efforts on the front end, because I felt that it was easier to grasp and implement than the server side aspects of things.

Working as group, was exciting and helped me make more friends from the course. At first I was excited to work in a large group as I felt that the numbers would make the project easier, but I soon realised that there was a lot of communication overhead which did reduced the efficiency of the group and prevented anyone from making large contributions, especially since there was not much to do. However, the large group size helped me to follow a professional workflow and gave me an insight into the real world.

In respects to the project, I worked on the frontend, mainly contributing to the gui, as well as working on some client-server requests in the beginning. I encountered many problems in the beginning, especially with setting up the environment as the version of java I had no longer supported javafx . Furthermore, I also experienced problems with scene builder and javascript, especially when it came to scaling scenes and linking the gui in the satart.

Over the course of the project I have learned to work effectively in a professional environment and corporate effectively with other, I also managed to improve my confidence levels and better myself in respect to coding.

Finally, I believe that I was successful in achieving my initial goals, additionally I have also achieved an understanding to help me become more creative and productive in future projects. I can say for sure the the project was successful and has also been a fun learning experience.

Alex

In the beginning of the project I was really enthusiastic about participating in a team-based project, especially since I had only worked in a team of at most three people, therefore collaborating with six other programmers imposed a new challenge that I had not yet faced before.

In order to be as efficient and as useful as possible I worked on improving my skills regarding communication, since one of my weak points was being too authoritative. Moreover, another aspect that I worked on improving throughout the entirety of the project was splitting my work efficiently, thus both me and my teammates directly benefiting from this.

In regards to the technical aspect of the project, rather than the personal development plan, I worked on the front end part, hence, I designed most of the graphical user interface and worked on some user requests as well, while collaborating with other teammates when joining our work together. The reason why I got assigned to do this was that I had some previous experience regarding user interfaces, therefore further improving my skills in this aspect of building an application. However, even though I had some prior knowledge when it came to designing graphical interfaces, I encountered some difficulties along the way whilst building everything using JavaFX. While it does feel a bit similar to CSS, HTML and Javascript, it requires a different way of thinking when it comes to things such as “alignment” or “scaling”.

I would say that overall, I have successfully achieved to improve my weak points while also focusing on my main programming skills. In addition, working in a team environment has helped me understand the importance of collaboration and teamwork even more than I used to, therefore I can say that this project was a success and I am sure my teammates agree with me as well.

Vetle

When starting this project I wanted to work on most of the components of the application (frontend and backend) to learn as much as I could. I started with backend, and when that was mostly complete, I started work on the frontend. Before the project, I thought that frontend development wasn’t for me, I wanted to work mainly backend. However, as the project progressed, I found a certain liking for it after all. I now realise that my distaste for frontend development was simply lack of experience, which I have more of now. As said in my personal development plan, flexibility is one of my strong points, which I believe I have shown. I have taken any task I’ve been given, either frontend or backend. As stated in the personal development plan, I said that planning was a weak point, mainly due to laziness because coding was more fun than planning. However, I realised that by not planning, you take away focus from the actual work. When planning as we did in this project, I could separate the planning and my work. This way, I would have a very clear idea what I needed to do. All in all, this project has taught me a lot of about how real-world applications are developed.

Omar

At the beginning of this course, I was doubting my ability in working with a team, since I have always preferred working alone and on my own pace. However after weeks of working with a great team I have learned how much better teamwork is. Splitting the tasks has significantly reduced the stress of work. Also having to keep up with others and being pushed to meet the deadlines have teached me to manage my time better and be more focused on working and learning.
The team was very helpful when I had problems with my code and was always willing to give me feedback to help improve on my work.
I have gained lots of experiences and I have also learned better techniques which resulted of me not just wanting to have functioning code, but more focused on keeping everything efficient and maintainable and try to test every single method.
I admittedly know that I still have a long way to go in terms of professional Java programming and real world work but now I have the skills to continue developing on my own.
Finally even though I was going through some rough times during the last month, I still pushed myself be responsible and not to let my team and friends down.

Kostas

Working on this project was very beneficial to me, mainly because I had the opportunity to work for a coding project in a group of 7 people. I improved on working inside a team and I became a better listener through the weekly meetings. I learned that I don't need to be involved in every part of a coding project in order to make a significant contribution, on the contrary I discovered that if I focus on a specific part of a product, I deliver work of better quality in every aspect.

My role in the development of the application was to research and decide on the logic behind the calculations regarding the carbon dioxide saved by each activity a user logs inside the application. I was responsible to find reliable sources, study them and transform their content into code that would provide accurate results. That way I was able to apply my analytical skills, which I consider to be one of my main strengths.

The most important skill I gained, is beyond any doubt the familiarity with the agile development process which the team had to follow to complete this project. I learned how to work with version control, scrum boards and most importantly to give and receive constructive feedback to each team member. I believe that this gave me more value to the job market, which was one of my goals at the beginning of the project.

Value Sensitive Design

The application was designed with the main purpose to help individuals reduce their carbon footprint. The environment is the main stakeholder and sustainability is the value this whole project is built to serve. To realise this goal, the team implemented a desktop application through which the users can first of all receive an estimation of their carbon footprint and afterwards perform environmental friendly activities to reduce it.

Moreover, in order to provide stimuli and actually motivate the users to use the application daily, the application rewards them with different types of prizes depending on the total carbon dioxide they save. In addition to this, the users can add friends and compare their progress with them. That makes the application engaging and helps users to adopt a more sustainable lifestyle. In order to engage the users even more, the application provides useful information in the form of tips, regarding the reasons why living a more sustainable lifestyle is important.

The sources used to create the final product included the study material provided by the teaching staff of the CSE 1105 and sustainability focused websites. These provided the necessary background to create an application that provides the users with useful insight on their carbon dioxide emissions and accurate calculations on the amount of carbon they can save by performing different activities.

Another important stakeholder which could be considered during the design process, is people of lower income or unemployed. The value associated with this stakeholder is inclusivity. The main problem with the adoption of a sustainable lifestyle is that it is at first a more expensive way to live, so people that belong to this category would not be able to support it. Therefore, the final product is not ready to include lower socioeconomic-status households.

So, if the team considered this new stakeholder during the design and development processes, the need to find new sources and to implement new features would arise. First of all, non-governmental organisations would be contacted to propose to the government the need for allowances to people with lower income, so that they become able to adopt a more sustainable lifestyle. The application would be modified so that it would be possible to log the receipts of sustainable products purchases (ie. organic food) or to connect the application directly with a user's account, so that their expenses on sustainability are tracked.

Another possible source would be the supermarkets of each area. The shareholders of the product could approach supermarket owners for which the team could develop electronic coupons so that the users can redeem them to receive discounts in their sustainable products purchases. This would also require the application to be transferred into mobile platforms.

Finally, the main conflict between the different stakeholders is privacy. The shareholders need data about the application's usage and the users' background, but the users need to be protected from malicious use of their information. The solution that was chosen, was to keep only the very necessary data of the users, so that no one can exploit possible security problems of the application.