OOP Project Report

Group 86

Waqas Abbasi | 4842529

Nafie El Coudi El Amrani | 4771338

Silviu Fucarev | 4801466

Kalin Kostadinov | 4904397

Anxian Liu | 4842316

Stef Rasing | 4956508

Gijs van de Meene | 4957881

Table Of Contents

Project Management **2**
- Planning 2
- Collaboration 2
- Communication 2
- Version Control 2

Project Decisions **3**
- Backend Server 3
- Frontend Interface 3
- CO2 Emissions and Calculating Score 3
  - Eating A Vegetarian Meal 4
  - Buying Local Produce 4
  - Using A Bike Instead Of A Car & Using Public Transport Instead Of A Car 4
  - Lowering The Temperature Of Your Home 4
  - Install Solar Panels 4
  - Calculating User Score 4
- Web Application 5

Points For Improvement **5**
- Improvement on our program 5
- Improvement on the process 6
- Improvement on the course 6
- How We Would Do Things the Next Time 6

Individual Feedback **7**
- Waqas Abbasi 7
- Nafie El Coudi El Amrani 7
- Silviu Fucarev 7
- Anxian Liu 8
- Kalin Kostadinov 8
- Stef Rasing 9
- Gijs van de Meene 9

Responsible Computer Science **10**

Project Management

Planning

The first thing we always did at the start of every meeting was to make a plan for the coming week and we all made sure as a team that we were all able to accomplish what we were assigned and if someone was struggling with a task or an assignment, we would voluntarily help out. This made sure that we were always on track with the project and were on track with the schedule. However, some weeks, due to other courses sometimes we were off track but we always ensured that we would finish the assigned tasks.

Collaboration

Throughout the project, we were always on the same page. Sometimes, when there were decisions to be made, we were all open-minded and made sure to give everyone a chance to speak. Whenever we had disagreements, we would decide to explore the options in more detail with research before coming to a final decision and in most cases this would resolve the conflict. Due to this, our energy as a team was always high and we made sure to always stay positive even with setbacks, for example when someone accidentally deleted all the data.

Communication

During our first meeting, we decided that we would use WhatsApp as our form of communication and this worked really great. However there were some problems with it. We later realised that Discord would have been a better form of communication because it supports having different channels (GUI, Backend etc.) and this meant that we could easily organise everything. This however is not supported by WhatsApp and it meant that WhatsApp wasn't the best form of communication because in some cases, we need to share files and they would easily get lost in the chat. Alongside this, as mentioned before we were really open minded and gave everyone a chance to speak during the meeting.

Version Control

Version Control was one of the biggest challenges for us. None of us had ever used it before, however we were all keen to learn it. We spend majority of our time in the first meeting trying to figure out how gitlab worked, we did this by branching out and editing the readme. We learnt a lot about version control in our first meeting and this was because we received a lecture about it just before the meeting.

It also helped for all of us to work simultaneously on the project. We weren't sure if there was any other way of doing this (efficiently) and with version control this was achieved very easily. We decided to have a development branch that we would merge all our new edits and changes at the end of each coding sprint and when the development branch was stable, we would decide to merge into the master for a release. We decided to use SourceTree as our software tool to manage the git repository. We often ran into problems with SourceTree, for example when we wanted to merge big changes SourceTree wasn't the best tool to fix and edit conflicts because often time we would have to manually open the file and fix the merge conflicts instead of using the 'Stage Hunk' and 'Discard' feature present in SourceTree as it would not allow us to use those buttons.

Project Decisions

Backend Server

During the first two weeks, we were not really sure how we were going to create a backend server for the app. We initially thought to use Java Sockets for this, however this decision was made without any research (due to our exposure to Java Sockets during the OOP Course), however with more research we found out about the Spring Framework. We found that the Spring Framework provided a lot of benefits. The Spring Framework would provide us with Industrial level implementations of software and all it required was the initialization and configuration of the code, that we had to implement ourselves. This was a bit of a learning curve, however with time we were easily able to understand and become fluent with the Spring framework.

Another big decision we made was that we decided that we would host our server on Heroku. This would mean that no one person would be required to keep the server running for all of us to test. This was a really important decision because our application relies on Heroku Database and with each instance of the server running, a database connection is required and you are only able to have at most two database connections.

In terms of building the Client Backend for the GUI, we initially started with using HttpUrlConnection classes provided by Java. This meant that whenever we had to send a GET / POST Request we had to setup the entire connection class and initialise it properly to send a good request that would be accepted by the Spring Application, this was however a really bad solution because it was unnecessarily complex and hard to read, so we invested in looking into different options and we found that Restful Template was the best solution. The Restful Template solution would mean that the library would implement the basic requirements for a request by itself and we would only have to specify things that mattered the most such as the content body of a Post Request.

Frontend Interface

After spending the first two weeks on research, we concluded that Swing was the best GUI Framework for us to use. It was simple and easy to understand, however the biggest problem with Swing was that it was very outdated and wasn't the best looking, for this reason we used Swing up until the first demo in week 3.
On one hand JavaFX offered a richer collection of graphic objects and more freedom for customization. A strong point of this library is the ability to create html-like templates in the fxml language. This allows the development of the GUI with a graphic tool like the SceneBuilder. The benefits of the SceneBuilder are faster development time and a easier debugging process.
One the other hand JavaFX introduced new challenges. The biggest challenge is the fact that after Java 9 JavaFX is not part of the API. While it is not particularly difficult to install the required dependencies to make it compile locally, it was at first, a frustrating experience.

CO2 Emissions and Calculating Score

Figuring out the right way to calculate score and carbon emissions for our features was a really hard task because there was nothing online that existed that could provide help calculating carbon emission with the features we had in mind and because of this we decided to switch the approach. Our new approach was to research data about carbon emissions and build our features based on that and this included:

Eating A Vegetarian Meal

For this feature we found data about foods / ingredients that had the most impact on the environment. This data provided us information about the emissions per Kg, to simplify calculating co2 emissions, we made the assumption that whenever ingredients were added, they were added per kg basis. This would mean that sometimes the user would eat more and sometimes eat less but when adding the activity the application would use a kg of that ingredient. This would average out over time and it wasn't a big problem because it was still mostly realistic approach when compared with others. We also decided to add a Meal history where the user can see the last meals they ate. This would help the user keep track of their meals and look back at them to see how they can improve.

Buying Local Produce

We were really creative with calculating CO2 Emissions and Score for this feature. This was the only feature that we had to create numbers for because it was really hard to find something that would fit the need. We decided to add country information to the User, that the user would fill in when registering. When using this feature the user would enter the country of production of the product and the way we decided to calculate the score was to calculate it by determining how far each continent (of the user based on country) is from the continent the product was produced. This means we had data (score) set for each combination of each continent.

Using A Bike Instead Of A Car & Using Public Transport Instead Of A Car

For these 2 features, we found data online on how much CO2 is emitted per mile for each type of vehicle and so we decided to implement our version of these features and so we split them by their category: Using Personal Transport & Using Public Transport. Personal Transport included vehicles such as, Bike, Scooter, Car etc. The Public Transport feature included: Metro, Bus and Train. The user is required to enter how many miles he travelled.
This approach on these features would mean that it would provide a more realistic way to measure the CO2 Emissions for transport used by the user instead of a simple Yes/No.

Lowering The Temperature Of Your Home

Calculating CO2 Emissions and Score for this feature was relatively easy. For this feature we concluded what really mattered was the difference in the temperature inside and outside because this means if the difference was greater than more energy was required to heat / cool the house and so the co2 emissions are based on the absolute value of the difference between the two.

Install Solar Panels

For this feature, we found data online on how much the average energy consumption was and how much CO2 Emissions were emitted for that amount of energy consumed. So we decided that for this feature, the user would enter how much energy they use in their household and how much of that energy is provided by solar panels (percentage) and based on how much CO2 emissions they save, they get rewarded accordingly. This is the most impactful activity out of all the other features because on average installing solar panels reduces the most CO2 emissions.

Calculating User Score

For calculating the user score for gamification, we decide that it would be best if the score was related to your carbon footprint and to simplify calculating we decided that the user score would be multiplied and processed through a function which would calculate the score and we decide to use this function:

$$f(x) = \frac{-x*10*e}{3} + 3$$

This function would mean that the higher the CO2 Footprint the lower your score and vise versa. This function also means that there is a minimum amount of carbon footprint you can have but the score is unaffected by this limit as it still increases / decreases depending on the activities you add.

Web Application

One of the things we decided we wanted to add to this project was to build a web application. Since the backend (the server) was already ready and functioning, all we had to do was create the front-end for the application, and for this reason we used a separate NodeJS server. This would mean the the online application would be an entity of its own and will communicate with the Spring server using restful calls. For the design of the web application we decided to use ReactJS. We decided this because ReactJS enables you to easily create modular applications and this is what we envisioned our application to look like (modular). If we ever wanted to add more features to the web application we just needed to add more modules and Integrating them would be extremely easy because of React. Another reason we used React was because React is one of the most popular Frontend Libraries available and so debugging and finding solutions to common problems would be really easy to solve. Overall, the web application turned out to be very user-friendly and flows more seamless than our Java Application due to the design. However, the web application did not contain all the features of the Java Application due to time constraints. One of the features that we were unable to add was Friends. In the final build, you are only able to track your carbon emissions and your score and add features / activities. You're also able to view your history of the activities.

Android Application

During the first week of the project we had already decided that we wanted to make an android app, but we were also aware that it was a feature that shouldn't be a priority, so we postponed it to the last two weeks. We started setting up the skeleton of the app with Android Studio. It seemed like the perfect IDE, it functioned almost identically to IntelliJ IDEA and had amazing plugins to set up an android project with ease. One potential problem was that Android studio is Based on Gradle, and our project was based on Maven, but we didn't think of the problem to be something severe. As the skeleton of the project was set up it was time to connect it to the main project. This is where the severity of the Maven/Gradle problem became apparent. We spent over a week trying to get the android module to work with our main project in the correct repository but with no luck. With little time left we decided to try and finish the app separately from the main project. Aware that it wouldn't count towards the project grade itself, we decided to try and make a working app like this regardless. Mostly because we just did not want all this time spent to be in vain.

Points For Improvement

Improvement on our program

After developing our GoGreen application, we think that we could have still improved some parts of our code. First of all, we used the Screen Builder of Javafx which produces slightly complicated code. So we think that coding the GUI classes ourselves would have contributed on the quality of our code. Moreover, we thought about implementing a graph that tracks the user's CO2 emissions over the last 3 months but unfortunately we didn't get enough time to do so. Besides, one of the things we didn't succeed to add it to our app is the option to connect to other social media platform such as: facebook, instagram and twitter, this option gives the user the ability to add all his friends on the platform automatically to his account on our GoGreen app.

Finally, even though the line coverage was not a requirement, we think we should have increased this percentage by adding more tests to our code.

Improvement on the process

Getting used to git and version control was one of the biggest problems our team has faced during this project. At first our project on GitLab was a bit messy. Some of us created branches we didn't need and others worked on the same one at the same time which caused conflicts. However after 3 weeks of struggle, we managed to get around Git and use all of his features in a good and productive way.

Improvement on the course

As one of us already said during the feedback session with the head TA's, posting some tutorials on Brightspace on how to use the different tools such as: Git and SourceTree would have been better than assisting a non-recorded lecture on these tools because we needed all what one of the TA's showed us but we didn't find it. We understand that one of the goals of this project is to train us to search and look up things on different sources such as: the internet and books. But helping us with some videos on Brightspace would have given the ability to start the project with a smooth launch.

How We Would Do Things the Next Time

One of the biggest problems we encountered during development was the server speed. Our server response was very low before we integrated Spring sessions. However after Integration our server became very slow at responding and this was bottlenecking our client application because a lot of features required data from the server and for this reason often times and place the application would hang (in places where http requests are sent to server) and to tackle this issue we decided since it was hard to improve the server speed, we decided on using threading however due to the lack of time we were not able to prioritise threading into our application, we were however able to add threads to the feature that took the longest for the server to respond to (friend search engine) and from this we concluded that if we were to do this project again we would make performance a big priority and would implement our own sessions instead of the ones provided by Spring, this way we would have more control of the server speed and we also decided we would also add threading to everything on the client application so that the application flows seamless.

Another important aspect that we struggled on sometimes was faulty code. Often times we had small typos / errors that would the cause the server / client application to crash and for this reason if we were to do this project again, we would try to implement code reviews every time someone added new code. With code reviews we would produce efficient and quality code that would last and not cause problems in the future.

Individual Feedback

Waqas Abbasi

One of the goals I aimed to achieve was to attain experience in working in a realistic environment for Software Engineering. I achieved this because I was one of the few people responsible for building the Backend of the Server and this meant that I learned alot about the Spring Framework.

Working Backend was one of my stronger points because I was easily able to understand most of the concepts of building a restful server which would receive and respond appropriately. I also wanted to learn Version Control and I believe I achieved this goal. This was an important goal to achieve because version control is used by many software companies..With the help of this project, I was finally able to fully understand the important aspects of version control such as Branching, Pulling, Merging, Commits and Pushing. Through Gitlab I was also able to take part of the Scrum Process, which is another goal I hoped to achieve.

At the start of project I had high expectations of myself to be able to communicate with the team properly to effectively get all the points across while doing my best not to upset anyone and I was really successful in doing so because the very few conflicts we had never escalated to arguments.

One of my weak points throughout the project was working in the Frontend Team. This was because I personally didn't enjoy working in the Frontend but I'm happy that I had the chance to work in this division to explore the option for myself.

Nafie El Coudi El Amrani

When I started the GoGreen project, I didn't have any experience with most of the tools we used during these past 10 weeks. As mentioned before in our report, each one of us had to do some small part of the project. Mine was to create the database on the Heroku server and then connect it to the backend using Spring Boot.

During the first weeks, I have worked extensively on the database part . I started by creating it using PgAdmin. Then I had to connect it using Spring Boot to our model classes.
After establishing the connection, I helped two of my teammates coding the controllers in the backend and test them. And by the end of the project, we all had to help each other on all different parts of the program in order to finish on time and make our product better.

One of the many difficulties I encountered is using Mockito and Spring Boot framework to test all the different methods in order to get a good result on our branch coverage. Besides testing, getting to work with version control was a bit hard for me. I had to ask my teammates for help several times and I think that i still have to get used to it even more. This project helped me develop my programming skills and be a better team player.

Silviu Fucarev

Overall I had an amazing experience contributing to the GoGreen project. I learnt a lot about the structure of a desktop application and about the details entailing its development and it was a really fun process.

Particularly, I would like to point out that I understood the place of libraries such as Javafx and Spring in the development workflow. I have also got a shift of perspective, as before I considered app development a rather simple one-man job which could be done with not so much effort. The project showed the importance of teamwork and knowledge of popular tools such as spring-boot. I came to value good quality libraries as they not only make the developer's lives easier but are also a guarantee for the application's safety, security and it well being.

If I was to do this project again, then I would definitely improve my knowledge about git, stick more to the Sprint practices, and would get more involved in the process of programming the business logic. A better knowledge of git would facilitate a better level of teamwork and Sprint practices help save time and effort. As for programming the business logic, this is something I would like to further expand my knowledge on.

Anxian Liu

In this project, I am working on the GUI Part. For the team work, I think I still have a lot of the points which I can improve in the future. As I mentioned in the Personal Development plan, I needed to improve my poor English skill, communication skill and coding skill. Indeed I improved my English speaking skill but still bad. Actually, during the meeting in every weeks, we talked a lot with each other, it improved my communication skill. The most important part I learned is that I know how to apply Model–view–controller model in the real program. My stronger point in the team is that I can follow the plan and finish my job on time step by step. My weaker point in the team is that I can not explained others my detailed opinion clearly because of my poor English skill. And my second weaker point is that I am not good at the web and databases part in Java. For the conflicts with other team member, actually before we did the GUI Part, we discussed which tools we can use in the first meeting and I suggested to use Swing to finish it. But Silviu thought JavaFX is better than Swing. Firstly we did it using Swing but found out that it’s not good. Then we used JavaFX to do the same thing and finally figured out that JavaFX made it look more beautiful. Scene Builder also made our work easier because we can focus on the controller part. It also made the view and controller separately.

Kalin Kostadinov

When I look back to my personal development plan from the beginning of the project I see that the goals that I have set were realistic and I did manage to achieve everything I had planned.

My contribution towards the project was mainly in the backend side of the application. After reading a lot of resources I managed to understand how autoconfiguration, security and data management worked with Spring Boot so I was able to develop methods and classes that connect the client with the server in a secure and efficient way.

I was also responsible for the deployment of the final product. I took advantage of GitLab Continuous Integration and I was able to develop a custom pipeline that builds, tests and deploys the application with every push.

Unfortunately I experienced a lot of difficulties with Maven and the way it manages dependencies and plugins. It took me quite some time to figure out how to create a pom.xml file that can successfully run all of the project’s goals. I also developed a class that allowed communication with the server over both http and https protocols to make testing easier. Git was not difficult at all to understand, but I think this was because I spent the first week of the project reading every resource possible.

This project had definitely laid foundations for my further development in the field of computer science. I gained a lot of experience that can be used in later projects. Now I have understood that computer science is not about writing code. We have to find and solve real world problems and the code is only our tool.

Stef Rasing

During this project I mostly worked on the GUI part of the application with Anxian and sometimes some help from Silviu. Furthermore I did some small tasks outside the GUI. Those tasks were things like creating a first version for the database schema and resolving a lot of the CheckStyle issues almost every week.

In the personal development plan we had to make, I said one of my stronger points was planning out activities. My planning this project was not the best I can. It could have been better. Still, it was sufficient to finish tasks on time. The other point was investing time in learning things. I think this went great. For example, I had to learn how JavaFX worked and other things like CheckStyle. One of the weaker points in my personal development plan was keeping an oversight of the whole project. In this project I think I kept a good oversight of the whole project; I knew what was happening and where for specific parts of the code were. The other one was getting better in using API's. I could have done this some more, since I did not use a lot of API's. We used one API for the GUI part of the project, but no more than that.

The collaboration between the group was pretty great. There were no issues or problems between any of the group members. Also the communication between our group members was good. We also had almost every week one more meeting with each other. I mostly worked together with Anxian, this went great.

Gijs van de Meene

In the beginning of the course I was not entirely sure what to expect. I knew I'd learn coding, but I also expected that it wasn't the only skill I'd enhance during the project.

As expected I had a lot of practice when it comes to coding, but what I noticed is how much you can learn with just the internet as a resource. Learning how to use the internet as a resource for many tools and learning how to do stuff on your own is something I noticed as being a very helpful and surprisingly doable ability. I was also surprised by the variety of aspects of a project. From back end to GUI and everything in between. Even though it's all programming the mindset is more different than I expected it to be. What I also didn't expect to spend much time on was setting up the structure of a project. While working on the android project most of the work that had to be done was actually getting two projects to work to work together by restructuring directories, modules, dependencies, libraries, SDK's, etc. These aspects of a project were things I didn't know even existed before the project started.

I am really surprised about how quick this project teaches me things I didn't even know existed, which is is a pleasant surprise. I am also quite impressed what can be achieved with a competent group, because I feel like my group members are very efficient I feel like I have learned a lot from them as well

Responsible Computer Science

During the project there were many things we considered when making ethical (in our opinion) design choices. One of the things we considered was the User Data. The user by using our application trusts that we can handle and keep their data safe, for this reason any vulnerable information such as their password is encrypted and stored. It is encrypted with a One Way Function, this means that even the development team cannot decrypt the password and only the user has information on what their password is.

Another thing that could make user data vulnerable was the use of the usernames to communicate to the server. We were storing the username of the client on the Client interface and sending it every time the client made a request so that the Backend could Identify the user making the request. This was a really bad decision because it left the information about the user vulnerable as anyone could simply just make a request with someone else's username and get access to someone else's data. To solve this problem we decided to make use of Spring Sessions. The challenge with using Spring sessions was that they were designed for web applications and not Desktop application and so a lot of configuration was needed before we could use them in our application. Using Spring Sessions also meant that we would be able to use a login authenticator provided by Spring. This meant that the level of security would be much higher than what we were using during Login. However the registration controller was still handled by us.

Overall, we think that we made responsible decisions to ensure an application that can be deemed Ethical by the general public. We made sure that the integrity of the user data was protected and that any information leaks that we were aware of were patched quickly.