Delft University of Technology

Object-Oriented Programming Projects CSE1105

Software Project: GoGreen

Authors:
Aleksander Sinanev (asinanev) - 4830040
Daniel van den Boogaard (dvandenboogaar) - 4783433
Gian Marco Forghieri (gmforghieri) - 4922115
Mahmoud (netid) - 4838068

Contents

1. General: How did the project go?

1.1 Did you manage to stick to the planning (why (not))?

- **Gian Marco**: I have managed to stick to the planning by essentially giving up all other responsibilities in the first 6 weeks of the quarter to be able to meet all the deadlines and make sure that we receive the points for the demos. Considering that I did not have any previous experience, I believe this was a necessary step to take in order to succeed in meeting the deadlines and ensuring the the team stays on track.

- **Alex**: I could have done better. During the first weeks of the project I was organised and tried to do as much as possible. After that I dialled back some of the effort that I was putting into the project in order to get up to speed with the other two courses. Hopefully I didn't make it harder for the rest of the team.

- **Daniel**: I managed to somewhat stick to the planning. In the first couple of weeks I did not complete everything on time. But in the later stages of the project I upped my game, got most of the tasks completed before or at the deadline and subsequently also spend more time on the project.

- **Mahmoud**: To some extent, I have managed to stay concurrent with weekly planning "week spring", but in some cases I had to catch some feature before keep going with my associated task for that week. As an example, in week 6 I had to delay some transportation feature to standardize the UI across all scenes/pages in order to keep the workflow organized. Surely, this has been done after quick discussions with team members.

1.2 How did the collaboration in the team go?

At the very beginning of the course there was some reluctance in the team, but that was soon resolved. Afterwards everything just flowed, there were a lot of ideas on what to implement and just enough enthusiasm. Granted not everyone was willing to work as a team, either because of personal issues or pure lack of commitment to the course and the program. Thanks to the effort of some of the team members we were able to push forward and start making good progress. Everyone had found his spot in the team and the work was flowing.

1.3 How did you communicate?

Our primary mean of communication was WhatsApp. We used this platform because all of us already had it, so it was the fastest to set-up. Another advantage was the instant notifications, which helped with resolving problems faster and more efficiently. We also used the suggested platform of communication by the university, namely Mattermost, to communicate with our TA.

1.4 How did version control help (if at all)?

At the very beginning version control was kind of a problem for our team. But we soon realised the potential behind it. After a couple of weeks of playing around with it, most of us got somewhat comfortable and started using more and more of the available features. Thanks to version control most of our work was organised and systematic, it became easier for us to help each other out. Overall, we believe that using version control helped us a lot and the acquired skill that is working with version control will be a valuable asset throughout our studies and careers.

1.5 What did you learn?

- **Gian Marco**: This course has definitely helped me to learn a lot, especially because my only experience in coding was from the OOP course. I had a general lack of understanding on how a fully functioning app capable of CRUD operations, should be built from scratch. I really couldn't connect the dots between having a UI which the user interacts with and somehow a database is updated. I had problems understanding how the communication between the client and server is built or why is it even necessary instead of just having everything accessible on the client side. I had absolutely no clue about how to build a GUI, or what tools are available for such purposes. I also didn't know how a database can be connected and made functional.

- **Alex**: Personally, this course has taught me a lot more than I anticipated. But most importantly it has shown me what to expect if I choose to become developer after finishing my studies. Building an app from scratch isn't as easy as it sounds, going through that process helped me understand how an application with CRUD capabilities is built. Getting more familiar with version control has been great and probably the most important thing that I learned is to trust your team.

- **Daniel**: The project has taught me more that I beforehand expected. I expected the project to be quite annoying, to some extent it was, but for the most part it was fun, challenging and altogether a good experience. Especially seeing how the app slowly, piece by piece, method by method, was being built to the final product. To be slightly more concrete, I learned how to properly use version control, in this case git. As well as dealing with setbacks like leaving team members, or a function that won't or a framework that is, "softly expressed", quite annoying.

- **Mahmoud**: in my view, the best part of any project is the learning process. During this project, I have learned about the APIs and how to communicate with such services. In addition to many other things like working with maven, version control (git), scene builder and introducing myself to reliable java libraries (JavaFx and JFoenix). Not to forget, the essential skill in this course "teamwork".

2. Design decisions

2.1 What are some of the major decisions that you have taken as a team?

- Divide the work and the 6 features to be implemented in the first week on team members based on the previous experience each one has and his desire.
- Decide on using Spring framework instead of VertX because of the lack of resources for the VertX framework
- Decide on using Relational Database where we can guarantee the reliability of the database and all the data on it to be completed

2.2 Which technological choices did you make and why?

- JavaFX
    - modern GUI library and has all the functionality of Swing
    - the abundance of resources (tutorials)
    - availability of great tool (Scene builder) to simplify the workflow and have more time to concentrate on design.

- JFoenix
    - library for any GUI desktop project
    - modern material design
    - easy to integrate it into scene-builder
    - cool animation for some component
    - the abundance of resources (tutorial)

- MySQL
    - ability to get SQL hosting for free
    - being familiar with relational DB from another course
    - no need to go more than that for simple database

- Use Google APIs for route information
    - ability to get trial (300$) to be spent on all Google APIs.
    - very good documentation.
    - Google Map APIs is considered being the best in the market among all route API services regarding accuracy, simplicity and good reputation.

3. Points for improvement

3.1 How can your software be improved?

(testing, GUI, code quality, features, …)

There are always things that one would like to change about their finished product. Same goes for our GoGreen app.
With regards to the GUI, one of the improvements can be that the middle circle is actually a progress bar, which show how far you are to the next goal/achievement.
One of the security improvements is hashing the password. Since this is a simple app that will not be distributed it is not a high priority to make hashed passwords. But if it were an app that would be distributed I adding a hashed password would definitely be a high priority and an improvement for our app.
One major feature we would like to add, but decided it would take too much time was making an android version of the app. We decided against it as it would be an extra feature in which we have to sink hours and hours in to make it happen. And at the point the app was ready to make an android version there was too little time to make an android version.

3.2 How can the process/collaboration be improved?

Collaboration could be improved by having more meetings, for example one on monday and one on friday. This would increase our contact time, we would be more up to date on what our team mates are doing and how much progress they have made during the midweek. This might also help those team members that get stuck on their part of the weekly sprint. Resulting in possibly more work getting done and/or a more even spread of the work getting done. This will also relieve some of the weekend work so the weekend will be more of a weekend.
Another thing that could theoretically be improved is that every members works almost full time on the project. Thus every member has more time to get their things done and als more time to spend on additional features. This is a theoretical improved since in practice this is not really an option as each of us also has to follow the other two courses.

3.3 How can the course be improved?

An improved for this course would be a more concrete description of what the report and presentation should contain. As for now it is, “The full requirements of these can be found in the introduction slides”, which is fine but having it in the Rubric would make it more concrete and easier.

4. Individual feedback

4.1 Each member reflects on their own contributions towards the project (min. 200 words per member)

- **Alex**: At the beginning of the course I tried to kickstart our work by designing most of the UI and trying to get everyone on the same page regarding what has to be done and what can be done to extend the main functionality of the application. In the first two weeks of the project we had some troubles with maven and the setup of the project since none of us had ever worked with it. The following weeks I worked on the UI with Mahmoud and Pedro, the latter of which decided to quit the course and didn't contribute to the project in any way. We designed and implemented most of the scenes in the UI and given that we had no previous experience with JavaFx I believe we did good enough of a job. After we lost two team members, we had to ramp our production up, thankfully we managed to work almost everything out. After most of the UI was done I started working on an upgrade of the solar panel feature, but it was too technical to be worth it and we scrapped the idea. Since Pedro didn't contribute to the project I had to deal with the things that were assigned to him. After that I focused my effort on making an achievement system for the application, which involved designing badges and the logic behind them.

- **Daniel**: In the first few weeks, I was going to my 'standard' attitude, which is know in dutch as "zesjes cultuur", which is doing just enough to pass. Usually this means only doing things on the day before the deadline or the same day. I soon realized that using this mentality for this course, would not only result in me not passing the project, but that would also mean that my teammates had to do extra and I would disappoint them. Which to be fair is not something that I would like. So I knew I had to step up my game, which was amplified by leaving team members and team members that did, to my knowledge, too little. So instead of just doing what we decided to assign to me in the meetings I tried to do something extra or finish well on time. Working with vertx was somewhat annoying and therefor I was not really motivated to work on it. Once we switched to spring, everything was easier and it motivated me to do more then when we had vertx. I glad we made the switch. What also did not help with the motivation when working with vertx, was that I worked together with David and I had to try to explain how vertx worked to him, while I did not fully understand vertx myself.

- **Gian Marco**: Before the start of the course, I knew that building an app from scratch with almost zero experience in programming is going to be a remarkably difficult, time consuming and a frustrating challenge, therefore I have prepared myself mentally in advance to be able to handle the workload needed. I was aware that I will not have problems with the organizational aspects of the project, however I was quite afraid of the technical part, so I hoped that there will be someone in the team who has previous experience. Unfortunately or fortunately, there was no one in our team with experience in programming, therefore I began to realize that I will need to push myself to my limits in order to manage the requirements of this project. Looking back at my progression and contribution, I still find it surprising that I have managed to work through all the frustration

and obstacles that I had to face. Whenever, I assigned a feature/task to myself I had absolutely no idea how I am going to be able to complete, however I have proven it to myself multiple times that with effort and commitment, I was able to ultimately complete the task. The first challenge was building the login screen and just starting the development process, I started off with researching how to build a UI in java which led me to find JavaFX and the Gluon Scenebuilder which seemed to be a very efficient and intuitive tool to use for UI Design. However I had a lot of problems with being able to even launch the app because I had no clue how maven worked or how to run the app with maven. After I managed to run the app in the IDE, I had to figure out how to work with the MVC design pattern and make the login actually functional. These were definitely the main problems and features that required me to push through my frustration to get it working. I thought from here onwards it would be simpler because I know the database operations required to make the other functionalities as well, however we were not able to combine the server side which was built in vertX with our client side because of several reasons, therefore I have decided to switch to the spring framework because I found a lot of resources and tutorials online. This was definitely the hardest part of the project for me, from here onwards we pretty much just had to replicate and tweak the existing code the make the basic feature requirements.

- **Mahmoud**: At the beginning of the project, I was looking forward to be the responsible member in the team, but after that, I knew Marco where he was great with such stuff and taking care of the project as a whole (by this I mean the deadlines, contacting the team members and make sure everything is going as intended). So, my contribution was more on doing the transportation features and the UI, where I have made this feature such that the user can specify the origin place and the destination, then the program will show a snapshot of the route with some CO2 reduction calculations; and in case the user commit to travel with public transport or with bike, the CO2 reduction of that trip will be add to user score in database. Therefore, I had first to embed an image of Google Map in the program marking the origin place and destination. Also I had to retrieve route information “distance and duration” to do CO2 calculation for either public transport and bike. This all done by using some Google APIs. Fortunately enough, I have got a free “trial” budget to be spent on those services from Google. Regarding the CO2 calculations, I have implemented a floating panel which illustrate all the details. I was making sure that all the CO2-emission calculations is based on scientific artikels not arbitrary numbers. Turning to UI works, beside doing the UI of the transportation scene I have polished the UI of the whole program using “JFoenix” library with some cool animations and took care of CSS stylesheet to standardize the looking of the whole program. As last contribution, I am currently working on theming feature to allow user customize the scene style which is still under developing.

4.2 What were your stronger/weaker points during this project?

- **Alex:**
  - *Strong points*: I think I was somewhat organised and disciplined, but most importantly I was willing to do any job assigned to me.
  - *Weak points*: I had some difficulties working in a team and having next to none experience with application development didn’t help.

- **Daniel:**
    - *Strong points*: Attention to detail, critical thinking and work ethic.
    - *Weak points*: proactivity, not very confident in sharing or communicating ideas to the rest of the team.

- **Gian Marco:**
    - *Strong points:* Being organized and staying on top of things and issues, making sure that deadlines are met and most of the rubric criteria are fulfilled. Willing to put in a lot of effort and commitment, thus ensuring that things get done.
    - *Weak points:* Not having prior experience in application development, therefore I needed a lot of time to understand concepts and be able to apply it.

- **Mahmoud:**
    - Strong points: responsibility in work, where I make sure that all my tasks is done as expected on time.
    - Weak points: spending so much time in some additional feature in which not all members may find it that much important. Unfortunately, many of those is not shown in program now and I had to buried them.

4.3 Did you have conflicts with other team members? How did you solve them?

- **Alex**: I'm happy to say that we didn't have any conflicts whatsoever. Maybe we had some misunderstandings and arguments, but if there was one it was resolved in matter of minutes. Overall I'm very happy with the chemistry in our team.

- **Daniel:** I did not have any conflict with any team member, the closest to a conflict was a not responding team member, but he left the group. But with the remainder of our group I can not think of any conflict between us, other than some different opinions followed by healthy discussions.

- **Gian Marco**: I believe we are a quite well functioning team, I don't remember having actual conflicts. Of course we had some misunderstandings but we were all open minded enough to take a step back and evaluate the matter in an objective and reasonable matter.

- **Mahmoud:** No, I found the workflow in the team was great despite 3 of us has quite the course but there wasn't any real conflictions.

5. Value Sensitive Design

Chosen not obvious stakeholder: Car Manufacturers

Since the main value/design pattern for this project was given as sustainability that's in the core of our application, however to be able to achieve that we have created an app which focuses on user gratification, since it is the gamified, somewhat addictive approach, which is going to actually make the user login on a daily basis and complete activities which reduce their CO2 emissions and contribute towards the greater good which is in essence the goal of this application.

As required by the assignment, if we would consider the addition of another principal stakeholder such as car manufacturers, additional features which promote the interest of the car manufacturer should be created. Since there is an inherent strife between the interest of a general user, whose goal is to reduce CO2 emission as much as possible and a car manufacturer whose primary interest theoretically lies in making a profit. Of course, in our case, we shouldn't look too far ahead in the future as popularity of electric cars is rapidly growing making(technavio, 2018) the conflictive state between these stakeholders much less of pressing issue if at all. Nevertheless, our application encourages travel by bike or public transport rather than car, which still might cause a clash between the two parties, even if the car is electric, taking into account that the change of zero emission public transportation vehicles will most likely occur before all individuals acquire a zero-emission car.

To be able to decide what additional features could be made, consultation with car manufacturers and discussions about their plans on tackling the worldwide movement against greenhouse gas emissions, could prove to be beneficial as the features of our application could align with the plans of the car manufacturer, thus helping to promote their interests. Furthermore, the recruitment of scientists, whose research area concerns the emergence of mass efforts supporting the reduction of CO2 emissions and their effect on the sales of car manufacturers could provide further insight into how our software could balance the scales.

A few options/features we were able to come up with to potentially relieve the tension with the hypothetical stakeholder (car manufacturers) would be the creation of a tab which lists different electric cars, so that the user would have greater awareness of the available cars on the market and could also compare different brands and their unique properties as well as prices. This would greatly help the car manufacturer, by reducing their marketing and promotional costs as well as possibly giving a competitive edge over other traditional cars or other manufacturer's cars.

Another different feature, which could be combined with the one mentioned previously, would be a discount schema based on the CO2 emission prevention of the user. Therefore, certain amounts of points could be exchanged for % or flat reduction of the car price. Alternatively, it could also be that if the user purchases an electric car, for instance a Tesla, the user could get price reduction for Tesla solar panels or whatever product is in the range of the manufacturer and has relevance to the application and CO2 emission prevention.

References

Still need to edit references formatting
https://www.technavio.com/research/electric-car-market-size
https://techcrunch.com/2015/05/29/urban-transportation-will-go-all-electric-sooner-than-you-think/
https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/fast-transit-why-urban-e-buses-lead-electric-vehicle-growth