Natalie Stein
Anthon Wirback
Lucas Gyllensvaan
Leo Mirzajanzadeh
Alexander Lisborg
Oscar Ranhage

Final report - Group Hoth

1. Introduction

During these last 8 weeks, we in group Hoth have developed an application called *Learno*. *Learno* is a desktop application designed to help younger students learn and practice terms and definitions. It's our way of working towards the UN goal number 4 : *Ensure inclusive and equitable quality education and promote lifelong learning opportunities for all*. The main goal of this project however, has been to learn about and gather experience implementing the agile mindset when working as a group. In this report we will discuss our project from the perspective of A (what is), B(what could be better) and A->B (how we could improve).

2. Discussion (Step A, Step B & Step A -> B)

2.1 Customer Value and Scope

- *The chosen scope of the application under development including the priority of features and for whom you are creating value*

A Since this project's targeted audience is mainly middle-schoolers and secondarily teachers, the value lies in a simple and easy to understand interface as well as the different studying techniques. The scope we chose was three studying techniques, flashcards, spelling and multiple choices, an ability to create, edit and delete sets, and a filesystem to be able to share created sets. The last two features benefit teachers more than students but as these sets that the teachers create are used for and/or by students we still argue students are in the focus.

B If it is possible for our future project we would like to have an even more specific target audience resulting in only having to consider that group's wants and needs. Which would make it easier to create a priority list regarding user stories.

A->B One way to achieve this is to conduct a survey before starting the project and thus get an understanding of what target audience we would get with the product. As well as get wants and needs from the consumers. It would also be good to have a few people from this target group who can review the product ever so often, to get feedback along with criticism from them.

- *The success criteria for the team in terms of what you want to achieve within the project (this can include the application, but also your learning outcomes, your teamwork, or your effort)*

A The success criteria for this project is to have the program runnable, working as intended and meeting the requirements of the DoD (definition of done). The program should be able to display the different studying methods such as flashcards, have it intractable as well as create and edit sets. When it comes to the part of the project that doesn't regard the program itself i.e. learning, teamwork and effort:

- Learning: We have learned agile practices such as the multi layer cake model and how we as a group can work more vertically instead of horizontally. We used less agile practices in the beginning of the project, since some of us had conflicting views on some of them, for example vertical and horizontal.
- Teamwork: Our teamwork has improved throughout the project due to us learning new ways for us to cooperate and communicate. We have good teamwork between the members of the backend and between the members in frontend, but we could further improve it between the two. We had little insight into each other's work and were more focused on ours. If someone in backend were to modify the work of someone in frontend, they would find it difficult to do so.
- Effort: We have put in a decent amount of hours into our product, and enough for us to reach beyond the minimal requirements and feel pleased with ourselves.

B For future projects we would improve the teamwork between frontend and backend. Additionally, use as many agile practices as possible from the beginning of the project until its end.

A -> B We would reach our goal of improving teamwork by giving more overviews of the code in backend to frontend and vice versa, so that people comprehend what everyone is working on and give their opinion on it. So that everyone feels more included and doesn't just worry about their part, but every team member's contribution. Given the agile practices we have learned throughout the course, we should be able to apply these early on into our work ethic. So that for the next project we would use these agile practices through the whole project.

- *Your user stories in terms of using a standard pattern, acceptance criteria, task breakdown and effort estimation and how this influenced the way you worked and created value*

A During our project we altered how we structured our user stories a couple of times. We started by following Henrik Kniberg's structure of user stories. Which included: Id, name,

prioritization, estimated time, how to demo and notes. Where “how to demo” corresponds to acceptance criteria. Later on we decided on dividing the stories into multiple small ones to make it easier for us to see what needs to be done and distribute the work. These partial stories were labeled according to which group would work on it. Which was also done partially due to an inability to keep to Kniberg’s structure. So for short, we prioritized the groups ability to read the scrum board and view the project as a whole then creating compact user stories.

B Following Henrik Kniberg’s structure would be what we want to follow in the future, since all parts of it are quite useful. We avoided this though because we wanted to avoid confusion during our project. We would also like to spend more time deconstructing our user stories during preparation to fully understand their requirements both regarding the users value and the work that needs to be done. With this we can more easily improve our workflow and the quality of our end product.

A->B To achieve this better user story structure, the team needs to spend more time analyzing user stories and constructing them in a detailed fashion that is also easy to understand. This would eliminate confusion between the team whilst keeping the information Kniberg’s structure allows.

- *Your acceptance tests, such as how they were performed, with whom, and which value they provided for you and the other stakeholders*

A We have used Trellos checklists for each user story, which represent the acceptance criteria for the story. This work ethic made it easy for us to work on the same user stories and at the same time follow what others were working on. Throughout the project we ran into smaller bugs and errors that were quickly fixed, but nonetheless we created items for these in Trello, which were not fully user stories. This led to confusion among the team about what items were user stories and what were just fixes. We had to rewrite these and merge together some so that they could be viewed as a complete user story and not just a quick fix item. When we deemed a user story that we had worked on as done, it needed to meet the requirements of the DoD (definition of done). Every individual user story had an acceptance criteria, if the user story met the expectations of the acceptance criteria it would progress into the “review state” where it would later be reviewed by the PO (product owner) at the last meeting of each week. If the PO deemed the user story as done it would progress further into the “done state” where it was completely finished, if not it went back to “in progress”. The overall PO satisfaction for the product was set by the PO testing and interacting with the application.

B For future projects we would not unnecessarily add new quick fix items into trello and later rewrite them, but instead write complete user stories from the start to mainly avoid confusion.

A->B We would reach our goal of limited confusion while in the writing of our user stories by being more distinctive between the definition of them and ordinary quick fixes. We should discuss and agree upon what differentiates them, their different structure etc. which would eliminate most uncertainties. Making it moderately straightforward for us how to create user stories that makes sense to everyone.

- *The three KPIs you use for monitoring your progress and how you use them to improve your process*

A The three different KPI:s we used in this project were: stress level, assigned- and completed stories and PO satisfaction. We think that these 3 KPI´s were useful and suitable for this project. They give us some very important information, such as if we need to reduce or increase our workflow, both in terms of stress level and assigned- and completed stories. From the PO´s satisfaction KPI we could see if we put enough effort in, or if we needed to do better.

B We were very pleased with the KPI´s we chose and we think that it worked out really well. If we did this again we would probably choose the same KPI´s.

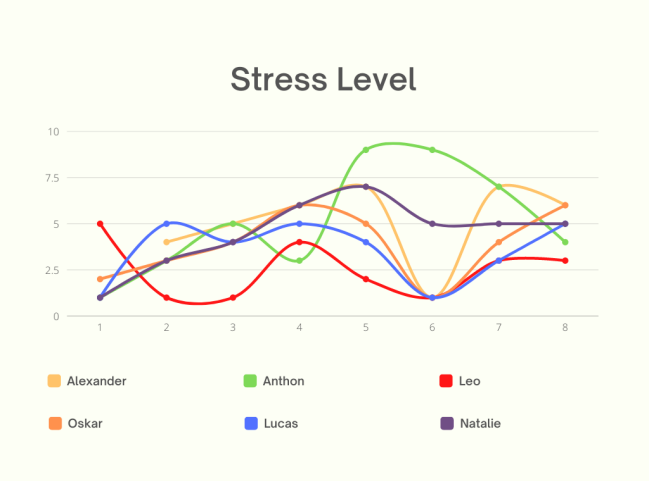


Image: KPI "Stress level"

Image: KPI "User stories"

Image: KPI "PO´s satisfaction"

A->B We believe that these 3 KPI´s gave us a good insight of various aspects in this project, and the KPI´s were most useful when we used them together. Our KPI´s covered an individual's progress or the sprints. But when comparing them with each other we shift from individual perspective to the project as a whole. Through this we could see what needed to be changed for the next sprint in order to improve our overall structure and well being. This was a very important tool that we had, and would like to keep having in future projects.

2.2 Social Contract and Effort

- *Your social contract i.e., the rules that define how you work together as a team, how it influenced your work, and how it evolved during the project (this means, of course, you should create one in the first week and continuously update it when the need arrives)*

A We created a social contract the first week. Which we then updated a few times during our project. The first social contract contained mostly rules about how we should behave among each other and in the team. And rules connected to individual effort and how many meetings we should have per week. We later updated our social contract with more detailed information on how we should structure our meetings. We tried to follow our social contract, and in general it worked well.

B Even though we updated our social contract, our meetings still were a bit unorganized sometimes and we noticed that we still had some lack of communication. In one week in particular we got a quite big misunderstanding among the members of the group, and that was really when we understood the importance of good communication and meeting structure. If we did a project like this again, we would therefore like to have better communication and a better structure for our meetings.

A->B To accomplish this we would need to spend more time on reflecting about our social contract. One improvement we could do is to add a specific "point" for one meeting once a week, where we discuss our social contract, whether we think it works out or if there are some things that need to be updated. By going through the social contract once a week we could have added more rules among the structure of the meeting. By doing so we could hopefully prevent misunderstandings like this from happening in the future.

- *The time you have spent on the course and how it relates to what you delivered (so keep track of your hours so you can describe the current situation)*

A The time each group member spent on this project was approximately around 15 hours a week. A bit less in the beginning, and a bit more at the end of the project. We also had a sprint in the middle of the project after many members of the group had been sick (and therefore not completed as much work as we had hoped for), where we put in more work hours just to get back on track. We believe that 15 hours a week were a good amount of work hours for us. The reason is we noticed that those weeks where we had more than 15 hours all of us were more stressed, this can be seen in our KPI´s.

B We're satisfied about the amount of time we spent on this course, and we believe that it was just the right amount. We completed our project, with the minimal requirements included, and even some extensions, for example the "edit set" feature and the animations of the play flashcards. If we did it again we would probably try to even out the hours a bit, so that we don't have to rush in the end. However, this is something that is very hard to estimate in the beginning, and it gets better the more you do it.

A->B To achieve a more evenly distribution of the time spent, in the next project we would like to focus more on the estimated time of each story in the beginning and apply the work in the beginning of the next project to even out the later weeks of the project.

2.3 Design decisions and product structure

- *How your design decisions (e.g., choice of APIs, architecture patterns, behavior) support customer value*

A At first we strived towards implementing the MVC design pattern into our programme, but then we met some hiccups with integrating the JavaFX into the MVC model. We realized this would take time to solve so we chose to instead spend on fulfilling the project's minimum requirements, since that was higher prioritized.

B For future projects we wouldn't change much about our design decisions, since they worked out and we had great customer value. We would however, try to implement the MVC model into the completed product, if we found the time for this.

A->B We would reach our goal of implementing the MVC model into our final product by using our time even more efficiently in order to integrate JavaFX into the MVC model.

- *Which technical documentation you use and why (e.g. use cases, interaction diagrams, class diagrams, domain models or component diagrams, text documents)*

A We used a web application called "Lucidheart" to model an UML diagram over our project Learno, which helped us to structure up our code. In the beginning of the project we used normal comments in the code for documentation. But in the middle of the project we figured that we should use JavaDoc comments to make even more detailed comments.

B If we did this again we would try to do more documentation in the code, as well as more detailed documentation in the code, since we noticed that good documentation was really important for understanding and to avoid misunderstandings.

A-> B To achieve better technical documentation we would have used JavaDoc comments from the beginning of the project, since they are more detailed and give us a better understanding of what some piece of code contains.

- *How you use and update your documentation throughout the sprints*

A We have used our documentation files throughout the project for different end goals. The individual reflections have served as a purpose for catching up and reminding ourselves what we have worked on last week or several weeks ago. It's very useful for reflecting and understanding what we would have liked to learn or understand better, what my contribution to the team in scrum or deliveries was etc. We have used a team reflections document in order for us to see what we improved from the earlier stage of the programme to a later one.

B For future projects we would like to improve the overview of the project through documentation.

A->B In order to reach our goal of improving the overview of the project through documentation, we would include the amount of hours we put into our user stories, meetings and overall the time we put down into our project each week. We would also add more done states in trello so that every week has a done state where the user stories completed that week lies, which gives a further overview of what went down that week.

- *How you ensure code quality and enforce coding standards*

A We've always had a main rule that when pushing or committing code to git it should be runnable. General object oriented programming standards is also something we have always strived towards, such as low coupling, high cohesion, implementing the usage of design patterns in our solutions and design principles like the SOLID principles. At the start of the project we didn't comment on our code, making it difficult for others to understand it. We later fixed this by adding comments to everything within the programme, mostly to all the methods.

B For future projects we would make it easier to understand each other's code, to ensure that everyone would be able to read and understand the entirety of the code within the programme. So that we would have an easier time enhancing it by for example making it more compact and less complex, modify and expand the functionality etc.

A->B We would reach our goal of easy understood code in the program by commenting on the code from the beginning of the project. We could also further achieve easy understood code by enforcing the rule of 100% javadoc coverage at the end of each sprint. We should also try to implement solutions that are not too long and complex, but simple and compact to ensure clean and easy to read code.

2.4 Application of Scrum

- *The roles you have used within the team and their impact on your work*

A We created several roles for members to play and experience first hand, one of these roles was the role of Scrum Master and in the beginning we worked collectively as the “scrum master” we later changed it to be one person that got swapped between every sprint. This person was placed in a more active role by being in charge of each meeting, making sure we followed our meeting protocol and so that the stories we chose were good. We also have one person each week that has the role of being the PO, who´s job is to rate the product from 1-10 each week and review our finished user stories whether they are done. We can see how many deliverables and effort we put in that week by looking at the rating that PO gave. We also split up in frontend and backend developers, backend being four people and frontend being two people.

B It took us two weeks before we started to assign roles between different members each week and if we were to do it again, we would have done it from the beginning. Otherwise it worked well so we would continue with this structure, with the exception of not switching between POs and Scrum Master in the beginning.

A->B It's difficult to say if switching between roles had an impact, maybe some POs were more leanet and some Scrum Masters not able to control the meetings as well as others. but to get a general feel for everyone

- *The agile practices you have used and their impact on your work*

A We’ve been working according to scrum throughout the project, meaning we have been using a product backlog, a scrum board that gets updated each sprint during the planning phase, and then a sprint where we may update the backlog. Our structure for meetings was not the best at the beginning of the project. And this is something we improved a few weeks in the project via the social contract. We organize our scrum board by different states that user stories enter depending if they are done or in review. However the structure could be further improved.

B For future projects we would have improved the structure for our meetings and followed it all the way through, given that we now know what defines a good structure for agile meetings. We would also improve the organization and structure in the scrum board.

A->B To achieve a better structure for our meetings we should have stated the meeting points from the beginning in the social contract. This was something that we added to our contract a few weeks in, and from there the meetings started to be more efficient and less time consuming. In order for improving the structure in the scrum board, it would’ve been helpful to save which week what stories were completed, as it is now all completed user stories get bundled up into “done” which makes it harder to look back and compare each sprint.

- *The sprint review and how it relates to your scope and customer value (Did you have a PO, if yes, who?, if no, how did you carry out the review? Did the review result in a re-prioritisation of user stories? How did the reviews relate to your DoD? Did the feedback change your way of working?)*

A The sprint reviews were used as a reflection of our work each week. With the help of our PO we could look at our work from a critical perspective and see future improvements. These improvements didn't affect our prioritization of the stories since we had a strong understanding of what needed to be done in which order. Except regarding newly added bug fixes or changes. With that said, with every sprint review we improved our workflow. Whether that be how we structured our work week or small changes like how a feature should be. Before each story could be shown to the PO it had to first be accepted by our DoD. This meant that a story would not be reviewed until we deemed it complete enough to show it.

B The best way we could have used a PO is to solely improve the project's features and customer value, and not things like bugs or small changes. Which couldn't really be done since we ourselves were the PO for each sprint review and thus already having the same opinion/view on the project as us. With that being the case, for future projects when we have a proper PO communications with them is key to understanding exactly how they want the product to be. Even if that would result in re-prioritising the user stories or changing our DoD.

A->B For future projects if we have a proper PO communication with them is essential. We would like to have frequent meetings and an easy way of contacting them. Outside of reviews, if something were to arise.

- *Best practices for learning and using new tools and technologies (IDEs, version control, scrum boards etc.; do not only describe which tools you used but focus on how you developed the expertise to use them)*

A During our project we had to tackle using some new and some old tools. The old tools were a github, figma, scene builder and a new tool was a scrum board which were created in Trello. Even though most tools were old for some members they were new to some, we all had different levels of understanding regarding these tools. This gave a great result as we were able to teach eachother how to use them, which is one of the best ways of learning, for both parties. The one getting taught can ask from someone who knows while the teacher further improves their knowledge. Regarding new tools, with trial and error we were able to learn and understand new things throughout this project.

B In future projects it would be preferable to have a structure where everyone can learn the most of as many different tools as possible without slowing down workflow.

A->B To achieve this improvement we would like to have, at least, one expert in each tool. That way everyone can turn to the corresponding person for help, with that tool.

- *Relation to literature and guest lectures (how do your reflections relate to what others have to say?)*

A During the course we reflected as a team and an individual, we reflected at the end of every sprint, and this worked great for the most part in the team reflections but varied in the individual reflections because it was difficult to pinpoint what we learnt as an individual but was great for filling in what we did during the week and how well we worked together. Our reflections have helped us work more agile and our application of scrum has followed in the same way that Matthias discussed, but with a few differences in manitorial stuff like when the meetings are, how long of a sprint and such. In the beginning it was hard to visualize how to work with a PO, and the guest lecture helped with Matthias saying "team is in control of its process and tool -> we decided how we wanted to develop" this way it showed us that the PO is a guideline while we get to develope how we want just as long it follows scrum and the stories gets done. But how we reflect on the weeks have stayed the same since the lectures. We noticed that not getting anything out of the reflection, not changing behavior after a sprint would make the project spiral out of control with no clear structure.

B After Matthias' guest lecture it was clear to us how important it was and why the structure of scrum and working agile is necessary for success, or more like how not working agile fails the longer the project goes on, mostly because of communication but can also attribute some towards reflection. Not reflecting or getting anything out of it, becomes detrimental when something is not working.

A->B Because we followed his example quite well, we would probably work the same way with reflections as we did in this project, but with the exception of going into more detail in the individual reflections, because they became quite repetitive in the end.

2.5 Git inspector

Github names:

"Alex" &"AlexLisborg" = Alexander Lisborg

"Anthon", "Bookah0" & "Kadant" = Anthon Wirback

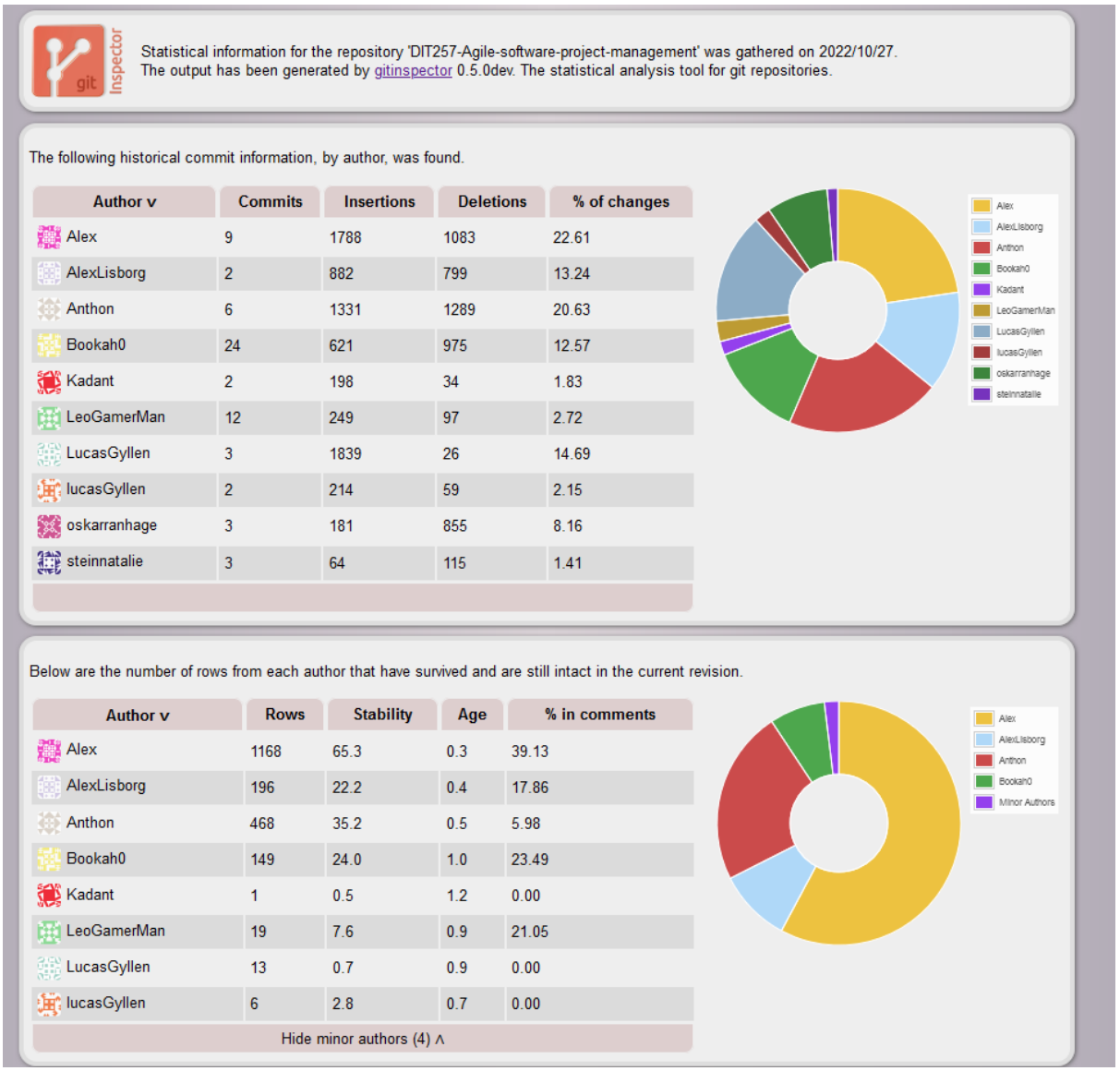
"LeoGamerMan" = Leo

"LucasGyllen" & "lucasGyllen" = Lucas Gyllensvan

"oscarranhage" = Oscar Ranhage

"steinnatalie" = Natalie Stein

This is how our git-inspector looks:



Statistical information for the repository 'DIT257-Agile-software-project-management' was gathered on 2022/10/27.
The output has been generated by gitinspector 0.5.0dev. The statistical analysis tool for git repositories.

The following historical commit information, by author, was found.

Author v	Commits	Insertions	Deletions	% of changes
Alex	9	1788	1083	22.61
AlexLisborg	2	882	799	13.24
Anthon	6	1331	1289	20.63
Bookah0	24	621	975	12.57
Kadant	2	198	34	1.83
LeoGamerMan	12	249	97	2.72
LucasGyllen	3	1839	26	14.69
lucasGyllen	2	214	59	2.15
oskarranhage	3	181	855	8.16
steinnatalie	3	64	115	1.41

Below are the number of rows from each author that have survived and are still intact in the current revision.

Author v	Rows	Stability	Age	% in comments
Alex	1168	65.3	0.3	39.13
AlexLisborg	196	22.2	0.4	17.86
Anthon	468	35.2	0.5	5.98
Bookah0	149	24.0	1.0	23.49
Kadant	1	0.5	1.2	0.00
LeoGamerMan	19	7.6	0.9	21.05
LucasGyllen	13	0.7	0.9	0.00
lucasGyllen	6	2.8	0.7	0.00

Hide minor authors (4) ∧

Image: Gitinspector

This is how our gitinspector looks, however we do not think that it represents everyone's work accurately.

For example we did much work in groups, which resulted in some people not having as many commits as others, but that did not mean they didn't contribute.

For example, Oskar, Lucas and Leo do not appear to have many rows written in the gitinspector because they did joint coding with Alexander who would push the code. Therefore Oskar, Lucas and Leo's contributions to our project don't show up as much in the gitinspector.

For the design-part, some things were not uploaded directly to github. Natalie and Anthon did pair up in the design-making, and many times Natalie sent the design in SceneBuilder directly to Anthon, which Anthon then uploaded to github. The reason behind that is that Anthon often did the “merge” of the design and code, so it was easier that Anthon just uploaded it when he was done. That is the reason why Natalie don´t have as many commits. It seems that SceneBuilder (design) commits are excluded in the total commits count, which takes away the majority of Natalie’s work from gitinspector.

We do not think that gitinspector truly represents our individuals of the group, but we do think that it gives some good information about our group as a whole. When analyzing our gitinspector we would rather say that the numbers of commits, and numbers of deletions says more how we worked as a team, back and forth trying to come up with a good solution, rather than looking at the commits of an individual.

3. Conclusion

Before taking this course, none of us had a good idea of what the “agile”- work process was. We had never used a scrum board or worked vertically before, so there were many new things for us. In the beginning of our project there was confusion regarding how this agile work process should be executed, but this confusion sparked curiosity, and every week we understood a bit more and it became easier and easier to work agile together. In the end we delivered a product which contained all the minimal requirements that we had in mind at the beginning of the project and then some more. In general our project worked out really well, but there are of course a lot of things that could be improved if we did a project like this again, for example:

1. More structured meetings (more detailed planning of the meetings)
2. Better code comments from start to avoid misunderstandings

And these are the main things we believed worked out well and that we would like to bring to our next project:

1. The trello board (Scrumboard), it was a very good way of structuring our weekly tasks. Even though we still could improve how we used it, we liked the way the whole team easily could get an overview of the tasks that needed to be done.
2. Our KPIs, the KPIs that we had helped us to evaluate our progress and how well we worked as a team.