2020-10-30
Chalmers University of Technology

Final report

for the project GreenTips in DAT257, Agile software project management

by

Team Marilyn
Endrit Karpuzi
Smedra Touma
Alexis Aguilar Tello
Mohammed Al-fatlawi
Firel Issa
Cezary Januszek

CHALMERS

Table of contents

1. Introduction

The goal of this course was to create a project that would aid in the achievement of one of the UN's sustainable development goals, that had to be reached by 2030. All of the goals were worth focusing on, but as we had to choose only one we decided to go with the sustainable development goal (SDG) number 13 - "Climate Action". The goal consists in taking urgent actions to combat climate change and its impacts.

Climate change is, nowadays, one of the main issues the whole world is facing and trying to find a solution for. More and more people become aware of this problem that starts slowly influencing our lives and will have an even bigger impact if people don't change their approach to this topic. In order to change their habits people need to know how to make this simple first step to fight against environmental changes. We can all get easily information we are looking for about this issue in tv, newspapers or articles on the internet. However, it is not the most efficient way to reach people (especially the youngest group, on which lives today's changes can have the greatest impact) with simple advice they will remember and apply in their lifestyles.

Our team wanted to attain as many people as possible and make the information campaign about climate change as approachable as possible, so we decided on creating an app, that provides tips in different categories, that everyone could easily follow and implement in their daily routine to make their way of life more eco-friendly. We as a collective can have a great impact on saving the environment if we change a few of our habits or just pay more attention to some elements of everyday's life.

This is how the idea behind our app "Green Tips" was born. We developed an android app that functions as your personal guide book on how to take these small steps to slow down climate change and contribute to a greener world.

2. Customer Value and Scope

What is the situation? (A)

Our project consisted in creating an application called “Green Tips”, which main purpose is to provide simple and useful tips for users to start a more sustainable lifestyle. The team decided to focus on 5 categories (transport, outdoors, household, social life and work life) in relation with our chosen SDG. In each of these categories the user can discover up to 7 tips, varying in length, on how he can change his lifestyle in the chosen aspect of his life. Thus the possibility to discover those tips, provided by reliable sources is the main feature of the app.

As we wanted our app to be reachable by people around the globe, we decided to support 7 different languages spoken by the team members (english, swedish, french, spanish, arabic, polish and albanian). Therefore the user can choose the language he is the most comfortable with or change it at every moment to discover tips in another language he knows.
In our app the user can also set self-written reminders for some tips he discovered or he thought about himself. At first we wanted to add the notification option to each tip card, to be able for the user to add notifications, while he is discovering some new advice. Unfortunately in the end we decided to give up on this concept, as we did not manage to implement it correctly on time. We found that the reminder system adds some important value to the users of our app, as they won’t just read all the tips at once to forget about them after that. The notifications they are able to set by themselves will truly remind them how to change their habits every day.
Another small but still important element we decided to add in our app was the contact form. In this way every user can write and send a mail to the developers to suggest some new tips or share their opinion on others that they don’t find useful at all.

When planning our app in the beginning we also were considering having some additional features, like a system of achievements related to the tips or grouping all the tips in a database to allow users to more easily add their own tips in the app. However due to some issues we encountered implementing those and time limits we gave up on those ideas, or changed their implementation. Those elements could be still added as a later update of the app!

At the start of our project we had clearly defined goals to achieve. As none of us had previous experience in developing an Android app, we wanted to discover together how to work in this new environment to be able to deliver in the end our first self-made app. As we were also using other new tools for some of us to support our project development, like Trello (for the Scrum board) or Git with Gitkraken (for version control), our goal was to become more confident when using those and be way more efficient. As we finished the project we can all state that we have accomplished this goal. We all discovered how to work with Android Studio and how to manage boards in Trello. Some of us started from the beginning with using

git when others had the opportunity to improve their abilities in using its features, like branches and merging.
As we did not all know each other, when starting the project and it was the first time for us working on a project in a bigger team, our goal was to have productive teamwork and meetings, especially as they were all remote and not live. At first we had some troubles with being productive during these Zoom meetings, but we managed to improve our teamwork and become a lot more efficient than during the first weeks of our project implementation.

Understanding the differences between user stories, themes and epics was a first bigger issue we faced. After receiving feedback from our supervisor we truly got the sense of these notions and we were able to define a proper product backlog using them. Based on it we were able to set the priorities of our user stories and split each one of them into smaller tasks. During a big part of our implementation we were modifying the user stories, trying to make them more vertical and adding to them acceptance criteria and effort estimations, that we were often forgetting to state clearly when adding user stories to our Scrum board. The verticality of our user stories was a part of our work that we had some troubles with and were continuously improving in our approach. In the end the user stories were clearly defining what features we have to implement and what should be its acceptance criteria. When the acceptance criteria was met as well as the other conditions form our Definition Of Done the team could consider the user story as finished. The effort estimations of the user stories allowed us to have an idea of how much time we need to deliver a new element of the app and set a given rhythm of work during the sprint.

The acceptance tests we performed were quite simple, but adapted to our app. As we did not have to implement any complicated algorithms or special behaviors, we decided to not have unit tests. Our app functionalities could be tested directly by a user. When a user story was completed, another team member was checking if the finished functionality behaves as it should (if a button works properly, the slide layout works or if the contact form truly sends a mail). By doing these personal tests we discovered some bugs that had to be fixed to provide the best experience in our app. One of the major functionality that we tested and fixed several times was the access to a given tip-card in the slide system from the corresponding card in the topic page. Only when the testing member was accepting the user story, it was considered by all as working and tested.

Our three KPIs we focused on during our project development were: productivity, stress level and team satisfaction. We were measuring them weekly by taking the average of our individual grades between 1 and 5 for each of them. To improve the way we monitor our progress using them, we created a KPI chart showing the changes from a week to another. We were able to notice that the team satisfaction was changing proportionally to the team productivity, as well as with the stress level in some period times. We found this reasonable as bigger stress motivates most of us to be more productive and in the end satisfied of what we have done.

What should be the situation in a similar future project? (B)

In a future similar software development project, we all agreed within the team that we would have to improve some aspects of our work. However, already now we found that we would keep some of the decisions and approaches we made, so let's start with the positive aspects to keep in our mind.

We think that we defined well from the beginning the goal of our app and for whom we are creating value. The goals we set as success criteria for the team were also appropriate for this kind of project and we could easily see the progress in achieving them. We all agreed that setting these goals is essential in each next project we will have the opportunity to work on. Thus we would make sure that we have set them clearly before starting the implementation, in order to keep a clear idea of the wanted result while progressing towards it.

We succeeded in delivering an app, meeting our expectations, despite being forced to give up on some ideas and features we considered to have in the beginning of this project. Therefore if we had the possibility to improve this same app, we would add some of the features we thought about to enhance the user experience. In another project we would more realistically evaluate our capacities to implement all the elements we want, instead of having to adapt, during the process, our application scope.

The main aspect we would improve in our agile development concerns the verticality of the user stories. We were struggling a lot at a certain point of this project to respect the implementation and delivery of vertical user stories. Early in the project we managed to change the definition of the user stories to make them more vertical, but later on we were still missing the correct way to deliver them. As some of our user stories were similar (those concerning the categories of tips), instead of delivering one category at a time, we were implementing them somehow in parallel. After feedback from a supervision session, we understood this main mistake in our approach. Thus, now we all know that implementing at first a “skeleton” of the app to add the content afterwards is not the agile way of developing. We learnt a lot from this mistake and criticism we received. Making sure that we deliver a feature that brings value to the user each time we complete a user story should be the main focus of our agile development in a future project.

Our simple acceptance tests were sufficient in this case, but they should always be adapted to the specific project. Of course if the project would require implementing some algorithms or special behaviour, the unit tests could prove to be necessary. As we didn’t have explicit documentation of the tests aside from the acceptance criteria for the user stories, we believe that next time the tests should have more documentation, in order to be able to follow them more easily.

Regarding the KPIs we would surely keep the stress level and team satisfaction, as we thought that those are important elements to measure in the team during a project of this kind. They easily show when the progress is going well or if on the contrary it is not the case.

However we found that measuring productivity was quite difficult. To replace this KPI we would add the motivation level to the other two mentioned before. Motivation is also an important aspect in a project development, that we thought should be measured in the same way we did it with stress and satisfaction. Our team agreed that making a KPI chart was a nice idea to redo in other projects, as it truly shows the changes in progress and draws the team members attention to some important points in the development timeline.
In a future project we should take more time to evaluate those changes during the meetings to know exactly on which points to focus on when evaluating the KPIs during the following sprint.

How can we make the change come true? (A->B):

The main outcome of this project helping us to change those aspects of our work we mentioned previously is the ability to learn from our own mistakes. All of us understood better the purpose of the elements that were not always on point in our methodology, by experiencing and understanding those mistakes. The obvious way to improve our abilities to work on such projects is to learn from the mistakes we made, but at the same time keeping the good practices we set, while developing this project.

We believe that one of these positive practices we have already had was knowing what are the goals that the team wants to achieve with this project and what value it will bring for people. This element should not change in our work methodology. However in order to have the possibility to deliver all the features we plan, the team has to evaluate objectively their capacities and time they dispose of. For that taking enough time for constructive meetings before starting the implementation is essential and should be kept in mind when starting a project.

The whole team learned the most from their own mistakes concerning the verticality and deliveries of user stories. In order to not redo the same mistakes we know that we all have to focus on ensuring that each delivered user story brings new value for the user. Keeping that in mind makes it directly more natural to deliver one user story after another instead of working on several at the same time. This approach allows to better see the progress and motivates for further work when the finished results can be seen by all the team members.
A plan for keeping track more easily of the user stories progress would also be to define from the beginning their acceptance criteria and effort estimation, in order to strictly stick to them and update them when needed. Splitting bigger user stories into smaller ones and dividing them into simple tasks is also a good way to deliver faster and more efficient value to the app.

As stated before, the acceptance tests should be adapted to the needs of the project. In order to have a clear test documentation we should agree from the beginning within the team on documenting regularly each test. Another idea to keep better track of the progress of the tests would be to state clearly and in advance, who is responsible for testing which user story. Making main features be tested by two different team members could also be a better and more efficient practise to adopt in the future.

In the end the KPIs should be chosen in such a way that their measurement could be easily performed. As we had troubles with objectively measuring productivity, a better solution would be to choose a more convenient one, such as motivation, that can be easily evaluated by the team members. A good plan would also be to not hesitate to add or change a KPI in the middle of the process when we notice that it is not representative as it should be. As we introduced our KPI chart only in the middle of the project, updating it from the beginning would make it easier for the team to observe the changes throughout the whole process and to draw conclusions from it.

For all these aspects the plan for improvement reduces to taking advantage of this first agile development experience to learn from the mistakes we made or to complete our approach with elements we did not consider from the beginning.

3. Social Contract and Effort

What is the situation? (A)

In this project we have been active when it comes to delivering what is expected from us, we were not very strict with the deadlines which we think now it had its effect on the delivery time. Most in the group were very good in raising questions, if they had issues to discuss and to get help with, maybe because we are already friends and that made it easier for us to communicate with each other.
The part that we did less good at was having regular meetings, some days we just skipped it at all and others we had it daily. The days that we didn't have meetings at all we can say that it went less well, we didn't know what each member was doing and how it went for them. Most in the group were good at telling each other if something was too hard, but even though it felt like somebody in the group took hard tasks which usually have to be done by more than one person.We had zoom meeting and some of us from the group had even real meeting at the campus and that facilitated the hard tasks.
Lastly we can mention that it went slowly in achieving the goal. It was difficult in the beginning to know all the expressions and to find the way to work towards the goal, after a few weeks it went as expected and everybody was satisfied.

The effort put into this project was not always the same from one week to another. During some weeks some members of the group were more active than others and in consequence more productive. There was a week where most of the group put less effort than usual, due to different reasons, and we had to motivate ourselves to catch up the delays; this situation made some members also more stressed. As mentioned before when speaking about the verticality of the tasks we also had sometimes issues with delivering valuable features, despite the amount of effort we put to progress in the implementation. However the effort we were putting was always related to making things go forward and approaching our goals. The

whole team put the most effort in the last two weeks of the project, when we had daily meetings and we were aware of mistakes we made and had to change some elements of our work approach. This time was the most productive and we were also very satisfied with our progress.

What should be the situation in a similar future project? (B)

In the similar upcoming projects, we in the group make sure that everyone follows the rules (Social Contract) to the point so that each group member does the same amount. The group thought that the five rules we had in Social Contract were very helpful and effective but could definitely be improved.For similar future projects we would be more strict in delivering in time, this part affected our project since somebody's task was based on others and here we had to wait till the delivery. This part has also stressed some people in the group and that has spread some negativity among the group.
In addition, it is good that the group members let us know in advance if they encounter any problems that have occurred so that the others in the group can help contribute with smart solutions.
This part that should be improved for future projects is that the group becomes more strict in holding meetings where the whole group is involved and gives suggestions for improvements, setting a time that is perfect for all group members. That will help each member to improve theirself fastly and do their best.
It is good that the group members tell each other if they need help or encounter any problems that they find difficult to manage on their own. This facilitates the work for the whole group and contributes to an efficient work environment.

Concerning the effort put into the project, we are aware that in a future project we would have to be more regular in our work to split the effort equally on all the weeks. The relation between effort and deliveries could also be improved, in order to always have something considered as done after a set amount of time, instead of taking more time to just make progress in complex implementations.

How can we make the change come true? (A->B):

Each group member can plan their time in a smart way so that they spend 4-5 hours a day working on the project so that they can reach the best deliveries and result. The scrum master should put a compulsory deadline for the group members to be on time.
The question that faces the group should be solved directly so the group doesn't think about it more and move to another question. The zoom meetings have really helped and we would like to continue working by having meetings on zoom where we discuss problems/questions that we face. Regular and some important compulsory meetings should be done to make sure that it goes well for each member, by having a schedule that for example says a meeting everyday at 18:00, a time that is perfect for the group members, and if somebody couldn't join the meeting the scrum master or others member should make sure that the person who misses the meeting get the needed informations.

When it comes to telling each other if something is too hard, in this point the group should make sure that the task is divided fairly, for example when it comes to the difficult levels, if the task is too hard then the group member or the scrum master should divide the task on more than one person, and if someone is finished in good time before the deadline they should take the responsibility and with the approval from the scrum master to start help the other group members who need more help.

To achieve the best results, the group must have a great relationship, divide big tasks into smaller ones and by following the instructions we wrote above the work will be perfect and we will achieve the goals we set.

4. Design decision and product structure

What is the situation? (A)

Before even starting the implementation the team decided that the application had to be easy to navigate to attract most people. We made some simple sketches of how we wanted the app to look, we wanted a few categories that the user could choose from, where each category offered tips that would help reduce their negative imprint on the environment. The tips are offered in two different ways, the first is a short summary of the tip that is there for people who only need the essence of the information and not the whole content, and the second view that you can choose is a slide view that shows the tips with more information for people that want to learn more.
This design decision of the interface is there to make it as easy for the user to navigate and find the tips they need or want to follow.

We wanted to reach as many people as possible, so we thought of making the application available in different languages so that the user gets the information they want in the language they are most comfortable with. So a couple of weeks into the project we implemented the function to change language on each page of the application. We chose to have english as our main language of the app (due to english being a worldwide spoken language) but we also added all the languages that the group knew such as: swedish, arabic, spanish, french, polish and albanian. The group is very proud of this implementation because it's a feature not always considered by the app creators.

Later on in the development we also developed a "contact us page" where users could suggest tips they wanted in the app or simply contact the developers. This is a very good way for users to interact with the developers of the app and influence the app's future look. The contact page is also very easy to navigate where the user types in the subject of the message and then write down the message (or a suggested new tip to add) and send a mail on the app's email address that we created.

All of these implementations were put in place in order to make the support of customer value our main focus. We wanted the customer to receive as much value as possible, while investing also their own time in this app. Everything is there to make it as easy and informative for them to use the app, so that we would attract as many people as possible and make the most change possible.

Our app is designed in a very simple way which means we didn't want any unnecessary long pages of documentation. However the welcome page of the app contains an “about” icon redirecting to a page that talks about who developed the application and gives a basic idea of the use of the app as well as its main goal. This should be sufficient for any user to understand what he can find in it and how to be able to discover our tips. We are not trying to sell a product but offer it for free, this is why the about us page is not complex at all nor developed as much as in some other applications.

The main document support of our app’s implementation was our product backlog which was representing the overview of every feature we want to have in it. We also used some diagrams to decide on how the tips cards layout should look like. We assembled our 3 main concepts in a document “Tip cards layout”, from which we chose the final version that was implemented in our app.

We also had weekly reflections on an individual and a team level. There we discussed and reflected on what we did (A) , what we want done in the next week (B) and how we can get from A to B. This was a very good way to see how the individuals in our group felt compared to how the group as whole did, if we felt that we achieved what we had set out to do and what the goals were for the entire team and for each individual.

As required we set a product backlog, our KPIs, social contract and other needed definitions at the beginning of the project development. After some supervision feedback we added the Definition of Done, to which we could refer when considering a feature implementation as done. As our work was progressing we had the need to update some of these documents. When reconsidering our user stories and how we splitted them into smaller tasks we had to appropriately update our product backlog in order to have an updated base of user stories to assign to the current sprint on our Scrum board. To reinforce the use of the measurement of our KPIs we added at a certain stage of the project a KPI chart representing the changes in measurements throughout the weeks. We were then updating this chart each week ,after the team reflection meeting, where we evaluated our KPIs. This allowed us to have a better look on how they were changing at different stages of the project and how they were related to each other.

At the end of each week we had a big meeting where we talked in depth about the code each one of us had produced. We checked if there were any inconsistencies with the code before we merged the different parts. If we found something that went against our standards we fixed it together so that everyone would learn from the mistake. We had a basic standard for

the code quality, we needed the code to be compatible for the merges that we did using gitkraken. In this goal some of us were doing pull requests adding a reviewer to check the compatibility of the code. We also wanted the code to be understandable by all with relevant definitions.

What should be the situation in a similar future project? (B)

We feel very good about the way we did some of things in this project because after all this was our first project that we tried to approach in an agile way. We think that we have already managed to make some good design and structure decisions which main aim was to bring as much customer value as possible, and we believe we would repeat this way of taking decisions in future projects.

However if we ever get the chance to work on a similar project there would be also still room for improvement in this aspect. We should make faster decisions concerning some essential elements in the design, in order to not lose time on hesitating and changing the concept several times (this was the case for the tips card layout).

Another aspect, we feel we should focus on more in the future would be the technical documentation to support our ideas. We would try to use it more as a work support and start documenting our concepts and different ideas earlier on (from the first week of developing the project), because it would offer us a considerable advantage over not doing it.

We would also try to have a better defined product backlog and precise definition of done in the early stages of working on the project to eliminate some of the confusion we felt among the team and changes we had to make during the process. We will also need to keep working on better defining user stories in our scrum board to make it easier for us to complete them and have deliveries at the end of each week in a consistent way.

What concerns our way of ensuring code quality, we believe that we would keep the practice of talking in group meetings about the way a feature was implemented, as it was an efficient way to make sure that all team members have more or less the same coding standards. The team should also make more use of pull requests before merging, in order to add other teammates as reviewers of their changes. We were already partially using this method in our project, but we are aware that we could take more advantage of this git feature to ensure code compatibility and quality. Last but not least, we would make the code more understandable to all with some more documentation in it.

How can we make the change come true? (A->B)

When it comes to the focus on supporting customer value by our design decisions we should continue as we have done already and maybe have the ideas better represented by having clear and achievable goals for each of the weeks. We did this but if we do it in the future we would have to start a lot earlier in the development progress.

When it comes to documentation we would use more class or layout diagrams to represent all of our ideas for the implementations which would make things a lot easier to follow and navigate between concepts. We would know what we have set out to do and in what way, there would be a lot less confusion among the group on what is supposed to be done and in what way. It would also be easier to see which ideas would most likely work and would not work in regards to the project. A good plan would be to document which ideas we gave up on, to have feedback on how we were making changes in our concepts and which ideas did not convince the whole team. This would allow us to draw conclusions on ideas we have and maybe save us a lot of precious time in the future rejecting some concepts that we know did not work before. The time saved could be put to better use, such as further developing new features beneficial for the user and not having to stress over and scratch the developed ideas that did not work in the end.

In order to improve our coding standards and quality a good plan would be to make sure that every team member does pull requests for the merges and add at least one reviewer to it. Creating a document where each pull request will be stated with the reviewers, who checked the changes for assuring the compatibility and quality, would allow all the team to follow the progress in this aspect. In this way we could keep track of who was reviewing each part and would be able to alternate between different team members to make sure that all have the same coding standards. This solution would be also helpful to prevent serious merge conflicts, as with time our code would become more consistent.

In the end to have an understandable code, each team member should add comments to his implementation from the beginning and do it every time that he finds that some code could not be obvious to all. Regular documentation of the code would make it easier to review and would improve the team's coding standards.

5. Application of Scrum

What is the situation? (A)

In the beginning of the project we chose to have one person (Endrit) as scrum master and the rest of the group as team members/development team. Other than that, we didn't have a product owner so our sprint reviews might have differed from the standard, which we'll get more in depth to a bit further down. We were all rather inexperienced with the scrum methodology/agile development process so it came as a learning opportunity for all of us. Our inexperience showed itself when we, in the beginning, didn't follow the principles and did things a bit in our own way, both in regards to the development itself but also the relation between the developer, the customer and the stakeholders. This is something we lacked in the beginning of the project, where more specifically our tasks were lacking in value for the

customer, and usually were extremely big and could span over a complete area of the development rather than a specific task. Around half-way of the project we realized that it could be improved and chose to do so and immediately our ideas became more clear, which made it a lot easier to choose one smaller, specific task and develop it and see something concrete.

Regarding the use of Scrum, as mentioned, we had one scrum master and the rest the development team. We didn't have a PO (Product owner) but instead the whole team acted as one. Ideas and wishes were given by everyone and discussed together. We implemented a Scrum board on the tool Trello and continuously improved the structure there. We also had a Product backlog which in the beginning wasn't satisfactory. This was brought up during the first supervision meeting with Hannes and was quickly fixed.

Our sprints were divided in weeks of two (each sprint taking two weeks) and were planned in the beginning, and discussed in the end. This could however have been improved to instead span over one week and be in sync with the individual and team reflections, but we have some doubts about whether or not it would be preferable in the future, which we will discuss more in the (B) section.

We tried to reach our goals and finish our deliverables for each sprint by having continuous meetings where we discussed how things are going and whether we are having any issues (something being too difficult or time-consuming). Most longer meetings, usually in the beginning and the end of the sprint were over Zoom where we in depth could discuss current tasks and difficulties. Some smaller meetings, for example in the morning of most days we could finish our discussions and have our questions answered by discussing things in our Group chat. We feel that all group members were exceptionally good in the discussion and in getting their voices heard, but we also feel that we lacked a bit in sticking with the schedule, as some meetings were improvised and some scheduled meetings didn't have all group members as participants. This was of course due to legitimate reasons, but the group as a whole feels like this is an aspect which could be improved upon greatly.

The sprint reviews were carried out at the end of each sprint where we hopefully had finished a bigger area of the project. We mostly managed to finish what we had set out to do, but some areas of development (such as the slide cards and notification system, which were main areas of development) were delayed. This was primarily in the beginning of the project where we still hadn't defined our Scrum board correctly which resulted in us taking upon us more than we could manage to complete. After splitting up those big areas of development even further it was a lot easier to "build upon" existing code and adding/removing functionality (tasks) a lot easier with more possibilities. It did also heavily result in re-prioritisation of stories where it became a lot clearer of what was critical and had to be done during a specific sprint, and what wasn't as critical and we could wait to see if or when we had time to implement it. This area was also greatly improved by implementing a DoD (Definition of Done), which can be found in our Documentation on GitHub.

Our sharing of expertise is also something we feel has been a positive area of the project. When members had issues, no one hesitated to help one another and try to teach them. If we acquired any knowledge in the upcoming tools we used, we made sure to make the team know of it and a few examples for these things are the different ways to use Git (branching and merging etc.) and how to develop in Android Studio.

As for technologies and tools, we used GitKraken for our version control, Trello for our Scrum board and Android Studio as our development environment. Most of us were inexperienced in these tools with the exception being Cezary who had some knowledge in GitKraken; which was very positive since the group felt that this expertise was easily and helpfully passed on the other members. GitKraken is a program which provides a graphical interface to the use of Git. We felt this was very well suited for us although we understand the points being made for using the actual terminal (which we'll get into at (B)). Initially Cezary provided the group with a crash course in using GitKraken but most of us have developed useful skills in using the program by continuously using the application, reading official documentation and searching for help when something didn't work or if we didn't know how to solve something.

We initially used a google document as a Scrum board since it felt "good enough" at the time, but we soon realised we had underestimated its importance in the project and soon switched to Trello. In the beginning we had one board spanning over the whole project, which we updated each time added, started, finished and removed tasks, but this was changed to having one board for each sprint which made it a lot more comprehensible.

Lastly, regarding the use of Android Studio, it was a completely new tool for us. While it's still Java, the environment is completely difficult which Hannes (our supervisor) made us aware of since he himself had used it for his project. While it was a completely new tool, it is something all members have grown to appreciate and gain knowledge in. We've realised it's a very common tool, especially today with so much app-development going on. When we had issues we usually found hints by searching for it and there was a lot of knowledge available for us. We feel that this is without doubt a tool we will continue to use in future endeavors and are quite happy to have chosen to develop in this environment.

A thing that was very helpful, especially for people lacking experience in these programs, was the inter-functionality of all three main tools; Trello could be implemented in GitKraken, whereas Git could be implemented in Android Studio.

What should be the situation in a similar future project? (B)

If we speak about future projects, there are some things that we would consider a requirement that we wouldn't have before this project. Regarding the scrum roles, we argue that we preferably would like to have a Product owner that can put some more definite requirements and wishes on the project. We think this could result in less time discussing small details and changing the outlook of the project too many times, or too far in the project. Having a clear

view of who the intended customer is and what is expected from us in terms of time and efficiency will be an important step to take for future projects.

Another thing we consider detrimental for future projects is having a well defined basis for the project, and this includes product backlog, estimated deadlines, DD's and KPI's before the actual development starts. This will greatly reduce inefficiency and makes the development a lot smoother in terms of both deadlines and unnecessary fixes along the development.

Discussing the sprint structure, the group primarily believes that while 1 week/sprint during this project might have been optimal, a somewhat longer sprint structure might work better for larger projects in the future. Besides the de-sync by having 2-week sprints and weekly reflection, we weren't too bothered by that structure and felt it allowed us to work on some things more in peace rather than feeling enormous pressure from a sprint ending.

The scrum board, deliverables and stories on the other hand are perhaps the areas we feel could be improved with quite a bit. We struggled a bit with our structure for stories, how we define them and the approach we took for developing. This resulted in developing an application but rarely having anything concrete to show specifically. Due to the stories being ill-defined it meant spending a lot of time finishing just one story. This can be improved and a few of the ways to do that will be discussed in the next section.

Another thing we feel should be a standard procedure and something working without issue is the scheduling. In future projects, our hope is that a set schedule that is followed will be in place, with impromptu meetings occuring on top of that. This would give a clearer view of the project more often and create routines that help us with meeting deadlines. While we held constant and constructive meetings throughout the whole project, they weren't always following a certain pattern.

The learning process for different technologies and tools, primarily GitKraken, Trello and Android Studio, is something we feel has been overwhelmingly positive and an area of great efficiency for us. We learnt by trial-and-error, searching information, reading documentation and sharing expertise and that is something we'll undoubtedly carry on to future projects.

How can we make the change come true? (A->B)

As previously mentioned, on how we think we should prepare ourselves for better planning in future similar projects, we will start the respective project by deciding who the Product owner should be (in case we don't already have one). This is something we clearly lacked in this project which made it very unclear for some of the Scrum team to certainly know how the product should specifically look like.

In order to be able to define a better basis for the project, we need to implement a more efficient way to just put much more weight on the backlog's, DD's and KPI's structure. For

that, each member of the group will need to make a more consistent and organized personal schedule where the Scrum team should be able to see the needed hours that this first section of the project will require.

This method will simultaneously affect and improve other important issues we've actually been having during this course and that have already been mentioned previously on this report, such as the sprint structure or the delivery for the different user stories. This will make it easier for the group members to have extra time that can be used for other issues instead of fixing problems or redoing parts of the project because of a lack of planning.

By knowing the exact amount of hours each sprint will have, at the same time as knowing the amount of hours each user story will individually require, we can just combine and adapt the two so that we can fit a user story in (for example) one sprint or split the story in two or more sprints such that they'll have a better cooperation together.

Another important issue to resolve before starting the development of future projects is the schedule for the weekly Scrum meetings; something that, we have talked about in this report, hasn't been either professionally or well structurally managed. For this issue to be fixed, we'll need to strictly implement a better organized schedule where we stick to the 10-15 minutes meetings.

We mentioned that our way of learning the different tools has been very positive and is something we'd like to carry on to future projects. But since this is the first time we work with all these new tools, we definitely will find a way to improve it even further. A very efficient way to do so will probably be by having more sessions using them, which will be added to those we just had now in order to gain more experience and share knowledge with others about these different tools.

To conclude this report, the group feels that while there is room for improvement, we also see positives in the way we have worked and what we've learnt, and our hope is that by being willing to listen to each other, ask questions and try to learn we can stimulate a good learning experience and a healthy developing environment.