Team Reflection W6

Customer Value and Scope

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

  This week we reprioritized some of the tasks so even though we weren't completely done with a user story we started on a more urgent one. PO also wrote a priority list so that we can, in the future, be even more effective when it comes to adding value.

- **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)**

  This has not changed since last week. We continued working in the same way and it’s going well.

- **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**

  We have tried to standardize our process of breaking down tasks. We found that in former sprints the tasks sizes were very different, ranging from very small to taking days to implement. Therefore we have focused on breaking down tasks to as small parts as we possibly can. We also have an effort estimation for tasks now as well. Not only user stories. This worked well since it is now clearer how much effort every task requires.

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

  This is basically the same as last week but we plan to make a document that specifies the criterias for GUI testing that is hard to test in code. This is something that will make our GUI testing more consistent and it is of our opinion that it will improve the value for the stakeholder.

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

  It has not changed since last week. We have mainly used the well being KPI to motivate changes, for example keeping meetings as short as possible. At the moment our velocity KPI is not that accurate since our task breakdown is not consistent. This is something that we are changing at the moment of writing but it also means that for the first few sprints productivity is hard to measure.

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) There is a survey you can use for evaluating how the team is perceiving the process and if it is used by several teams it will also help you to assess if your team is following a general pattern or not.**

  Last week we planned to clarify roles, and made a “Definition of roles” document as an extension to our social contract. This seems to have worked well.We used the fact that our social contract says that every major decision is to be done democratically to decide our meeting times this week. A majority voted to have meetings at 8:00. This makes it seem to us that our social contract is well written and that voting is a good way to make decisions if you’re not in consensus.

- **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)**

  The reflection part of the course is very extensive, the individual and the team reflection every week takes many hours which makes you work less on the project. Even though it is helpful to evaluate previous weeks in order to improve your way of working it feels like to be more effective the sprint should be more than 20 hours. In other words, for the hours worked we have not delivered what we expected project wise because we thought that the no coding parts would take less time. We tried to rationalise the team reflection part and it's now a smaller part of the course even though it still sometimes takes almost a fifth of the course hours every week.

Design decisions and product structure

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

  We found an API for the calendar function in our app that has sped up that part of development substantially. It was chosen due to being fairly popular compared to other calendar API and having a very well documented page. We decided to sacrifice speed in the program to make sure it is type safe which is something that is going to make the program more stable which is preferable for our users.

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

  Nothing new here since last week. Github wiki for saving and loading functionality is still planned but not done.

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

Nothing changed since last week.

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

  Our code has been made fairly modular, which makes it easier to change specific parts when needed, and the code is also more reusable that way. It is also easier to corporate and in the future we will try to contain this modularity. We also peer review the finished code before moving it to done. This has taken some extra time but is ensuring code quality. The previous week we even tried to do peer review on all written code by text in a document but realised that this was an ineffective way of working and we will stay to feedback in person (on zoom).

Application of Scrum

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

  The product owner decided to spend more time on tasks associated with the PO role instead of coding to make sure that the team had a better understanding of the external stakeholders wishes and requirements.

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

  We have been more agile in who does what task. Since we started we have splitted into three groups while working and these have not been decided according to how well you know the program but only who you knew from before. This week we splitted up those groups and started working together with someone from the other section. This was more efficient since it diffused the knowledge and experience for coding in all the groups and was an efficient and more developing way to work. This is in essence pair programming, which we have used before, but we have mixed the more experienced android with the beginners.

  The focus of what tasks/user stories needed to be worked on was shifted towards being more valuable towards our customer. Previous weeks there had been more of a focus on getting everyone used to the work environment and preparing some central backen stuff in the coding.

- **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?)**

  The sprint review made us reprioritize our user stories. The team was told to focus more on core functionality, such as data representation and the quiz, and pause the work with less prioritised functionality. We also agreed with the PO that the team should move user stories to “done” before input from PO to make it easier to see what the team has done. It still needs to be reviewed by the PO to be really done though but this makes it less stressful for the PO. There was some confusion if PO

had to move tasks to done as well but we decided that was not the case and makes us finish tasks quicker since everyone has the authority to move it and further removes stress from PO.

- **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)**

  We've split up our work in a way so that those who are new to Android Studio programming projects in general can work together with those who have more previous experience. Some of the groups did more pair programming than others but it might be better to try pair programming to a bigger extent. Some parts of the app development seem to be more suitable for different ways of working.

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

  We did not have any guest lectures yet. Some Implementation decisions were made from what the group could find online regarding best practices in Java. It is our goal to try to do that in the future.