Team Reflection: Week 3

Team Ginger

Customer Value and Scope

- *The chosen scope of the application under development including the priority of features and for whom you are creating value*
  We have further developed the application from what we have described in team reflection from week 2. This week we focused on what we felt was most important for the user and for us as developers. We decided that some sort of GUI as well as a component for registering that the user has collected garbage is important. It was also important to have the possibility for this information to be stored locally on the device so that the user could turn the app off and be able to progress from where they left off upon reopening the app again.

- *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)*
  We have learned principles on how to apply SCRUM and agile ways of working. During the supervision session, we received feedback on our Scrum approach. We have used this input to redesign our user stories. Before the supervision, our user stories were developer-focused which, although it provided value to developers in "horizontal slices", did not provide much immediate value to the user. Therefore, the stories for our first sprint were reprioritized to be more user-facing.

- *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 were accurate in our time estimations this sprint. We had two stories that three people worked on both of medium size. We decided that medium is one sprint for one person and it took three people one sprint to complete them. We also had one story which four people worked on in parallel, meaning we got four different results. This story was of size medium and as it took one sprint for each assignee so the time was accurate. We have had some difficulty in breaking down the user stories into tasks that are more vertical, as our natural tendencies is to have them be horizontal i.e. do one thing to completion before starting development on another part. This hinders us from providing immediate customer value but we got a lot of help during the supervision in how to reason and manage this.

- *Your acceptance tests, such as how they were performed, with whom, and which value they provided for you and the other stakeholders*
  As part of our Scrum process, we have decided to have a review stage for each

user story. Since our git workflow was not established yet this week we have not been creating pull requests for the review stage. Therefore, the only type of acceptance testing we have performed is to check the acceptance criteria of the user story assigned to ourselves while working on it. We worked in two large groups and provided feedback to each other within the groups. This feedback, most likely, got us to produce a better result. In upcoming sprints, we would like to have the review process be well-established so it is clear who will be checking each story and what they will be checking before accepting it. We hope this will provide useful and detailed feedback to whoever worked on the user story so they can improve their work and provide more value to the user.

- *The three KPIs you use for monitoring your progress and how you use them to improve your process*
  From the results of this week's sprint we can see, based on the KPI: Target / Actual time per Sprint, that almost all members did not put in the estimated amount of hours that are expected to be spent working on the project. This was somewhat reflected in the KPI: Planned features / developed features per sprint, as some features were completed but some were half-finished. In our final KPI: Satisfaction level with project progress, we can also see that while we met expectations, represented by a 3 on a scale from 1 to 5, there is still room for improvement. So if the team as a whole were to be able to spend the estimated amount of hours during the sprint results might improve.

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)*
  Everyone has been following the social contract created at the beginning of the project. We have not made any changes so far as there has been no need yet.

- *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)*
  We are keeping track of our individual time spent and we sum it up at the end of each sprint. Below is an example of how we keep track of the time spent per week.

  Target/Actual Time per Sprint
  Formula: -(14h - consumed time per sprint - consumed time for individual reflection):

  - Sophia: -1h
  - Alexander: -10h

- Emanuel: -2h
- Snjezana: -3h
- Benjamin: 0h
- Bashar: 0h
- Isabella: -2h

Total: -18h

A negative amount of hours means we have not spent the expected amount of time this sprint. This was somewhat reflected in the KPI: Planned features / developed features per sprint, as some features were completed but some were half-finished.

If we look at what we delivered this sprint (after a modification to our DoD), we were able to finish all user stories. Regarding the time spent, the time was slightly lower than the estimate but we were still able to produce something of customer value on time.

Design decisions and product structure

- *How your design decisions (e.g., choice of APIs, architecture patterns, behaviour) support customer value*
  When deciding UI implementation we have chosen to have one of the most relevant pages of the application visible from the main navigation bar. We do not want to clutter the user's screen with things that may distract from the overall purpose of collecting trash. In the UI design we also decided to have the reward mechanisms clearly presented to the user from the home screen in order to encourage picking more trash. Our decision to use React Native to be able to have our code run on both iOS and Android has proven to be a good one. We have continued to only use dependencies which are compatible with both platforms for our customer base to be as broad as possible.

- *Which technical documentation you use and why (e.g. use cases, interaction diagrams, class diagrams, domain models or component diagrams, text documents)*
  We have added a short workflow document that describes what VSCode extensions we should use to keep a uniform code-style. Otherwise, the technical documentation remains the same.

- *How you use and update your documentation throughout the sprints*
  This sprint we have written documentation for what IDE extensions should be used. We were lacking guidelines for code style and we chose to solve this problem through extensions. We will create and modify our documentation as needed, when we find flaws in the documentation, we will fix them.

- *How you ensure code quality and enforce coding standards*
  We are using the extensions Prettier, React Native Tools, ES7 React/Redux as well as JavaScript (ES6) snippets in VSCode. These extensions handle indentations and such, thus making all team members code look similar. In the future, when we have written more lines of code we will be able to see if there are any major differences in how each member writes their code. From what we discover there we will update our standard. As soon as we have established a better workflow for code reviews, this will aid us in making sure the code we write is of reasonable quality.

Application of Scrum

- *The roles you have used within the team and their impact on your work*
  This week we worked in parallel which worked quite well because nobody was dependent on the other. Half the team worked together on fixing the functionality to register trash while the other half worked on their individual UI prototypes and then we presented our ideas to each other and picked out the best features to combine.

- *The agile practices you have used and their impact on your work*
  We are still using our stand-up meeting strategy. This strategy is flowing more naturally now and we are still discussing if we have faced some troubles at those meetings, or if we need some kind of help specifically. Overall, those kinds of meetings add more understanding to how the team is performing. We would like to add more time to our big Friday review & planning meeting by 2 hours so we can both do our team reflection, review and assign new user stories/tasks for the coming week. We did consider having it in the middle of the week but it seemed more convenient to have it at the end of the sprint all on the same day so there is no gap in the sprint cycle. So this is what we will plan for the coming week.

- *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?)*
  As we do not have a PO this sprint-review was carried out similarly to our last, where all team members go through what they have done during the sprint and what problems have occurred.

  This sprint we discovered that one of our user stories could not achieve our DoD because this particular user story could not be tested properly. We have now agreed to change our DoD as a result of this particular case. A meeting will be held for the user story in question where the team gets to vote on whether to exclude it from testing or not. This will prevent other user stories from not being able to achieve our DoD in the future.

As a result of what we have worked on this sprint and the review, we have decided upon the general look of the GUI and added some tasks for the next sprint as a follow-up to what happened this sprint; such as merging code in order to bring on new features for the customer faster.

- *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)*
  What was new this week was that we had to learn some new React features and React libraries like React Navigation, Async Storage, Expo Sharing, Jest and more. Additionally, we added some plugins for Visual Studio Code to simplify our coding. We are still learning how to use flexbox styling for UI. We learned a lot by watching YouTube tutorials and learned that a lot of them were outdated (which ended up wasting a lot of time) and some were up to date. This caused our different React versions to be mixed and gave some of us complications and errors. To learn Async Storage, Expo Sharing and Jest we read a lot of documentation and searched for code examples of the functions we wanted to use. To make it easier in the future we will have some sort of guide in place when someone has discovered how to do something that has previously caused issues.

- *Relation to literature and guest lectures (how do your reflections relate to what others have to say?)*
  No guest lectures have been given and we have not read any literature.