Team Reflection Course Week 7 (V21)

	A	B	A → B
Scope & Stakeholder	We had another meeting with our stakeholder to confirm what we will do during the last sprint and what functionality that should be of focus.	Focus on the tasks that the stakeholder wants us to prioritise.	Complete the tasks in our scrum-board whis is most interesting for the stakeholder and perform testing of previous tasks.
Priority of features	The priority of features is to fix all the bugs that the app has, so that we can show a decent working app on presentation day. The focus has shifted from coding new code to coding bug fixes and testing of existing code.	The aim is to have a finished MVP so that our stakeholder and examiners will be satisfied with the final product. For future projects we want to have more meetings with the stakeholder in the beginning of the project in order to pinpoint which features should be prioritized earlier.	To achieve a MVP we have to spend more time in meetings with the whole group and with the stakeholder so that we together find all the bugs and minor faults. Minor touches on the design and the interfaces needs also taking care of so that the app is both working and aesthetic.
Success criteria	The success criteria of this week is to finalize the MVP. We as a group want to be satisfied with the product that we have been working on for the last 10 weeks.	For future projects, have in mind that success criteria can change during the project because of the stakeholder but also because of the group members opinion.	Have more discussion during sprint reviews about our success criteria and if they need to be changed.
User stories	The user stories which we decided (together with our stakeholder) added the most value to the application have been finished. Unfortunately, we had to leave some user stories unfinished due to time constraints of this project.	In a future project, we want to be able to complete all the user stories we've written. We also want to improve how we write tasks from the user stories and be able to work on these independently during the sprints. So, have more loose coupling between the tasks we work on during the same sprint, so we don't have a lot of conflicts between different tasks.	It's difficult to have exactly the right amount of user stories when the project is in its initial phase. A lesson we can take from this project is maybe to have less user stories when the project is in its first phase, and add more as the project progresses. This was difficult for us though since we felt that every user story we created in the early stages added value to the application. It will be easier in a future project to complete every user story if we don't

			have a tight time schedule before the application should be released.

We got increasingly better with having loose coupling between tasks in the same sprint during the time period of the project. As we now have better knowledge of scrum, we don't feel like this will be a huge problem in a future project. |
| **Acceptan-ce tests** | Our stakeholder expressed that everything in the application looked good. This last sprint has been more focused on fixing bugs and making finishing touches of the application, e.g. finishing the user stories we started last sprint. | Our acceptance tests should maybe in a future project involve the opinions of more stakeholders, to get a broader range of feed-back. However, the stakeholder we had was very good at giving feed-back, so we didn't experience this as any big problem. | Contact more people with different interests and backgrounds, but that are still potential future customers. |
| **KPI** | The KPI:s have retained high values, similar to the last sprint's. | We are still improving the project even in the last few days to reach the best version possible. We want to have a more even project burndown in a future project. | If we set points more correctly in the start of the project, we think it will show in a more even project burndown graph. We should also aim for less coupling between tasks the we do during the same sprint. |

Social Contract and Effort

	A	B	A → B

The rules that define how you work together as a team, how it influenced your work, and how it evolved during the project	The rules haven't changed since last week, we are continuously following the rules written in our social contract, which have been helpful for our collaboration. We are still following and respecting the same rules written in our social contract.	For future projects, have in mind that the social contract should be reviewed more often and maybe changed during the project in order to make everyone feel comfortable with the project.	During sprint reviews analyze what parts of the social contract that work well, and what parts need to improve.
The time you have spent on the course and how it relates to what you delivered	The group spent a lot of time on the course this week. The reason behind it is that we were trying to finalize the app so that we have an MVP for next week. There have been many commits this week and a lot of 'merge conflicts' have been resolved. There has also been a lot of collaboration with team members and cross coding.	Next week is the presentation so we want to spend more time on working on the presentation and to fix minor faults in our code.	Same as last week: As mentioned last week, we get more data from the KPIs, which makes it easier to evaluate our velocity. By continuously monitoring our progress we'll see what improvements we can make. Hopefully, these improvements will result in a better workflow and a delivered product of high value.

Design decisions and product structure

	A	B	A → B
How your design decisions (e.g., choice of APIs, architecture patterns, behaviour) support customer value	There have been minor changes in our design decisions, but of those we have changed, most of them have been on the aesthetic part of the app. We have also talked more about following certain code principles and code design, but we have not fully implemented those. These are documentation of the code, comments and removing TODO:s.	<u>Same as last week:</u> Our ambition is to design a customer friendly application and layout so that we can create more customer value. For that we want the design to be customer friendly.	The focus has shifted from creating new features of the app to creating functioning features so that we can make an MVP. In order to accomplish this, the workload distributed between the group will be focused towards making the final app and documenting the app and so on.
Which technical documentation you use and why (e.g. use cases, interaction diagrams, class diagrams, domain models or component diagrams, text documents)	We use Google Drive for e.g. KPI, team reflections, and other text documents that everyone in the group needs access to. - Diagrams.net for UML. - Miro for our mockup. - Github. - Spreadsheets. - Trello for scrum-board. -Figma This week we added: -The tasks have pdf:s of the layout so that everyone can see what the others have done so far and what the layout looks like. This helps us with documenting the design work, and figma for design.	For future projects we want to explore more documentation methods.	Creating a project with a different focus. Maybe towards a company and explore technical documentation we can use to conversate with them.

How you use and update your documentation throughout the sprints	This week's TODOs in our sprint contains the last fixes of our application. We have prioritized the tasks depending on our stakeholder's interests. The scrum board looks better this week than the previous one, since we've done a lot more planning and we've especially been more realistic regarding the time we have left. We still have more user stories in our sprint backlog that can be added in the future.	As we continuously develop our application, there'll always be new features to add, in the form of new user stories and tasks. We want our scrum board to be as helpful as it can be and easy to understand. We want our documentation to be more sleek and to have less cards so that we can more easily estimate how much work we have until the app is completely finished. Furthermore, we also want to apply the same work technique for future projects.	For adding new features to further development of our application, we can make sure to work and continue following the rules of the scrum board and perform the needed tests of the tasks to remove the excess cards from the testing column. In the case for future projects, we can try to apply the same rules and our experiences of working with this application.
How you ensure code quality and enforce coding standards	We have been following the MVVM design pattern during the whole project. The JavaDoc documentation has also been better during this sprint compared to earlier ones.	In a future project, we want to add more documentation to our code, so that everyone can understand more easily what the code does.	From the beginning of a future project, set aside time to add JavaDoc to our code.

Application of Scrum

	A	B	A → B
Roles	Same as last week: We have tried to distribute the same amount of	Same as last week: We want each role to have the same amount of	Same as last week: During sprint reviews review the KPI and to bring up if any

| | workload for each role. | workload | role caused any reason for the grade in the KPI.

Reviewing during daily check-ups with role related activities. |
|---|---|---|---|
| **Agile practices** | We have had several conference calls for everyone to write code together. | For future projects we want a more detailed recap of what each team-member has programmed each week | Have longer sprint reviews and daily checkups for everyone to communicate what we have programmed |
| **Sprint reviews** | During sprint reviews we focus on the workload and how the scrum process has worked during the week | For future projects we want to have a more in depth review of each team member's written code. | Have a short presentation during sprint reviews about functionality in the code. |
| **Best practices for learning and using new tools and technologies** | We use the opportunity of reviewing code in github to understand other team members code before merging. We only have one official reviewer for each person. | For future projects we might want to assign more official reviewers for each code before merging for a chance to understand a greater amount of the done work during the week. | Assign more official reviewers for each team member for each week and continue to randomly generate what code each team member reviews. |
| **Relation to course material** | We have a great understanding of the application of scrum. | For future projects, understand more conflicts that can occur when working with scrum. | For future projects, work on another platform or create a different app so new conflicts can occur because the material would be less familiar. |