"ProteQ - SimpleGDPR" – 2nd year project

Group N - Elite
Alberte Kjær Wærens - albw@itu.dk - 11/03/1999
Benjamin Jarom Lüthje- bely@itu.dk - 22/10/1998
Harry Singh - hars@itu.dk - 09/02-2000
Isabella Drest Rasmussen - iras@itu.dk - 01/06/1999
Jimmy Müller - jimy@itu.dk - 16/02-2000
Lauge Kjærgaard Jensen - lakj@itu.dk - 21/05-1998
Louise Götzsche - logo@itu.dk - 28/02/1997
Simone Jakobsen - hasj@itu.dk - 22/05-2000

May 26, 2021

Table of content

1 Introduction

1.1 Description of the target project

In this project, there was a cooperation between students from the Software Development program at ITU and the company ProteQ. The target project was to build a website to guide and help smaller companies through the rules of GDPR. The final product should be a website called SimpleGDPR, with guides, tutorials, and a forum. This should help a user to comply with the rules of GDPR within their company. The requirements that the Product owner (will during the report be referred to as PO) provided for this project were:

- Step by step guide to indicate how far the user is in the GDPR steps
- A forum for the users to communicate with each other and the owner of Simple GDPR
- An independent login function that can differ between a normal user and an admin user
- Implementation of a payment service so that a user can pay once for the GDPR guide steps.

1.2 Description of the approach to the project.

When choosing which project to work with throughout this course, the team valued liberty to choose the programming technologies and realistic viability. Once the project was chosen, the backlog and epics were defined and presented to the developers by the PO. Due to this project containing many new aspects, which the developers were not familiar with, it seemed to be too ambitious for the team to reach the goal the PO had in mind for the project.

This table will elaborate on the different risks that have been taken into account during the project. The likelihood and impact are rated in a range from 1-10 and the risk is a calculation of the likelihood * impact.

Ref	Hazard	Likelihood	Impact	Risk
R1	Not being able to master Heruko	4	8	32
R2	Not being able to master React	3	9	27
R3	Not being able to master Quick pay	6	4	24
R4	Significant staff dropping out of the SWU program	5	7	35
R5	Significant staff sickness due to Covid-19 affecting critical path activities	8	4	32

These risks are soft risks due to this being a school project, and therefore it did not include a budget, which often brings higher risks with it. Therefore, the risks for this project are more internal within the group and therefore has a more human perspective.(7)

1.3 Description of the team.

The team within this project worked with Scrum, and there were several roles within the team. They are as follows:

Product owner
The Product Owner for this project was *Matias Aabye*, he is the CTO of ProteQ. He played a very important role throughout this process since he was responsible for creating a Product Backlog for the developers, communicating Product Goals, and prioritizing different items in the backlog.

Scrum master
Our Scrum master (will during the report be referred to as SM) was *Mikkel Agerlin*. The scrum master played an important part in the team. He was responsible for establishing a functioning scrum process. He coordinated and prioritized with the PO, and he helped the developers with various problems in the daily work. He was equally in charge of planning and organizing the sprint planning, daily scrums, sprint reviews, and sprint retrospectives. Mikkel is a previous SWU-student, and therefore he had a lot of competencies and experience that he shared with us throughout this project.

Developers
The developers played a very important role in the project, as they were the people building the actual project. In this team, there were eight developers. All have experience working with Java, C, F, HTML + CSS, JavaScript, and PSQL. Besides that, all team members were adding different values to the team:

- Alberte
  - Brings a very positive mood to the team and always strives to learn more. Values good communication high so a certain structure can be kept.
- Benjamin
  - Strives to learn new things, contribute as much as much as possible and keep a positive mindset.
- Harry
- Isabella
  - Very Structured and organized. Brings a very positive mindset and social personality to the group.
- Jimmy
  - Good at keeping the team spirit alive. I strive to keep the social integrity and bring a positive outlook on the project every day.

- Lauge
  - Good at focusing in on specifics tasks on an individual level. Brings a good amount of irony to team and enjoys a good laugh. Works best in smaller teams.
- Louise
  - I value structure and good communication within the team. Therefore i try to come up with different ideas to improve the structure in the group.
- Simone
  - I value a good atmosphere and flexibility in the team. Brings new ideas to get a broader experience with working in a scrum team.

Agile coach
The agile coach for this team was *Anders Muhola*. As he was an agile coach he was not a part of our specific scrum team. This meant that he joined our sprint planning, sprint reviews, and feedback sessions as often as he could, to help us and provide more guidance, besides the one our SM were providing.

2 Initiation and planning

Description of the main activities, work packages, milestones.

2.1 Main activities

The different activities that were consider the most valuable are described below.

2.1.1 Release planning

Before the Sprints began the developers met with the PO to get an overview of the Product Backlog items that the PO expected to be completed for the final product. In this activity the developers and the PO aligned expectations of what could be done in the following Sprints, where the outcome of this activity was a long term estimate of what could be expected to get done in the Sprints.

2.1.2 Sprints

There has been six Sprints in the project, that all had a start- and end date. An overview of the Sprints' duration, start- and end date can be seen in the Gantt chart in section 2.2. In the following sections (First Sprint - Fifth Sprint) the outcomes of each Sprint are described. It should be noted that there is no section called "Sixth Sprint" because the team had not finished the sixth Sprint when this report was written.

2.1.3 First Sprint

3. - 12. of March
In the first sprint the team managed to setup the project and got a common understanding of the shell that was going to be the foundation for the entire project. Another outcome of the sprint was to experience working in a Scrum team with the associated agile methods. The first sprint was very chaotic and the team realized what not to do in the following sprints, and to set a new direction of the ways of working.

2.1.4 Second Sprint

15. - 26. of March
In the second Sprint the Scrum team gained experience in communication with the PO about tasks in the Sprint Review, and thereby decided to have more focus on Sprint goals and communicating more with the PO in future sprints. The team also had an improvement in ways of working as a Scrum team, and it was discussed in the Sprint retrospective.

2.1.5 Third Sprint

5. - 16. of April
In the third sprint a lot of increments were made through the Sprints compared to previous sprints. The scrum team also had a thorough Sprint retrospective and decided to implement the following in future sprints:

- Extra working days
- Rotating groups and tasks more often during the Sprint
- Refine the Scrum sprint backlog during the Sprint
- Keep track on product backlog items, estimations and increments for the purpose to track progress e.g. a burn up chart

2.1.6 Fourth Sprint

19. - 30. of April
In the fourth sprint, the team completed all the Sprint backlog items. The team experienced an increased understanding of estimating tasks during the sprint. The need to prioritize the tasks by "need-to-have" and "nice-to-have" became clear through this sprint. This was also the first sprint where the progress were tracked and a burn-down chart were made.

2.1.7 Fifth Sprint

3. - 14. of May
The fifth Sprint was very chaotic and a lot of things did go wrong. This was

among other things communication between developers. This problem was discussed in the Sprint retrospective. Here, the team agreed to have a better communication within the group. Furthermore, the developers should try to estimate how much time could be used on the project in the following sprint, so that the sprint planning event could be more realistic in terms of the amount of story points that could be finished through the sprint.

2.2 Activity Network diagram & Gantt chart

2.2.1 Activity Network diagram

The Activity Network diagram is used to find the critical path of the project, which will be used in the Gantt chart (in section 2.2.2)

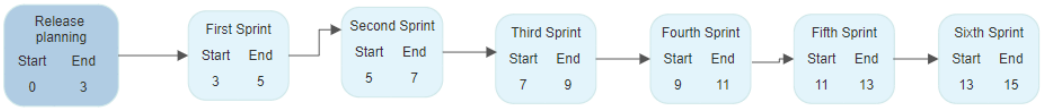


Figure 1: Activity Network diagram

Since there is only one path through the activities, this path is the critical path. The unit that is measured between Start and End is weeks. For example the Release planning would take 3 weeks based on the Activity Network diagram.

2.2.2 Gantt chart

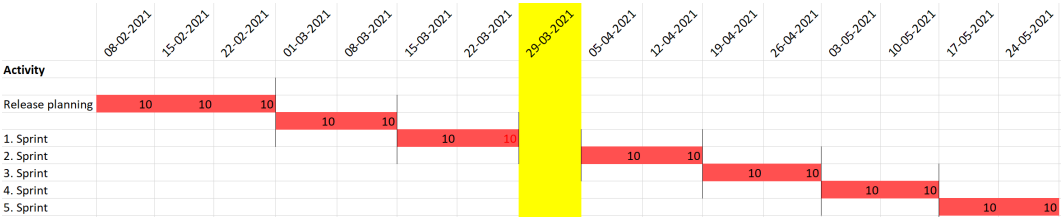


Figure 2: Gantt chart

The Gantt chart shows an overview of the project's timeline. Here the different activities' start and end dates can be seen. It is measured in hours pr. student pr. week.

2.3 Milestones

The team have experienced/gained different milestones. Some of them are more obvious than others for the developers, and some milestones have come through

the project. All the types of milestones are described in the following subsections.

2.3.1 Sprint goals

The team had some milestones defined as Sprint goals. These milestones were more obvious for the team, and were set in each Sprint, where each sprint goal achieved was one step closer to the product goal. Some achieved Sprint goals were more important milestones for the team than others. For example the first sprint goal was:

- Setup basic shells and hosting

The first sprint goal was important for the team because the developers felt that now the "real" development could begin. All developers had a common understanding of the shell, and a shared starting point.

2.3.2 Feedback sessions & assignments

Other milestones through the project were the feedback sessions & assignments in the course Software Development in large teams. This is because both the feedback sessions and assignments were done at specific times in the project, e.g. making an assignment after every other sprint.

Furthermore, when the team worked with the assignments, the developers discussed and reflected upon what was learned and experienced through the sprints.

The feedback sessions gave a lot of information and feedback that the team tried to implement in the future sprints. The team learned a lot from the feedback sessions and considered them as important milestones in the project.

3 SCRUM

3.1 Description of the scrum.

3.1.1 Scrum and alternatives.

For this course and project it was required to implement SCRUM as the process model.
Scrum creates a self-managing team, and within the Scrum environment the team consisted of one SM, one PO, one agile coach and eight developers.

There was several alternatives to Scrum, that also works projects, but would have added other values than scrum, which this project wouldn't necessarily have benefited from.

Extreme Programming is another process, that is a more prescriptive way of agile compared to Scrum. Here specific technical practices are required, including pair programming and test driven development.

Lean Software Development is a process with no specific roles, and is more relevant for repetitive industrial work. This process has the principles of eliminating waste, amplifying learning, decide as late as possible, deliver as fast as possible, empower the team, build in integrity and seeing the whole picture.

There are several options other than Scrum, and both Extreme programming and Lean Software Development have great qualities. With that being said, Scrum would still be a better fit for this specific project. The team has benefited tremendously from having specific roles, since the developers are students working with an outside company, it was important to establish roles. Within the process great use of pair programming was used, but the developers also made use of swarming, which was an option within Scrum, since it has no technical requirements.
Therefore, although Scrum was not an optional choice in this project, it has proved to be very effective, and a great way for the developers to work as a team.(4)

3.1.2 Processes of Scrum - events

In the process of Scrum, the team went through and used several elements. The project had a duration of 6 sprints. Within these sprints there activities included sprint plannings, daily scrums, sprint reviews and sprint retrospectives. The artifacts used within these process elements were a product backlog and sprint backlogs, which will be further extended in section *6. Initial backlog.* Each sprint lasted approximately two weeks, all with the same constant duration.

Sprint planning
Every sprint within this project started off with a sprint planning. At the sprint planning it was discussed why this sprint would be valuable. The PO would explain what he thought would be important to get done in the upcoming sprint, and what he hoped the result would look like at the end of the sprint. Then there would be a mutual discussion of what could be done in the following sprint. This included how many tasks, the team would be able to take from the product backlog and put in the sprint backlog. The tasks would be evaluated in importance as to reach the expectations of the PO. When this was done, each task was estimated with the numbers 1,2 and 3. This was done to clarify an overview of the difficulties of the tasks, and to create an idea of how much time a developer would need to complete a given task. Lastly a sprint goal was established, which contained the overall goal of the tasks that should be done

until the next sprint. By having a sprint goal it made it easier to track the progress of reaching the goal for each sprint.(3)

The sprint planning was an efficient way for the developers to gain an understanding of what the PO wanted for the exact sprint. It was a great starting point, and it helped knowing that everyone was on the same page. Equally it made the working hours more sufficient, since there was no questions as what to work on, since this was clearly clarified during the sprint planning.

An alternative to sprint planning is a *CheckPoint Meeting*. Within this meeting the agenda focuses on the endgame, celebrating small wins, identifying lessons learned and risks, reviewing a budget, managing issues, and planning the work to come. This is equally as nice approach, and it sums up several of the points from a sprint planning, and a few from the sprint retrospective. However the *CheckPoint Meeting* approach would be a better fit for a company, due to the focus on planning with a budget, which would not have been relevant for this project.(1)

Daily Scrum

The daily scrums occurred four times within each sprint. The purpose of these daily scrums were to inspect the progress in the sprint. These meetings had a duration of approximately 15 minutes, and each developer explained what they had done so far, what they were going to do, and if they had any obstacles with their task. These meetings were an essential and important part of the process, since it gave the developers a chance to update each other and help out with possible obstacles. Within these meetings, the sprint backlog was adjusted if necessary, meaning that if a task was estimated wrongly, or if product refinement was necessary.
Usually the daily scrum should occur every day. Due to other school projects the developers were not able to work on this project every day, and therefore it made sense to narrow it down to 4 meetings per sprint. In this way the team used an alternative to the Daily Scrum Stand-up meeting by having less frequent meetings.(3)

Sprint review

The sprint review happened at the very end of the sprint. The purpose of this meeting was to inspect and adapt the product. The developers presented their work to the PO, and a discussion of whether the sprint goal was reached or not would begin. This was a great way to conclude the work done in the sprint, and make sure it was as the PO expected it to be. This was a key to maintain a healthy communication with the entire scrum team and to make sure that the project was moving in the right direction.(3)

Sprint retrospective

Like the sprint review, the sprint retrospective also occurred at the end of each sprint. Where as the sprint review was meant to inspect the product, the retrospective was meant to inspect and adapt the process. During this meeting the entire scrum team would get together and examine the previous sprint, with regards to each individual developer, the interactions within the team, the process, tools, and definition of done. Every retrospective was divided into some discussion topics, *What worked well*, *What did not work well*, *What should we improve* and *Things we wished we knew at the beginning of the sprint.*(3)

These meetings led to a nice discussion and reflections upon the previous sprint. The entire team benefited from these meetings since it created a good foundation for the upcoming sprint. The retrospective guaranteed that each sprint was an improvement regarding to the topics discussed.(3)

3.1.3 Processes of Scrum - Artifacts

In the process of Scrum, we went through and used several artifacts. The artifacts used in this project included a *Product Backlog & Sprint Backlog*, *User stories* and *Definition of Done*, which all proved to be very helpful for the project.

Backlog

An important artifact of the Scrum process is the product backlog. The product backlog contains all the requirements for the project in progress. This list of tasks is prioritized in what is the most important or urgent task to be completed in that sprint. Tasks from this product backlog is moved to the sprint backlog during sprint planning. Having a product backlog is a very helpful tool, since it helps making it possible to keep track of all the tasks the project required, and it creates an idea of how far along in the project one is. The specific for this project's product backlog will be extended in section *6. Initial Backlog.*

User stories

One of the process elements of scrum is *User stories.* The user stories works as a communication channel between the clients, and the developers.
A user story is written for every product backlog. This is done to make sure that the developers know exactly what the given task is suppose to accomplish. At the sprint planning it is possible to have a conversation with the PO about the user stories, to make sure that they are correctly understood.
There are several alternatives to user stories, including *Initiatives*, and *Problem stories.*

Initiatives is a very great tool, especially when working across different teams, since different teams can split up the work in Epics and smaller backlog items.

Problem stories explain a problem that needs to be solved, rather than a feature a user would like to do. It is very important that one would understand the problem, before creating a problem story, when using this approach.

These are both great tools, although user stories is the best process tool in this case. This project consisted of only one team and there has not been problems, but rather features that would be nice to implement for a user. Thereby making user stories a great process tool for this particular project.(6)

Definition of Done
This is a very important artifact of the scrum process. It is a description of the meaning of being done with a project. For this project the definition of Done was the following

- A working site with a functional login system, payment function and GDPR information for the customers
- Rather a working system with main functionalities, than having a lot of half working "nice to have" features
- Easy to use (Usability)

Each task in the product backlog was measured against these descriptions, and once a task met the definitions it became an increment. In this project, the first two statements were more related to the final product rather than the definition of done for a task. Where as the last statement regarding usability was meant to be taking into consideration when working with a task from the backlog, and a way to define when it was done. The definition of done has therefore proved to a very nice tool, since it makes it easy to keep a clear guideline for how the tasks and product should be once they are done.(9)

3.2 Configuration of the software process model.

3.2.1 Product owner

The product owners (PO) is accountable of creating and communicating the product backlog. The PO also writes out user stories and helps the developers get a better understanding of what is expected for the product.

In the first sprints the developers did not communicate enough with the PO. This meant that various questions that the developers might had would get answered at the sprint review. As a result to this, many tasks were unfinished at the end of the sprint. The PO requested that the developers should contact him if reaching a wall or being in doubt about specifications of user stories. By increasing awareness of this issue the communication between the developers

and the PO became more frequent.

The team also had a very open communication about how much work would be realistic in each sprint. And quite early on developers and PO decided to cut out one of the less important epics since it would not be realistic that the group would be able to finish the task in time.

3.2.2 Scrum master

The scrum masters accountability is to establish scrum as it is defined in the scrum guide. The SM also makes sure that the team are communicating and help the developers engage with the PO. The SM serves the PO by helping find different techniques, product backlog management and product goal definitions. Furthermore the SM ensures that all scrum events are performed in a positive manner and brings value to the team and the work.

Our SM were present at most stand up meetings and would run them by asking questions like "Which task are you working on?", "What is holding you back?" or "Who can help you overcome this problem". This often helped us vocalise our problems and if needed get help from other team members.

In addition to the traditional SM tasks the SM of this project would also help the developers if they got stuck with programming aspects of the project. The developers were also able to contact the PO regarding technical issues since the PO of this project had a an IT-related education. In the first sprint the SM also came with suggestions as to how the website could be structured and how to connect front-end, the API and the database together.

3.2.3 Agile Coach

An agile coach is responsible for creating and improving agile processes within the team.**??** The agile software process model is about reducing overheads in the software. This can be by limiting document and thereby fostering communication. The focus is also more on the code rather than the design. It is an iterative approach to software development and is intended to develop software quicker and evolve by meeting changing requirements.

In this project most contact with the agile coach was at the end of each sprint when the team had the sprint review and retrospective. Here the agile coach would arrange different ways to evaluate the completed sprint. If a developer had some problems with how the teamwork had been the agile coach would come with suggestions to how this problem could be solved. This method would be evaluated at the next retrospective.

3.2.4 Developers

A team of developers typically consists of 5-9 people, in this case the team consisted of 8 developers. The developers are responsible of planning the sprint from the sprint backlog in collaboration with the PO. They also have to be able to adapt the plan each day if necessary. The team members also have to hold each other accountable as professionals and make sure that every one is working and sticks to the agreements made in the team.

Amongst the team of developers each developer possessed a specific skill within different areas of the program. Some were better at setting up the database, and others were better at front-end and some were better at back-end. The first sprint the team worked in bigger groups so everyone had knew the foundation of the program. In sprint 2-4 the developers worked mostly with the parts each were more skilled at and a result of this was that the smaller groups didn't rotate. In the last two sprints (sprint 5 and 6) the team decided to split the fixed groups to invite each developer getting familiar with new parts of the program. The intention of this was to increase the learning experience for all the developers.

3.2.5 Scheduled work days

The agreed schedule for the sprints was that the sprint plannings would be every other Monday and the sprint review and the retrospective would be the last Friday in the sprint. In the beginning of the project the team came to an agreement of scheduled work days, Wednesday and Friday. In this arrangement it was stated that the work days weren't mandatory because of other courses the developers had to attend to. The developers introduces two more working days since they quickly realized that two working days was not enough. Therefore Tuesday and Thursday also became working days.

In the first three sprints the stand up meetings was scheduled on Thursdays and Sundays. The developers decided change this since it was considered more convenient to discuss the work that had been done on that particular working day rather than waiting postponing it. This gave more value since the work that had been done was fresh in memory.

3.3 Overview of the plan.

Each sprint cycle had a duration of two weeks. Where the sprint started on a Monday and ended on a Friday. As for the different consecutive stages the sprint planning always took place on the first day of the sprint (on a Monday) and the Sprint review and retrospective took place on the last day of the sprint (on a Friday). The daily SCRUM did not take place every day of the sprint since the developers also had other courses to attend to. Therefore the daily-scrum meetings started by taking place every ...

4 Technology ecosystem

4.1 Discord & Messenger

For communication the team has been using Discord and Messenger. Messenger has mainly been used by the developers with a more casual approach whereas discord has been the official communication platform for the whole team.

Discord has allowed us to gather all important communication on one platform. A discord text and voice channel were created, where the text channel has been used for communication with the PO and the SM.

The voice channel has been used for daily scrums, sprint reviews, sprint retrospectives and sprint plannings. The discord voice channel has been a great tool for meetings and screen sharing, and the team didn't at any point consider switching communication platform.

It did however have its shortcomings. A single voice channel was never enough, as the group often worked in pairs when doing pair programming. This resulted in some disorganized teamwork, as some pairs were in other servers and some used zoom calls. It became a challenge to figure out where the different parts of the team were located, and what tasks had been completed, which became a problem when the team had to aggregate the work done in team work sessions. To avoid this problem, and to make sure that the whole team was on the same page, the group rescheduled the daily scrum stand ups to be at the end of every work sessions, in the main voice channel.

Another shortcoming of discord was having only a single text channel. The developers quickly discovered that having all text communication in the same channel became cluttered. Important messages would drown in unimportant messages and often left unseen resulting in miscommunication. The solution to this was to use messenger. We decided that all communication that was not meant for the PO, would happen over messenger. The developers created two group chats, one with all developers and the SM, and one with only the developers. All developer related communication happened in the messenger group of developers, and all scrum related messages, that did not involve the PO, happened in the other group.

4.2 GitHub

GitHub was used by the team for version control and code collaboration. The whole team had some degree of experience in using GitHub from previous projects and were fairly confident in the ways of utilizing this tool. GitHub ensured that the whole team had access to the most recent version of the program, and let all developers work on different parts of the project simultaneously. It was only until the version control induced more struggle than gain, that the

developers questioned the ways of working with GitHub.

To begin with, the team of developers had no collective idea about how to use GitHub and the developers all used it in their own way. This resulted in struggles as everyone worked on their own branches for the entire sprint and then the day before the retrospective, the developers tried to collect all pieces of code in a master branch.

The struggles were expressed in a scrum retrospective, and the SM planned a GitHub guide in the resulting sprint planning.

The GitHub guide changed the way the developers worked with Git. A lot was learned, and a agreement of how to use GitHub was established within the team. All coding sessions should begin with a pull from the master branch and a creation of a new branch. When a usable function had been implemented on that branch, it should be merged with the master branch. This should in most cases happen multiple times during a coding session, and at least happen at the end of the coding session. This ensured the shortest possible branches, and little to no merge conflicts. Improving GitHub customs made the work a lot more effective, as less time was spent on merge conflicts.

4.3 Trello & Forecast

In the first 3 sprints Trello was used as a scrum board. Trello helped organizing the sprints with cards on lists with different labels and descriptions. The lists were divided in epics, sprints, product backlog, sprint backlog, doing and done. At the beginning of the project, the PO filled the epics list with cards that described the different epics, that were the bigger features of the program. (See 10.Appendix 10.1)

The epics were broken down into smaller tasks with user stories, and labels that connected them to their Epics. At each sprint planning tasks were moved from the product backlog and into the sprint backlog. The tasks were given estimation points, and during the sprints the developers could assign their name to the specific task. The cards were moved to the doing section when developers started to work on them, and done when they were finished.

This was how it was supposed to work out, but it did not quite go that well. The Board quickly became cluttered when a lot of cards were added to the board and it was a challenge to visualize which tasks belonged to what epic and sprint.

There were also no means to visualize how much work had been done in the different sprints, as opposed to how much work that should have been done to be on time. To overcome this issue, our SM introduced another scrum board after the third sprint called Forecast. Switching scrum boards in the middle of a project were a risk, since it meant spending time getting accustomed to a new

tool set, when the team had already become familiar with Trello. It was a risk that the developers were willing to take as it could potentially result in better sprints.

Forecast essentially works in the same way but has some key features that differentiate it from Trello: The ability to filter tasks by sprint or epics, so that the team could focus on a single sprint instead of seeing the whole project at all times. Different visualization tools that showed the amount of work that had been done and how the work were distributed in the sprint in form of burndown and burnup charts. This made it easier to tell if the developers were doing less work than usual, and it also helped the group pin point a mistake that the developers tended to make in all sprints. The developers forgot about the scrum board during the sprint, and then the day before the retrospective, all cards were moved to done.(See 10.Appendix 10.2)

4.4 Miro

For this project Miro was used for the sprint retrospective which has been a great tool. Miro is a white boarding platform which can be used in a range of different ways, and the agile coach introduced different boards, all with the same goal - retrospective. The way Miro was used within this project was by writing on sticky notes and placing the notes in different fields. An example of this could be a board with fields with the topics *more of*, *keep doing*, *start doing*, *stop doing*. The team would write bullet points about what they wanted to do more of or less of, and after a discussion, this could turn into action points for the next sprint. (See 10.Appendix 10.3)

5 Cooperation practices

In the following section the collaboration within the scrum team will be elaborated. This includes different social events and also basic cooperation techniques used within the project.

5.1 Social physical events

Since Covid19 has had its fair share of dominance on physical meetups during the process of creating the website for ProteQ, the developing team decided to work mostly via Discord. This meant that all questions for the PO and SM had to be conducted online. After the restrictions eased up, the developers decided to host a group event. The developers got tested and on the 24th of April, our first and only physical meetup during the Second Year Project was held.

This event was an improvement for the team of developers, and the way of working. All developers knew each other before this project, but had never

worked all together on a project of this size. During the first three sprints, some communication difficulties and a lot of old habits residing from working with the waterfall model started resurfacing. Some of the developers were more experienced in some specific part of software development while others were not. This resulted in the developers only doing the specific tasks they excelled in.

The physical meeting the developers had, introduced a much needed physical dimension to the very separated lifestyle that Covid19 enforced. Along with finally being able to meetup physically, another important aspect of working in larger groups came up. Socializing did inevitably make the teamwork and cooperation between developers prosper. This was especially evident in the way of engaging in the physical event where the team discussed in detail about the Agile process, thereby enforcing the agile process instead of being in the waterfall induced thinking.

5.2 Social online events

Another important part of making a large group of many different people cooperate is to make sure fun is incorporated. Since the first and second sprint were not as organized and successful as the team would have hoped, the SM and PO suggested an interesting idea during the Sprint review of Sprint 2. Instead of being held back in some particular problem, the PO suggested that during the Easter Holiday, the group should consider doing a team building activity to enhance our cooperative skills.

Since Covid19 was at its peak during the Easter Holiday, it was not allowed to meet up physically. Instead, the developers arranged to meet on Discord with our cameras on and play a game of Skribbl.io. Even though it was an online event, it really helped the team spirit to see the other developers in a more relaxed context instead of only getting together in a professional matter.

5.3 Recaps after each session

The first Sprint was a very chaotic experience. The developers were not communicating correctly with the PO and SM and there was a lack of cooperation between the developers. Most of the developers had a good grasp of how the website was supposed to turn out, but the basic question always arose: "What have we done today?". To eliminate this query after every single work session, the team decided to do the Daily Scrum at the end of the work session. This change was implemented after our first Sprint review. The cooperative coherence in the group automatically increased because of this slight change in the way of working.

5.4 Swarming a task

The traditional approach of task delegation was during the Sprint, where the developers would split into smaller groups to do different items from the Sprint Backlog. For the first sprint, the team decided work together to get a shared understanding. This included the swarming approach to software development, where all of the developers worked on one particular big task. Swarming helped by setting up the fundamentals for the rest of the Sprint which allowed a deeper understanding of the basics, so the smaller tasks would be easier to complete. The tasks, the team decided to swarm was *setting up the shell* and *configuring the basic hosting solutions*, which were especially important to swarm since it laid out the foundation for the rest of the project.

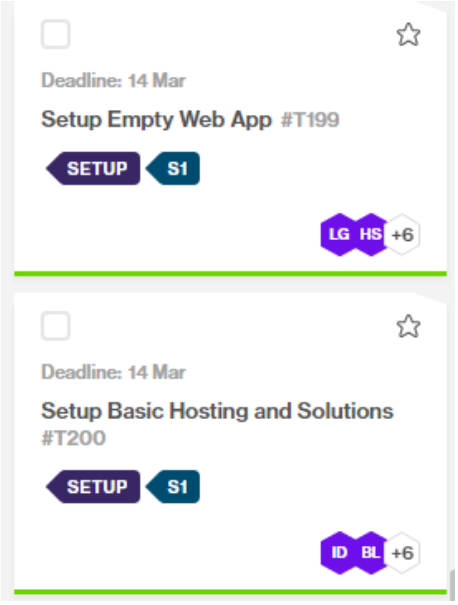


Figure 3: The tasks "Setup Empty Web App Setup Basic Hosting and Solutions" each assigned to all the developers.

After the first sprint, pair programming was the preferred way of working, consisting of each pair looking into their own specific part of the website. Still, some problems were apparent. There was some misunderstandings regarding the database and how to connect it to the API. Therefore, for the Sprint Planning in the second sprint, the developers, SM and PO all agreed to take a look at the swarming pattern again before splitting up into pairs.

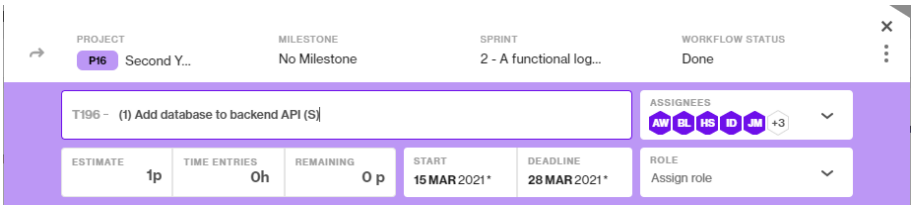


Figure 4: The task "Add database to backend API" with all 8 developers assigned to it

The task *Add database to back-end API* was especially important to swarm since our basic understanding of databases were not up to par compared to other parts of the program.

Another use of swarming a large task was used during the fifth sprint of the project. This particular task included the research of how to store the guide contents.

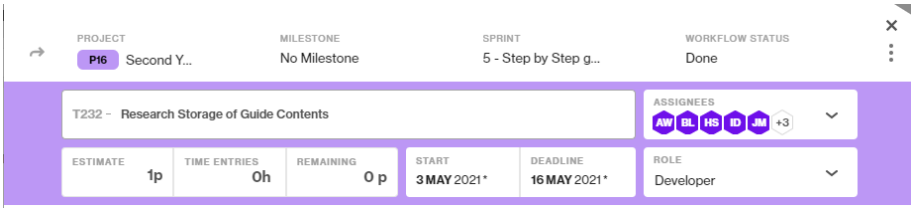


Figure 5: The task "Research Storage of Guide Contents" with all 8 developers assigned to it

The main reason for changing up the cooperation techniques was because of the lack of a deeper understanding regarding the basics of the product. Some of the developers were naturally more skilled in the database department, while others struggled. The swarming pattern really helped establishing an even foundation for all the developers and also allowed for an easier handling of the next tasks in the sprints to come.

6 Initial backlog

The initial backlog was created and prioritized by the PO. Before the kickoff of the project the initial backlog was discussed and refined at a meeting among the developers, the PO, the SM, two stakeholders and the agile coach. At this meeting the goal for the developers was to gain a better understanding of the business domain and the project in general.

The PO began the meeting by presenting the backlog where the most important items subsequently was up for discussion. After an agreement was reached within the team the refinement of the product backlog took place.

After all the details of the backlog was agreed upon, a Scrum board was formed with the tool "Trello" which later was exchanged to "Forecast". This change was elaborated in chapter *4. Technology ecosystem.* The epics of the backlog was therefor as follows:

Epics:

- User to mark step as completed.
- Admin dashboard.
- Document layout and document editing.
- User dashboard.
- Forum.
- Architecture and hosting.
- Invoices.
- Registration.
- Design fixed document layout.
- Step by step Guide to GDPR.
- Compliance calendar.
- Site membership system.

The entire backlog can be seen in chapter *10. Appendix 10.4*

7 Sprints

Introduction?

7.1 Sprint 1

The goal of the first sprint was to set up basic shells and hosting and understand the software stack that the team chose for the web application.

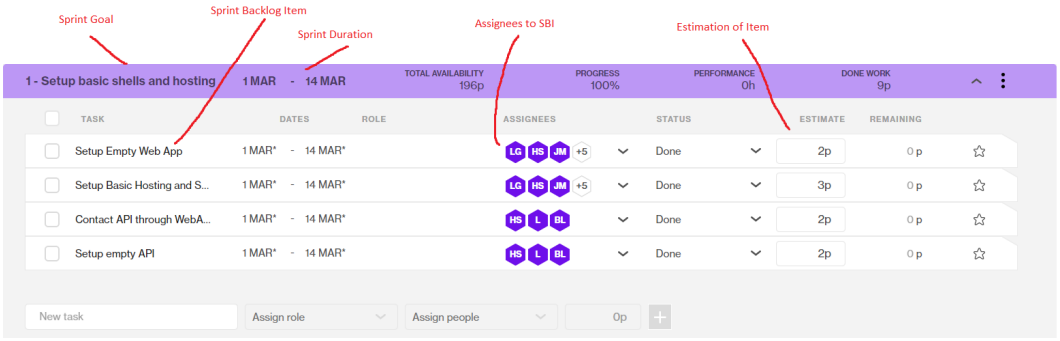


Figure 6: Sprint Backlog & Sprint Goal

In the first sprint, the developers worked together as a team to get a common understanding of the fundamental code and components that had to be implemented and used. In the sprint planning, the developers decided to go with a swarming approach when setting up the basics for the project. When and how long the developers worked in the first sprint was very unstructured. There was no tracking of time and the overall communication was informal and unstructured - the developers used to call or write to each other on their social platforms and ask if everyone was available to meet and work on the project. Often all the developers' schedules did not allow for everyone to work synchronously, so therefore it was decided to work on days where most people had the opportunity to join in.

Developers agreed on working individually on the project if it was desired. The agreement was that the developer would have to inform the developers and demonstrate the work that was produced from an individual session. Did the team reach the sprint goals? The answer was no and the scrum team concluded that it was the lack of communication between the developers and not having mandatory workdays. This was agreed upon in the sprint review and therefore it was decided that sprint backlog items would be split into smaller groups of developers. Also, a schedule was set for more mandatory workdays. Referenced from the first retrospective was that the team should use our Scrum board on Trello to increase the process and from that increase the quality of work. Our PO Matias made the point that our first sprint was very similar to the scrum simulation where dividing resources was our weakness.

Events

- Sprint Planning - the 3rd of March
- Daily Scrum - the 4th of March
- Daily Scrum - the 7th of March
- Daily Scrum - the 11th of March

- Review & Retrospective - the 12th of March

7.2 Sprint 2

In the second sprint, the developers were split into smaller teams which resulted in the working process being much more efficient. An explanation for this was that everyone had a clear idea of what to work on. The communication had also been improved since the first sprint which resulted in almost reaching the sprint goal for the second sprint. Since setting a fixed schedule for developing hours it has been easier to keep account of the time spent on the project. But despite getting a better result in the sprint the developers still encountered problems with specific sprint backlog items that caused developers to be in a limbo of not knowing what to work on instead. For instance, implementing a QuickPay API had problems because of bad documentation and integration with the software stack, especially the front-end in React. A lot of time and effort was put into QuickPay which caused frustration since the developers couldn't implement it. Therefore the developers looked into implementing payment in another framework which was then later discussed with the PO. The item of implementing QuickPay therefore became a backburner.

In the retrospective the use of pair programming was reflected upon and which pros and cons it caused. It was decided that communication was especially more crucial since developers could lose sight of the project as a whole when only specifying with parts of the project.

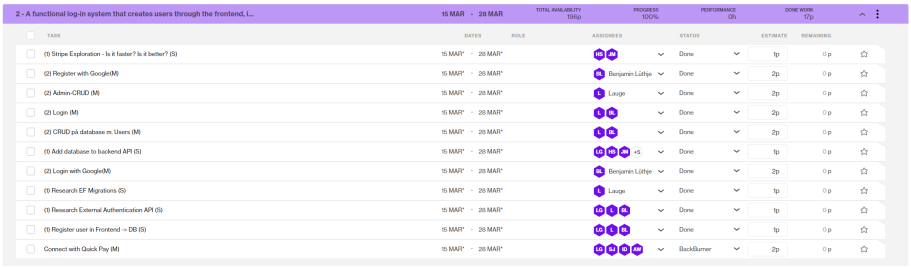


Figure 7: Sprint Backlog & Sprint Goal

Events

- Sprint Planning - the 15th of March
- Daily Scrum - the 18th of March
- Daily Scrum - the 21th of March
- Daily Scrum - the 25th of March
- Review & Retrospective - the 26th of March

7.3 Sprint 3

The third sprint started with some loose ends from the second sprint. From the last sprint retrospective, it was decided that developers should have more than only one task per team so in case a team hit a wall they could start on something new and then get back to that task or ask other group members for help. Therefore we had a large number of tasks in the sprint. The goal was to get a functional Login/signup flow for the user, with payment and password functionality. Since developers had never implemented such a user and login system, there was much confusion on how to approach this task. Much time was spent on researching this and trying to implement login with a google framework.

This was dropped later since the PO wanted a user to be able to log in with any email which developers hadn't gotten clarified during the sprint planning. The sprint goal wasn't reached and a lot of sprint backlog items were still in the doing phase on Trello. It was therefore decided in the review to move them to the next sprint. In the retrospective, it was concluded that developers needed to clarify sprint backlog items more since developers wasted time on features that were nice to have. Therefore it was decided that developers should communicate more with the PO to clarify what was nice to have and what was need to have. Since our PO also has technical expertise he encouraged developers to question him on how features of the web application could be implemented.

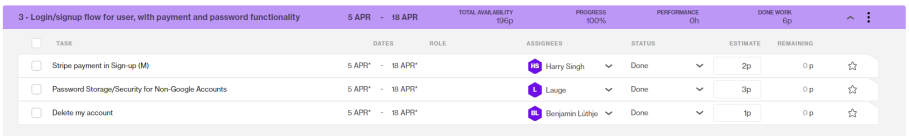


Figure 8: Sprint Backlog & Sprint Goal

Figure 8 shows the sprint backlog which was completed during sprint 3. The rest of the sprint backlog was moved into sprint 4 as shown in Figure 9 and -10.

Events

- Sprint Planning - the 5th of April
- Daily Scrum - the 8th of April
- Daily Scrum - the 11th of April
- Daily Scrum - the 14th of April
- Review & Retrospective - d. 16. of April

7.4 Sprint 4

The fourth sprint started with a few unfinished tasks from sprint three. Again we focused on clear communication and delegation of the different tasks. In

this sprint, we decided to have a daily scrum at the end of each work session Wednesday and Friday, which improved the communication within the group. This made it easier to know what other team members were doing and made it easy to reach out for help.

During this sprint, we finished our entire sprint-backlog and overall felt really on track with our work. In the sprint review, developers got positive feedback on their work and great compliments from the PO. During this sprint, we also exchanged the Trello-board to Forecast. This made tracking our progress a lot easier since we were now able to see graphs, and other visualizations of our work process (see burn down chart Figure 11). Although the Scrum team had a great sprint, several developers were concerned with the fact that we mostly worked with the same people in smaller groups. This led to a nice reflection during our retrospective, and it was decided to try to mix up the groups a bit more for the next sprints.

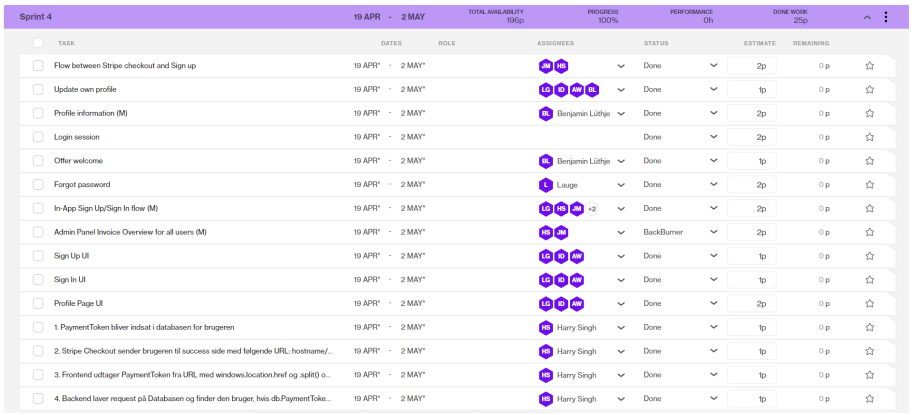


Figure 9: Sprint Backlog & Sprint Goal

Figure 10: Sprint Backlog & Sprint Goal

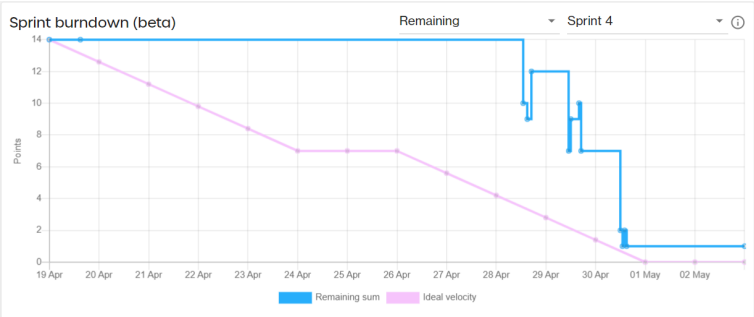


Figure 11: Burndown Chart

Since developers were new to the Forecast framework Figure 11 does not show the full picture. Since developers hadn't moved sprint backlog items to the done column when finishing them, the burn down chart shows that nothing was finished between the 19th of April - 29th of April. But, on the 28th of April, the SM told the developers to remember to mark sprint backlog items as done when finished with the task.

Events

- Sprint Planning - the 19th of April
- Daily Scrum - the 21th of April
- Daily Scrum - the 23th of April
- Daily Scrum - the 28th of April
- Review & Retrospective - the 30th of April

7.5 Sprint 5

The sprint goal of the fifth sprint was to implement Step by Step guides, storage, navigation & displayed. It was decided in the last retrospective that developers who had focused on ex. front-end or back-end should switch to improve the overall learning of the project. Therefore the sprint started with developers spending extra time on getting accustomed to their new field of work.

The sprint goal was not at all reached this sprint and the feedback from the SM and PO was very negative. This was accumulated due to a chaotic review with missing developers and a presentation that didn't please the SM and PO. Overall the sprint was very chaotic and problems with the design of the database also came to light. It was therefore decided from the review and retrospective that the developers should improve the communication which had especially been lacking during the 5th sprint. As a start to sprint 6, it was already decided in the review that the database design should be clarified by implementing an ER diagram for the whole scrum team.

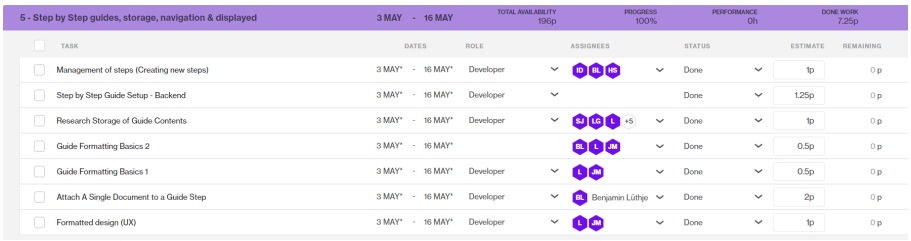


Figure 12: Sprint Backlog & Sprint Goal

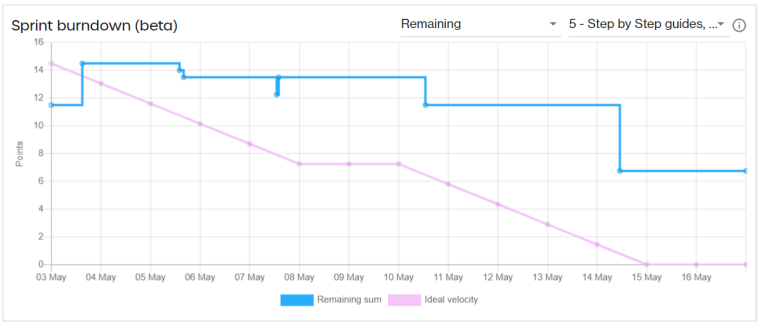


Figure 13: Burndown Chart

Events

- Sprint Planning - the 3rd of May
- Daily Scrum - the 5th of May
- Review & Retrospective - the 14th of May

7.6 Sprint 6

With sprint 6 being the final sprint, all loose ends had to come together to create the final product. Since the sprint had not come to an end, at the time this report was written, it is difficult to write about it.

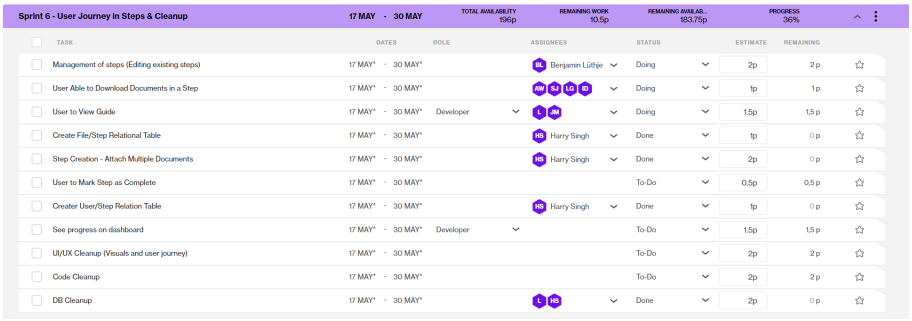


Figure 14: Sprint Backlog & Sprint Goal

Events

- Sprint Planning - the 17th of May
- Daily Scrum - the 23th of May
- Daily Scrum - the 26th of May
- Review & Retrospective - the 28th of May

8 Reflection

8.1 Communication among the developers

The retrospectives were useful to reflect upon the work process in the finished sprint. Here it would be decided what tasks the developers should work on and try to implement in the next sprint.

A topic that came up in almost every retrospective was the communication among the developers. At the first couple of retrospectives it was especially the communication about who would attend the different meetings and stand ups.

It was also a problem that some developers were always late. To try solve this problem it was decided that if a developer was prevented to attend a meeting the person should communicate this to the group. Furthermore, if a person was more than 3 minutes late it would be perceived as being late and they would there by owe the group a round of drinks. This was a fun way to ensure everyone being on time. It also worked really well for the first couple of sprints but in sprint 5 and 6 some developers again began to be late and the communication among the developers got worse. This could be because of the pressure and stress from other exams.

One of the developers took the initiative to make a spreadsheet where all the developers could write which meetings they were able to attend and what

time they were able to attend it. This was useful for the developers who had work on the site and therefore couldn't attend all meetings or could only attend after a specific time. It made it easier for the rest of the group to know who would be present and they didn't have to ask in the group before every meeting who would attend. (See 10. Appendix 10.6)

8.2 Communication with the Product owner

In the third retrospective the PO also requested that the developers would communicate more with him through out the sprint. This was especially if the developers had problem solving a task or didn't understand what they were supposed to do. This was useful to the developers because they had been hesitant to ask throughout the sprint with fear of being a burden to the PO. After the PO voiced the desire to be more involved in the ongoing work the developers began to ask more freely in the discord chat when being stuck or needing guidance. This also made the process go faster because the developers wouldn't be stuck on a problem for to long.

In the sprint planning in sprint 5 the team experienced communication problems once again. The developers were not fully prepared and therefore they didn't know what they should ask about. This resulted in a lot of confusion in the first week of the sprint and a lot of communication afterwards with the PO. To avoid this in sprint planning of sprint 6 the developers met an hour before the sprint planning to go through the backlog and discus if there were the same understanding of the tasks and what confusion about.

8.3 Scheduled work days

In the third sprint the developers finished far less tasks than expected and therefore a lot of the sprint backlog got moved to the fourth sprint. A reason for this was that a lot of the tasks were almost finished but didn't live up to definition of done and could therefore not be moved to the "done"-section on the scrum board. It was decided after this that the developers should have more than two scheduled work days so there were more dedicated time to work on the project. Another reason for the added work days were that some of the developers had work on one of the two work days and were therefore not able to attend these scheduled days. The effect of these added work days were that the next sprint, sprint 4, were the most successful sprint where every task in the sprint backlog were finished.

8.4 Mix of groups & tasks

The developers knew each others' skills and interests through the 'Group Agreement' document. In the start of the project the team focused a lot on the project over the team and the team's processes. So for example a task was assigned to a team member that could fix the task fast and then move on, and by that prioritizing to finish as many tasks as possible, rather than prioritizing the potential

learning outcome for another team member. This problem was first discussed in the third Sprint retrospective, where the group decided to start rotating groups and thereby tasks more often, so that every group member could get a broader learning outcome. Furthermore, during the sprint the group began to focus more on the team rather than the project. An example of this, was in the sixth sprint where the team assigned tasks to team members that wished to learn more about a specific component that fitted with the tasks.
(See 10.Appendix 10.5 Group Agreement Template)

Although the team made some modification in the team's ways of working, it was sometimes difficult to uphold. One of the modification was to mix up the working groups and tasks more often, but since some of the sprint backlog items were not yet finished during the given Sprint, they were moved to the next Sprint. This led to difficulties mixing up the pairs, since the familiarity of a specific part of the project hindered change in the programming pairs. It was more natural to continue working on the Sprint backlog item that was not yet finished for the upcoming Sprint, instead of suddenly switching up the pairs.

After Sprint 4 had concluded, a new epic was introduced which was not a follow-up from an older product backlog. It was a fresh start and therefore easier to change up the pairs meaning that the new pairs could work on something that was the opposite of what they had done in the earlier sprints. This gave each developer a better insight of how the different components worked together.

8.5 Individual- & pair-programming

The first sprint began by swarming a task, which made sure that all the developers understood the fundamental code. After swarming the first sprint backlog item the developers split into smaller teams. By implementing pair-programming into the working methods the developers gained a more personal connection to each other. This lead to a stronger engagement from all the team members and increased the amount of tasks being done.

9 Conclusion and final thoughts

The second year project has been a very educative experience for all the developers. Working in a large scale software development team was more challenging than it first seamed to be, but the team gained confident in being more capable and efficient in the agile way of working.

As a team it was a rocky start, but lessons were learned from each mistake and the sprints became better and better. Throughout the first sprints there were a lack of communication both internal between the developers, and with the PO and SM. This was brought to everyone's attention within the sprint

retrospectives, and it was then improved in the next sprints.

The communication became more sufficient, and there were a switch to technologies that worked better for the team. Equally there was a deeper focus on retrospectives, sprint plannings and daily scrums.

The team experienced the sprint planning to be a great aspect of organizing and it made it easy and more manageable to start a new sprint. Equally the daily scrums turned out to be of great use for the developers, to keep a clear picture of how far along they were in the project and what the other developers had accomplished since the last meeting. Through sprint retrospective and review the team gained great knowledge in reviewing their own work and work with self-criticism, to help make the upcoming sprints even better. This agile way of working all resulted in more efficient work of better quality.

Working with full stack development was a great experience as it tied the different aspects of the courses in the bachelor together. Equally the developers gained more experience within different parts of software development.

This cooperation between students from the Software Development program at ITU and the company ProteQ has been a great experience. It has created a great understanding of the use of Scrum, and the importance of using a great process model, when working with projects of this size and importance.

References

[1] Smartprojx: Why CheckPoint meetings are a better alternative to sprint panning meetings
https://smartprojex.com/better-alternative-to-sprint-planning-meetings/
Visited: 22-05-2021

[2] Lecture on Scrum and agile project management
https://learnit.itu.dk/pluginfile.php/281678/mod_resource/content/0/%5BBSUP21%5D%20-%2002%20-%20Scrum%2001.pdf
Visited: 22. May 2021

[3] Scrumguides: The Scrum Guide.
https://scrumguides.org/scrum-guide.html
Visited: 22-05-2021

[4] ExtremeUncertainty: Alternatives to Scrum.
https://www.extremeuncertainty.com/alternatives-to-scrum/
Visited: 22-05-2021

[5] Principles behind the Agile Manifesto.
https://agilemanifesto.org/principles.html
Visited: 22. May 2021

[6] Unheralded Alternatives to User Stories
https://medium.com/serious-scrum/unheralded-alternatives-to-user-stories-f1d787fc2278
Visited: 24. May 2021

[7] Bob Hughes Mike Cotterell: Software Project Management, Chapter 7 Risk Management
https://agilemanifesto.org/principles.html
Visited: 24. May 2021

[8] What Does an Agile Coach Do and How Can You Become One?
https://www.toptal.com/project-managers/agile/what-is-an-agile-coach
Visited: 22. May 2021

[9] Lecture: Scum a detailed overview
https://learnit.itu.dk/pluginfile.php/281678/mod_resource/content/0/%5BBSUP21%5D%20-%2002%20-%20Scrum%2001.pdf
Visited: 24. May 2021

10 Appendix

10.1 Trello Board

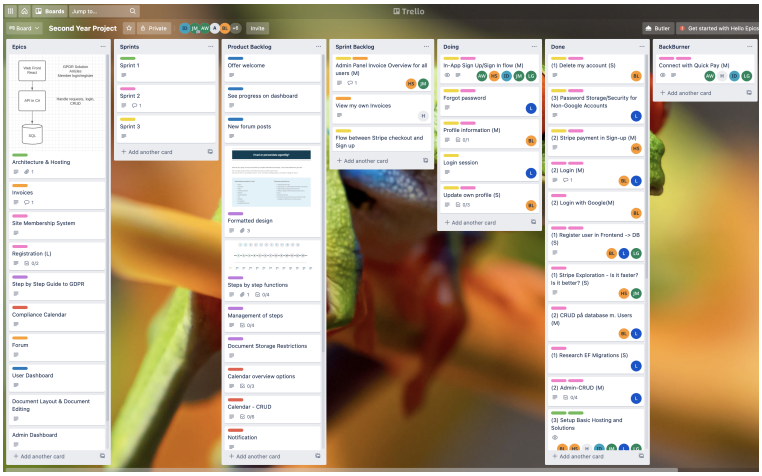


Figure 15: Trello Board was the first scrum board used for this project

10.2 Forecast Board

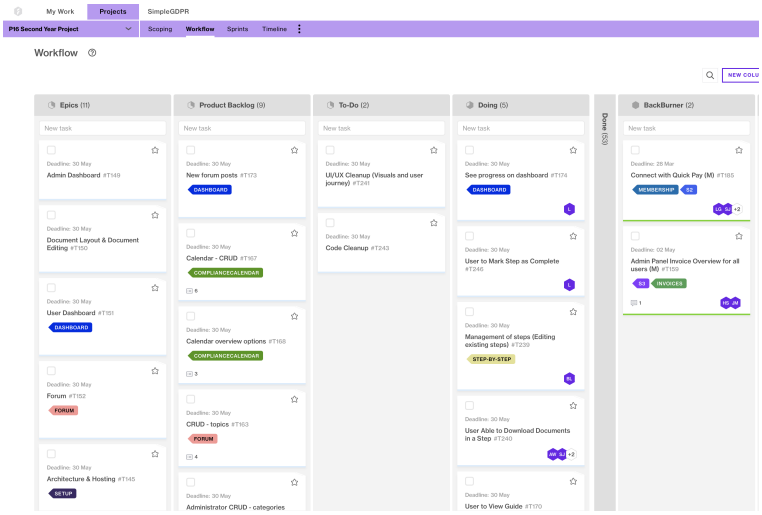


Figure 16: Forecast Board was the second and final scrum board used for this project

10.3 Miro Board

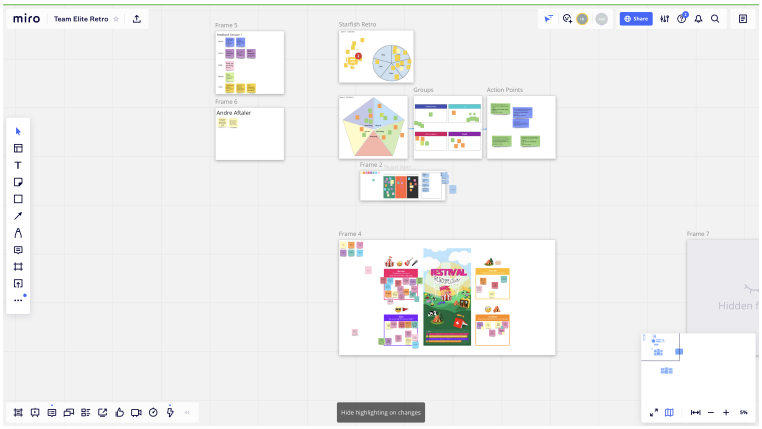


Figure 17: Miro Board was the board used for the retrospectives in this project

10.4 Product Backlog

Admin dashboard

Document layout and document editing

User dashboard
• Offer welcome
• See progress on dashboard
• New forum posts

Forum
• Administrator CRUD - categories
• CRUD -topics
• CRUD . reply

Architecture and hosting
• Setup Basics Hosting and Solutions
• Setup Empty Web App
• Setup empty API
• Contact API through WebApp

Invoices
• Admin panel invoices overview for a user
• View my own invoices

Registration
• Register user in frontend -> DB
• Stripe payment in Sign-up
• In-App Sign Up/ Sign in flow
• Stripe exploration
• CRUD database m. Users

Design fixed document layout

Step by step Guide to GDPR
• Steps by step functions
• Management of steps
• Document storage restrictions
• Formatted design

Compliance calendar
• Calendar overview options
• Calendar CRUD
• Tasksystem

10.5 Group Agreement Template

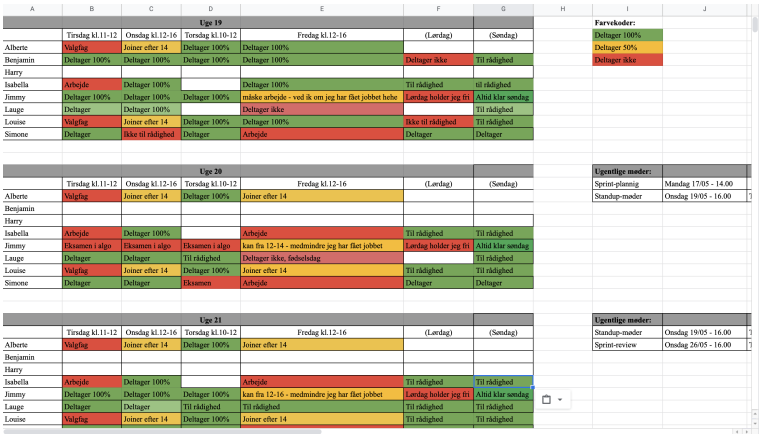
Group Agreement Template

1. Group Name	*Something cool - get creative and make a logo*
2. Group Learning Goals	*What are the collective learning goals of the Group, and/or areas of improvement?* 1. Better at coding 2. Know your role in a large team 3. Mastering Scrum • Get better at developing large projects • Mastering individuel coding parts of a large project individuel • Teamwork in larger groups • Collaborating with a company • Getting better in database
3. Strengths and Skills	*What can each student contribute with?* • *Harry: Like to code. Always happy. I hate beer but I like to bring drinks to social gatherings. Flexible, motivated.* • *Benjamin: Motivated. Flexible. Beer.* • *Lauge: Good vibes and chill to work with (hopefully). Would like to code and drink on Fridays for a more chill work environment.* • *Jimmy: love to share a beer after a long day of work (Brooklyn IPA) Good, social and talkative boi - retaking algo so I love algorithms and i am very good at them obviously <3<3 Very flexible, can work on sundays. 12 in systematic design* • *Louise: Creative ideas, graphic design, happy mood, openminded* • *Alberte: Flexible, like to code, lots of curiosity, motivated and good vibes ;)* • *Simone: Flexible, prioritize the work as well as the group (socially).* • *Isabella: Structured and organized, very positive mindset and social*

4. Preferred Tasks and Responsibilities	*What would each group member prefer to work with in the project?* *I like working with databases, but I also like doing the frontend because I am a very creative individual I like structure - Jimmy* *Harry: Backend coding sounds fun. I do not prefer anything tho.* *Lauge: Doesn't prefer backend or frontend but wants to code.* *Simone: Do not prefer to work with something specific (flexible)* *Benjamin: Implementing database. Backend. (flexible)* *Louise: wants to code both frontend and backend* *Alberte: to work with both frontend and backend to get some more experience with both* *Isabella: Would like to work with both the backend and frontend.*
5. Team-building and Celebration	*How will we have fun together?* • *Due to code we must get creative: Scribble.io and beers with group, 1v1 with Jimmy in typewriterRace (jeg skriver 140 wpm harry, prøv mig):)* • *Meet-ups and try not to spread coronavirus* • *https://www.youtube.com/user/ImDanishAndImProud (watch together, Fortnite lessons from Jimmy)* • *We could also do a wikirace or jeopardy if the group wants to do some fun BUT ALSO getting some common knowledge along the way* *✨BEER✨* *How do we want to celebrate successes?* • *Getting drunk* • *En lille skarp +3* • *Making social events* • *Eat food together - jimmy laver en mean lasagne føj* • *Mette Blomsterberg drinking game* *Jimmy moved out recently - so he has a lot of space for drinking beer in his new house (He also has a ✨pladeafspiller✨ with a michael jackson vinyl!!!)*
6. Ideal Work Hours	*When will group members be available?* *Available on:*

	• *Monday: Not possible* • *Tuesday (- 1 ppl): 12-16* • *Wednesday (-1 SM): 12-?* • *Thursday: 14-?* • *Friday (- 2 ppl): 8-16* • *Sunday: if necessary*
7. Ways of Working	*How will we carry out the project work? For example, will we work in pairs or smaller teams? How will tasks be chosen (e.g. self-delegated)?* • *Use trello (https://trello.com/) to distribute the tasks between the team.* • *Self-delegated where people are flexible around others priorities* • *Work in pairs (have someone to discuss things with)* • *Sometimes work in pairs, other times work alone.* • *Sometimes coordinate as whole team before working in pairs/alone to get some kind of consensus*
8. Norms & Guidelines	*What code of conduct do we want to have pertaining to: events, decision making, communication, conflict resolution, workload and collaboration. Be specific, for example, what will we do if someone is delayed or unavailable for an event or work session?* • *Inform the group in good time if you cannot show up. Messenger for Developers, Discord for Official matters with SM.* • *Conflicts are resolved between those in conflict. A judge (SM?) can be called if necessary* • *If you don't show up - buy everyone 1 beer - Brooklyn IPA (gælder jimmy)* • *Disagreement → voting* ○ *If two parties are in a disagreement that cant be resolved then the parties each have to do a pitch and then the voting will be done.* • *Decisions are reached in consensus.* • *You are expected to communicate with the group and create a good atmosphere. Team Spirit.* • *Don't be too egotistical - make space for others, it is a group project not solo project*

10.6 Group calender



	Uge 19							**Farvekoder:**	
	Tirsdag kl.11-12	Onsdag kl.12-16	Torsdag kl.10-12	Fredag kl.12-16	(Lørdag)	(Søndag)		Deltager 100%	
Alberte	Valgfag	Joiner efter 14	Deltager 100%	Deltager 100%				Deltager 50%	
Benjamin	Deltager 100%	Deltager 100%	Deltager 100%	Deltager 100%	Deltager ikke	Til rådighed		Deltager ikke	
Harry									
Isabella	Arbejde	Deltager 100%		Deltager 100%	Til rådighed	til rådighed			
Jimmy	Deltager 100%	Deltager 100%	Deltager 100%	måske arbejde - ved ik om jeg har fået jobbet hehe	Lørdag holder jeg fri	Altid klar søndag			
Lauge	Deltager	Deltager 100%		Deltager ikke		Til rådighed			
Louise	Valgfag	Joiner efter 14	Deltager 100%	Deltager 100%	Ikke til rådighed	Til rådighed			
Simone	Deltager	Ikke til rådighed	Deltager	Arbejde	Deltager	Deltager			
	Uge 20							**Ugentlige møder:**	
	Tirsdag kl.11-12	Onsdag kl.12-16	Torsdag kl.10-12	Fredag kl.12-16	(Lørdag)	(Søndag)		Sprint-plannig	Mandag 17/05 - 14.00
Alberte	Valgfag	Joiner efter 14	Deltager 100%	Joiner efter 14				Standup-møder	Onsdag 19/05 - 16.00
Benjamin									
Harry									
Isabella	Arbejde	Deltager 100%		Arbejde	Til rådighed	Til rådighed			
Jimmy	Eksamen i algo	Eksamen i algo	Eksamen i algo	kan fra 12-14 - medmindre jeg har fået jobbet	Lørdag holder jeg fri	Altid klar søndag			
Lauge	Deltager	Deltager	Til rådighed	Deltager ikke, fødselsdag		Til rådighed			
Louise	Valgfag	Joiner efter 14	Deltager 100%	Joiner efter 14	Til rådighed	Til rådighed			
Simone	Deltager	Deltager	Eksamen	Arbejde	Deltager	Deltager			
	Uge 21							**Ugentlige møder:**	
	Tirsdag kl.11-12	Onsdag kl.12-16	Torsdag kl.10-12	Fredag kl.12-16	(Lørdag)	(Søndag)		Standup-møder	Onsdag 19/05 - 16.00
Alberte	Valgfag	Joiner efter 14	Deltager 100%	Joiner efter 14				Sprint-review	Onsdag 26/05 - 16.00
Benjamin									
Harry									
Isabella	Arbejde	Deltager 100%		Arbejde	Til rådighed	Til rådighed			
Jimmy	Deltager 100%	Deltager 100%	Deltager 100%	kan fra 12-16 - medmindre jeg har fået jobbet	Lørdag holder jeg fri	Altid klar søndag			
Lauge	Deltager	Deltager	Til rådighed	Til rådighed	Til rådighed	Til rådighed			
Louise	Valgfag	Joiner efter 14	Deltager 100%	Joiner efter 14	Til rådighed	Til rådighed			

Figure 18: A group calendar used by the developers to organize working days

10.7 Meeting diary