Design decisions and product structure

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

When designing the application we are always doing so with user friendliness in mind. Filtering and sorting of the results is the main purpose and thus needs to be easily understood and used. One example is that when sorting the list, it should default to be sorted in ascending or descending order depending on what sorting variable is chosen. Or when filtering, only available options should be shown, and chosen filters should be persistent while navigating the page. This behaviour is something we have thought about and implemented.

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

We currently only use code documentation as our only means of documentation and have been using it since the start of the project. We had communicated with TA and our teacher as well about this during the first weeks of the course and was informed that it was sufficient. We as a group believe that this is enough and wish to continue in this way to the last week of the course.

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

We believe as a group that we have been very clear about handling this question. Everyone is making documentation for their own part of code before pushing it to Github. If it’s a really hard and difficult part to understand, we are trying to make a live demo where everyone gets the opportunity to ask questions and understand the specific part in the last meeting of the week. However, the answers for this question have been consistent during the whole project time, and therefore we are really happy to have been successful with it.

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

Every week we have a checkup meeting, where we all check if we have any questions and we present what we have done so far. At the end of the week we have time to review each other's code where we also merge our codes. This ensures that we all work together and have a similar quality and the standard is ensured by MVC by ASPNET. This we believe had worked very well and if we do more projects, we might do something similar.

Social Contract and Effort

- **Your social contract, the rules that define how you work together as a team, how it influenced your work, and how it evolved during the project (this means, of course, you should create one in the first week and continuously update it when the need arrives)**

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

The social contract has been working well throughout the project which is a testament to each member's commitment to the project. We have been keeping our rules in mind and working around them to satisfy all members' needs. When a member does not have time to attend the meetings for example we created compromises beforehand to ensure that no bottlenecks would occur because of their absence.

The general working structure provided by the social contract is something that most definitely helped in productivity and mood from the get go and is something that should be emulated during future projects.

To emulate this type of environment a balance between strictness and flexibility should be kept in mind. Since we are only four members in our group the strictness part is not really as demanding since four people can keep each other in check more easily however given a larger group more strict rules would probably be a requirement in order for expectations and communication to not fall apart.

- **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 have used three data variables to keep track of our time we spent or should spend on each user story: Story points, effort estimation and actual effort. Where the story points is the amount of value this story should give to the customer. Effort estimation is the amount of time we expect to spend on this story. Actual effort is the amount of time we actually spent implementing the story. We believe this format has worked well as we have a clear overview of how prioritized a user story should be to implement and how much time we should be expected to spend on it. The time we spent on the course amounts to the actual effort variable, and in some cases it was less than the expected which made us happy, or it was significantly more in some cases. Here we had to put in more effort into those. We believe we have put a fair and reasonable amount of time into the project and we believe we have delivered. In future projects, we should do something similar.