Workshop 2 - Design

Peer Review of Erik Hamrin (eh222ve)

Architecture & design

Good separations between the model, view, controller classes.

The application is nice and simple. I like to be able to click compared to the console I worked with.

The list of members would be better to show the name, id, boats and interaktion choices in a cell. Now it's hard to get a good overlook of the list.

It would also be more efficient of the name of the member actually was clickable to view the member. I believe that our associations to go down in a structure is actually by clicking the header of that object.

What unit is length? That is the most troublesome think in the design. It doesn't say either if it's meter, centimeters or inches.

It would also be a good think to add some validation of the input for names and social security number. The name should only accept letters and the social security number should only accept numeric values.

UML diagrams

The Class diagram shows good use of dependencies, associations and also relations between the classes. The model is not separate from the view, but I guess it's a read-only association. Shouldn't it be a read only dependency instead? I might be wrong.

Class diagrams do not contain attributes or operations, which probably is more of a benefit in this case. I still must say that CRUD-operations would help me understand the class diagram better.

Good naming of the class names with the extra help to show if it's a model, view or controller.

The sequence diagrams are well made and I understand them. It gives me good clues how the code is made.

However, when looking at the code it looks like the c.Program is dependent to the m.Boat due to it's method EditBoat(), but it doesn't show in the diagram?

The HTMLView isn't in the class diagram, which make my wonder if the developer did update the class diagram at the end of the project. I know I didn't. ;)

Code quality

The code is well divided in mvc classes and does only focus to fulfil their duties, except for the v.Member that validate use input. Shouldn't that be a task for the controller?

The code is selfexplanatory due to the good naming but some method does not start with a capital letter, which unfortunately makes the code looks less nice.

Great work with the switch case implementations. It's really nice done with each case refer to operations instead of making a humongous method to handle everything.

As a developer would the diagrams help you?

Yes, it’s clean and gives a good overview between the relations and attributes.

What are the strong points?

The code quality was good and it was easy to read and self-explanatory.

What are the weaknesses?

I must say the list view when running the application and the lack of validation of user input.

Passed the grade 2 criteria?

Yes, of course! The application and diagrams fulfilled all requirements. The weaknesses were not a part of the criteria.