WS2 - Changes

Add instructions with examples (README.txt) -> How to use it ? (It’s my own suggestion)

- Sequence diagrams:

- I believe that the sequence-diagrams can be simplified a bit to make it even easier to understand them.
- The student has implemented all possible inputs and outputs instead of only one for each that was required.
- Nothing wrong with that but from the start I wondered why so much information was included in the diagrams.
- Shorten the sequence-diagrams to a single input and output.
- In the output sequence diagram, (alt (input=3)), I didn't understand why you get MemberList in the end of loop when you get the information of each member!!!
- The sequence diagrams are very good, I can't find any suggestions for improvements.
- But you did not need to display all requirements in the diagrams, only one input requirement and one output requirement, as the workshop 2 requirements states.
- The Input diagram is so big and distorted, that I could not read it even if I zoomed in. Output was a bit better, but still distorted.
- What I find lacking is that none of the input and output diagrams shows who the actors are and who sending what information.
- All I see are lines that go back and forth. Obviously you can guess which part goes where, but they are not clearly shown.

- Class diagram :

- I didn't get why the association (View-Register) has (addMember) as a notation in the class diagram while it shouldn't be specified for one use case.
- The class diagram is, in my opinion, somewhat lacking when it comes to names of associations.
- The 'member' label on the association between View and Member is a bit odd, maybe 'views' would be more clear.
- The same goes for the 'addMember' association between View and Register.
- One thing that seems to be missing from the Class diagram is the FileHandler. The implementation does not represent the model 100%, in my opinion, since the FileHandler is not included in the model but in the implementation.

- Class diagram looks fine, however you may consider having a dotted line from Register to member, as the register is dependent on the member class.
-
- The filemanager class that you have created is not shown on your diagrams, and what I would consider to be a hidden dependancy.

- Code :

- I believe that the "toString"-methods in the model-classes should be moved to the view-classes because no output belongs in the model.

- The layout could be better if you add a margin to the sides.

- Package names should be moved to rectangle in the top of the package.

- For code every thing is ok but the code has not code has no comments. I think it would be a good idea to add some comments to the code. At least comments describing the responsibilities of classes in the file headers.

- I could run the system but I don't know why I couldn't save data.

- When delete or update a boat and the member has no boats, the program still asking for the boat id. Otherwise, all other requirements are working properly in the run.

- What if the register has 2000 member? Should I scroll down and look for an id between all those members? You could use the personal number as an id while its a unique number. You could make a method for the printing command instead of writing (System.out.println) each time.

- You merged the controller and the view in one class I don't know if that is possible but the design will absolutely be more high level if you separate them in different packages.

- I also encountered a few majors bugs in the program. Firstly, I could not save the member lists but got "Could not save data!!" when selecting the save option. I tried this on both Linux and Windows and neither worked. I also got the program to crash by entering letters when it prompted me to input Member ID. It only crashed for option 7 (updating a member) and option 8 (deleting a member).

- I do not quite understand why you have the ViewInterface though, as it does not seem to be used anywhere.

- Another suggestion is to try to break down the View class a bit more, since it's a bit on the big side in number of lines and it seems that some things, like handling the input of member IDs for the commands 4, 5, 6, 7 & 8, is handled in slightly different ways. (Since the program crashes when letters are entered instead of member ID in 7 & 8 but not in 4, 5 & 6.

- I did see that in the view class, in the addmember method, you catch any exception. This should probably be narrowed down to the expected exceptions to prevent unexpected crashes.