Received reviews and my feedback

Review #1 on my file

Comment on Clarity

The language utilized is English and the grammar within the UML diagrams is properly used. The diagram files provided are in common formats and their size is fitting for almost all computer screens nowadays. Both the sequence diagrams and the model diagram are presented with a clear and intuitive design, making it easy to follow each one of them when reading. The nomenclature used for classes is also appropriate, the font is clear and readable and the connections between classes is evident and easy to understand. Likewise, the symbols used in the diagrams comply with the UML standards mentioned by Larman on chapter 16 (Class Diagrams) and chapter 10 (Sequence diagrams), which makes the navigability of the diagrams efficient. On the class diagram there is a lack of multiplicity between the classes, which can be easily remedied since the diagram itself is not too extensive presenting merely seven classes. The class diagram depicts the three general parts of it: the model, the controller and the view. Each one is clearly separated and obvious, as well as the communications between them. It could add value to the diagram if the attributes of each class were also added, which would give an even more clear view of the design. The sequence diagrams use the correct notation for navigability and the classes' interactions are well exemplified with method names and sometimes parameters. The flow of each diagram is easy to follow and the the feedbacks are clearly depicted as well. With regards to the code, the diagramming is flawless, with indentations, spacing, formatting and name rules being clear, correct and easy to follow. However, there is a lack of comments in the classes, which is inadvisable, specially when there is a larger team or peer review involved. The method and classes names make sense and represent real actions/objects, which helps the understanding of the code even more. Overall, the code is simple, clean and easy to read. A minor modification could be to, instead of using "MemberID" in the member's details, to use Member ID as it reads better.
Grade 3.

Comment on Completeness

The program runs with the executable and is easy to understand. However, the fact that the user is required to type "help" every time an action is completed could probably be simplified by always presenting the menu after each action is finished. Requirement 1 is implemented and runs smoothly. However, and despite the personal number format instructions in the creation of a new user, it is still possible to add a member with a personal number that does not correspond to the provided example. Requirement 2 executes and runs detailing all the required information from the member list. The formatting is also well thought and easy to understand, even with the boat information. Perhaps the addition of a "Title" to the printout would make it even more clear since the name list starts right after the "lv" or "lc" command. It is a minor change that could improve usability. Requirements 3,4,5,6,7 and 8 could all benefit from a change of usability. They all require a user's personal number, which requires the call of another function to check it or the input of 10 numbers, when it could

be one letter or one number picked from a list. Requirement 3 is implemented and works well, but the necessity to provide the member's personal number could be troublesome for it requires the user to output the verbose list, write the number down (or copy), and then delete. Perhaps the addition of a member list that is numbered could facilitate the deletion process. Requirement 4 is well implemented and executes its purpose. I believe a confirmation that the member's details were successfully changed could aggregate value to the function. Requirements 6 and 7 are well implemented and could just use a confirmation that the new boat was successfully registered and that the boat's details were changed. All modifications are saved in an external file and persist upon closure of the program and reopening.
Grade 4.

Comment on Content
The system utilizes a strict, clear-cut model and follows it to the brink. There are no efforts for elaborate or even simple printouts in the model classes and there is a clear distinction between the three (controller, model, view) sections of the design. The program is clearly object-oriented, with objects communcating, from my perspective, through associations alone and not keys. The design reflects the domain model directly and classes are correctly represented in the diagrams. the small, but sufficient number of classes demonstrates a high cohesion, with every class' responsibilities being handled well by each of them. The structure is clear and the hierarchy suits the system's purposes well enough, with every class having a handful of obligations and not turning exceedingly long. The connections between classes are well-though and communication between them is efficient and only kept to the necessary. From what I observed there were no hidden dependencies and fields and classes were well encapsulated. Overall, the software runs smoothly, it contains a small number of well-defined and organized classes and I cannot personally see much room for improvement.
Grade 4.

My feedback on review #1:
I agree to all of the suggestions you gave me to make my application better. Even though the focus of the workshop wasn't to create a usable or fancy interface, I see the points you're making as something that would make the application and documentation even better than they already are. I am going to make the suggested changes and rework my class diagram and add some comments to the code. I will also rework my source code to accommodate the suggestions you wrote under completeness, this would make the application more user friendly and is not too hard to add since it is mostly feedback messages to the user that the thing they wanted to do was successful or not.

Overall I think your review was truthful, helpful, thorough and complete. It ticks all the boxes for how a good review should be and I got some good suggestions on how to make my final submission better. Thank you!
Grade 4.

Review #2 on my file

Comment on Clarity

The diagrams were clear to be followed and conform well. The naming of the classes completely follows the rules of thumbs as asked. The associations that have been used in the class diagram made it even easier to understand the relations between the classes. Sequence diagrams were clear and described the functionality of the methods briefly as well.
The codes were easy to follow even though I am not so familiar with C sharp but it was all clear since the name of the classes and the methods were easily understandable. More comments in some methods were required but the comments that already was included could be quite enough to describe them.
Grade 2.

Comment on Completeness

The submission included A UML class diagram as asked and tow sequence diagrams for input and output requirements. The source code was as well included in the submission and it was good structured and easy to follow. There was an executable file but as I mentioned, I am not familiar with C sharp, so it was not easy to run the system in the beginning but after reading the instruction I could get the idea. What to mention is, that the instruction are somehow not clear since the link that was included was not that easy to follow and in the end it was easier to run than what was included in the instruction document.
All the requirements are included in the system and working good but some but I expected getting some warning messages at some point. for example, while creating a member I suggest that the input should not accept integers but strings, and the same for the personal number as I could add a member with a personal number that contains only letters. Moreover, while choosing a boat type, the user gets 4 choices and choose the type by choosing a letter that refer to one type. However, I tried typing another letter that is not exist in the list but still I was able to register a boat with a type "None". I suggest haveing a method to ask the user to choose only one of the 4 options.
Grade 2.

Comment on Content

I believe that there was a strict model and view separation as it supposed to be .
The design itself I can consider it an object oriented as the classes have high cohesion with no big responsibilities, which made the quality of the design good and well structured. Hidden dependencies were avoided in the design. In general, the relation between the classes are really clear and it is like it is shown in the class diagrams so the design artifacts is in sync with code
Grade 3.

My feedback on review #2:

The grading of the clarity is a bit unclear. The only "bad" feedback you provided on the clarity is that there are too few comments in the source code. I would like to know how just the lack of comments in the source code could result in grade 2. I would have wanted a more thorough motivation to why the grade was so low.

On the completeness, I agree that I should have added the link for the executable files in the instruction.md file. However, my other two reviewers has not found it difficult to download an executable file for their OS using my instructions and the link provided. But I understand that this is why the grade for completeness on your behalf is so low.

Also, on the completeness, I did not understand the sentence "What to mention is, that the instruction are somehow not clear since the link that was included was not that easy to follow and in the end it was easier to run than what was included in the instruction document.". What does this mean? What "was easier to run than what was included in the instruction document"?

More on the completeness, I agree that there should be validation for what has to be included in the personal number. Letters should not be acceptable as a personal number, you are right.

Regarding the boat type, the "None" type is there to make it impossible to save a boat without anything in the type field. It sort of works as a fail-safe, but I agree that when the boat type is "None" the user should be prompted to choose one of the 4 options provided instead.

Lastly I would have wanted a more thorough motivation to why the grade was a 3 and not a 4 on the content. I could not see any feedback on the content that would motivate the grade 3 and not a 4.

In summary:
- I would have wanted a motivation to the grading on clarity and content. Like, what is wrong and how can I fix it to get a higher grade from you. Helpful clues.

Since your review fails to motivate the grading I would have to give you:
1. Failed - Most of the review is not correct and fail to motivate the grading.

This does not mean I don't think your review is correct, because I think it is correct. It just means that I would have wanted a motivation on the grading you gave me on clarity and content. The grading on completeness, however, I understand completely and it is motivated.

Thank you for your review!
Grade 1.

Review #3 on my file
(Handed in via mail and not via the csquiz)

Comment on Clarity
The provided diagrams uses a correct language, i can't find any issues related to spelling or grammar.

A correct UML notation is used, and the diagrams are easy to read and understand. Great job making the MVC separation clear on both the class and sequence diagrams.

Classes have been given good names (noun in singular form) that are naturally inspired by the problem domain.

When it comes to the code, almost everything is on point. The formatting and indentation is correct and consequent, good names are given to both classes, methods and variables. I could however make use of some more comments in some places, as well as some refactoring of some of the quite large methods into smaller units to avoid loads of nested statements (for example in the Boat.cs controller).

Overall: Great job!
Grade 3.

Comment on Completeness
The handed in zip-file contains the required UML-diagrams (one class diagram and two sequence diagrams). The source code is included in the submission.

The zip file did not contain an executable, but there was good instructions on how to download an executable for my OS from a GH repo. The downloaded executable did run as expected on OS X 10.11.1 - all of the requirements are working in the running system.

One note to this is that since the executable for my OS is downloaded from a GH repo, the submission is not anonymous.
Grade 4.

Comment on Content
GOOD:

- There is a strict model/view (in this case even MVC) separation. All of the printing is done from the view, and the model strictly handles the data.

- Both the design and the implementation is object orientated. I cant seem to find any hidden dependencies, and the classes have low coupling.

- Given the problem domain, a natural design is used where it seems like the domain model have inspired the design

- Overall, great job both with design and implementation

THOUGHTS ON IMPROVEMENT:

- The design artifacts are in sync with the implemented code, but one could argue that the class diagram is a bit to simplified and that some key attributes/operations could be added.

- Some of the classes are in my opinon to large and have to much responsibilities. This is true for the Database.cs (model), as well as the Console.cs (view). I think that these classes could be split up into something like:

MODEL:
Database.cs (Handle the more generic stuff like reading/writing/parsing JSON)
Member.cs (Inherit/use the Database.cs and be in charge of all the member related business logic: DeleteMember(), GetMember() etc..)
Boat.cs (Inherit/use the Database.cs and be in charge of all the boat related business logic: DeleteBoat(), ChangeBoatInfo() etc..)

VIEW:
Console.cs (Generic stuff like WriteException(), DisplayInstructions(), GetEvent() and so on)
Member.cs (Handle member related view stuff)
Boat.cs (Handle boat related view stuff)

Discussed in chapter 16.9 page 232 Applying UML and Patterns second edition under the heading ”High Cohesion”
Grade 3.

My feedback on review #3:

(Handed over via mail and not via the csquiz)

I agree that the source code should have more comments and that the larger methods in Boat.cs controller should be refactored into smaller methods to avoid the nested statements. The omitted comments are more a reaction to the parallell course 1DV610, and I have tried making the code self explanatory to avoid adding all these comments that no one is going to read anyway. But I will go through my source code and see where comments could be added to make it more clear.

I also agree that the class diagram is a bit simplified. I mostly omitted the attributes and operations for the classes because I was unsure on what to add and what to omit. This is something that I will consider fixing, since it would make the class diagram more transparent and connected to the source code.

The refactoring of the model class Database and the view class Console is also something I agree would make the code easier to follow and to manage. I will consider implementing this.

Thank you for the excellent review!

Grade 4. The review is truthful, thorough and motivates the grading in a good way. Also the review was helpful and I learned how to write better artifacts(strategy/plan/cases/report) from it.