Peer review for hg222dv
Workshop 2

Architecture

- **Is there a model view separation?**
  - Yes. The *Model* contains the functionality that the application should process. The *Controller* contains the functionality to handle and respond to the events in the *Model*. And then the information is presented in the *View* as it should.

- **Is the model coupled to the user interface?**
  - Yes.

- **Is the model specialized for a certain kind of UI (for example returning formated strings to be printed)**
  - Yes.

- **Are there domain rules in the UI?**
  - No. The code only contains information that should be seen and controlled by the user.

- **Is the requirement of a unique member id correctly done?**
  - Yes. Each member has a unique id.

Code quality

- **Code Standards**
  - Code is easy to read and understand. Comments are good and explains what each method do.

- **Naming**
  - Naming is good in general. But there are small parts were only a letter is used of shortened words, which can make it harder to understand.
    (Example 1: ShowMessage(string.Format("{0}.{1}", i,b.ToString()));)
    (Example 2: string rl = Console.ReadLine();)

    What is *"b"*? or *"i"*?. And what's *"rl"*?
    It would be much easier to understand if it was explained in words.

- **Duplication**
  - I could not find any unnecessary duplication in the code.

- **Dead Code**
  - No.

What is the quality of the design? Is it Object Oriented?

The code is well written and it's easy to read and understand. All the methods are commented and explains what it's goal is, which helps alot.
The code is seperated as it should and it's well organized.

However, while testing the application I immediately found a pretty serious bug which ruins one of the most important requirement. It seems like registering a new member does not work.
I filled in all the required fields (name, last name, personal number etc) and then went back to the menu to check the member list. I could not find any new member in the list. Later, I found out that I had not typed in the personal number correctly.
I know it has to do with user friendliness, but without any feedback (error message) I thought it was a bug, not my fault.
There is also a bug when trying to edit a boat on a member that does not have any boat(s). The application crashes with an unhandled exception.

As a developer would the diagrams help you and why/why not?

The class diagrams does not help me that much. Sure, it gives a good overview of what each class contains. It might come in handy if a code is badly written. Otherwise I see no point of it really.
The sequence diagram on the other hand is really helpful.
When looking thru the code you can understand what it does and what happens. The problem is that it's much harder to see the whole picture of how the application work. Because when looking at code in classes you only see smaller parts of the whole system.
The sequence diagram helps me understand how the whole application works and how all the parts of the code working together.

What are the strong points of the design/implementation, what do you think is really good and why?

Like I have already mentioned before, the code is organized and well commented which makes it easier to understand. The sequence diagram is very good and clean: I have no issues understanding it really,

What are the weaknesses of the design/implementation, what do you think should be changed and why?

I can't really find any weakness. Therefore I can't say anything about it.

Do you think the design/implementation has passed the grade 2 criteria?

Absolutely!