Test the runnable version of the application in a realistic way. Note any problems/bugs.
When register boat and presses a number that is not on the list causes a crash.
if new member is created and the program crashes then that member wont save unless the program is quited before.
When editing a boat and presses a number that is not on the list causes a crash.

Try to compile/use the source code using the instructions provided. Can you get it up and running? Is anything problematic? Are there steps missing or assumptions made?
It works fine.

Does the implementation and diagrams conform (do they show the same thing)? Are there any missing relations? Relations in the wrong direction? Wrong relations? Correct UML notation?

The UML and code is showing the same thing, but the view should not have a relation with the models like view to m.boat and view to model [1, chapter 16.2 - 16.5].

Is the Architecture ok?

- model and view is not seperated.
- Model is coupled to the user interface.
- No the model is not specialized for any kind of UI.
- No there is no domain rules in the UI

Is the requirement of a unique member id correctly done?
Yes it is implemented correctly.

What is the quality of the implementation/source code?

- They code standard is good [2].
- The naming could be better, because the controller has the same name as the model so it could get confusing when reading the code [2].
- There is no duplication of code.
- There is no dead code.

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

- Yes they are connected using associations [1, chapter 17.2]
- It looks like they have used GRASP correctly [1, chapter 17.9].
- The code have high cohesion but perhaps some is to connected to other like view is connected to the models.
- Classes are more coupled than they need to be.
- No use of static/global variables
- As far as we can see there are no hidden dependencies
- The code is encpasulated
- The domain model is not detailed enough to have so much inspiration for the code.
- No primitive data types

As a developer would the diagrams help you and why/why not?
Yes the diagrams would be helpful because they are detailed, informative and relatively easy to understand. However some of the relations in the class diagram are missleading because they should not be there [1, chapter 16].

What are the strong points of the design/implementation, what do you think is really good and why?
Design and structure of the code was really good implemented. The sequence diagrams was also good and informative [2] [1, chapter 10].

What are the weaknesses of the design/implementation, what do you think should be changed and why?
The weakest point in the code is that the view class has a direct relation with the models [1, chapter 13.7].

Do you think the design/implementation has passed the grade 2 criteria?
Although there were some minor issues and flaws in the code and design we still think that this project should pass grade 2.

References:

1. Larman C., Applying UML and Patterns 3rd Ed, 2005, ISBN: 0131489062
2. Oracle, Java Coding Conventions ,2015-10-09, http://www.oracle.com/technetwork/java/codeconventions-150003.pdf