Can you get it up and running? Is anything problematic? Are there steps missing or assumptions made?

There was no problem to get it up and running by using the instructions. There are some bugs.

Bugs

- In the main menu, if a number is not entered the program will crash. Example letter a.
- Program will crash if the user try to add a boat to a member id that do not exist.
- When updating a member, personal number is not updated. Sometimes the personal number will be removed.
- Members can get the same member id.

Class diagram and sequence diagram

The class diagram is easy to follow and the main components are there. Some parts can be a bit misleading. Memberregistry in the code instantiate a boat class in addBoatToMember, but in the class diagram there is no association between memberegistry and Boat class.

The sequence diagram are also easy to follow but some details are missing. A member class should be added. Since the the member class is left out some parts gets a bit confusing. In the diagram Console has a method addMember to call MemberRegistry but that is not the case in the real code. In the code MemberRegistry has a method addMemmber to call member class.

The diagrams should be split up, one for add a member and one for list members. Larman states on page 279 a sequence diagram shows one scenario of a use case.

Quality of source code

is there a model view separation?
Yes the separation is achieved by using a mvc pattern. View and model do not have any knowledge about each other. According to Larman on page 329, model objects should not have direct knowledge about the view

Code standards
Good naming for variables and methods. When reading the methods they are self explaining and I can understand what they do without reading the code. Duplication and dead code is avoided.

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

Objects are connected using associations and not with keys/ids.
Objects are connected by using associations. The association between MemberRegistry and Boat can be avoided. This makes the code harder to reuse. Larman states on page 444. A class with high coupling relies on other classes. Such classes may be undesirable.

Is GRASP used correctly?
Yes I believe GRASP is used correctly in this project, except with the example above making it not so low coupling.

Classes have high cohesion and are not too large or have too much responsibility.
Larman states on page 461, a class with low cohesion does many unrelated things. I do not believe that this project suffer from low cohesion. Classes do what they are supposed to do and stick with that.

Classes have low coupling and are not too connected to other entities.
Yes, except for the problem already mentioned with MemberRegistry.

Avoid the use of static variables or operations as well as global variables.
Static variables are used in some classes.

Avoid hidden dependencies.
I could not find any.

Information should be encapsulated.
Yes by using private fields and getters and setter. I'm not sure about the static variables.

As a developer would the diagrams help you and why/why not?
Yes it would help me to get good overview of the project but can be a bit confusing because important parts are left out.

What are the strong points of the design/implementation, what do you think is really good and why?
Very good naming for variables and methods, easy to understand them without reading the code. Model and view separation achieved. Classes are not to big and do what they are supposed to do. Follow the rules to achieve GRASP.

What are the weaknesses of the design/implementation, what do you think should be changed and why? I believe that some work need to be done for the diagrams, right now they are missing information and are bit confusing. Fix bugs. Avoid unnecessary association.

Do you think the design/implementation has passed the grade 2 criteria?
At moment no. Bugs need to be fixed, especially updating the member and id bug. They are part of the requirements for this assignment. The diagrams also need to be fixed, they don't feel complete. As mentioned before remove the unescarry association. After this I believe the assignment should meet the requirement to pass grade two.

Reference Section

1. Larman C., Applying UML and Patterns 3rd Ed, 2005, ISBN: 9780131489066 (https://aanimesh.files.wordpress.com/2013/09/applying-uml-and-patterns-3rd.pdf)