The following requirements are to be met:

1. ~~Create a new member with a name, personal number, a unique member id should be created and assigned to the new member.~~
2. ~~List all members in two different ways:~~
   1. ~~“Compact List”; name, member id and number of boats~~
   2. ~~“Verbose List”; name, personal number, member id and boats with boat information~~
3. ~~Delete a member~~
4. ~~Change a member’s information~~
5. ~~Look at a specific member’s information~~
6. ~~Register a new boat for a member the boat should have a type (Sailboat, Motorsailer, kayak/Canoe, Other) and a length.~~
7. ~~Deleta a boat~~
8. ~~Change a boat’s information~~
9. Persistence (the registry should be saved and loaded for example from a text file.)
10. Strict Model-View separation (The model should not depend on the view or user interface in any way (direct or indirect) the user interface (view) should not implement domain functionality)
11. Good quality of code (for example naming, standards, duplication)
12. An object oriented design and implementation. This includes but is not limited to:
    1. Objects are connected using associations and not with keys/ids.
    2. Classes have high cohesion and are not too large or have too much responsibility.
    3. Classes have low coupling and are not too connected to other entities.
    4. Avoid the use of static variables or operations as well as global variables.
    5. Avoid hidden dependencies.
    6. Informations should be encapsulated.
    7. Use a natural design, the domain model should inspire the design.
13. Simple error handling. The application should not crash but it does not need to be user friendly.
14. Participate in the peer review process

**You should have the following in your portflio and all parts should match.**

* A well tested runable version of the application. For example an .exe or .java files, link to website etc. If it is not easy to run the application you must include instructions on how to run it.
* The source code of the application, with possible instructions on how to compile it, external dependencies etc.
* A class diagram for the entire application, focus on relationships of the classes and important details (do not add every single attribute or operation)
* Sequence diagrams that covers one input requirement (create member, register boat or Change Information) and one output requirement (List members or Look at member info)

Some tips on tools to create diagrams are yuml.me and websequencediagrams.com that seem to cover the needs we have and are quite agile. Some versions of Visual Studio has a number of tools for diagrams.