First Part:

Unfortunately the source code is not included, so I cannot see how the names / comments and so on is in the source code. So the comments here are only regarding the diagrams.

I don't think the class diagram is using correct relation notation. I.e. there is not a single association line, which means no class variables. Otherwise, I think the class names have good naming standard, except a missed character in "CosolePresentation".

Second Part:

As mentioned there was no source code included in this submission.
However the .exe is working, so I could do some manual testing at least.
At first glimse, it looks nice, loads nicely from .json file and shows the members and boats alright. Some part of the crud is working, but maybe the biggest missing part is that neither Boat nor Member can be deleted. Also I think it's not good to let the user type in an userID. The system isn't handling if you type in the same ID for two members, both is added, but you can just handle the first one added. Typing a string character in the personalID or memberID is giving an Exception. It would have been nice if that was handled to give a proper error message that the id should only be digits instead. I think the boat type should have been an Enumeration to make a more stable, and later searchable system.

If trying to edit a boat when there is none, first gives you instruction to write boat info, and then gives you an exception. Lastly, to edit a boat I need a boat name, but I cannot find what I should write to be able to edit the boat.
I wanted to edit:
Boat: Boat 1: Boat Type: Motorflyer, Boat Length: 123
I tried using ”1”, ”Boat 1”, ”Motorflyer”. All gave me Exceptions.

Third Part:

This part is very hard to peer without the source code. But I studied the diagrams and could see a couple of things that violate the MV separation. Firstly, the sequence diagram of add boat shows that :view::ConsoleOptions send a message to the controller which is strictly forbidden in an MVC architecture. Secondly the :model::Member returns a returnAsJson() or is that a message since the parantesis?, anyhow if it is a return I assume it is a formatted string that will fit the JsonParser, and I think if you have the Json parsing in the controller you should format the string there as well. If it on the other hand actually is a message, it violates the rule that model should not depend on controller or view.

Secretary in the sequence diagram is a bit unclear, but I guess that is an Actor. If so you should probably not specify the return as a String. A little notation is that return from a message should be showed with a dashed line and thin arrow. Larman shows this in some diagrams[1,p229].
Larman C., Applying UML and Patterns 3rd Ed, 2005, ISBN: 0131489062