Assessed Exercise 2

This is a follow up to Assessed Exercise using the same teams. It involves design software using UML class and sequence diagrams, and a small amount of implementation.

You should implement the following small part of the PTT specification used in AE1

Before the start of each term or semester, the class directors produce a list of teaching requirements which we must try and fill. Our administrator will then attempt to find suitable staff and organise training for them.

You should use an Agile approach to designing your software and Trello to communicate between team members.

You should find appropriate User Stories, they don't have to be the same as your stories from AE1. Note, only include stories related to the limited part of the software described above.

You should use sequence diagrams for each User Story to design your classes and methods, producing a Class Structure Diagram. Each class must be owned by one team member who is responsible for implementing it. Your stories will involve several different objects from different classes. Each class owner should implement the part of the story related to their class. This is how stories are split into tasks.

You should use a simple implementation and develop a standalone app that does not require a database. All permanent information should be stored in a single file and this file should be read in when the program starts and the information written to the file when the program finishes. Your program will have an internal representation of lists of class, teachers and so on. They will be similar to the LoP class in the lab work, and so you have had some experience of implementing them. Don't try and make them too elaborate, you don't have much time! These lists are empty when the program starts and then filled with data by reading from the file. The updated version of the data is then preserved by writing to the file before the program finishes. You should design this part of the code so that a better implementation, using a database, could be provided in release 2 with minimum changes to the code base. You should not try and write a database version for this exercise!

You will gain marks for using good design principles and using patterns, when appropriate. Marks will also be awarded for working functionality.

Note that a well designed program where some things don't quite work can score just as many marks as a program that does everything but without good design principles.

Submission details

Every group needs to submit:

1- One team report (preferably in pdf format) , which should contain the following:
   - Your team name and a list of each team member's contribution, including classes they were responsible for.
   - A list of User Stories, as in AE1.
   - Your class structure diagram.
   - The sequence diagrams for each of the user stories.
   - Screenshots showing your code running.
   - A retrospective on what went well and what was challenging while completing this assignment.

2- A zip file including your code

   I will run each of your programs to see how well they work. Please comment your code so that I can understand what you were intending so that you can get marks if something doesn't quite work.

Other than group reports, every student needs to submit an individual report on your team members as in AE1.

Please submit your files through Moodle by **4:40pm on Friday 11 March 2022**