SOFE 4790U: Distributed Systems (Fall 2021)

Instructor: Dr. Q. Mahmoud/H. Singh

Individual Programming Assignment #2

[Monday, Nov. 1 -- Wednesday, Nov. 10]

The objective of this individual programming assignment is to get a flavour of the effort involved in designing and developing distributed applications using Java RMI. You will practice designing and developing an innovative distributed application of your choice using Java RMI. **The application idea for this assignment can be based on or extend what you have done in Assignment#1**.

The Task

Design and develop a novel, easy to use, and useful distributed object-based application of your choice. It must accomplish something useful and has some novel features. Your Java RMI remote interface must provide 5 unique remote methods (services) to clients.

Guidelines

- Your application should continue to handle clients' requests until it is manually terminated, or you have a UI for run/shutdown.
- Your application must be fully functional in order to get reasonable marks on the other assessment items (see grading rubrics on next page).

Important Notes

- **Deadline**: Assignment#2 must be submitted **by 11:59pm (night) on Wednesday, November 10**. *No extensions, so plan accordingly*.
- Your solution must be designed and developed by yourself (your own work).
- While students are encouraged to discuss the assignment and general ideas for solutions, each student must design and develop his/her own solution and code. No code sharing is allowed, and no two or more students can have the same application. **JPlag will be used for detecting code similarity**.
- The assignment will be assessed based on the grading rubrics provided on page 2 of this document.

Submission Guidelines (note the 2-step submission)

1) **Source & class files, and a README file**: Submit your assignment solution source code (*.java) and bytecode (*.class) files, along with a README file on Github **by 11:59pm (night) on Wednesday, November 10** as per the following instructions:
    a. Go to the following link for Assignment2: https://classroom.github.com/a/wXJtIcpv
    b. Your submission must include a README file with a brief description (one paragraph) of the application, and instructions on how to run your application.
    c. If your application requires any resource files, make sure you include them in your submission on Github.

2) **Report**: Submit your assignment report through Canvas by **11:59pm (night) on Wednesday, November 10** (look under Home -> Assignments –> Assignment #2 Submission). Your report must be in PDF or Word and must include – please use the assignment report template from assignment#1:
    a. One full-page (approx. 500 words) detailing your application idea, novel feature(s), challenges and solutions. You may include one clear diagram but no screenshots of the application.
    b. A description of the tests you have run to demonstrate the functionality of your application. You must describe the actions with screenshots, and clearly demonstrate this was done by you on your own laptop (e.g. show command-line prompt with your account name).

Grading Rubrics

Item (%)	**Excellent (full mark)**	**Good (75%)**	**Satisfactory (50%)**	**Unsatisfactory (25%)**	**Zero (zero)**
Report (20)	Clearly documented and well organized with novel features, sample runs with description & screen shots, challenges and solutions.	Readable but not well organized or missing parts.	Documentation is minimal, but clear sample run.	Documentation is minimal, with no sample run.	Non-existent.
Usefulness and usability (20)	Useful and intuitive to use.	Nothing special.	Requires a manual to use.	Not useful or usable.	Non-existent.
Novel features (20)	Creative and offers novel functionalities.	One novel feature.	Nothing special.	Cannot be considered as novel features.	Non-existent.
Functionality (20)	Fully functional with no errors or warning.	Functional but nothing special and sometimes no response.	Basic functionality beyond code covered in class.	Error messages during run.	Does not compile or run.
Source code (20)	Follows coding standards (name, date, title, meaningful variable names, whitespaces, etc.) and code is fully documented.	Readable source code. Does not follow coding standards.	Spaghetti code.	Code provided is incomplete or does not make sense.	No source code provided or the link to the source code is not accessible.