CSS 543
Program 4: Multitier Server Application

Professor: Munehiro Fukuda

Due date: see the syllabus

1. Purpose

This assignment designs and implements a multitier server application using Servlets as well as RMI or Java CORBA. This is an open project where you will choose an interesting application from your career background or interests, design the software architecture best fitted to your application, and implement using whatever you have studied in this course as well as whatever you are specialized in. Only three restrictions in this project are: (1) use Servlets, (2) use RMI or Java CORBA, and (3) design a multitier (3 or more tiers) server system. The goal of this project is to demonstrate your understanding of multitier server systems.

2. Application

Choose an interesting application fitted to multitier client-server architecture. Some examples are but not limited to:

(1) A doodle-like voting website
(2) A ticket/room reservation system
(3) A shared bulletin board
(4) A scientific-job submission system (like Qsub, MapReduce, Condor, etc.)
(5) A filter system that priorities or choose what a database returned, based on a client's preference.

Note that your choice of applications is part of the project grade.

3. Specification of Your System

Your system must follow the specifications below:

(1) **No need to develop a client system:** Your client may be an Internet browser such as IE, firefox, or safari. However, you may still implement your own client system, which gives extra credits to you. Your client program must facilitate GUI (rather than depend on text-based user inputs).
(2) **Servlets running on a web server:** Your middle tier must be a web server that runs Servlets. As far as you use Servlets, you may design fancy JSP code or applets, either displaying GUI input menus on a client-side browser. Your Servlets may be either generic or HTTP-based.
(3) **Third tier server:** The middle tier, (i.e., your Servlet) must contact the third tier, which you can implement by yourself, use a Microsoft-dependent software tool (such as .NET), or even use an existing server (such as www.youtube.com).
(4) **Communication link between a client and the middle tier (namely the front-end link) as well as between the middle and the third tier (namely the backend link):** You must use RMI or CORBA for either the front-end or the backend links. Needless to say, the very first contact from a client user to your Servlet is achieved with an HTTP Get/Post message, however you may use any protocols such as RMI, CORBA, JDBC, or even TCP to complete this client's transaction. For instance, an applet downloaded from your web server can establish a TCP connection to the server so as to send the client's additional information, in which case you must use RMI or CORBA for the backend links.
(5) **The size of your program:** Your code must be equal to or larger than 400 lines including at most 20 percent comment lines, (i.e., 320 lines excluding comments).
(6) **Each tier running on a different machine:** Although you may develop and test all the tiers of your system on the same computer, (i.e., your notebook or home computer), the final deliverable

must be capable of running each tier on a different machine. As far as you follow the above guidelines (1) ~ (5), you may use any platforms.

4. Teamwork

You may work on this final project independently or in teamwork with another student, (i.e., a group of up to two students including you). If you work in a team, you must comply with the following guidelines:

(1) **The size of your program:** The entire code must be equal to or larger than 800 lines including at most 20 percent comment lines, (i.e., 640 lines excluding comments).
(2) **The work assignment table:** All team members may turn in a joint report. (I am reluctant to repeat reading the same reports.) However, the report must include the work assignment table that details who worked on which portion of code, functionality, and/or tiers.
(3) **Confidential collegial evaluation:** Please evaluate your partner's contribution to the project, envelop it, and turn in this evaluation with your joint report.

5. Presentation

Plan a presentation for your project work in week 10. Your presentation should include:

(1) Overview of your application: explains what a user has to give inputs to your system and what results s/he will obtain.
(2) Architectural design of your multitier server: gives the details of your server in terms of what each tier does and how each tier communicates with the others (using RMI, CORBA, etc.).
(3) Up-to-date project status: summarizes the latest status of your project or even the final outcome if you are done.
(4) Demonstration: Please demonstrate your program even if you are still in the middle of your system development.

Please send me the following information of your project within a week after this assignment was made available:

(1) Your project title
(2) Your preference of presentation date: 1st lecture day or 2nd lecture day in week 10. First come and first assigned.
(3) Your project partner if this is a joint project. In that case, choose the team leader who should inform me of this project information.

6. What to Turn in

This programming assignment is due at the beginning of class on the due date. Please turn in the following materials in a hard copy. No email submission is accepted.

Criteria	**Grade**
Presentation of your project that should include an overview of your application, an architectural design of your multitier server, up-to-date project status, and demonstration. 5pts: a good presentation with a demo of your program. 4pts: a good presentation without a demo.	5pts
Selection of your application that should be well fitted to a multitier client-server system with multithreading.	5pts
Documentation of your project that details your presentation in up to 5 pages (firm). Ambiguous or insufficient documentation receives 4pts.	5pts
Source code that adheres good modularization, coding style, and an appropriate amount of commends. 5pts: well-organized and correct code that satisfies all the design guidelines 3-(1) ~ 3-(6) as well as 4-(1) and 4-(2) receives 5pts 4pts: messy yet working code or code with minor errors	5pts

3pts: code that does not satisfy at most one of the guidelines 2pts: code with major bugs, incomplete code, or code that does not satisfy two or more of the guidelines.	
Execution output that verifies the correctness of all your implementation as well as covers many test cases. 5pts: a correct output with a good GUI 4pts: a correct output with minor errors or with a poor GUI (such as a text-based output) 3pts: an insufficient output that does not perfectly verify the correctness of your program 2pts: a quite poor output that cannot verify the correctness of your program at all	5pts
Discussions about the performance, usability, limitation, and possible functional improvement in up to 5 pages. Ambiguous or insufficient discussions receive 4pts.	5pts
Lab Sessions 7 through 9 counts 1pt for each. If you have not yet turned in a hard copy of your source code and output or missed any session(s), please turn in together with program 4.	3pts
Teamwork is evaluated if your project is achieved in collaboration with another student. No work assignment table receives -2pts. A poor collegial evaluation receives -2pts.	(-4pts)
Total Note that program 4 takes 25% of your final grade.	33pts