Homework 3, RMI And Databases

Network Programming, ID1212

1 Goal

- You can develop a distributed application which uses remote method invocation for inter-process communication.
- You can develop a distributed application which persists data in a database.

2 Grading

The grading is as follows:

Not accepted Your work has not been accepted, and you have no score.

0 points Your work has been accepted.

1 point Your work has been accepted before or on due date.

2 points Your work has been accepted before or on due date. Also, it has an acceptable layered architecture and is well designed. This means it follows the guidelines of the lecture on architecture, and of the programming examples on the course web.

3 Auto-Generated Code and Copying

You must be able to explain and motivate every single part of your code. You are *not* allowed to copy entire files or classes from the example programs on the course web, even if you understand it and/or change it. However, you are allowed to write code which is very similar to the example programs on the course web. You are also allowed to use GUI builders and other tools that generate code.

4 Task, A File Catalog

Develop a client-server distributed application which allows storing, retrieving and removing files to/from a file catalog. A client is controlled by a user and provides a user interface. The server allows clients to perform the following actions:

- A user can register at the catalog, unregister, login and logout. To be allowed to upload/download files, a user must be registered and logged in.
- A user specifies username and password when registering at the file catalog server. On registration, the server verifies that the username is unique. If it is not, the user is asked to provide another username.
- When logging in to the server, a user provides username and password. The server verifies the specified username and password.
- A user can upload a local file to the catalog and download files from the catalog to the local file system. A file at the catalog is identified by its name, thus there can not be two files with the same name. A file has the following attributes: name; size; owner; public/private access permission that indicates whether it's a public or private file; write and read permissions if the file is public.
- A private file can be retrieved, deleted or updated only by its owner. A public file can be accessed by any user registered at the file catalog. The write and read permissions for a public file indicates whether the file is read-only for other users than the owner or if it can be modified, i.e. deleted or updated, by any user. Owners have all permissions for their files.
- Users can inspect what files are available in the file catalog, i.e. list the files in the catalog and their attributes. Note that if a file is marked as private, it can be listed only by its owner.
- A user can request to be informed when other users access one of its public files. The user tells the server for which of its files it wants to be informed. When that public file has been read or updated by another user, the server tells the owner who performed the action, and what action was taken.

The files in the catalog are stored on the server's file system under a specified directory.

Requirements on Your Program

All of the following requirements must be met in order for your solution to be accepted.

- Client and server communicate only using remote method invocation, except for file transfer, which is performed using TCP.
- Only the server is allowed to register itself in an RMI registry. When the server makes a call to a client, it must be via a remote reference passed to the server by the client.
- The server uses a database to keep records on each user (user name and password) and on each file in the catalog (name, size, owner, public/private access permission, write/read permissions).

- The clients do not store any data. All data entered by a user is sent to the server for processing, and all data displayed to a user is received from the server. Here, *Data*, means only username, password and file information (name, size, owner, public/private access permission, write/read permissions, and content). Other parts of the user interface, like instructions to the user, are best generated by the client.
- The user interface must be informative. The current state of the program must be clear to the user, and the user must understand what to do next.

What is NOT Required of Your Program

Below is an explanation of things that do not affect your score.

- You are free to decide how files are stored at the server. A suggestion is to store all files in a directory dedicated to that purpose.
- Minor changes or misunderstandings of the functionality are allowed, as long as your program does not become notably simpler than a program implementing the intended functionality.
- Your are not required to create a graphical user interface. A command line UI is sufficient.
- You are not required to make the client multithreaded.