comp1206 programming ii - assignment two

(Submission Deadline: 4pm, Friday 13th May 2016)

Please direct any queries regarding these instructions to Dr Julian Rathke (jr2@ecs).

You can expect to receive your mark by: Monday 6th June 2016.

assignment instructions: a lightweight auction system.

A small community group wishes to raise some funds for a local charity and in order to do so they have decided to hold a simple auction. Members of the community can donate goods for sale and other people can bid on these. The group has decided to run the auction over a few days with the aid software but unfortunately the group aren't particularly skilled with technology. In particular, the group are not running a web server or database however the computer in which the Server for the auction will run does have a known static IP address. All handling of exchange of goods and payment will be done outside of the software system. In this sense, the aim of the system is to match up a list of users with items won after a period of a few days.

Your task is to write the software to manage the auction. You will write a Java application for the Server that receives requests from a number of Client applications and processes them appropriately. The Server is responsible for managing the auction. You will also write a Java application for the Client that allows users to participate in the auction. More detailed specifications follow below.

Part One (Users)

Write a class User to represent users of the system. Users must have a given name, a family name and a unique user id amongst all other users stored in the system. Users will initially register with a password of their choosing.

Part Two (Items)

Write a class Item that represents an item for sale in the auction. It should contain a short title, a full description of the item, a category keyword for the item, the user id of the vendor, a start time for the bidding on this item, a close time for the bidding, a reserve price for the item and a list of current bids. You should also identify each item with a unique id code.

The keywords for the categories may be drawn from a pre-defined list of your choosing.

Part Three (Communication Layer)

Write a class Comms that handles communication between the Server and Client applications. Provide a 'sendMessage' method (or methods) that allows each client to send a message object to the server and the server to send a message to each client. The Server and Client applications will also need to check for incoming messages by calling a 'receiveMessage' method (or methods) of the Comms class. The types of these may vary depending on the types of message being sent/received.

For the purposes of this assignment, you may assume that Clients and Servers run on the same machine and communication may be achieved by making use of the local filesystem. For example, messages sent by the server can be stored in 'mailbox' style file(s) dedicated to a particular user.

If you prefer you may use Socket communication in the Comms class instead of files. In either case, the Server and Client applications should be oblivious to the actual communication mechanism used. That is, all I/O operations must reside in the Comms class. Access to them is only allowed via your send/receive methods above.

For this part you may find it useful to create a Message class with possible subclasses for different types of message.

Part Four (Client Application)

Write a Java application for the Client that initially creates a GUI inviting the user to login to the system (possibly by creating a new registration). Once logged in, the application must allow the user to view all

- Items for sale
- Items from a particular seller
- Items of a particular category
- A specific Item given by its unique id code
- All, possibly sold, items (created after a given date/time)

The application must also allow the user to be able to submit a new item for sale, make a bid on any open item, see their own bids, see the status of their own items for sale, and receive and display notifications upon winning items.

Part Five (Server Application)

Write a Java application for the Server that repeatedly checks for messages from Clients and processes them appropriately by validating and authenticating log in and new registration requests, creating new items for sale, responding to view requests, and updating and validating bids on existing items. The Server must also manage the closure of bidding when items have reached their specified closing time and the highest bid has met the reserve price. The Server must notify the winner of the item.

The data representing the items for sale and current bids in the system must be made persistent in case the Server application needs to be restarted at some point during the auction period.

Finally, provide the Server with a GUI consisting of a button that, when pressed, produces a report consisting of the users' full names listed against item titles that they have won, along with a console window that displays a log of the Server's activity.

Part Six (Persistence Layer)

Provide a service class DataPersistence that provides functionality to ensure that all data stored in the system is stored on disk in case the Server application needs to be restarted at some point during the auction period. This should include user data, item data, bids and activity log files.

Modify your Client and Server classes to make use of your DataPersistence class.

If you have made it this far and completed everything well, then you could expect a decent mark in the 70-80% range. To

obtain higher marks than this, read on: Part Seven

Extend your system with extra functionality for further marks: possibilities include

- Use Java Sockets in the Communication Layer to allow Clients to interact with the Server across a network.
- Improve security of the system by encrypting all stored data and transmitted messages.
- Allow images to be included in the item descriptions
- Implement a star-rating system for users with separate ratings for buying and selling. Buyers can rate sellers and vice-versa. Implement ways of penalising a user by reducing their star-rating for engaging in activity that is detrimental to the success of the auction site. For example, withdrawing items for sale from an auction before the closing time when there are bids higher than the reserve price already made.
- Implement a search facility to search for (near) keywords in an item's description.
- **YOUR OWN IDEA HERE** - feel free to add your own extensions to your solution. Marks will be given in accordance with

how difficult we perceive the extension to be to implement.

These extensions are worth up to a maximum total of 3 marks.

Good Practice

Make sure that your code is well organised, concise and well commented. I will award a half mark for well presented code that compiles as submitted.

mark scheme

In total there are 17.5 marks available, which will contribute 35% towards your overall course mark. The breakdown of available marks is as follows:

- Good practice: 0.5 Marks
- Parts One-Three: 4 Marks
- Part Four: 4 Marks
- Part Five: 4 Marks
- Part Six: 2 Marks
- Part Seven: 3 Marks

additional information

Submission Information:

Please place of all of your source files in to a single directory and make a zip archive of it.

- Zip or tar files only. No RAR files please.
- No package declarations in your code please.
- Only .java files, no .class files please.

You should submit your zipped directory using the automated ECS hand-in facilities found at:https://handin.ecs.soton.ac.uk

Originality of work:

Please be aware of and adhere to the University regulations on collusion and plagiarism, as outlined in the University Calendar on Academic Integrity.http://www.calendar.soton.ac.uk/sectionIV/

Page maintained by Julian Rathke.