SOFT8023 – Assignment 2 – Option B

Due: 7 Dec 2022

Value: 30%

Part 1 (15%)

Overview
Using Python Sockets, create an application that allows a remote client to send commands to a server component to get employee details.

Client:
Develop a menu-driven console-based program (using Sockets) that will allow the user to get information about the modules offered at MTU. Of the query data listed, only the learning outcomes can be edited by the user.

Example run:

```
Module System 1.0
What is the module id? SOFT8023
(L)earning Outcomes, (C)ourses, (A)ssessments or e(X)it? L
1. Evaluate and apply design patterns in the design and development of a
distributed system.
2. Assess and apply different architectural patterns in a distributed system.
3. Critically access and apply threading in a distributed application.
4. Debug a distributed client/server application, identifying object
properties and variables at run-time.
5. Create a distributed object application using RMI, allowing client/server
to communicate securely via interfaces and objects.
(A)dd, (E)dit, (D)elete or (R)eturn? D
Enter LO #: 3
Updated LO List:
1. Evaluate and apply design patterns in the design and development of a
distributed system.
2. Assess and apply different architectural patterns in a distributed system.
3. Debug a distributed client/server application, identifying object
properties and variables at run-time.
4. Create a distributed object application using RMI, allowing client/server
to communicate securely via interfaces and objects.
(A)dd, (E)dit, (D)elete or (R)eturn? A
Enter new LO description: Learn to play the saxophone.
Updated LO List:
Evaluate and apply design patterns in the design and development of a
distributed system.
2. Assess and apply different architectural patterns in a distributed system.
3. Debug a distributed client/server application, identifying object
properties and variables at run-time.
```

```
4. Create a distributed object application using RMI, allowing client/server
to communicate securely via interfaces and objects.
5. Learn to play the saxophone
(A)dd, (E)dit, (D)elete or (R)eturn? E
Enter LO #: 5
Enter new text: Learn to play the flute.
Updated LO List:
Evaluate and apply design patterns in the design and development of a
distributed system.
2. Assess and apply different architectural patterns in a distributed system.
3. Debug a distributed client/server application, identifying object
properties and variables at run-time.
4. Create a distributed object application using RMI, allowing client/server
to communicate securely via interfaces and objects.
5. Learn to play the flute
(A)dd, (E)dit, (D)elete or (R)eturn? R
What is the module id? SOFT8023
(L)earning Outcomes, (C)ourses, (A)ssessments or e(X)it? C
CR_KSDEV_8
CR_KDNET_8
CR_KCOMP_7
What is the module id? SOFT8023
Learning Outcomes (LO), Courses (C) or Assessments (A)?A
Project (An example assessment would be to create a simple client server
application using sockets) 20.0% Week 6
Project (Programming assignment(s) using the technologies covered in the
lectures. Example assignment(s) will access if a student can apply design
patterns to write distributed code, use technologies such as RMI and secure
communication and information in transit.) 30.0% Week 12
(L)earning Outcomes, (C)ourses, (A)ssessments or e(X)it? X
Goodbye
```

Server:

Using Socket programming, accept a connection from a client and handle the options given in the example run, i.e.

- Verify the existence of a module id. The client sends a value such as “SOFT8009” and we return a Boolean result.
- Knowing the employee id is valid, now it can receive commands and options.
- The server must also be able to send back a “not recognised” response for invalid commands / options (and the client must handle it).
- Use 1 or more data structures (hard coded is fine) to store some module details. Lists, dictionaries or whatever you want can be used. You could use tinydb for a JSON file, but it is not necessary.

Part 2 (6%)

Modify the sockets server program to handle multiple simultaneous client connections.

Part 3 (6%)

It would be good to keep track of access to the system (i.e . an activity log). Modify the server so that each time a request is received for a module, a message is sent to a message queue with the module id, the command and options sent from the client, as well as IP address details. Then write a simple script to print out the activity log details. Use RabbitMQ. There is very similar functionality in the Darts application – see lab05.pdf for the stats example.

Part 4 (3%)

Put the server into a Docker container.

Form

There will be a separate form to fill in to indicate what you got finished and sample runs of your programs.