**CS6006 Distributed, Cloud, and IoT Systems**

Coursework 1: Distributed Application

***Submission Deadline: 31st Jan 2024***

# Summary

This coursework is worth 35% of the total module mark. It requires developing of a prototype of a distributed system for exchanging information over chosen network protocol between clients, servers and/or peers and a written report to be submitted by the deadline.

# Tasks

### Design the application you would like to develop after picking one of each of the following alternative choices.

* 1. Choice of one of the following architectures to be designed: *client/server, peer-to-peer* or *mixed* (***Examples:*** entirely peer-to-peer chat application which allows direct chatting between the users, client/server chat application which uses central server to exchange the messages between the users, or a chatroom, which allows to use private rooms for exchanging messages as well as common meeting room, etc.)
  2. Choice of one or more of the following data structures to be used for exchanging information in the distributed application: *text messages, multi-media messages, text files, binary files, text feeds, multi-media streaming* (***Examples:*** text messaging with binary file attachments, multimedia messaging, uploading/downloading binary files to/from the server, broadcasting multimedia messages with file attachments, etc.)
  3. Choice of one of the following protocols to be used: *TCP, UDP, HTTP, IIOP* or *RTP* (***Examples:*** TCP for desktop messaging application with attachments, UDP for multimedia broadcasting over the Internet, HTTP for instant chatting on the Web, etc.)
  4. Choice of one of the following interfaces to be used: *command prompt input/output* (using System package), *GUI-based input/output*, *Web form-based input/output*, *Applet-based* input/output, or any applicable combination.

### Develop the software to match the design.

* 1. Develop the basic server-side application, test it using suitable test client and create deployable version of it. (***Hint:*** Place all files of the client in a single Eclipse project).
  2. Develop the client-side application and test it using the server-side. (***Hint:*** Create a separate project for the client application, different from the server application project).
  3. Extend the server-side and the client-side applications to allow any additional functionality whenever applicable (i.e., graphical UI, multi-threading, password control, attachments, etc.)

### Test the system produced

Some tests can be already reported as part of the development process (i.e., unit tests), but this section should include the tests of the separate components (integration testing) and the entire application (functional testing)

### Write the report

The report structure should follow the process of working - titlepage, table of contents, introduction, design, development, deployment, application walkthrough, testing, conclusion.

# Requirements

1. Your choices must be registered with the teaching team by **Monday 16th Oct 2023**. Your submission must match the registered choices.
2. The deployed software should run using the Java runtime engine only. All non-standard libraries must be included in the archives of the deployed version.
3. The written report must provide information on the following issues:
   * **Application design** with suitable illustration of the software architecture (at least one UML diagram).
   * **Software Development** (at least one class/package/component diagram for the structure and one collaboration/sequence/statechart diagram for the functionality).
   * **System Deployment** (how to install, configure and run the application).
   * **Application Walkthrough** of the working application (set of screen dumps from the client and the server).
   * **Software and/or System Testing** with suitable test cases (test strategy, testing scenarios, test data, test results and analysis).

# Deliverables

1. Written report, submitted by the deadline to WebLearn (one .pdf file)
2. Java application, submitted by the deadline to WebLearn (Eclipse, NetBeans, or any project files from your chosen IDE, with all sources plus deployment files in .jar and/or .war format – all in a single .zip archive)

## Notes:

1. All code files must be submitted to WebLearn in a single .zip archive (any other archive formats are not supported and using them may cause the entire submission to be discarded)
2. The report must be submitted in a single .pdf file

# Marking Scheme

* + Basic server-side functionality up to 5 marks
  + Basic client-side functionality up to 3 marks
  + Additional functionality up to 20 marks
    - GUI extra 4 marks
    - Multi-threading extra 4 marks
    - User-to-user delivery extra 2 marks
    - Multi-part communication (attachments) extra 3 marks
    - Session Contextualization (chatrooms, folders, boxes, etc.) extra 2 marks
    - User authentication (file or database registration) extra 3 marks
    - Operation Auditing (log file) extra 2 marks
  + Design, Development and Testing up to 5 marks
  + Deployment, Documentation and Quality up to 5 marks

**Total: max 35 marks**

## Notes:

1. The basic functionality may be sufficient to pass but without additional functionality the max mark will be 51% (no first class)