ITCS 222L Final Project
Distributed Application in CORBA
Due Date: 15 May 2025

Objective: To create a distributed application using CORBA technology where the server and client programs are implemented in the Java programming language and another client application is implemented in Python programming language.

Requirements:
Students will work in teams. Each team must create a distributed application where players (client application) must login using appropriate credentials. The server verifies the credentials by checking a database (note: client program does not have access to the database). The connecting client will be refused if the account is currently logged in (single session only) or if the credentials provided are invalid. Upon successful connection, the client will be provided with the following options (home screen):

a. Start "**What's The Word**" game (modified Hangman - you may search about this game if you are not familiar with it).
b. View the leaderboard.
c. Logout from the game.

If a player chooses to start a game and there is at least another player that has opted to start the game within 10 seconds when a game was first initiated, then the game shall begin. Note: the server can send an exception that will be handled by the initiating player in case there is no other player that is willing to play within the specified waiting time. Once players have joined, the game round shall commence where the server will send a digit, representing the number of letters of a random word, to be guessed by the players. A player must guess the mystery word by sending out letters to the server. Every missed letter will be counted and the player is given a maximum of five guesses or else he will lose in the given round. Also, a game round lasts for 30 seconds and if a player will not be able to guess the word correctly within the duration then he will lose the round. Each correctly guessed letter will be shown its position(s) on the mystery word on the player's side. The player who guesses all letters of the word correctly in the shortest time wins the round. No winner will be declared if all players have lost a given round. The first player to win 3 rounds will be declared the eventual winner for the game and it is at this point that the game ends placing the players at the home screen.

There is a possibility that two or more games (with different sets of players) will be concurrently running. In each game there will be a separate randomization procedure to be done in coming up with the mystery words to be guessed. Once a word has been selected for a given round in a game, it shall no longer be used in the succeeding rounds. The games are to be treated independent of each other. Note further that the server must be able to verify the game where a player is participating when he/she sends out a letter (as a guess).

When a player chooses the option to view the leaderboard, a window shall appear showing the top 5 players with the most wins.

For this project, there will be another client account type which will serve as the ***administrator*** with the privilege to perform **CRUD+Search operations for the player/s of the game**. Also, the administrator will have the option to change the time configurations:

- Waiting time for players to join a game (default of 10 seconds)
- Duration for a game round (default of 30 seconds)

Note further of the following:

- The server will use the file "***words.txt***" to randomly select the words to be guessed in a given game round.
- All pertinent information shall be stored in a MySQL database.

Required:

1. Create CORBA IDLs needed for the scenario above.
2. Implement the server-side component given the CORBA IDLs using the Java language.
3. Implement the client-side components given the CORBA IDLs using:
   a. Java programming language (player and administrator)
   b. Python programming language (player only)
4. The user interfaces for the Java client applications (player and administrator) must be graphical but not necessarily for the Python client application.

Deliverable:

1. Source codes: CORBA IDLs, Server program (Java), Client programs - player and administrator (Java), Client program - player (Python).
2. Documentation: Process of creating the Python client program and a write up of the difficulties/issues encountered in the process as well as how these issues have been resolved.
3. Accomplished consolidated peer evaluation forms.

Other details:

A. For the source code submission:
   Project folder naming convention: ClassCode-Team##_FinProject
   (e.g., **9300-Team01_FinProject**)
   Have the following items be placed inside the project folder:

Item	***Type***	***Description***
CORBA IDLs	Files	Shared files to be used by both server and client programs. No naming convention to be followed.
Server_Java	Folder	This directory contains all the source codes as well as other resources needed for the server (implemented in Java).
Client_Java	Folder	This directory contains all the source codes for the client applications and other resources needed for the programs to run properly.
Client_Python	Folder	This folder has all the source codes and resources (images, fonts, other files, etc.) needed to run the client program in Python.

Python_Installer	Folder	This folder has all the necessary installation files needed to run the Python client program.

Submit the archived form of this folder (**9300-Team01_FinProject.zip**).

B. For the documentation:

File naming convention: ClassCode-Team##_FinProject-Doc.pdf

(e.g., **9300-Team01_FinProject-Doc.pdf**)

This file must contain the step-by-step procedure on how to run the Python client program. Issue/s encountered and how you resolved this/these problem/s must also be indicated in this document.

Note:

Provide a cover page for this document with the names of the members as well as the subject and schedule of the subject must be indicated. List the names of the members in alphabetical order based on the last names.

C. For the consolidated peer evaluation forms:

File naming convention: ClassCode-Team##_FinProject-Peer.pdf

(e.g. **9300-Team01_FinProject-Peer.pdf**)

Use the provided template file in prelim/midterm for this.

Rating Scale:

5 - Excellent 4 - Very Good 3 - Adequate 2 - Needs Improvement 1 - Very Poor

Criteria:

- Administrator application features (35%)
    a. CRUD+Search operations - 25%
    b. Time configurations - 10%
- Distributed application functional features (40%)
    a. Single session - 5%
    b. Game creation - 10%
    c. Words are not repeated in a given game - 5%
    d. Isolation of concurrent games - 15%
    e. Leaderboard - 5%
- Other considerations for the application (25%)
    a. Use of MySQL database - 5%
    b. Java applications (GUI) - 10%
    c. Python application - 10%