Assignment-1: Student management system

Operating System

August 3, 2023

1 Context

You are required to create an application that offers functionalities/services required in an academic setting involving student activities. The main actor in this setting is 'student'. The application is to be developed using C-language, and on Linux-based environment.

A student at an academic institute can be part of various activities that are loosely related to each other and have their own seaparate environment. For example, a student can be part of a programming club, sports club, or cultural club. Each of these clubs may have different kinds of expectation from the student. Assume that, these clubs have their own application/website that manages student's association from registration to termination of membership, history of participation in various activities, or other kinds of statistics involving activities or members of the club.

2 Problem overview

You are required to develop an application for one of these clubs. This application will provide services such as registration (operation_id = 0), search (op_id = 1), update (op_id = 2), delete (op_id = 3) operations. There will be three kinds of role, administrator, manager, and members. This application will use command line interface. The application's working will require to follow the constraints listed below:

1. A file-based database, called as 'disk', will be used to store the records.
2. You will need to register 100 students. This registration will not be manual. You will need to create file(s) with 20 first names, 20 lastnames, 5 hostel names, 5 courses (BTech/MTech/MS/PhD/MBA). Now, complete student record name, hostel, course, and remaining details such as room number, date of birth, year of study will be generated using radom function. Once the whole record for a student is generated, registration function will be called and student's record will be stored in database.

3. Your application will, now, start a function called 'server'. This program will be listening to the requests from any user or any other program. These requests may involve the operations mentioned above (registration, serach, update, or delete). These operations will be requested by user or other program via 'operation_id'.

4. Everytime an operation is performed, the corresponding student record is kept in a linked-list called 'main-memory', except for registration. This linked list can have a maximum of five nodes.

5. Assume that it takes 2 seconds to access the 'disk' file; it takes one second to access 'main-memory'.

3 Performance Observation

1. **Request Generation**

   (a) **Request_generator_regular** This program generates a total of 50 pairs of random numbers (student_id=[1-100], op_id=[0-3]), at an interval of one second. Consider first number as student_id, and second number as operation_id. If operation is update, then choose a random value for Hostel room as update candidate.

   (b) **Request_generator_random** This program generates a total of 20 pairs of random numbers, like the first program. However, after generating each pair, this program waits for a random amount of time between one to 3 sec before generating next pair.

   (c) **Waiting Queue** Each request will be stored in a file called as 'operation_queue' as a four tuple ('regular/random', student_id, operation_id, time). First entry is an id of the program that generated the coressponding request, and last entry is the time when the request is being queued in file.

2. **Scenario1** Measure the throughput and mean response time for both request generators.

3. **Scenario2** Measure the impact of increasing the main-memory size by 20% and then 50%

4. **Scenario3** Measure the impact of faster memory technology, if both the memories perform twice as fast. Measure this performance for original memory size and increased memory size.

4 Code Execution/submission

The code should be executable with one single command 'sh run.sh'; all the remaining commands such as make or other commands should be mentioned in ‘run.sh’ script. Clearly, identify different componenets, and divide among yourselves. Aim to develop a modular code. Each file/module must have on top a description mentioning which method is implemented by whom. Keep the variable names readable, and same with method names.

5 Group Details

This is a group assignment. A group of three or four is permitted. The skills required are Linked list and File Handling, makefile, and shell scripting. Each member will implement and maintain a piece of code (assessment will be on your code; not planning or documentation). While implementing the project, keep thinking how the process can be improved, if required in future.