EECS 495
Introduction to Database Systems
Fall 2018
Instructor: Mas-ud Hussain
Project No. 3
Due: Wednesday, November 21, 2018

Guidelines:

- This project is supposed to be done in groups of 2. Please send an email to all the TAs (cc: mmasud.hussain@gmail.com) and let us know your groups whenever you form them. I'd suggest you to form a group within the next Monday at latest.
- The project is due by the midnight on the day mentioned on the top. Please submit project electronically in Canvas by uploading your files. As you'll have most likely more than 1 file to submit, please zip the files into a single file and upload the zip file. *Only one submission per group is required.*
- **Late policy:** Usual, 1 day is 10% off. 2 days is 20% off. We won't grade the projects submitted more than two days late.
- **There will be two demo sessions.** The project will be graded based on the demo. For the demo, we will use the same database schema but may use different data. Demo dates will be announced by your TAs – most likely the demo sessions will take place in the week after thanksgiving (*November 26 – 30*). During the demo, you'll have to download your submitted zip file from the canvas in front of the TAs and demonstrate all the attempted functionalities of your program (TAs will tell you what to do during the demo).

Assignment:

Develop a small database client that implements something along the lines of Northwestern Caesar's academic system. It should implement the following subset of the functionality provided by Caesar:

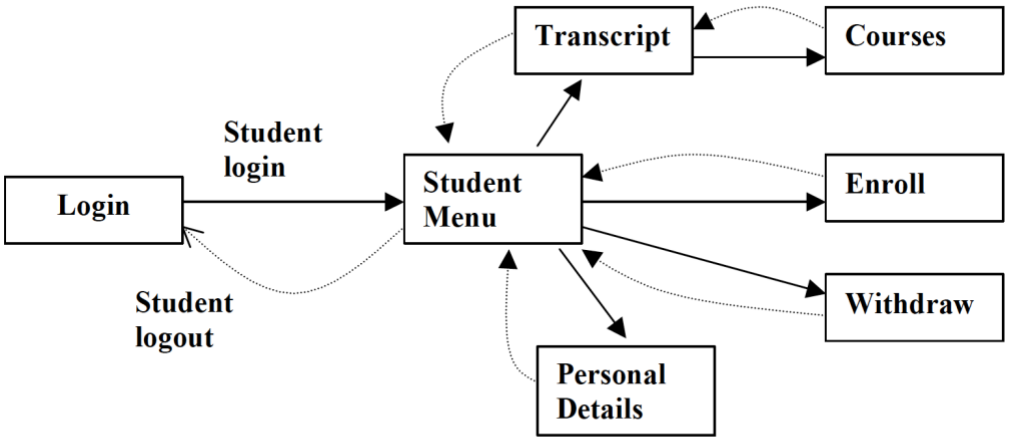

Figure 1: Screen flow of the database application to be implemented.

Database Schema:

STUDENT (ID, NAME, PASSWORD, ADDRESS)

FACULTY (ID, NAME, DEPTID, PASSWORD, ADDRESS)

CLASSROOM (CLASSROOM_ID, SEATS, TYPE)

UOSOFFERING (UOSCODE, SEMESTER, YEAR, TEXTBOOK, ENROLLMENT, MAXENROLLMENT, INSTRUCTOR_ID)

UNITSOFSTUDY (UOSCODE, DEPT_ID, UOSNAME, CREDITS)

WHENOFFERED (UOSCODE, SEMESTER)

REQUIRES(UOSCODE, PREREQ_UOSCODE, ENFORCED_SINCE)

TRANSCRIPT (STUDENT_ID, UOSCODE, SEMESTER, YEAR, GRADE)

LECTURE (CLASSROOM_ID, UOSCODE, SEMESTER, YEAR, CLASS_TIME)

Note: You will see Q1, Q2, etc. referred in the given database schema to denote quarters. Following is the meaning of the quarters:

Q1 = Fall; Q2 = Winter; Q3 = Spring; Q4 = Summer

That means, Q1 2018 is the current quarter, and Q2 2019 will be the next quarter.

In case of grades, "CR", "D", "P", etc. means the student passed and completed the course. "NULL" means the student's grade is incomplete or failed.

Requirements:

(a) The system should first ask the students to login using their username and password. Those two input arguments must be matched against the database. When successfully logged in, students shall be directed to the STUDENT MENU screen. If username or password is incorrect, an appropriate error message shall be given.
[10]

(b) The STUDENT MENU shall list the student's current courses of this quarter and year (use clock to programmatically determine the current year and quarter instead of hardcoding it. The courses currently being taken are also in the transcript but the grades are NULL), plus the following options: Transcript, Enroll, Withdraw, Personal Details and Logout. Logout shall log out the student and go back to the state where it waits for a student to login.
[10]

(c) The TRANSCRIPT screen shall list logged-in students their full transcript with all courses and grades. The student should be provided with an option to see details of any of the courses listed in the transcript or go back to the STUDENT MENU. If the student wants to see details

of any course, then he/she should be asked to enter the course number and the details of the corresponding course should be shown. The course details should include: the course number and title, the year and quarter when the student took the course, the number of enrolled students, the maximum enrollment and the lecturer (name), the grade scored by the student.
[20]

(d) The ENROLL screen shall allow logged-in students to enroll in a new course. Students shall be able to select a specific course offering in this screen, but only subject offerings of the current year and quarter or the following quarter shall be presented (again: don't hard-code these years but determine year and quarter programmatically from the current date using the clock). Students can only enroll in courses whose direct pre-requisites they have passed (not failed or incomplete) and whose maximum enrollment number has not been reached (i.e. MaxEnrollment > Enrollment). On successful enrollment, a new entry in the Transcript table shall be created with a NULL grade, plus the Enrollment attribute of the corresponding course shall be increased by one. In case the student cannot enroll because he/she has not cleared the prerequisites, print those prerequisites on the screen. **Implement this part using stored procedures and call it from your database client program.**
[25]

(e) The WITHDRAW screen shall allow logged-in students to withdraw from a course that they are currently enrolled in. Students can only withdraw from a unit that they have not finished so far (i.e. grade is NULL). If successful, the corresponding Transcript entry shall be removed and the current Enrollment number of the corresponding course shall be decreased by one. Implement this part using stored procedures and call it from your database client program.
[15]

If the Enrollment number goes below 50% of the MaxEnrollment, then a warning message should be shown on the screen. **Implement this using Triggers.**
[10]

(f) The PERSONAL DETAILS screen shall show the current personal record of the student and allow him/her to change his/her password and address. On submission, the corresponding Student record shall be updated accordingly.

Students cannot change their student id or name.
[10]

For extra credit:

All database modifications (such as course enrolling / withdrawing or student update) shall be implemented as **well-defined database transactions**.
[10]

Other Trivial Requirements/Notes:

- All the screens mentioned above can be implemented using simple text and menus.

- Please make sure your program does not crash at any point (EXCEPTIONS should be handled properly), and can handle all types of inputs.
- Some inherent business logic should always be met, e.g., a student should not be able to takes a course they are already enrolled in or completed in some previous quarter, a student should not be able to withdraw a course that they have not taken yet or already have completed (i.e., has a grade), etc.

Logistics:

- We are providing you with schema and sample data for the required tables. Please look for the schema "project3-nudb.sql" file in Canvas – which contains the database for this project.
- You are *encouraged* to use C API to connect to the MySQL database and stored procedures to implement the functionality. You can consult the tutorial as an initial guide provided with this assignment to learn how to use MySQL C API, stored procedures and triggers. *Although, note that the tutorials are there just to give you an idea and may not be up-to-date.*
- Also, if you want, you can use any other major programming languages instead of C, such as Java, Python, and PHP. There will be some links provided at the end as a pointer to libraries for each programming language as well – just as a starting point (you can choose your own libraries too, as long as they work).
- Having a GUI is not mandatory for the project, but if you want to build one – you are welcome!
- As you can see, this is mostly a DIY project (except for SQL portions – which you should be comfortable with by now). So, form a group and start early, as there can be some learning curve initially.

Helpful/Starter Links:

- **C:**
  - https://dev.mysql.com/doc/refman/8.0/en/c-api.html
  - https://dev.mysql.com/doc/refman/8.0/en/c-api-function-overview.html
  - https://dev.mysql.com/doc/connector-c/en/
  - https://stackoverflow.com/questions/44860756/how-to-connect-and-insert-record-to-mysql-using-c-language
- **Python:**
  - https://dev.mysql.com/doc/connector-python/en/connector-python-example-connecting.html
  - https://opensourceforu.com/2009/05/database-programming-in-python/
  - http://www.mysqltutorial.org/python-connecting-mysql-databases/
  -
- **Java:**
  - https://www.javatpoint.com/example-to-connect-to-the-mysql-database
  - https://dev.mysql.com/doc/connector-j/5.1/en/connector-j-usagenotes-connect-drivermanager.html
  - http://www.vogella.com/tutorials/MySQLJava/article.html

- **PHP:**

- https://www.w3schools.com/php/php_mysql_connect.asp
- https://www.a2hosting.com/kb/developer-corner/mysql/connect-to-mysql-using-php
- http://php.net/manual/en/function.mysql-connect.php