EECS 495—Fall 2016

Program No. 3 - Database Application Programming
Due Date - 11/16/2016

Guidelines:

• **This homework is supposed to be done in groups of 2. Please send an email to Sarah Yang at baiyuyang2017@u.northwestern.edu (cc: peters@eecs.northwestern.edu) and let us know your groups.**

• Homework is due by the midnight on the day mentioned on the top. Please submit homework electronically by clicking on Program 3 and uploading your file. Be sure to send the files in a format that we can read. If you have more than 1 file to submit, then please zip the files into a single file and upload zip file.

- Late policy: 1 day is 10% off. 2 days is 20% off.
- We won't grade the solutions submitted more than two days late.
- **There will be two demo sessions on Friday, Nov 18th and Monday, Nov 21rd.** The homework will be graded based on the demo. For the demo, we will use the same database schema but may use different data.

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:

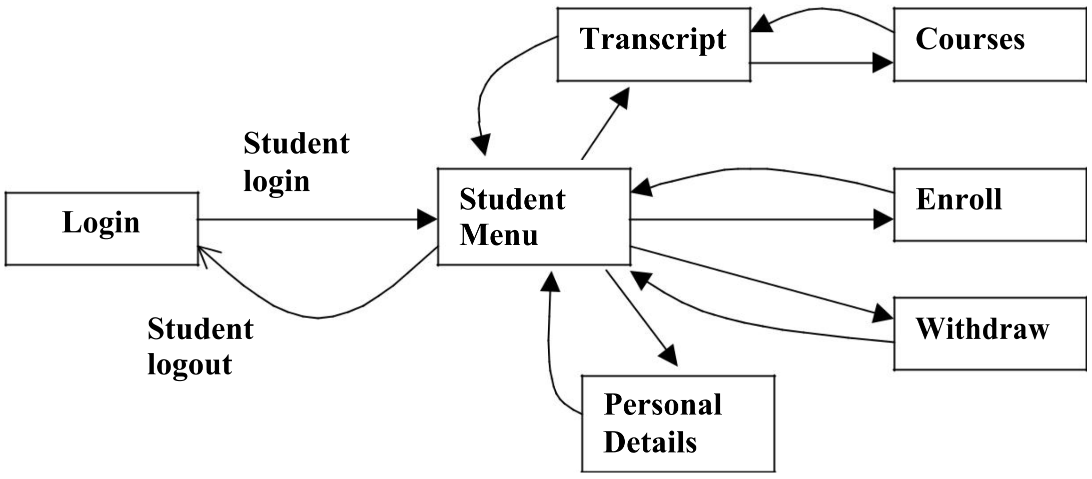

Figure 1: Screen flow of the database application to be implemented.

We are providing you with schema and sample data for the required tables. You are required to use C API to connect to the mysql database and stored procedures to implement the functionality. Please consult the tutorial provided with this assignment to learn how to use MySQL C API, stored procedures and triggers.

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. Everything of the username and password should match, i.e., caps, sequence of numbers, etc.
[7.5]

b) The STUDENT MENU shall list the student's current courses of this quarter and year (use clock/system 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. Note that, Q1 -> Sep-Dec, Q2 -> Jan-Mar, Q3 -> Apr-Jun, and Q4-> Jul-Aug. So, Q2 2016 session/quarter (in DB) means 2016 Winter.
[7.5]

c) The TRANSCRIPT screen shall list logged-in student's 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. Note that, in the database, there might be some courses which are taken in current quarter but already graded --- this is okay and made intentionally.
[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). *Also, there are some extra requirements that needs to be fulfilled, detailed in (h) part.* 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). Also, a student should not be allowed to enroll in a course, which they are currently taking, or have previously taken. 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.**
[20]

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]

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

g) The PERSONAL DETAILS screen shall also show and allow the student to specify a value for the "maximum preferred number of students" (e.g., 100) and "non-preferred classroom type" (e.g., tiered, flat, sloping, etc.). When configured for a logged-in student, in the ENROLL screen, exclude all the courses for which MaxEnrollment > student's "maximum preferred number of students", and also exclude all the courses which will be taken in a classroom with the same type as "non-preferred classroom type" automatically.
[15]

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

For extra credit:

database modifications (such as course enrolling / withdrawing or student update) shall be implemented as **well-defined database transactions** --- which means, a modification of the database will either go through fully, or fail fully.

[10]