

DATABASE MANAGEMENT APPLICATION DOCUMENTATION

PROJECT DESCRIPTION

A desktop application for the purpose of managing and viewing content information on the school's database via a Graphical User Interface.

This application has not been branded to a particular university for the sole purpose to allow usage for any university to manage and view content from their database by simply replacing the database username, password, and database name in the database connection structure.

DEVELOPMENT FRAMEWORK, DATABASE, AND LANGUAGE

The application was built using the QT Creator framework, C++ programming language, and MySQL database.

REQUIREMENT TO RUN THE APPLICATION

The application requires an existing MySQL database client on the computer.

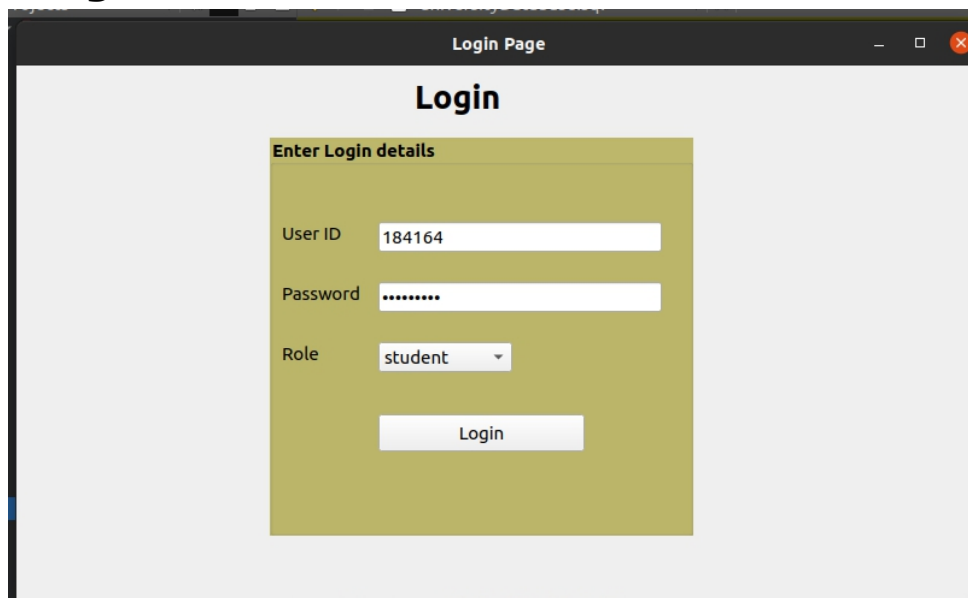
To connect the mysql client with the code, please go to the databaseconnection.h file and change the database connection details to mysql client details on local computer.

```
7
8  struct connection_details{
9      const char *server = "localhost",
10     *user="root",
11     *password = "9889",
12     *database = "universityDatabase";
13 };
14
```

The basics for the relations(including foreign key relationship and primary keys) is the schema below.

INTERFACES AND VALIDATION APPLIED

1. Login Page:

A screenshot of a web application window titled "Login Page". The window has a dark gray header bar with standard window controls (minimize, maximize, close). The main content area is light gray and features a central olive-green rectangular form. At the top of the form is the heading "Login" in bold black text, followed by a sub-heading "Enter Login details" in a smaller font. The form contains three input fields: "User ID" with the value "184164", "Password" with masked characters "*****", and "Role" with a dropdown menu showing "student". Below these fields is a "Login" button.

The login page has three input fields: user ID input, password input, and role input. The role refers to the type of user logging in and it has three fields in its dropdown. Users can log in as students, faculty coordinators, or instructors. The pages that will be displayed next will be based on the role entered.

The logic of the code:

When the login button is triggered, a query is executed to validate the user ID and password from the corresponding table. The table name is brought from the drop list of the role list.

Then, according to the type of user-selected, it displays the specified user page by launching a constructor of the user page. (Each user is implemented as a class).

Validations

Based on the role selected, a query will run in the background to validate if the ID and password input are in the tables. For example, when an ID and password are entered and the student role is selected, the query checks to make sure the ID and password entered are valid entries in the database. This also applies to the instructor and faculty coordinator. Only when this validation is true does it display the requested page.

The password is set as the user id prefixed with “std” for the student and “ist” for the instructor as the default behavior for the student and instructor.

Testing sample:

- Student: std187341.
- Instructor: ist983451.
- Faculty coordinator: 223144, password: 111111.

2. Student Page:

The screenshot shows a web application window titled "Student Page". It is divided into two main sections. The top section, "Student Profile", contains a form with the following fields: Name (Paula Umana), Std No (187538), Department (Computer Engineering), Faculty (Faculty of Engineering), Total credit (19), and CGPA (3.22). There is a placeholder for a profile picture and an "Edit profile pic" button. The bottom section, "Courses Registered", contains a table with the following data:

	Course Id	Group No	Semester	Year	Grade
1	COMP124	2	Fall	2020	B+
2	COMP117	1	Fall	2020	C-

Below this, there is a "Student Registered" table with the following data:

	Name	Student No	Course Id	Semester	Year
1	Tina Benjamin	199522	COMP117	Fall	2020
2	Carolina ...	199554	COMP117	Fall	2020
3	Stella Ambrose	185021	COMP117	Fall	2020
4	Denis Bolt	182945	COMP117	Fall	2020
5	James Peterson	181123	COMP117	Fall	2020
6	Carolina ...	199554	COMP124	Fall	2020
7	Tina Benjamin	199522	COMP124	Fall	2020
8	Venessa Miles	191554	COMP124	Fall	2020
9	James Peterson	181123	COMP124	Fall	2020
10	John Boston	182341	COMP124	Fall	2020

A "Logout" button is located at the bottom right of the page.

The student page has a profile page (and contains the student name, ID, department, total credit taken, and CGPA). It also contains the courses the student is registered to and other students registered in the class.

The logic of the code:

In the constructor of the student page, it accepts the student ID as a parameter and it runs multiple queries in the following order:

- Personal information query from the student relation.
- The registered course query from the takes relation using the student ID passed to the parameter of the student constructor.
- The classmates of the student in the courses registered; query from the takes and the student relations with the validation of student ID's, course ID, semester, and year.

The user has the ability to log out of the page. The logout button will close the current page and create a new object for the login page and launch the login page.

The student has the ability to set and change the profile picture.

Validation

The student profile shown can only be the student that logs in with a valid student ID and password.

3. Instructor Page:

The instructor Page is designed with QTabWidget and has three tabs.

The profile tab: Displays the profile of the instructor which contains his personal information; It also contains the logout button that will redirect the user to the login page.

The instructor has the ability to set and change the profile picture.

the instructor can fill in to register students into the courses he is assigned to.

Instructor Page

Profile Course info Register

Profile

PERSONAL INFORMATION

ID : 567123

Name: Dr. Cem Kalyoncu

Department: Software Engineering

Faculty Faculty of Engineering

Edit profile pic

Logout

The course info tab contains two QTableWidget: one displays courses the instructor is assigned, and the other, students registered to the instructor's assigned courses.

Instructor Page

Profile Course info Register

Courses Assigned

	Course Id	Group No	Semester	Year
1	COMP218	2	Spring	2020
2	COMP217	2	Spring	2020

Courses info

	Name	Student No	Course Id	Section	Semester
1	Venessa Miles	191554	COMP217	2	2020
2	Frederick Scott	184527	COMP217	2	2020
3	John Boston	182341	COMP218	2	2020

The registration tab displays a form where the instructor can fill to register students to the instructor's assigned courses.

The screenshot shows a web application window titled "Instructor Page" with a close button in the top right corner. Below the title bar is a tabbed interface with three tabs: "Profile", "Course info", and "Register". The "Register" tab is currently selected. Inside the "Register" tab, there is a form titled "Register Student" with a light green background. The form contains the following fields: "Student ID" (text input), "Course ID" (dropdown menu), "Group number" (text input), "Semester" (text input), and "Year" (text input). At the bottom of the form is a "Register" button.

Validation for the registration tab:

- The courses that the instructor can register students to are only the courses he is assigned (this takes into consideration the group number, semester, and year he is assigned). The course selection is ensured by using a dropdown for selection displaying only courses the instructor is assigned.
- Also, the instructor is limited to registering students only in the faculty he is in. For example, an instructor in the Faculty of Engineering cannot register students in the Faculty of Architecture.
- It is ensured that the student to be registered is already in the university's database by checking the student ID entered against all ID in the student table.

The logic of the code:

At the constructor of the instructor page, a query is run to obtain the list of all course IDs from the takes table using the instructor ID as the where clause condition. The result of this query is used to populate the QcomboBox on the registration page.

4. Faculty Coordinator Page:

The faculty coordinator page also is built with the tab widget and has two tabs.

Faculty Coordinator Page

Add

Assign

Add new course

Course ID:

Course title :

Dept_name:

choose...

Faculty:

choose...

Credit:

Add new course

Add new Instructor

Instructor ID:

Name :

Department:

choose...

Faculty:

choose...

Salary

Add new teacher

Add new Student

Student ID:

Name :

Department:

choose...

Faculty:

choose...

Total cred:

Add new Student

The Add tab contains different forms for the purpose of adding a new course, a new instructor, or a new student to the database.

The screenshot shows a web application window titled "Faculty Coordinator Page". It has two tabs: "Add" and "Assign", with "Assign" being the active tab. The main content area is olive green and contains two white form boxes. The left box, titled "Assign course to instructor", has fields for "Instructor ID:" (text input), "Course ID:" (dropdown menu showing "choose..."), "Group:" (text input), "Semester:" (dropdown menu showing "Spring"), and "Year:" (dropdown menu showing "2021/2022"). Below these fields is an "Assign Course" button. The right box, titled "Assign instructor to student", has fields for "Student ID:" (text input) and "Advisor ID:" (text input), followed by an "Assign Instructor" button.

In the assigned tab, forms to assign courses to instructors and assign advisors to students can be found.

Code Logic & Data validation for the Faculty page:

In the constructor of the faculty coordinator class, queries were entered to get all the department, faculty, and course ID. This information was used to populate the department, faculty, and course dropdown list in the respective form.

Due to the multiple access points that the faculty coordinator has to the database through this UI, measures have been made to ensure valid entries to the database. Validations done include:

- To add a new course, the course ID and course title field cannot be left null.
- The course ID to be inserted must be at most 8 characters and at least 5 characters. There also have to be at least two alphabets in the course ID. For example, C112 is not a valid entry because it has just one alphabet and it is less than 5 characters.
- The faculty of the course must be specified, but provision has been made in the database to allow the faculty coordinator to specify the department later.
- To add a new Instructor, the instructor ID, name and faculty must be specified.
- To add a new student, student ID, name and faculty must be specified.
- To assign a course to an instructor, the ID of the instructor must be an instructor already inserted to the instructor table in the database.
- Though a list of all courses can be seen in the course ID selection, it is ensured that the instructor cannot be assigned a course from a faculty he is not

registered to. For example, an instructor in the Faculty of Engineering cannot be assigned a course from the faculty of Pharmacy.

- The instructor to be assigned to a student as an advisor must be in the same faculty as the student.
- Both the new instructor ID and the new student ID(for adding new instructor and student, respectively) must be 6 characters.