Data Base Mid Project

March 10, 2024

Submitted by:

Abdullah Fasih, 2022-CS-6

Supervised by:

Prof. Nazeef Ul Haq

Department of Computer Science

University of Engineering and Technology Lahore, Pakistan

Table of contents

Introduction:

Welcome to our mid-project showcase where we explore the integration of database concepts within Windows Forms applications. In the contemporary landscape of software development, efficient data management is pivotal for creating robust and scalable applications. With the advent of relational database management systems (RDBMS) and the ubiquity of graphical user interfaces (GUI), developers often seek to synergize these technologies to empower users with seamless interaction and data manipulation capabilities. In this project, we delve into the realm of Windows Forms, a versatile framework provided by Microsoft for building desktop applications. We leverage the power of Windows Forms to craft intuitive and user-friendly interfaces that facilitate interaction with a backend database. Our objective is to demonstrate the fundamental operations of data management - Create, Read, Update, and Delete (CRUD) - within the context of a Windows Forms application.

Key Features:

1: We integrate a relational database into our Windows Forms application to store and manage data efficiently. Utilizing a database management system such as SQLite, MySQL, or SQL Server, we establish a robust foundation for data storage and retrieval.

2: Our project prioritizes user experience by designing intuitive interfaces using Windows Forms. Through carefully crafted layouts and interactive elements, users can seamlessly navigate through the application to perform CRUD operations effortlessly.

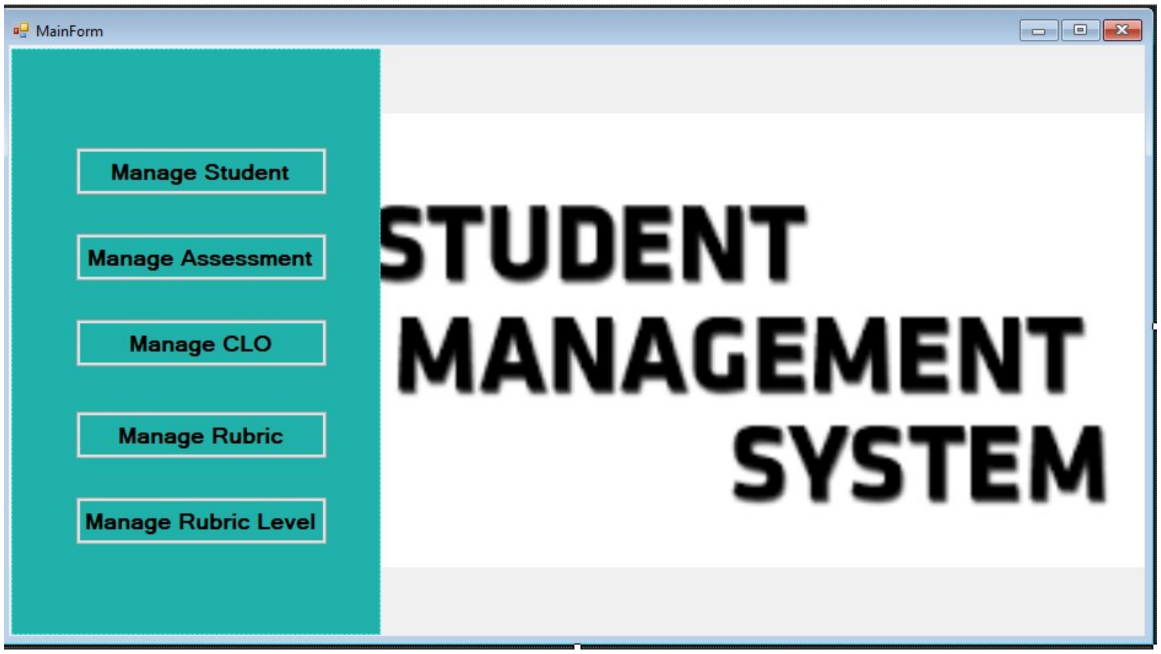
3: We implement the core functionalities of CRUD operations - Create, Read, Update, and Delete - within our Windows Forms application. Users can add new records, retrieve existing data, update information, and delete entries, all from within the familiar environment of the GUI.

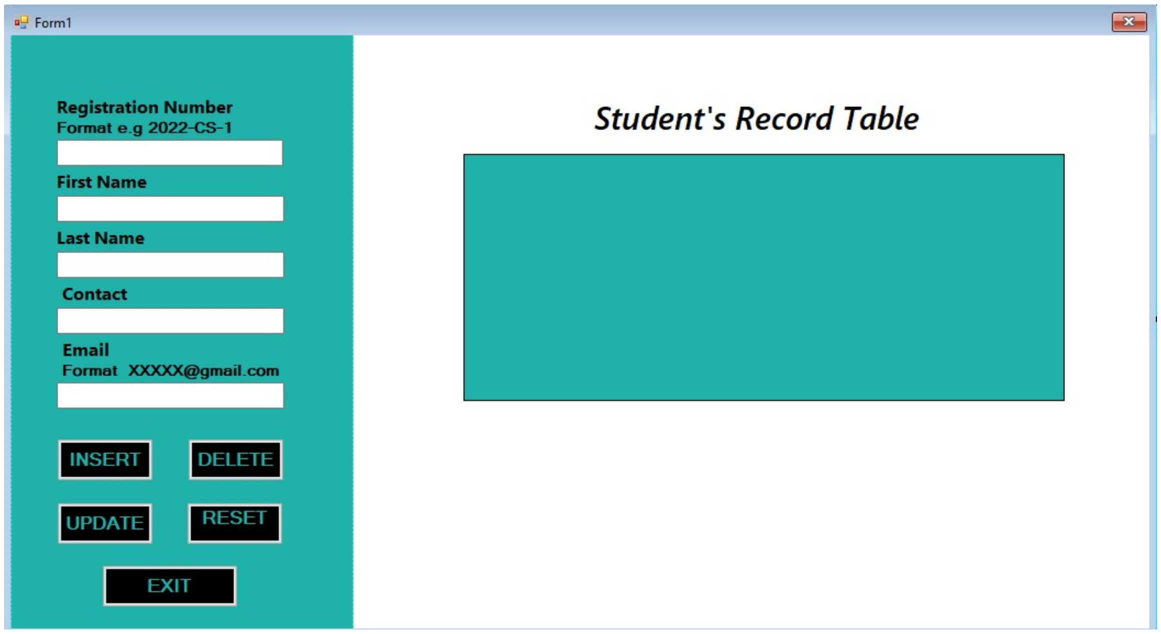
4: Ensuring data integrity is paramount in any application. We incorporate validation mechanisms within our Windows Forms application to validate user inputs and prevent erroneous data from entering the database, thus maintaining data accuracy and consistency.

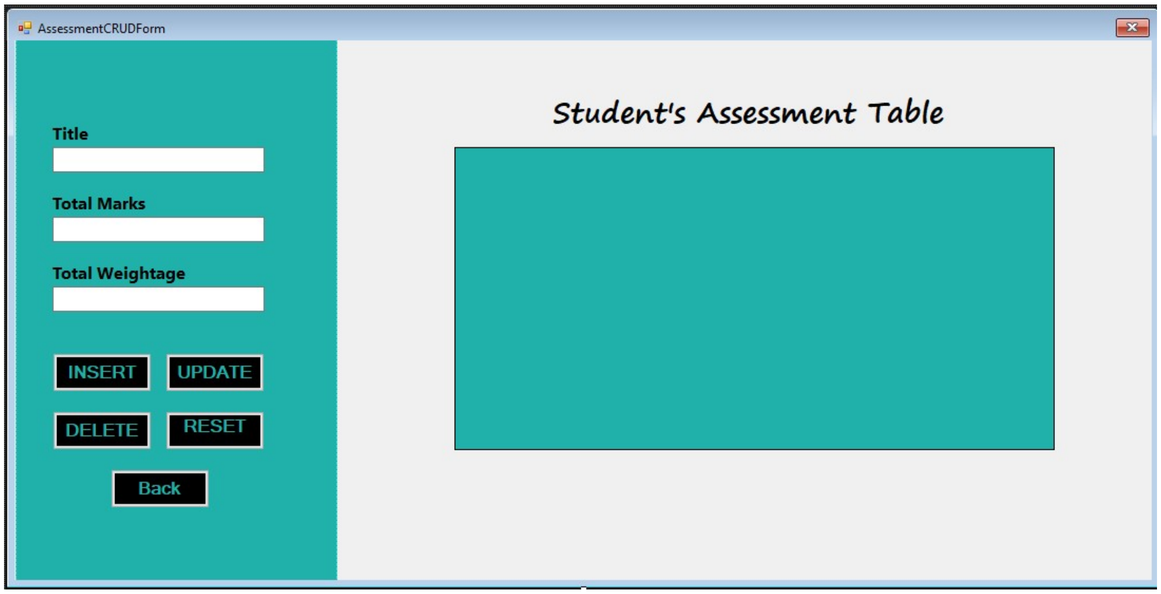
5: Robust error handling mechanisms are implemented to gracefully manage unexpected scenarios and provide informative feedback to users in case of errors or exceptions.

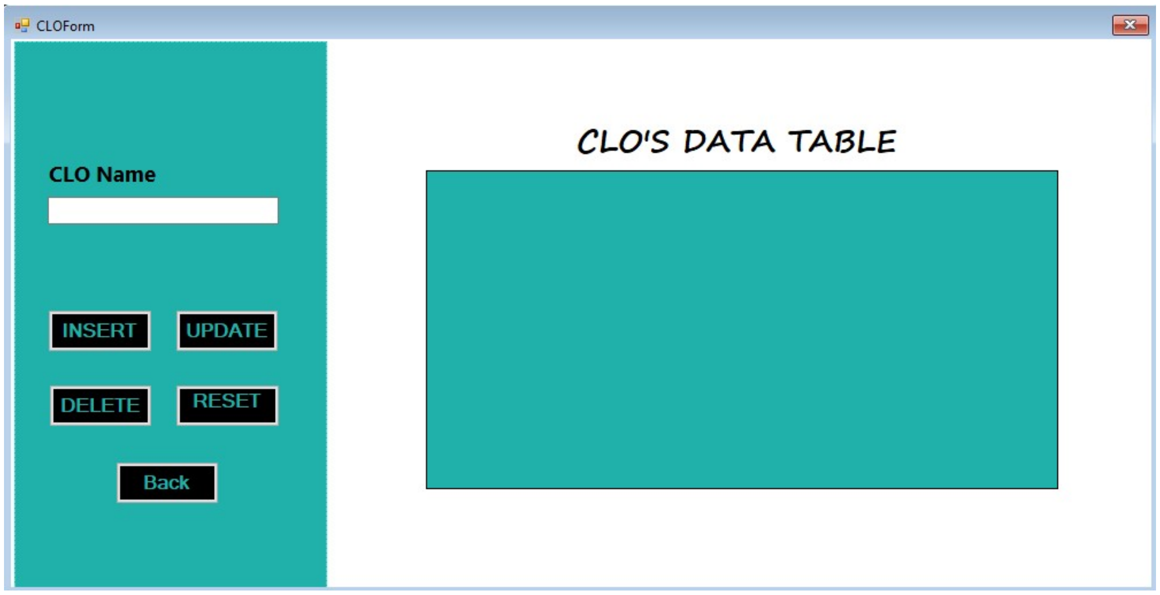
Interface layout:

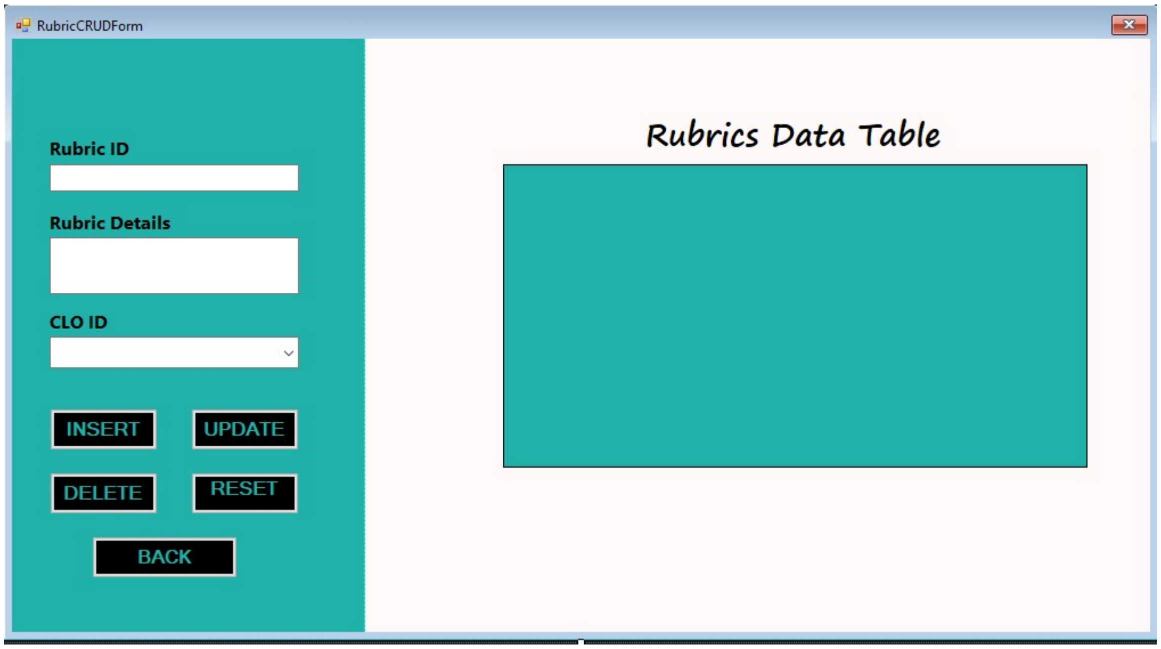
The UI of the application can be determine by the following pictures view:

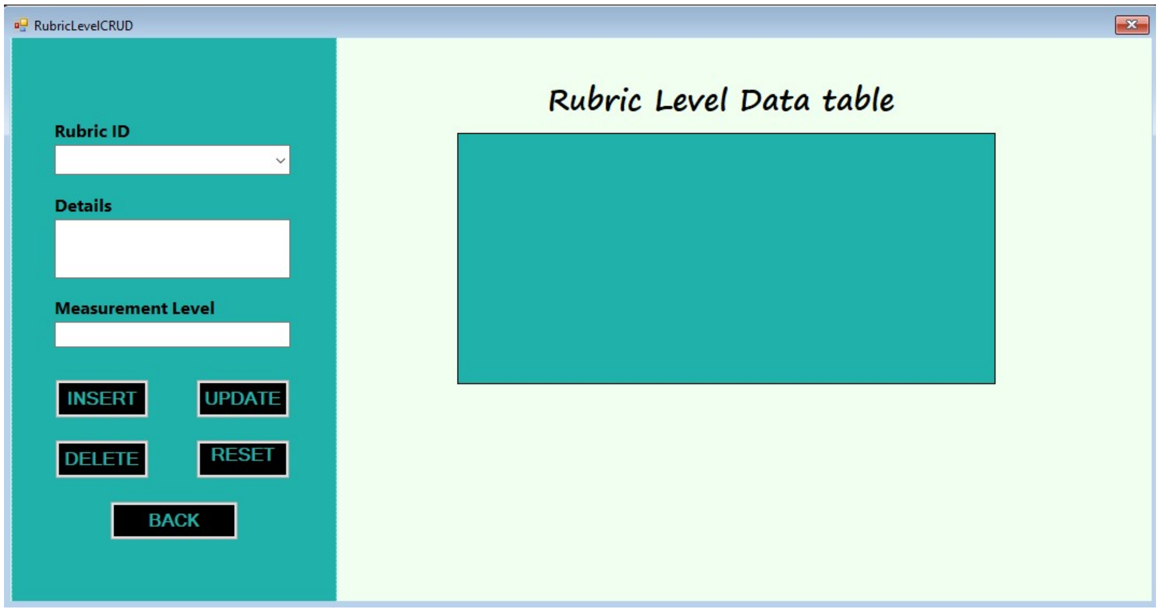





AssessmentCRUDForm
Title
Total Marks
Total Weightage
INSERT
UPDATE
DELETE
RESET
Back
Student's Assessment Table


CLOForm
CLO Name
INSERT
UPDATE
DELETE
RESET
Back
CLO'S DATA TABLE





Relations in data base:

The relation between different tables can be understood by the following:

Student:

Id is e unique identifier for each student. First Name is Student's first name. Last Name is Student's last name. Contact is Student's contact information. Emailis Student's email address. Registration Number is unique registration number assigned to the student. Status is student's current status (e.g., active, inactive).

Student Attendance:

AttendanceId is unique identifier for each attendance record. StudentId is Id of the student the attendance record is for (foreign key referencing the Student table). AttendanceStatus is student's attendance status for a particular class session (e.g., present, absent, excused). Lookupid is referencing a lookup table containing the possible attendance status values.

Class Attendance:

Id is unique identifier for class-level attendance records.AttendanceDate is date when the class attendance was taken. DateCreated is date the record was initially created. DateUpdated is date the record was last updated. One student can have multiple attendance records (one for each class session they attend).One class attendance record can include attendance information for multiple students.

StudentResult:

StudentId (int) is unique identifier for the student. AssessmentComponentId (int) is foreign key referencing the AssessmentComponent table. RubricMeasurementId (int) is foreign key referencing the RubricMeasurement table related to grading criteria. EvaluationDate (DateTime) is date and time the assessment was evaluated for the student.

AssessmentComponent:

Id (int) is a unique identifier for the assessment component. Name (string) is the name of the assessment component (e.g., question, section). RubricId (int) is a foreign key referencing the Rubric table (might define grading scales). TotalMarks (int) is the total marks achievable in this component. DateCreated (DateTime) is the date and time the assessment component was created.

Rubric:

Id (int) unique identifier for the rubric. Details (string) details or description of the rubric. CloId (int) foreign key referencing the Clo table.

CLO:

Id (int) unique identifier for the learning outcome (CLO). Name (string) is name of the learning outcome.

Rubric Level:

It has Id as a primary key and it takes Rubric Id From rubric table it main purpose to assign student different rubric level to gnerate result and find obtain marks.

Assessment:

It has Id as a primary key it has components . It has total marks and weightage and also Date created and updated it is that assessment upon which bases we generate results.

Obtain marks calculation:

The obtain marks can be caluculated as follow:

Obtain Marks = (Obtain rubric level/Mximum Rubric level) x Component marks

Relationships:

Assessment can have many Student Result objects (one for each student taking the assessment). Student Result is related to one Assessment and one Assessment Component. Assessment Component is related to one Rubric (defining the grading scheme). Rubric associated with multiple Assessment Components (across different assessments). A Rubric linked to a Clo (depending on how learning outcomes are modeled).

Error Handling in project:

In our project, error handling is meticulously implemented to ensure the robustness and reliability of the application. We employ a systematic approach to anticipate and address potential errors or exceptions that may arise during runtime. This includes validating user inputs to prevent invalid data from entering the system, implementing try-catch blocks to gracefully handle runtime exceptions such as database connection failures or SQL query errors, and providing informative error messages to guide users in troubleshooting issues. Additionally, logging mechanisms are employed to record errors and system events for later analysis and debugging purposes, enabling us to swiftly identify and rectify any issues that may occur. Through comprehensive error handling, we aim to enhance the stability and usability of our application, ensuring a seamless user experience even in the face of unforeseen challenges.

Conclusion:

In conclusion, our project demonstrates how we can create a user-friendly desktop application that interacts with a database. We've shown how to add, view, update, and delete data easily through the application's interface. By combining database skills with Windows Forms, we've made it simpler for users to manage information efficiently. With features like error handling and data validation, we've ensured the application runs smoothly and reliably. Overall, our project highlights the power of integrating databases into Windows Forms for building practical and user-centric desktop applications.