FYP Management System

Submitted by:
Omar Baig 2022-CS-43
Submitted to:
Mr. Nazeef-ul-Haq

University of Engineering and Technology Lahore

10 March 2024

Contents

Chapter 1

Overview

1.1 Introduction

The FYP management project is a comprehensive management system designed to streamline various aspects of academic administration within an educational institution. By managing the complex database and maintaining the schema of the database. It aims to digitize and automate processes related to student management, advisor allocation, project management, and group formation.

1.2 Purpose and Objectives:

The primary purpose of our project is to simplify and optimize the administrative tasks associated with academic management. The key objectives include:

- **A**utomating student registration and enrollment processes.
- **F**acilitating the assignment of advisors to students.
- **M**anaging project details, including allocation and tracking.
- **C**reating and managing student groups for collaborative projects.
- **P**roviding administrators with comprehensive data analysis and reporting capabilities.
- **Error Handling:** Graceful error handling, including try-catch blocks, contributes to a more robust and user-friendly system.

1.3 Scope of the Project:

The project encompasses several modules covering various aspects of academic management. These include:

- **Student Management:**Registration, enrollment, and profile management.
- **Advisor Management:** Allocation, update, and CRUD management.
- **Project Management:** Creation, assignment, and monitoring of academic projects.
- **Group Formation:**Formation of student groups for collaborative projects.
- **Reporting and Analytics:**Data analysis and visualization for decision support.

1.4 Technologies Used:

Our project leverages modern technologies and frameworks to achieve its objectives effectively. The key technologies used include:

- **Programming Languages:** C#
- **Database Management System:** Microsoft SQL Server.
- **Frameworks and Libraries:** .NET Framework
- **Frontend Development:** Windows Form (Asp.Net)
- **Version Control:** GitLab
- **Development Tools:** Visual Studio, SQL Server Management Studio

These technologies have been chosen for their reliability, scalability, and compatibility with the project requirements. They enable us to build a robust and user-friendly academic management system that meets the needs of our users efficiently.

Chapter 2

Database Schema

2.1 Description of the Database Structure

Here is the breakdown of the database structure:

2.1.1 Tables

- **Person:** This table stores information about individuals, including their first name, last name, contact details, email address, date of birth, and gender.
- **Student:** It contains additional details specific to students, such as their registration number.
- **Advisor:** Similar to the student table, this table holds information about advisors, including their designation and salary.
- **Project:** This table stores details related to academic projects, such as project name and description.
- **Group:** It represents groups formed by students for collaborative projects.
- **ProjectAdvisor:** This table establishes a many-to-many relationship between projects and advisors, indicating which advisors are associated with each project.
- **StudentGroup:** Similarly, this table establishes a many-to-many relationship between students and groups, indicating which students are part of each group.

2.1.2 Columns

Each table has columns representing specific attributes or properties of the entities they represent. For example, the Person table may include columns like FirstName, LastName, Contact, Email, DateOfBirth, and Gender.

2.1.3 Relationships

There are several relationships defined between tables using foreign keys:

- Students and Persons are related through a one-to-one relationship, as each student is also a person.
- Advisors and Persons have a one-to-one relationship, as each advisor is also a person.
- Projects may have multiple advisors, leading to a many-to-many relationship between Project and Advisor tables.
- Students can belong to multiple groups, and each group can have multiple students, resulting in a many-to-many relationship between Student and Group tables.

2.1.4 Constraints and Validations

- Primary keys: Each table likely has a primary key constraint to ensure each record is uniquely identifiable.
- Foreign keys: Foreign key constraints establish relationships between tables, ensuring referential integrity.

Understanding this database schema is crucial for developing a comprehensive FYP management system that effectively manages students, advisors, projects, and groups, as well as their relationships and interactions within the system. The Database diagram is also provided in the Images section of the report.

Chapter 3

Functionality

3.1 Overview

The FYP Management System offers several key features aimed at streamlining academic administration tasks and facilitating collaboration between students, advisors, and projects. Below is a detailed explanation of each major feature:

3.1.1 Student Management

- **Description:** This feature allows administrators to manage student information, including registration and profile details.
- **User Interaction:** Administrators interact with the system through a dedicated student management interface accessible via the dashboard or main menu.
- **Input:** Users input student data such as first name, last name, contact information, email address, and date of birth into the system.
- **Output:** Upon successful submission, the system stores the student's information in the database and provides a confirmation message to the user.
- **Validation:** The system performs data validation checks to ensure that all required fields are filled out and that the data is in the correct format (e.g., valid email address, correct date format).

3.1.2 Advisor Management

- **Description:** This feature enables administrators to manage advisor information, including assignment, update, and deletion of advisor records.
- **User Interaction:** Similar to student management, administrators access the advisor management functionality through a dedicated interface in the system.
- **Input:** Users provide advisor details such as first name, last name, contact information, email address, designation, and salary.
- **Output:** Upon successful submission, the system stores the advisor's information in the database and notifies the user.
- **Validation:** Data validation mechanisms ensure that the entered information meets specified criteria (e.g., salary is a numeric value).

3.1.3 Project Management

- **Description:** This feature allows administrators to manage academic projects, including creation, assignment, and monitoring.
- **User Interaction:** Administrators access project management functionality through the system interface.

- **Input:** Users input project details such as project name and description.
- **Output:** The system stores project information in the database and displays project details to users for monitoring and tracking purposes.
- **Validation:** Validation checks ensure that project data is accurate and complete, with appropriate error messages displayed to users if validation fails.

Chapter 4

Implementation

4.1 Explanation of How the Project was Implemented

The FYP Management System was implemented using a combination of programming languages, frameworks, and design patterns to achieve its objectives efficiently. The implementation process involved several stages, including database design, front-end and back-end development and testing.

4.2 Technologies and Frameworks Used

The project leveraged modern technologies and frameworks to ensure reliability, scalability, and compatibility with project requirements:

- **Programming Languages:** C#
- **Database Management System:** Microsoft SQL Server
- **Frameworks and Libraries:** .NET Framework
- **Frontend Development:** Windows Form (Asp.Net)
- **Version Control:** GitLab
- **Development Tools:** Visual Studio, SQL Server Management Studio

These technologies provided a robust foundation for developing the FYP Management System, enabling efficient data management, user interface development, and system integration.

4.3 Design Patterns or Architectural Styles Employed

The project followed a layered architecture, with clear separation of concerns between the presentation layer and business logic layer. This architectural style promotes modularity, flexibility, and maintainability, making it easier to manage and scale the system over time.

Additionally, the project adhered to object-oriented principles to enhance code organization and reusability.

4.4 Code Snippets for Key Functionalities

Below are code snippets demonstrating key functionalities implemented in the FYP Management System:

4.4.1 Student Management

Listing 4.1: Code snippet for adding a new student

```
private void AddStudent(string firstName, string lastName, string contact, string email,
{
try
{
// Code for adding a new student to the database
// ...
}
catch (Exception ex)
{
Console.WriteLine("Error: " + ex.Message);
}
}
```

4.4.2 Advisor Management

Listing 4.2: Code snippet for updating advisor details

```
private void UpdateAdvisor(int advisorId, string firstName, string lastName, string ema
{
try
{
// Code for updating advisor details in the database
// ...
}
catch (Exception ex)
{
Console.WriteLine("Error: " + ex.Message);
}
}
```

4.4.3 Project Management

Listing 4.3: Code snippet for creating a new project

```
private void AddProject(string projectName, string description, DateTime startDate, Date
{
try
{
// Code for adding a new project to the database
// ...
}
catch (Exception ex)
{
Console.WriteLine("Error: " + ex.Message);
}
}
```

Chapter 5

Conclusion

5.1 Summary of the Project

The FYP Management System represents a significant step forward in digitizing and automating various aspects of academic administration within educational institutions. Throughout the development process, the project aimed to streamline student management, advisor allocation, project management, and group formation processes, ultimately providing administrators with a comprehensive platform for efficient academic management.

5.2 Achievements and Challenges Faced During Development

5.2.1 Achievements

- Successfully implemented CRUD functionalities for managing students, advisors, and projects, providing administrators with full control over academic data.
- Utilized modern technologies and frameworks to ensure reliability, scalability, and compatibility with project requirements.
- Developed a user-friendly interface that facilitates easy navigation and efficient data entry and retrieval.
- Implemented robust error handling and validation mechanisms to ensure data integrity and system stability.
- Established clear architectural patterns and design principles, promoting code reusability, maintainability, and scalability.

5.2.2 Challenges

- Integrating multiple modules and ensuring seamless communication between different components of the system.
- Addressing complex business logic requirements and ensuring accurate data processing and manipulation.
- Testing and debugging the system to identify and resolve potential issues and errors.
- Managing project scope and timeline to meet project deadlines and stakeholder expectations.

Chapter 6

Project UI (Images)

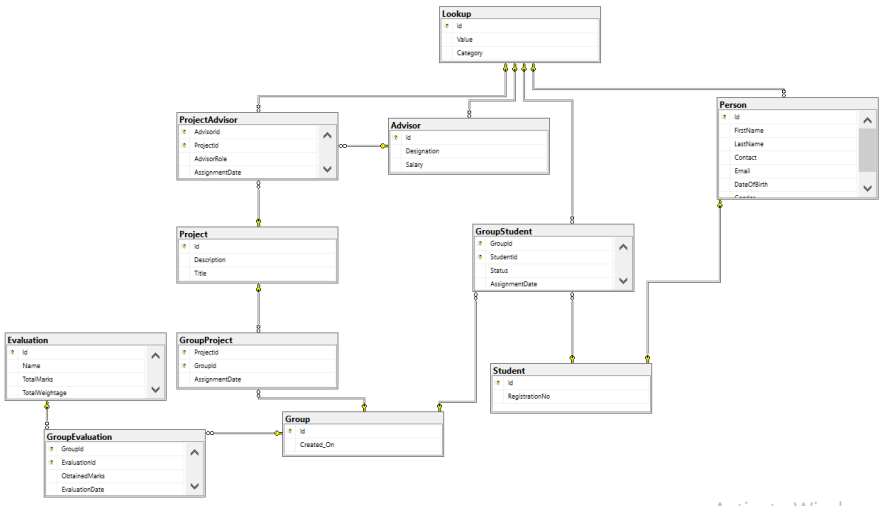

Figure 6.1: DataBase schema

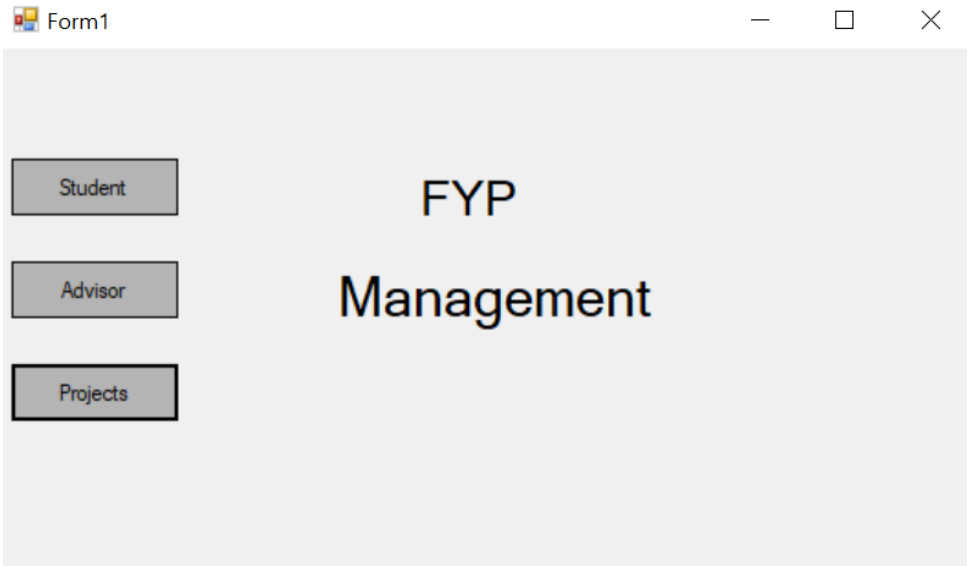

Figure 6.2: Main Page

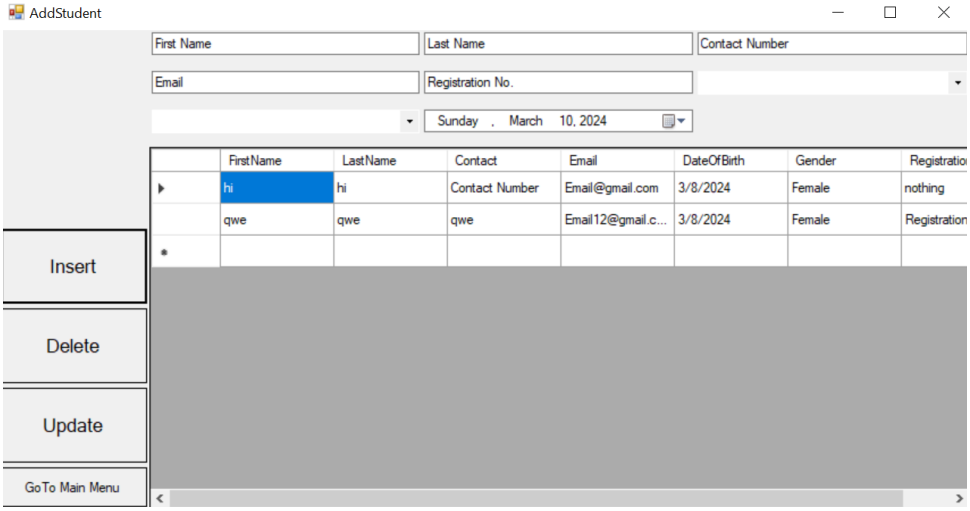

Figure 6.3: Student Management Page

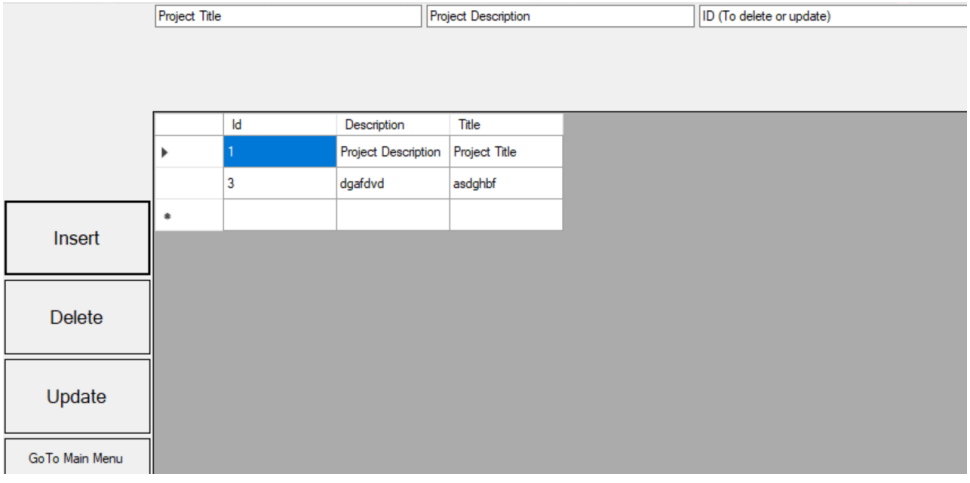

Figure 6.4: Project Management Page