**Student Grade Tracker Console Application with MySQL Database**

**Project Overview**

Create a console application to manage student grades using LINQ for data manipulation, delegates for custom sorting, events for notifying grade updates, and MySQL for data storage.

**Key Features**

1. **Student Class**
   * Create a Student class with properties: Name, ID, and Grades (array of integers).
2. **Data Management**
   * Implement a StudentRepository class to manage a collection of Student objects.
   * Use LINQ queries for operations like sorting students by name, ID, or average grade.
3. **Delegates for Sorting**
   * Define delegates (Comparison<Student>) for sorting students based on different criteria (e.g., by name, by average grade).
4. **Events**
   * Implement an event in the Student class that triggers whenever grades are updated.
   * Use events to notify the StudentRepository or other interested parties when grades change.
5. **Console Interface**
   * Create a console interface to interact with the StudentRepository.
   * Allow users to:
     + Add new students with names and initial grades.
     + Update grades for any student (triggering the grade update event).
     + Display student details sorted by various criteria (name, ID, average grade).
     + Subscribe to grade change events to receive notifications.
6. **MySQL Database Integration**
   * Use MySQL for persisting student data between sessions.
   * Implement CRUD operations (Create, Read, Update, Delete) using MySQL queries or an ORM like Entity Framework.

**Example Scenario**

* Users can add new students with their names and initial grades.
* They can update grades for any student, which triggers an event notifying the change.
* Users can sort and display students by name, ID, or average grade using different sorting delegates.
* The console application provides clear prompts and feedback for each operation.

**Benefits of the Project**

* **LINQ Practice**: Gain hands-on experience with LINQ queries for filtering, sorting, and manipulating data.
* **Delegates**: Understand how to use delegates for implementing flexible sorting strategies.
* **Events**: Learn how to implement and handle events for notifying changes in data, facilitating decoupled components.
* **Database Integration**: Practice integrating a relational database (MySQL) into a console application, learning about data persistence and retrieval.

**Additional Challenges (Optional)**

* Implement error handling and validation for input operations (e.g., validating grades to be within a specific range).
* Explore advanced features like transaction management in MySQL for ensuring data consistency.

**Technologies and Libraries**

* C# for application development.
* LINQ for querying and manipulating data.
* MySQL Database for data storage and retrieval.
* MySQL Connector/NET or Entity Framework for database interactions.
* Visual Studio or any preferred IDE for development.

This project provides a robust learning experience in handling data with LINQ, using delegates for sorting flexibility, implementing events for notifications, and integrating a MySQL database for persistent data storage, preparing you for database-driven application development scenarios.