Graduation Project G3

Deadline: 15 September

Overview

A structured ASP.NET Core MVC web application for managing **Courses**, **Sessions**, **Users**, and **Grades**, built with **Three-Tier Architecture** (Presentation Layer, Business Logic Layer, Data Access Layer) and the **Repository Pattern**, **Generic Repository**, **Unit Of Work**.

The project ensures **clean separation of concerns**, **reusability**, and **maintainability** With **Clean UI**.

**Requirements

1. **Three-Tier Architecture**
   - **Presentation Layer (UI)**: MVC Controllers + Views (Razor with Tag Helpers).
   - **BLL (Business Logic Layer)**: Services with business rules & validations.
   - **DAL (Data Access Layer)**: Repositories for CRUD operations.
2. **Repository Pattern**
   - All data access must be abstracted via repositories.
   - Example: `ICourseRepository`, `ISessionRepository`, `IUserRepository`.
3. **Validation**
   - Use **Data Annotations** for standard validation.
   - Implement **Custom Validation** (e.g., `NoNumberAttribute` for Course Name).
   - Implement **Remote Validation** (e.g., check if Course Name or Email is unique without page reload).
   - Enable **Client-Side Validation** with unobtrusive validation.
4. **Partial Views & Reusability**
   - Create **shared partial views** for repeated input forms (e.g., `_CourseForm.cshtml`, `_UserForm.cshtml`).
   - Use partial views in Create & Edit pages.
5. **Pagination & Searching**
   - Implement **pagination** for lists (Courses, Sessions, Users, Grades).
   - Allow **search and filter** by name, category, role, etc.
   - Example: Courses list with `search`, `pageNumber`, `pageSize`.

6. **User Feedback**
    - Show **success/error messages** after actions (Create, Update, Delete).
    - Example: "Course created successfully" or "Email already exists".
    - Use **Bootstrap alerts** or `TempData-based` notifications.

Core Features

1. Course Management

- CRUD: Add, edit, delete courses.
- Assign an instructor to a course.
- Search & paginate courses by name or category.
- **Validation**:
    - Name: Required, 3–50 characters, **unique** (Remote Validation).
    - Category: Required.
    - Custom: Name must not contain numbers (`NoNumberAttribute`).

2. Session Management

- CRUD: Manage sessions for a course.
- Set start and end dates.
- Search sessions by course name (with pagination).
- **Validation**:
    - `StartDate` : Required, cannot be in the past.
    - `EndDate` : Required, must be after StartDate.
    - `CourseId` : Required.

3. User Management

- CRUD: Add, edit, delete users.
- Users have: Name, Email, Role (`Admin` , `Instructor` , `Trainee`).
- Search & paginate users by role/name.
- **Validation**:
    - Name: Required, 3–50 characters.
    - Email: Required, must be valid, **unique** (Remote Validation).
    - Role: Required.

4. Grades Management

- Record a grade for each trainee in a session.
- View grades per trainee (with pagination).
- **Validation**:
  - Value: Required, must be between 0–100.
  - SessionId & TraineeId: Required.