Task : Library Management System

Project Overview

Develop a **Library Management System** using **ASP.NET MVC**, **Entity Framework (Code-First)**, and an **in-memory database** to manage **books, authors, and borrowing transactions**. Use an **n-tier architecture** with **services** to handle all business logic. No logic should be in controllers.

Features to Implement

1. Author Management

- Add, edit, delete, and list authors.
- Ensure the author's name is unique.
- Ensure each author has:
  - A **full name** (string, required, unique, must be four names with at least 2 chars for each one).
  - An **email** (string, unique, required, email validation).
  - A **website** (string, optional)
  - A **bio** (string, optional, max 300)
  - A read only list of the author books

2. Book Management

- Add, edit, delete, and list books.
- Each book must have:
  - A **title** (string, required).
  - A **genre** (required Enum from the following values: (Unknown, Adventure, Mystery, Thriller, Romance, SciFi, Fantasy, Biography, History, SelfHelp, Children, YoungAdult, Poetry, Drama, NonFiction)).
  - A **description** (optional, max 300).
  - An associated **author** (selected from a dropdown list).

3. Book Library

- Lists all the books with status (borrowed, available)
- The Ability to filter books based on the status / borrow date / return date
- Button for Borrowing a book (separate screen):
    - Set the `BorrowedDate` when a book is borrowed.
    - Prevent borrowing if the book is already checked out.
- Borrowed books only have the following action:
    - Return a book:
        - Set the `ReturnedDate` to current date & time.
        - Make sure that the book is marked as available.

4. JavaScript Feature

- On the **Book Library** Borrow Book action contains a **dropdown list** for selecting a book, which updates the availability status dynamically (e.g., "Available" or "Checked Out") and then borrow a book.

Technical Requirements

1. **Architecture**
   - Use an **n-tier architecture**:
     - **Presentation Layer**: ASP.NET MVC (for views and controllers).
     - **Business Layer**: All logic should be in services (e.g., `IBorrowingService`).
     - **Data Layer**: Entity Framework for database handling.
   - Implement **dependency injection** to decouple layers.
2. **Database**
   - Use **Entity Framework Core In-Memory Database**.
   - Seed sample data (e.g., a few authors and books).
3. **Controllers**
   - Create controllers for authors, books, and borrowing transactions.
   - Ensure controllers only handle requests and responses; all logic must reside in services.
4. **UI/UX Features**
   - **jQuery**: Use jQuery for dynamically updating the book's availability on the Borrowing Page.
   - **Pagination**: Add pagination to the system. **(Bonus)**
   - **MVVM and Partial Views**: Use partial views for displaying a book's details or borrowing status.

Submission Requirements

1. **GitHub Repository**
   - Include:
     - **Multiple commits** showing incremental progress.
     - A **README.md** with setup instructions and a brief explanation of the project and architecture.
2. **Code Quality**
   - Follow clean coding practices, with no duplication of logic.
   - Ensure **services** handle all business logic.
3. **DeadLine**
   - Project deadline is **3 Days (72 hours)** after receiving the email.

Send the github repository link with the subject [Your Name] [Date of the interview] [Task Name] to the following Email: career@codezone-eg.com