**Practical Examination**  
Subject code: PRN232   
Duration: 60 minutes

In this project, you are tasked with developing a robust and scalable ASP.NET Core Web API designed to efficiently manage life skills courses. This API will leverage the power of Entity Framework Core to facilitate seamless database interactions, ensuring data integrity and optimal performance. The system will be structured around a well-defined relational database schema, which encompasses key entities essential for course management. The database architecture is outlined as follows:

Solution Name: **SP25\_PRN231**

Users table

| **Column** | **Data Type** | **Constraints** |
| --- | --- | --- |
| **UserId** | int | Primary Key, Identity (auto-increment) |
| **Email** | nvarchar(100) | No duplicates |
| **Password** | varchar(100) | Encrypt (hashed) |

Courses table

| **Column** | **Data Type** | **Constraints** |
| --- | --- | --- |
| **CoursesId** | int | Primary Key, Identity (auto-increment) |
| **Title** | nvarchar(100) | Not null, do not leave blank |
| **Description** | varchar(100) | Allow null |
| **CategoryId** | int | Foreign Key Categories(CategoryId) |
| **UserId** | int | Foreign Key Users(UserId) |
| **Price** | decimal | Not null |
| **CreatedAt** | datetime | Default is the time of course creation |

Categories table

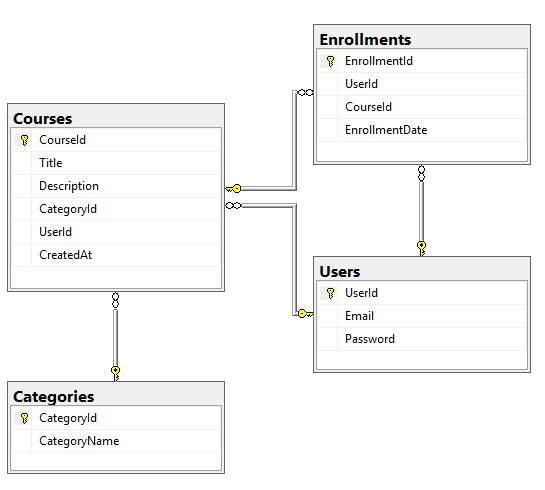
| **Column** | **Data Type** | **Constraints** |
| --- | --- | --- |
| **CategoryId** | int | Primary Key, Identity (auto-increment) |
| **CategoryName** | nvarchar(100) | Not null, do not leave blank |

Enrollments table

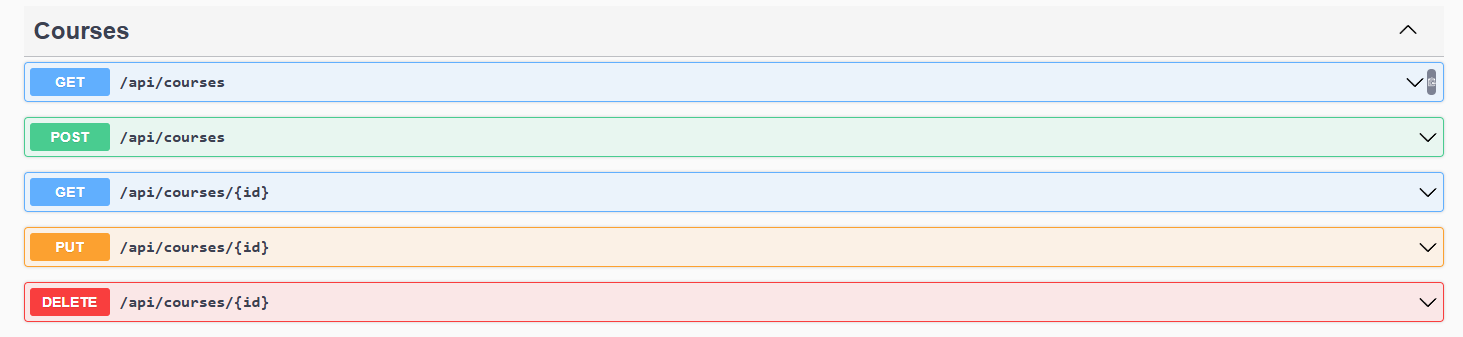
| **Column** | **Data Type** | **Constraints** |
| --- | --- | --- |
| **EnrollmentId** | int | Primary Key, Identity (auto-increment) |
| **CourseId** | int | Foreign Key Courses(CourseId) |
| **UserId** | int | Foreign Key Users(UserId) |
| **EnrollmentDate** | datetime | Default is the time of enrollment creation |

**1. Question 1: Build CRUD API and test with Postman (5 marks)**

* Create a database and Entity Framework Core model, can use Code First or Database First. (1 point)



* Create a CRUD API for the Courses table (2 points):
  + POST /api/courses → Create a new course
  + GET /api/courses → Get a list of courses
  + GET /api/courses/{id} → Get a course details
  + PUT /api/courses/{id} → Update course information
  + DELETE /api/courses/{id} → Delete a course



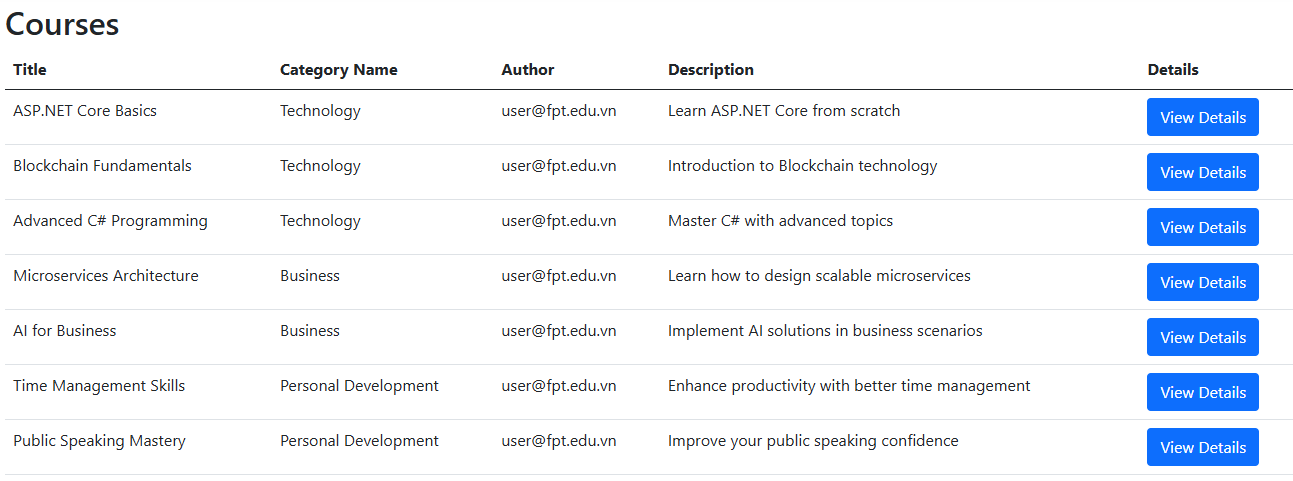
* Integrate OData to support dynamic queries in Course tables, including (1 point):
  + Filter data with price greater than
  + Sort courses by title descending
* Create a simple API for the Enrollments table to enroll (1 point):
  + POST /api/enrollments → Register a course

Note: Test the API using Postman

* + Send POST, GET, PUT, DELETE requests
  + **Take a screenshot** of each sentence's results using Postman and save it to a **Word file**.

**2. Question 2: Display API data on Web ASP.NET MVC / Razor Pages (3 marks)**

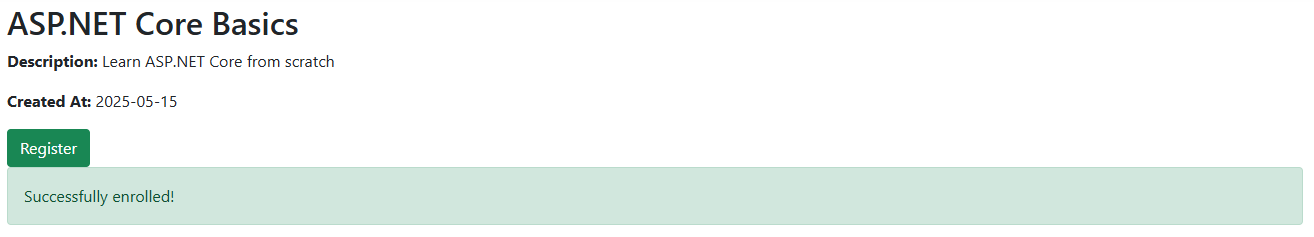
* Create a simple website that displays a list of courses from the API:
  + /Courses page displays a list of courses (1 point). Requires full display as shown below, including CategoryName, Author



* + Create a course details page (/Courses/Details/{id}), displaying information from the API (1 point)



* + Add course registration functionality by calling the Enrollments API (1 point)



**Bonus Points**

* Apply Repository Pattern, Service to Organize Code Better (1 point)
* Using AutoMapper to Convert Between Entities and DTOs (1 point)