Project Documentation: Course Management System

Overview

This project aims to develop a Course Management System using .NET Core for the backend and Blazor for the frontend. It includes features for managing courses, students, and enrollments with role-based authentication and authorization.

Technologies Used

- **Backend:**
  - ASP.NET Core
  - Entity Framework Core (MySQL)
  - JWT Authentication
  - Swagger (optional)
- **Frontend:**
  - Blazor WebAssembly (Student Portal)
  - Blazor Server or WebAssembly (Admin Panel)
- **Database:**
  - MySQL
- **API Documentation:**
  - Postman Collections
  - Swagger (optional)

Project Structure

Backend Structure

```
├── Controllers/
│   ├── AuthController.cs
│   ├── CoursesController.cs
│   ├── StudentsController.cs
│   └── EnrollmentsController.cs
├── Models/
│   ├── Course.cs
│   ├── Student.cs
│   └── Enrollment.cs
├── Data/
│   └── ApplicationDbContext.cs
├── Services/
│   ├── AuthService.cs
│   ├── CourseService.cs
│   ├── StudentService.cs
│   └── EnrollmentService.cs
└── Startup.cs
```

Frontend Structure

```
├── Pages/
│   ├── Index.razor (Student Portal Landing)
│   ├── Login.razor
│   └── Register.razor
├── Courses/
│   ├── CourseList.razor
│   └── CourseDetail.razor
├── Admin/
│   ├── ManageCourses.razor
│   ├── ManageStudents.razor
│   └── ManageEnrollments.razor
├── Components/
│   ├── CourseForm.razor
│   ├── StudentForm.razor
│   └── EnrollmentForm.razor
└── Services/
    └── AuthService.cs
```

Detailed Features

1. **Authentication and Authorization**
    - **Backend:**
        - JWT Authentication setup with ASP.NET Core.
        - Endpoints for user registration, login, and logout.
        - Role-based authorization (Admin and Student roles).
    - **Frontend:**
        - Login and registration forms.
        - JWT token handling for authenticated requests.
2. **Course Management (Admin Panel)**
    - **Backend:**
        - CRUD operations for courses.
        - Course entity with properties like CourseId, Title, Description, Duration, etc.
    - **Frontend:**
        - Admin panel pages for managing courses.
        - Forms for creating and updating courses with validation.
3. **Student Management (Admin Panel)**
    - **Backend:**
        - CRUD operations for students.
        - Student entity with properties like StudentId, Name, Email, DateOfBirth, etc.
    - **Frontend:**
        - Admin panel pages for managing students.
        - Forms for creating and updating student information with validation.
4. **Enrollment Management (Admin Panel)**
    - **Backend:**
        - CRUD operations for enrollments.

        - Enrollment entity with properties like EnrollmentId, StudentId, CourseId, EnrollmentDate, etc.
    - **Frontend:**
        - Admin panel pages for managing enrollments.
        - Forms for creating and updating enrollments with validation.
5. **Student Portal**
    - **Frontend:**
        - Visually appealing landing page.
        - Registration, login, and logout features.
        - Pages to display available courses and course details.
        - Ability for students to enroll in courses and view their enrollments.
6. **API Design**
    - **Backend:**
        - RESTful APIs designed for user management, course management, student management, and enrollment management.
7. **API Documentation**
    - **Postman:**
        - API endpoints documented using Postman collections.
        - Example requests and responses provided for each endpoint.
    - **Swagger (Optional):**
        - Integrated with ASP.NET Core for automatic API documentation generation.

Conclusion

This Course Management System integrates robust backend services with a modern frontend using Blazor, providing a seamless experience for administrators managing courses, students, and enrollments, and for students interacting with the portal. The use of JWT for authentication ensures secure access, while MySQL serves as the persistent data store.