# Exercises: REST Services

Problems for exercises and homework for the [“C# MVC Frameworks - ASP.NET” course @ SoftUni](https://softuni.bg/trainings/1537/csharp-mvc-frameworks-asp-net-march-2017).

## Courses Services

Create a REST Service with Web API for accessing a **Learning System.**

### Endpoints

Define the following endpoints in your service project. Do **NOT** use scaffolding (auto-generating endpoints)!

|  |  |  |
| --- | --- | --- |
| **Http  Method** | **Endpoint** | **Description** |
| GET | /api/courses/all | Get **list** of all courses with their **ID, name, trainer name, start date, end date** and the **count of enrolled students**. |
| GET | /api/courses?keyword={keyword} | Get the **names** and **IDs** of all courses who contains provided keyword |
| POST | /api/courses | **Add new course** by providing **name**, **trainer ID**, **start** and **end** date. That feature is restricted **only for admins**. |
| GET | /api/courses/{id} | Get **name**, **description,** **trainer name**, **start** and **end** **date** of a course |
| PUT | /api/courses/{id} | **Edit a course details** by provided id. Only for **admins**. |
| DELETE | /api/courses/{id} | **Delete** a course by provided id. Only for admins. |
| GET | /api/courses/{courseId}/students | Get a list of **ID**s and names of students **enrolled** in a course. |
| PUT | /api/courses/{courseId}/enroll | Enrolls the currently logged in user to the course with provided course ID |

Test the **endpoints** through [**Postman**](https://chrome.google.com/webstore/detail/postman/fhbjgbiflinjbdggehcddcbncdddomop).

## Articles Services

Create a REST Service with Web API for accessing a **Learning System.**

### Endpoints

Define the following endpoints in your service project. Do **NOT** use scaffolding (auto-generating endpoints)!

|  |  |  |
| --- | --- | --- |
| **Http  Method** | **Endpoint** | **Description** |
| GET | /api/articles | Get a **list** of all articles **titles** and **ID**s |
| GET | /api/articles/{id} | Get all details about an article **name**, **content**, **publish date** and **author** |
| DELETE | /api/articles/{id} | **Deletes** an article by provided id. **Only** the author of the article can delete an article |
| POST | /api/articles/publish | **Add new** article by providing title and content. **Only blog authors** can publish new articles. |

## Students Services

Create a REST Service with Web API for accessing a **Learning System.**

### Endpoints

Define the following endpoints in your service project. Do **NOT** use scaffolding (auto-generating endpoints)!

|  |  |  |
| --- | --- | --- |
| **Http  Method** | **Endpoint** | **Description** |
| GET | /api/students/{studentId}/courses | Gets a **list** of all courses (**name** and **start date**) ordered by **start date ascending**. |
| GET | /api/students/grades | Gets a **list** of **course** **names** and **grades** for the currently logged in user |
| PUT | /api/students/edit | **Edit the profile** of currently logged in student |

## Application Users

Implement two new models - **user** and **purchase for the BookShop**:

* **ApplicationUser -** has many **purchases**. It should extend (**inherit**) the default built-in **IdentityUser** class and add any extra properties. Users should be able to **login**/**register**/**logout** by using the standart identity API.
* **Purchase -** has **user**, **book, price, date of purchase** and **is-recalled**.

|  |  |  |
| --- | --- | --- |
| **Http  Method** | **Endpoint** | **Description** |
| PUT | /api/books/buy/{id} | Creates a **purchase** with the specified book and the currently logged user. Decrements the book's copies by 1. If there are no copies, return an appropriate status code. |
| PUT | /api/books/recall/{id} | Returns the book if **less than 30 days** have passed since the purchase. Increments the book count by 1 and sets the purchase to **recalled**. |
| GET | /api/user/{username}/purchases | Gets data about all **purchases** of the specified user, ordered by date of purchase. Returns **username** and for each purchase - **book title**, **purchase price**, **date of purchase** and whether it's **recalled or not**. |

## User Roles

Add an **Admin** **role for the BookShop**. Restrict access to all **POST**, **PUT** and **DELETE** routes only to users with **Admin** role (except buying and recalling books).   
Search online for ways to **authorize only certain roles**.

**Tip:** The database context should contain a **DbSet<Roles>**.

|  |  |  |
| --- | --- | --- |
| **Http  Method** | **Endpoint** | **Description** |
| PUT | /api/users/{username}/roles | Adds a new role to the user (passed in request body). |
| DELETE | /api/users/{username}/roles | Removes a role from the user (passed in request body). |

## Documentation

In the Learning System create full **documentation** describing **each method** in the CoursesController from the Web API. Consider documenting **description** of the method, its **parameters**, **returns**, etc.