Exercise 1a - Create a Web API Project

Create a new web API project. Either through the wizard, or command line interface, by:

dotnet new webapi

You'll probably also need to add packages for the Entity Framework, and the InMemory database. You can find both in the "nuget" package manager, or by using the terminal. Navigate to your project folder, and use:

```
dotnet add package Microsoft.EntityFrameworkCore
```

```
dotnet add package Microsoft.EntityFrameworkCore.InMemory
```

Exercise 1b - A Client for Testing

Run your web API and then perform a GET request to your web API using Postman (typically located on http://localhost:5000).

- What response do you receive?
- What is the content and format of the response?

The default example code should generate a Weather API, figure out how to request data from it.

If you use Postman, you may need to change the SSL setting:

File->Settings->General

Turn off the "SSL Certificate verification"

Exercise 3 - Building Your API

Create a C# `TodoItem` class with appropriate properties (long `Id`, string `Name`, string `Description`, bool `IsComplete`).

Below is an *example* overview of what your API should be able to do (You can choose a different approach, if you want):

API	Description	Request body	Response body
GET /api/todo	Get all to-do items	None	Array of to-do items
GET /api/todo/{id}	Get an item by ID	None	To-do item
POST /api/todo	Add a new item	To-do item	To-do item
PUT /api/todo/{id}	Update an existing item	To-do item	None
DELETE /api/todo/{id}	Delete an item	None	None

Include a way to GET specific items by providing arguments.

You can choose to use another model for your API. For example, if you like cats, create your API around retrieving and storing information about cats.

The main areas that you should keep an eye on when creating your API:

Accessing Data

Configure your API so that the data is stored in the in memory database.

Attribute Routing / Route Templates

Adjust your controller so that the routes match the above overview of the API. Use route tokens and extract route values when needed.

Model Binding and Validation

Apply data annotations to your model (e.g. the `Name` property should at least be 3 characters long).

Action Results

Change your controller actions to use `IActionResult` so that a status code is included in the response to the client. Remember to validate your model wherever appropriate.

For your POST request, you should use the `CreatedAtRoute` method (of the `ControllerBase` class) to tell the user where the data was stored.

If you're finding it hard to get started with building the API on your own, follow this official tutorial from Microsoft.

Exercise 4 - Help Pages (optional)

Create help pages for your Web API using Open API.

This is a good place to start.

Exercise 5 - A Client for the API

Create a basic console application that consumes your web API.

The client should perform a GET request to one of your API endpoints, download the data in an asynchronous manner and then write the response to the console.

Next, extend your client to also use the POST, PUT and DELETE endpoints of your API.

Make your client take input from the console.