TECHNOLOGIE KOMPONENTOWE GRUPA 2 - 06.11.2017

1. Sprawdzenie środowiska pracy:
   a. git --version
   b. dotnet --version
   c. Visual Studio Code
   d. Postman - przydaje się do testowania API, nie wymagany
2. Fork projektu:
   https://gitlab.com/lukasikpiotr/webapigr2
   Po ukończeniu zadania proszę o wykonanie Merge Request.
3. Stwórz aplikację opartą o interfejs komunikacyjny Web API (format JSON).
4. Obsługa akcji w klasie TodoController: (4 pkt)

Endpoint	Opis	Request body	Response Body
GET /api/Todo	Pobieranie wszystkich obiektów	brak	Lista obiektów lub pusta lista
GET /api/Todo/{id}	Pobieranie obiektu po Id	id	Model obiektu lub błąd

POST /api/Todo/	Dodawanie obiektu	np.
		{
		"id": **0**,
		"name": "name",
		"isComplete": true,
		"steps": **0**
		}
			Model stworzonego obiektu lub błąd

DELETE /api/Todo/{id}	Usuwanie obiektu po Id	id	Potwierdzenie usunięcia lub błąd

PUT /api/Todo/{id}	Zmiana wartości w obiekcie po Id	id
		oraz
		{
		"id": **0**,
		"name": "name",
		"isComplete": true,
		"steps": **0**
		}
			Potwierdzenie zmian lub błąd

5. Kody odpowiedzi (2 pkt):

Endpoint	Kod odpowiedzi
GET /api/Todo	OK
GET /api/Todo/{id}	OK lub NotFound

POST /api/Todo/	Created lub BadRequest
DELETE /api/Todo/{id}	NoContent lub NotFound
PUT /api/Todo/{id}	NoContent lub NotFound

6. Metody w kontrolerze powinny opierać się na komunikacji Async / Await. (2 pkt)
7. Stworzenie modelu ToDoItem w folderze Models, który powinien mieć następujące właściwości: (3 pkt)
    a. Id - typ long
    b. Name - typ string, nie może zawierać znaków białych, cyfr oraz innych znaków, może zawierać tylko duże i małe litery
    c. IsComplete - typ bool, zmienna wymagana
    d. Steps - type int, zmienna wymagana, zakres od 1 do 20
8. Użycie EF Core w projekcie (paczka nugetowa jest już w projekcie) (2 pkt):
    a. Dodanie kolekcji DbSet dla modelu do klasy TodoContext.
    b. Wstrzyknięcie InMemory Database w ConfigureServices
    c. Stworzenie konstruktora w klasie ToDoController wraz z konstruktorem przyjmującym ToDoContext.
9. Użycie biblioteki Swashbuckle.AspNetCore w celu stworzenia dokumentacji Swagger (referencja do Swashbuckle.AspNetCore znajduje się już w projekcie) (2 pkt)
    a. Wstrzyknięcie biblioteki Swashbuckle w ConfigureServices oraz Configure (klasa Startup.cs)
10. Spełnienie wszystkich testów w projekcie WebApi.Tests.