Kolokwium zaliczeniowe z przedmiotu Technologie Komponentowe (08.01.2018)

Warunki zaliczenia: Uzyskanie punktów 50% punktów tj. 10 pkt

Część 1

Jak wykonać pierwszą część kolokwium:

a) Wykonaj fork repozytorium: https://gitlab.com/lukasikpiotr/poprawa_cz_1
b) Po sklonowaniu repozytorium, otwórz VS Code
c) W VS Code w menu górnym kliknij w File -> Open, otwierając folder z pobranym repozytorium.
d) Otwórz konsole View -> Integrated Terminal.
e) W celu budowania projektu wpisz:
dotnet build
W celu uruchomienia testów wpisz:
dotnet test
f) Po wykonaniu zadań wykonaj marge request do głównego

Zadania części pierwszej (5 pkt)

1. Zadanie: (2.5 pkt)
Stwórz metodę, która zwróci silnie wskazanej liczby.
Proszę o umieszczenie metody w pliku:
MathService/MathService.cs

W pliku:
MathService.Tests/MathService_Factorial.cs
przygotowane są testy, które stworzona metoda musi spełniać.

2. Zadanie: (2.5 pkt)
Stwórz metodę o nazwie:
CollectionMaker
zwracającą listę liczb od 1 do n (gdzie n to argument metody), gdzie:
a) Wielokrotności liczby 2 zastąpi słowo „a”
b) Wielokrotności liczby 5 zastąpi słowo „b”
c) Wielokrotności liczby 2 oraz 5 zastąpi słowo „ab”

Proszę o umieszczenie metody w pliku:
StringService/StringService.cs

W pliku:
StringService.Tests/StringService_CollectionMaker.cs
przygotowane są testy, które stworzona metoda musi spełniać.

W tych zadaniach proszę o merge request tylko ze zmianami w plikach:
MathService/MathService.cs
StringService/StringService.cs
gdzie będzie znajdować się implementacja powyższych metod.

Część 2

Jak wykonać drugą część kolokwium:

a) Wykonaj fork repozytorium: https://gitlab.com/lukasikpiotr/poprawa_cz_2
b) Po skończonym zadaniu wykonaj merge request do głównego repozytorium.

Zadania części drugiej (15 pkt)

1. Stworzenie aplikacji w .Net Core 2.0 opartej o architekturę microserwisów związanej z wybraną tematyką (wyłączając temat aplikacji ToDo).
2. Stworzenie odpowiedniej struktury projektu (plik solucji, oddzielenie projektów aplikacji od projektów testów), (1 pkt)
3. Implementacja powinna zawierać przynajmniej 2 microserwisy, gdzie: (12 pkt)
   - 1 microserwis
     a. jednym z microserwisów będzie aplikacja oparta o interfejs komunikacyjny Web API (format JSON),
     b. model, który jest wykorzystywany w API ma zawierać co najmniej 5 właściwości (property)
     c. w warstwie „Data acces layer" należy użyć: Entity Framework Core InMemory
     d. aplikacja ma obsługiwać następujące akcje:

API	Opis	Request body	Response body (odpowiedź serwera)
1. GET /api/*model*/	Pobierz wszystkie modele	Brak	Obiekty w formacie JSON
2. POST /api/*model*	Dodaj nowy obiekt	Obiekt w formacie JSON	Obiekt w formacie JSON
3. PUT /api/todo/{id}	Zaktualizuj obiekt	Obiekt w formacie JSON	Brak

     e. akcje należy zaimplementować z wykorzystaniem async / await
     f. akcje mają zwracać odpowiednie kody odpowiedzi HTTP (według przyjętych standardów
     g. stworzenie testów integracyjnych oraz unit testów (XUnit, do każdej akcji przynajmniej jeden test, sprawdzający kod i zawartość)
     h. sporządzenie opisu wykonanego Web API, z uwzględnieniem adresów akcji oraz informacji na temat otrzymywanych kodów odpowiedzi
   - 2 microserwis
     a. implementacja aplikacji korzystającej z 1 microserwisu
     b. stworzenie testów integracyjnych
4. Dodanie pliku .gitlab-ci.yml z poprawną konfiguracją budowania projektu oraz wykonywania testów (2 pkt)
5. Po wykonaniu powyższych zadań należy wykonać merge request do repozytorium