Федеральное агентство связи

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики» (МТУСИ)

ООПСЗИ

## Лабораторная работа №6 Тема: Разработка уровня бизнес-логики приложения для Web-службы: вариант использования "Синхронизация данных с внешним источником"

Выполнил студент группы БПЗ1701 Меркушев М. А.

Проверил \_\_\_\_\_\_\_\_Барков В.В.

Москва 2020

**Задание.**

1) Добавить классы, реализующие возможность обновления БД.

Ссылка на используемые данные:<https://data.mos.ru/opendata/9710068955-perechen-prioritetnyh-produktov-i-tehnologiy-ispolzuemyh-v-otraslyah-gorodskogo-hozyaystva>

Ссылка на gitlab:

Листинг №1 – код файла ResultFromServer.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace InnovativeProduct.WebService.InfrastructureServices.Gateways

{

public class Cells

{

public string Effects { get; set; }

public string Name { get; set; }

public string Indicators { get; set; }

public string Tasks { get; set; }

public string TechnicalCharacteristics { get; set; }

}

public class ResultFromServer

{

public int Number { get; set; }

public Cells Cells { get; set; }

}

}

Листинг №2 – код файла GetJsonAndParse.cs

using System;

using System.Net;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Text;

using Newtonsoft.Json;

using InnovativeProduct.DomainObjects;

using InnovativeProduct.InfrastructureServices.Gateways.Database;

using System.Runtime.CompilerServices;

using Microsoft.Extensions.DependencyInjection;

using Microsoft.EntityFrameworkCore;

using System.IO;

namespace InnovativeProduct.WebService.InfrastructureServices.Gateways

{

public class GetJsonAndParse

{

public async Task ParseAndPush()

{

WebClient client = new WebClient();

client.Encoding = Encoding.UTF8;

string result = client.DownloadString("https://apidata.mos.ru/v1/datasets/593/rows?$top=3&api\_key=fee68e1ff9da6aa97c7deb04d48c82d1");

List<ResultFromServer> resultServer = JsonConvert.DeserializeObject<List<ResultFromServer>>(result);

var optionsBuilder = new DbContextOptionsBuilder<ProductContext>();

string newPath = Path.GetFullPath(Path.Combine(Directory.GetCurrentDirectory(), @"..\"));

string newnewpath = Path.Combine(newPath, "InnovativeProduct.WebService", "InnovativeProduct.db");

optionsBuilder.UseSqlite($"Data Source={newnewpath}");

var context = new ProductContext(options: optionsBuilder.Options);

context.Database.ExecuteSqlRaw("DELETE FROM TransferNodes");

using (context)

{

foreach (var item in resultServer)

{

DomainObjects.Product product = new DomainObjects.Product();

product.Name = item.Cells.Name;

product.ExpectedEffects = item.Cells.Effects;

product.Indicators = item.Cells.Indicators;

product.Tasks = item.Cells.Tasks;

product.Specifications = item.Cells.TechnicalCharacteristics;

context.Entry(product).State = EntityState.Added;

context.SaveChanges();

}

}

await Task.CompletedTask;

}

}

}