Импорт данных

**Импорт** - данные из внешних файлов переносятся в БД

**Этапы:**

1. Считывание данных из файлов

2. Десериализация данных(восст в изначальное состояние структуры данных из битовой последовательности)

**Примеры формата файлов:**

1. txr/tsv/csv

2. xls/xlsx/ods

3. json/xml

3. Запись десериализованных данных в БД

**Импрот данных из json:**

[]-массив

{}-объект

“”-названия параметров

: - значение параметра

* Значение может быть и массив и объект
* Для десериализации нужно создать класс модели данных
* В VS есть генерация класса на основе json и xml
* Json2kt.com – генерация классов, для языка Kotlin
* После создания классов модели данных выполняется имитация
* Для десеарилазации используется System.Text.Json или Newtonsoft.Json
* Сериализация данных в json
* Импорт csv: System.IO, CsvHelper

**Импорт файлов на сервер:**

1 вариант: загрузить в БД

public byte[]? Logo { get; set; }

**Минусы:**

* уменьшается рамер БД и резервной копии

**Приемущетсва:**

* Для запроисла вычисления лолтсуп только в БД
* 2 вариант копирование фала на сервер и соранине в БД только имени файла
* WebApi – dotnet8
* Кроме swagger можно использовать postman

**Методы Api**

**1. Get()-получение данных**

* 200 ok
* /api/users - список
* api/users/1 – объект
* 404 – not found
* 400 bad request
* Метод get – параметры в адрессной строке

**2. Post – создание ресурса**

* 201 crated
* Location – адрес ресурса
* Location /api/users/123
* 409 conflict
* Post для авторизации чтобы не передавать логин и пароль в открытом виде

**3. PUT – обновление**

* 204 no content
* 200 ok

**4. PATCH – частичное обновление, меняет часть данных а не все**

**5. DELETE – удаление ресурса**

* Post,put,path,delete позволяют выбрать через тело запроса, обычно в формате json
* Могут принимать Id

app.MapGet("/categories", (GameContext context)

=> context.Categories.ToList());

app.MapPost("/categories", (Category category, GameContext context)

=> {

try

{

context.Categories.Add(category);

context.SaveChanges();

return Results.Created();

}

catch

{

return Results.BadRequest();

}

});

* Для методов put и patch можно передавать идентификатор и получать его из оъекта

app.MapGet("/categories/filter", (string? name, string? sortBy, GameContext context, int page = 1, int pageSize = 5)

=>

{

var query = context.Categories.AsQueryable();

//фильтер

if (!string.IsNullOrEmpty(name))

query = query.Where(x => x.Name.Contains(name));

// сортировка

query = sortBy?.ToLower() switch

{

"id" => query.OrderBy(c => c.CategoryId),

"name" => query.OrderBy(c => c.Name),

\_ => query

};

// пагинация

var items = query.Skip(pageSize \* (page - 1)).Take(pageSize).ToList();

return items;

});