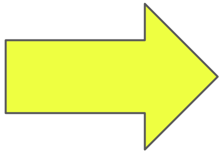
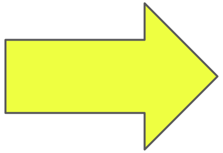
Consultas Avanzadas con ASP.NET CORE

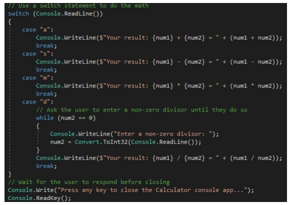
Specification Pattern



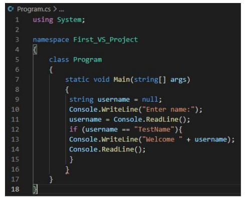


Biker's License

ID: 199 804 975 23
Jose' Roberto
2 Youngtodriveon Highway
Bike Park, US 97523
Gender: Male Eyes: Blue Hair: Brown
Class: High Issue: Jan 2013
Expires: Upon Issue of Motorcycle License
Personalized Bike Plates
Commissioner of Fun Powered Vehicles



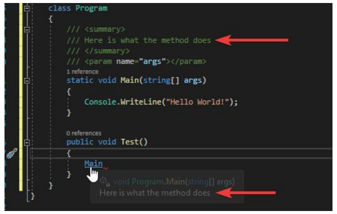
```
// Use a switch statement to do the math
switch (Console.ReadLine())
{
    case "a":
        Console.WriteLine($"Your result: {num1} + {num2} = " + (num1 + num2));
        break;
    case "s":
        Console.WriteLine($"Your result: {num1} - {num2} = " + (num1 - num2));
        break;
    case "m":
        Console.WriteLine($"Your result: {num1} * {num2} = " + (num1 * num2));
        break;
    case "d":
        // Ask the user to enter a non-zero divisor until they do so
        while (num2 == 0)
        {
            Console.WriteLine("Enter a non-zero divisor: ");
            num2 = Convert.ToInt32(Console.ReadLine());
        }
        Console.WriteLine($"Your result: {num1} / {num2} = " + (num1 / num2));
        break;
}
// Wait for the user to respond before closing
Console.Write("Press any key to close the Calculator console app...");
Console.ReadKey();
```



Program.cs > ...

```
using System;

namespace First_VS_Project
{
    class Program
    {
        static void Main(string[] args)
        {
         string username = null;
         Console.WriteLine("Enter name:");
         username = Console.ReadLine();
         if (username == "TestName"){
         Console.WriteLine("Welcome " + username);
         Console.ReadLine();
         }
        }
    }
}
```



```
class Program
{
    /// <summary>
    /// Here is what the method does
    /// </summary>
    /// <param name="args"></param>
    1 reference
    static void Main(string[] args)
    {
        Console.WriteLine("Hello World!");
    }

    0 references
    public void Test()
    {
        Main
    }
}
```

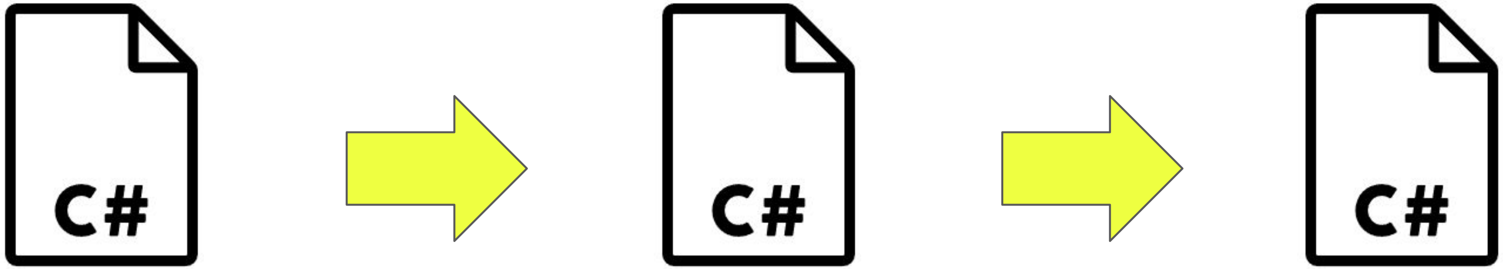
void Program.Main(string[] args)
Here is what the method does

Specification Pattern

Define Reglas

Implementa Expressions

Escribe la condicion logica



ISpecification.cs

ExpressionSpecification.cs

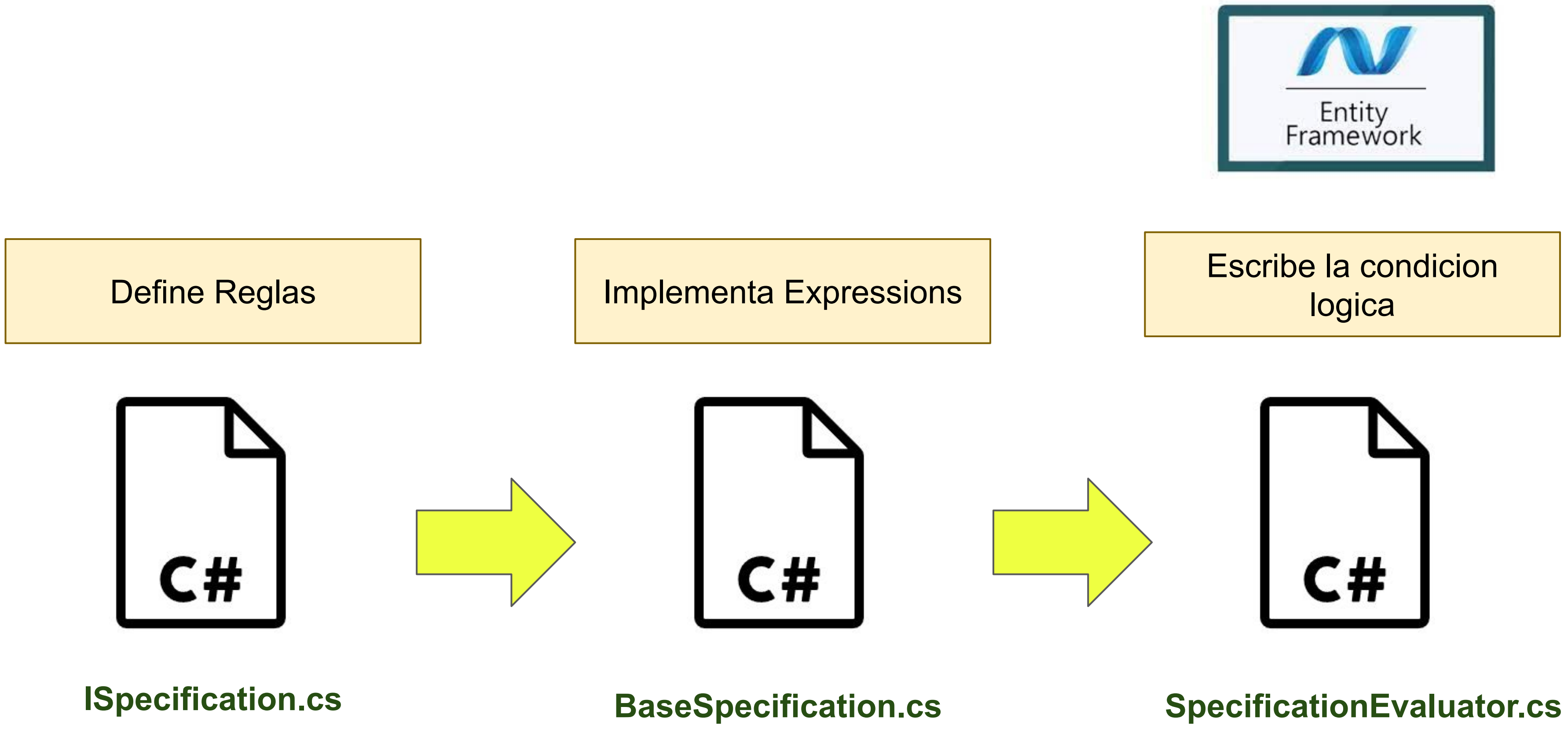
DriverLicenseSpecification.cs

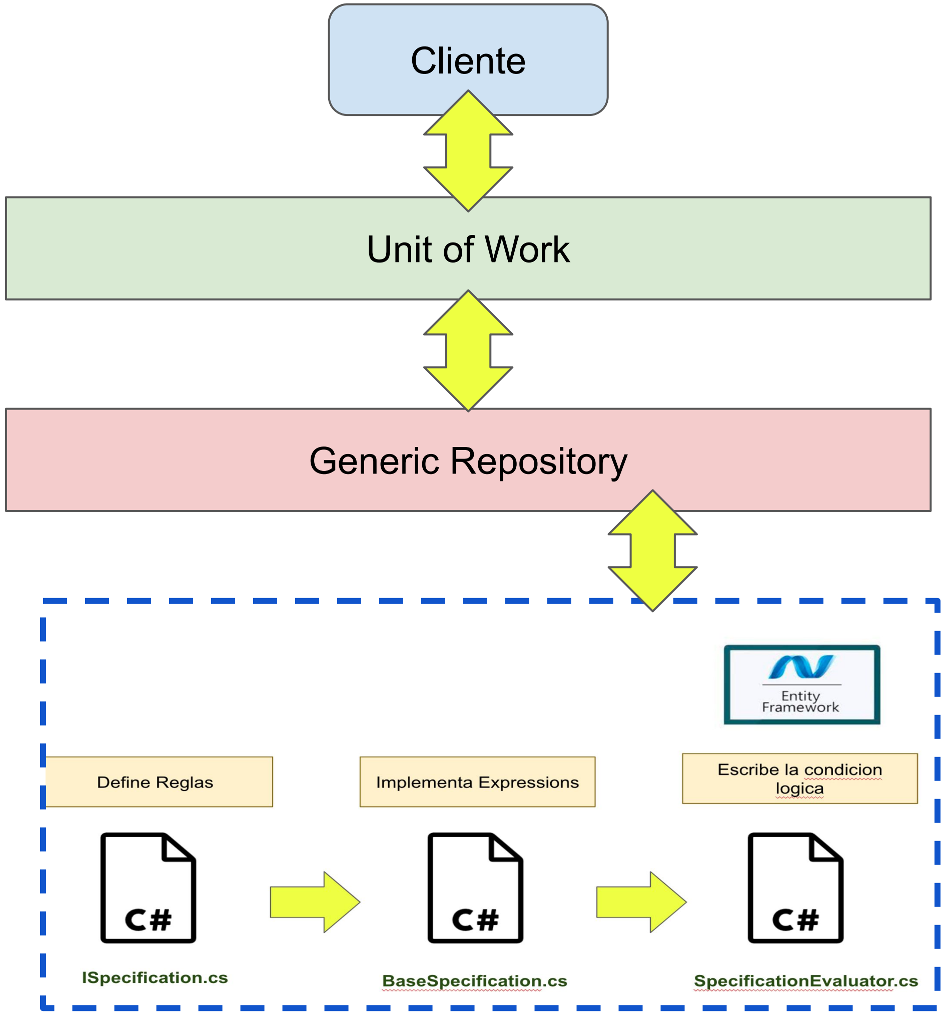
Entity Framework

Agregar criterio logico a una consulta de datos

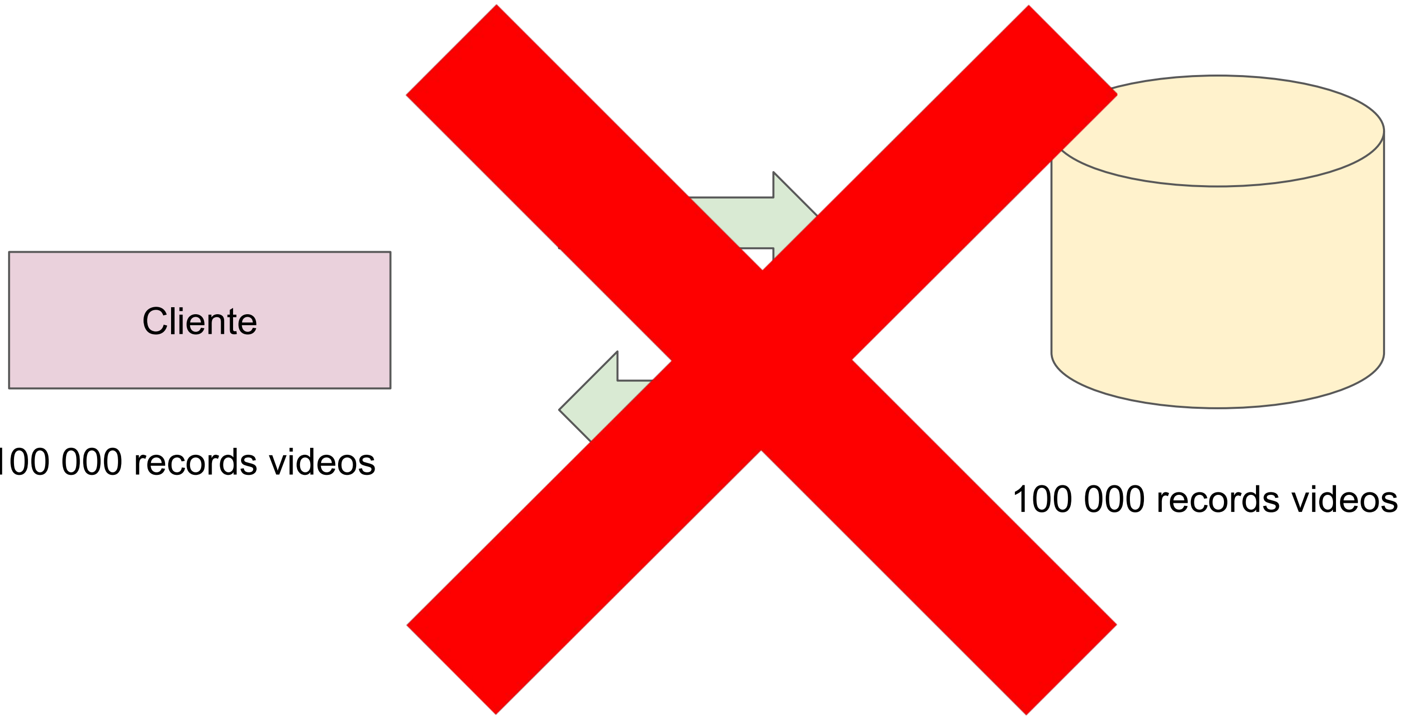
Ordenar resultados

Agregar multiples entidades a una consulta de datos

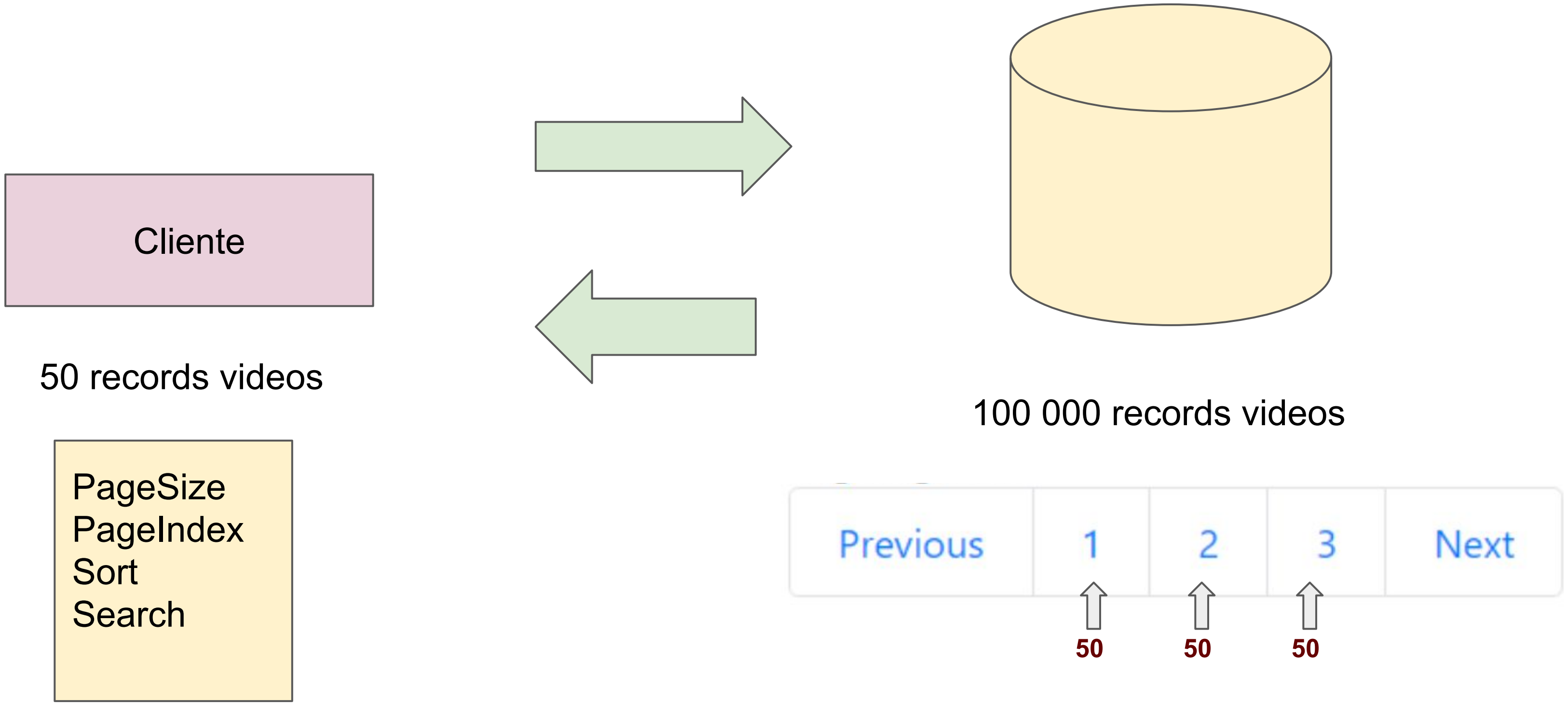

Entity Framework
Define Reglas
Implementa Expressions
Escribe la condicion logica
C#
C#
C#
ISpecification.cs
BaseSpecification.cs
SpecificationEvaluator.cs

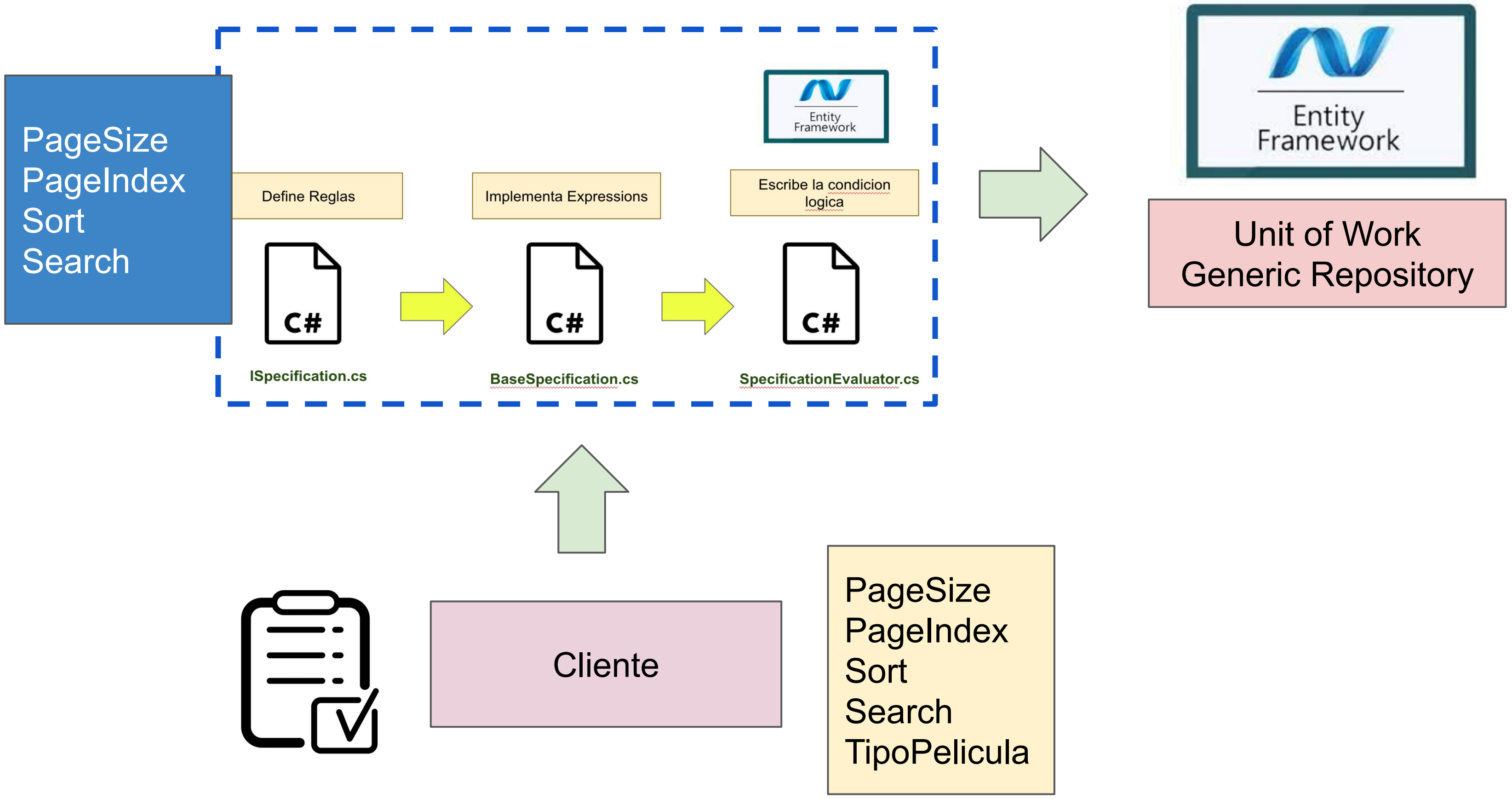

Cliente
Unit of Work
Generic Repository
Entity Framework
Define Reglas
Implementa Expressions
Escribe la condicion logica
ISpecification.cs
BaseSpecification.cs
SpecificationEvaluator.cs

Pagination con Specification

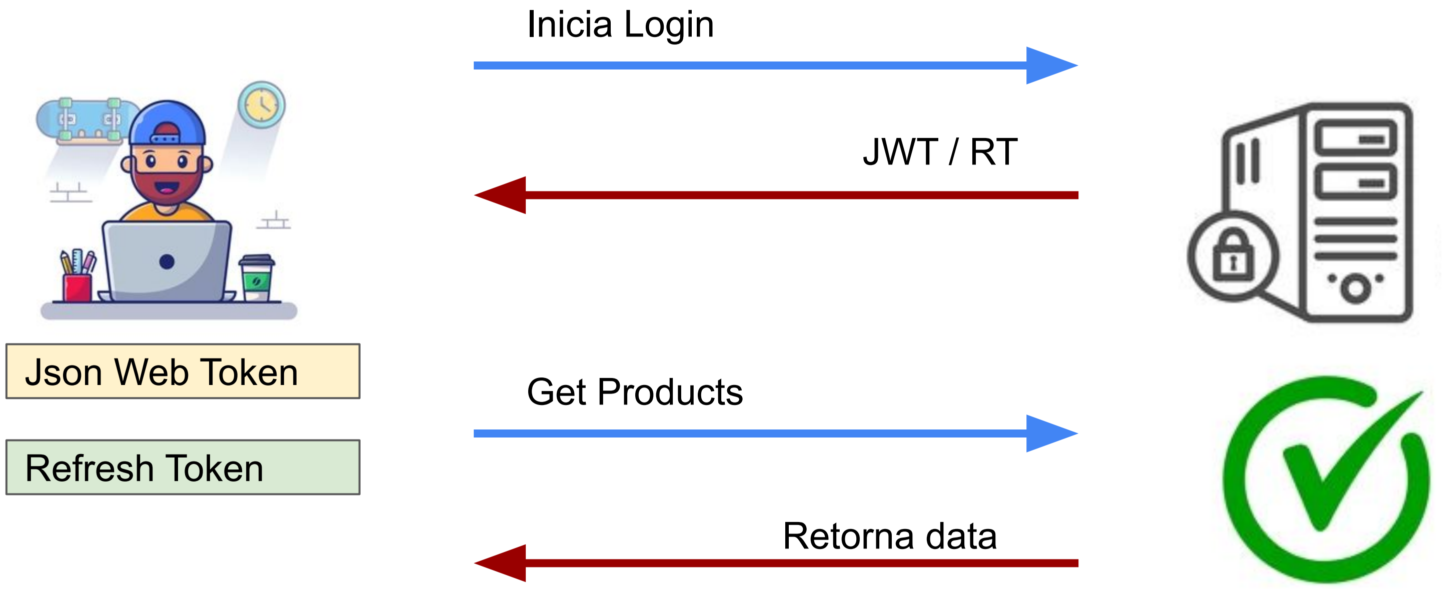


Pagination con Specification

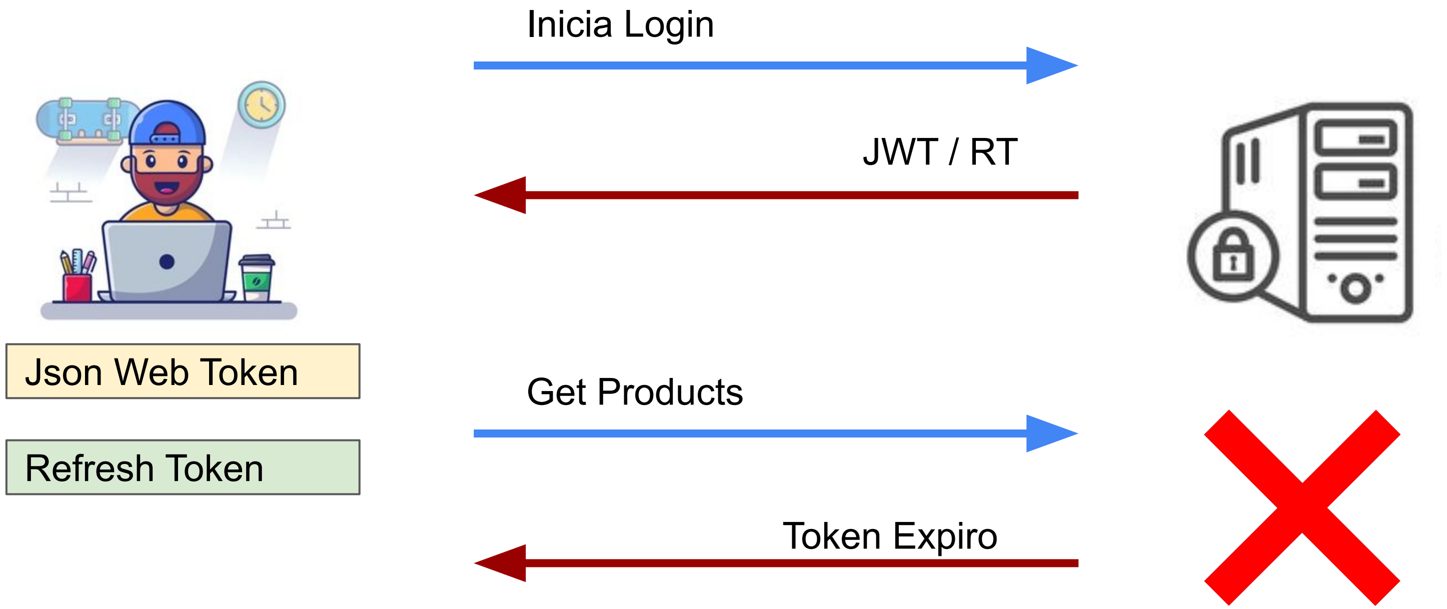



PageSize
PageIndex
Sort
Search
Define Reglas
Implementa Expressions
Escribe la condicion logica
Entity Framework
C#
C#
C#
ISpecification.cs
BaseSpecification.cs
SpecificationEvaluator.cs
Entity Framework
Unit of Work
Generic Repository
Cliente
PageSize
PageIndex
Sort
Search
TipoPelicula

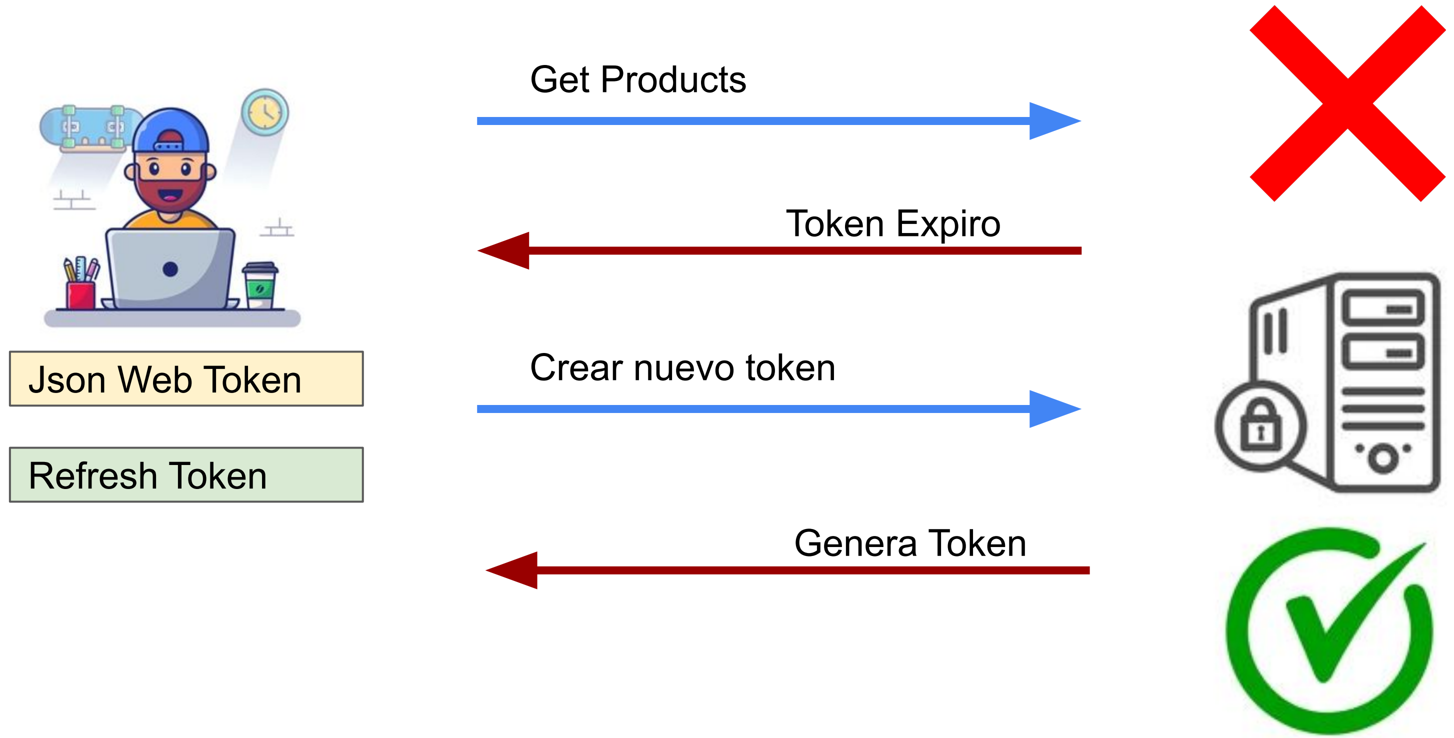
Refresh Token



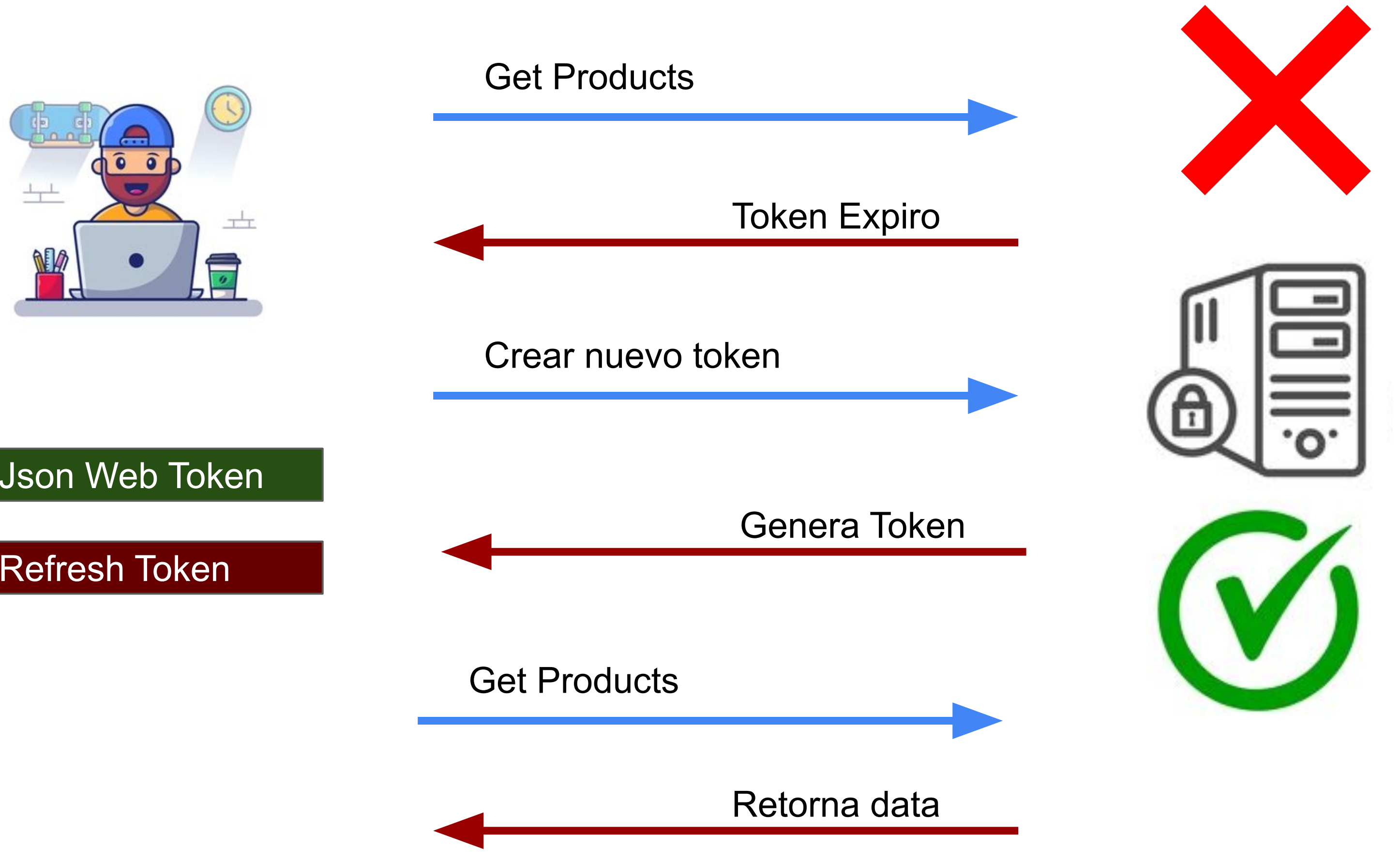
Refresh Token



Refresh Token



Refresh Token



Json Web Token

Refresh Token

Que es GraphQL?

Rest vs GraphQL

Solucion

Rest Api Request

https://facebook.com/user/id

https://facebook.com/user/id/amigos

https://facebook.com/user/id/messages

https://facebook.com/user/id/notifications

https://facebook.com/user/id/new-feeds

```
Query {
        User (id:1001){
                nombre
                amigos {
                 nombre
                }
        }
}
```

```
"Data": {
        "User": {
                "nombre": "Vaxi",
                "Amigos": [
                        {
                        "nombre":"ale"
                        },
                        {
                        "nombre":"ryan"
                        },
                ]
        }
}
```

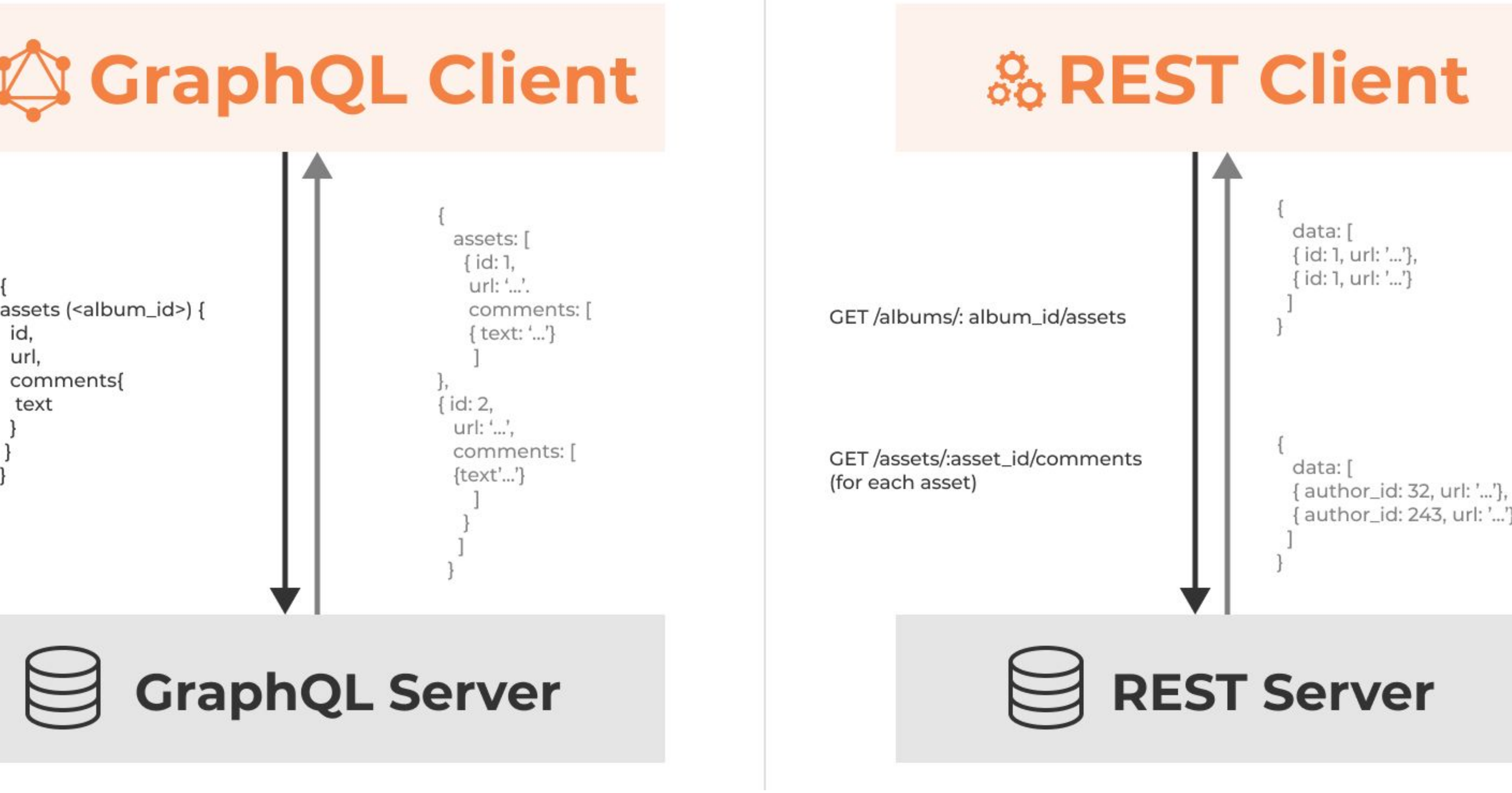
GraphQL vs Rest

```
GET /books/:id
GET /authors/:id
GET /books/:id/comments
POST /books/:id/comments
```

```
type Query {
  book(id: ID!): Book
  author(id: ID!): Author
}

type Mutation {
  addComment(input: AddCommentInput): Comment
}

type Book { ... }
type Author { ... }
type Comment { ... }
input AddCommentInput { ... }
```



```
{
assets (<album_id>) {
  id,
  url,
  comments{
   text
  }
 }
}
```

```
{
  assets: [
   { id: 1,
    url: '...'.
    comments: [
    { text: '...'}
     ]
},
{ id: 2,
  url: '...',
  comments: [
  {text'...'}
     ]
    }
   ]
  }
```

GET /albums/: album_id/assets

```
{
  data: [
  { id: 1, url: '...'},
  { id: 1, url: '...'}
  ]
}
```

GET /assets/:asset_id/comments
(for each asset)

```
{
  data: [
  { author_id: 32, url: '...'},
  { author_id: 243, url: '...'}
  ]
}
```

GraphQL Components

Schema

Types

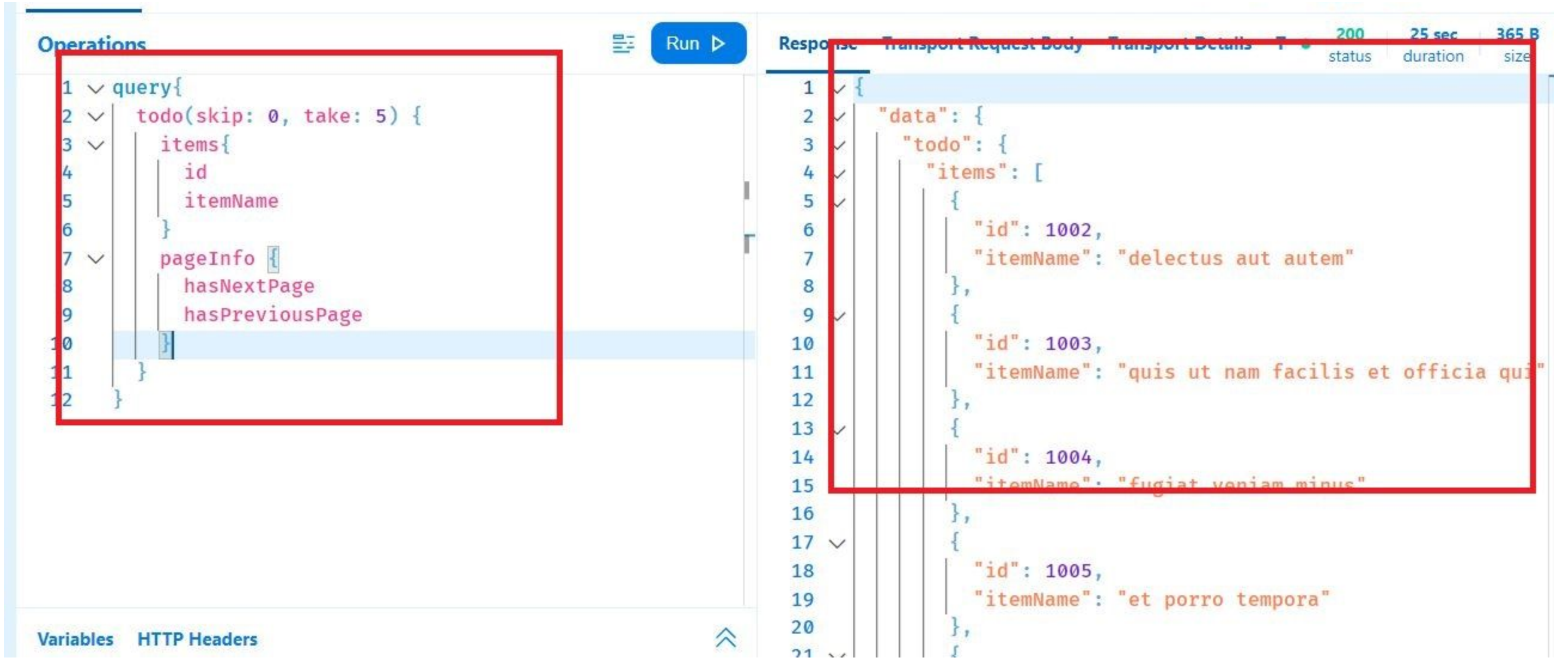
Resolvers

Query

Mutation

Subscription

GraphQL - Pagination



```
query{
  todo(skip: 0, take: 5) {
    items{
      id
      itemName
    }
    pageInfo {
      hasNextPage
      hasPreviousPage
    }
  }
}
```

```
{
  "data": {
    "todo": {
      "items": [
        {
          "id": 1002,
          "itemName": "delectus aut autem"
        },
        {
          "id": 1003,
          "itemName": "quis ut nam facilis et officia qui"
        },
        {
          "id": 1004,
          "itemName": "fugiat veniam minus"
        },
        {
          "id": 1005,
          "itemName": "et porro tempora"
        },
        {
```