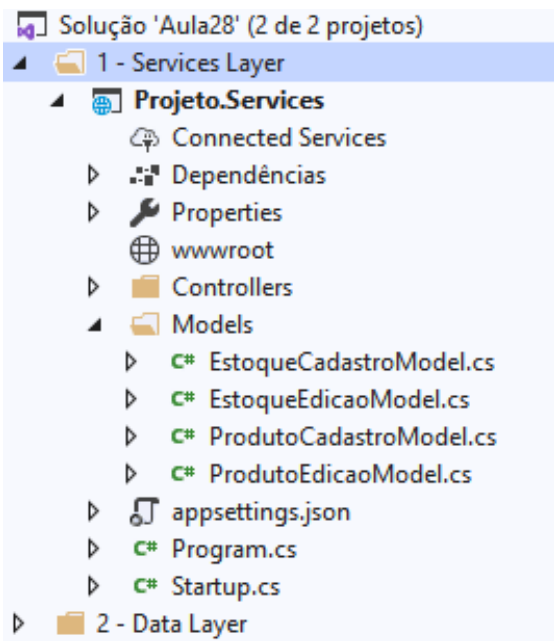


Criando classes de modelo para cadastro e edição de produto e estoque:



/Models/EstoqueCadastroModel.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.ComponentModel.DataAnnotations;

namespace Projeto.Services.Models
{
    public class EstoqueCadastroModel
    {
        [Required(ErrorMessage = "Informe o nome do estoque.")]
        public string Nome { get; set; }
    }
}
```

/Models/EstoqueEdicaoModel.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.ComponentModel.DataAnnotations; //validações

namespace Projeto.Services.Models
{
    public class EstoqueEdicaoModel
    {
        [Required(ErrorMessage = "Informe o Id do Estoque.")]
        public int IdEstoque { get; set; }
    }
}
```

```
        [Required(ErrorMessage = "Informe o Nome do Estoque.")]
        public string Nome { get; set; }
    }
}
```

/Models/ProdutoCadastroModel.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.ComponentModel.DataAnnotations; //validações

namespace Projeto.Services.Models
{
    public class ProdutoCadastroModel
    {
        [Required(ErrorMessage = "Informe o Nome do Produto.")]
        public string Nome { get; set; }

        [Required(ErrorMessage = "Informe o Preço do Produto.")]
        public decimal Preco { get; set; }

        [Required(ErrorMessage = "Informe a Quantidade do Produto.")]
        public int Quantidade { get; set; }

        [Required(ErrorMessage = "Informe o Id do Estoque do Produto.")]
        public int IdEstoque { get; set; }
    }
}
```

/Models/ProdutoEdicaoModel.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.ComponentModel.DataAnnotations; //validações

namespace Projeto.Services.Models
{
    public class ProdutoEdicaoModel
    {
        [Required(ErrorMessage = "Informe o Id do Produto.")]
        public int IdProduto { get; set; }

        [Required(ErrorMessage = "Informe o Nome do Produto.")]
        public string Nome { get; set; }

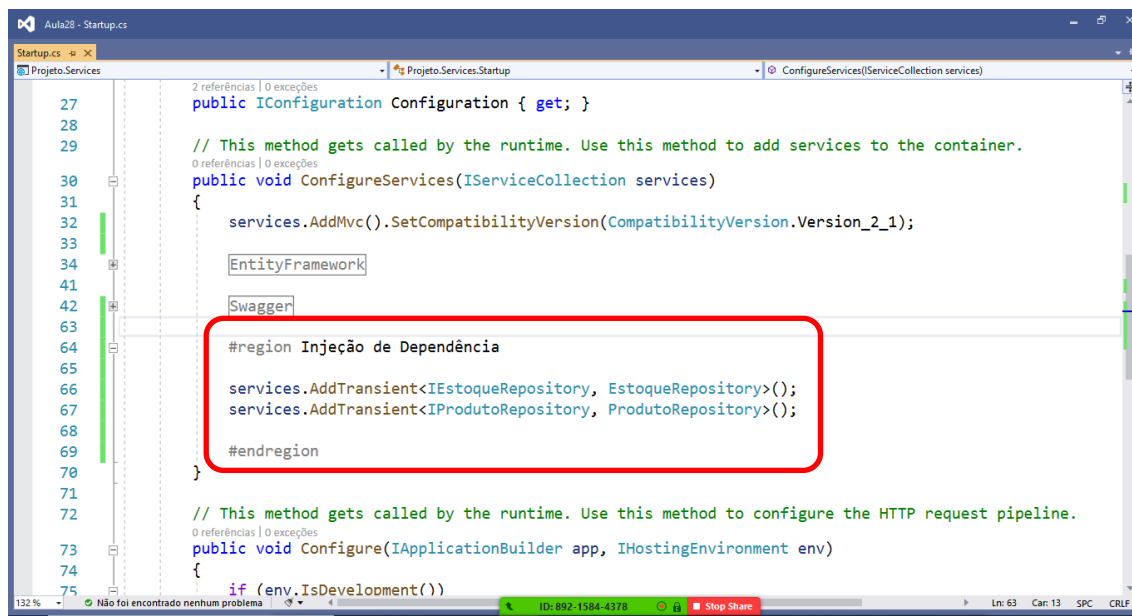
        [Required(ErrorMessage = "Informe o Preço do Produto.")]
        public decimal Preco { get; set; }

        [Required(ErrorMessage = "Informe a Quantidade do Produto.")]
        public int Quantidade { get; set; }

        [Required(ErrorMessage = "Informe o Id do Estoque do Produto.")]
        public int IdEstoque { get; set; }
    }
}
```

Startup.cs

Mapeamento de injeção de dependência para os repositórios de Estoque e Produto. (**EstoqueRepository** / **ProdutoRepository**)



```

27 public IConfiguration Configuration { get; }
28
29 // This method gets called by the runtime. Use this method to add services to the container.
30 public void ConfigureServices(IServiceCollection services)
31 {
32     services.AddMvc().SetCompatibilityVersion(CompatibilityVersion.Version_2_1);
33
34     [EntityFramework]
35
36     [Swagger]
37
38     #region Injeção de Dependência
39     services.AddTransient<IEstoqueRepository, EstoqueRepository>();
40     services.AddTransient<IProdutoRepository, ProdutoRepository>();
41
42     #endregion
43
44 }
45
46 // This method gets called by the runtime. Use this method to configure the HTTP request pipeline.
47 public void Configure(IApplicationBuilder app, IHostingEnvironment env)
48 {
49     if (env.IsDevelopment())
50     {
51         app.UseDeveloperExceptionPage();
52     }
53     else
54     {
55         app.UseExceptionHandler("/Home/Error");
56         // The default HSTS value is 30 days. You must do this before running the application.
57         app.UseHsts(30);
58     }
59
60     app.UseStaticFiles();
61     app.UseCookiePolicy();
62
63     app.UseMvc();
64 }

```

#region Injeção de Dependência

```

services.AddTransient<IEstoqueRepository, EstoqueRepository>();
services.AddTransient<IProdutoRepository, ProdutoRepository>();

```

```
#endregion
```

/Controllers/EstoqueController.cs



```

9 namespace Projeto.Services.Controllers
10 {
11     [Route("api/[controller]")]
12     [ApiController]
13     public class EstoqueController : ControllerBase
14     {
15         //atributo
16         private readonly IEstoqueRepository estoqueRepository;
17
18         //construtor para injeção de dependência
19         public EstoqueController(IEstoqueRepository estoqueRepository)
20         {
21             this.estoqueRepository = estoqueRepository;
22         }
23
24         [HttpPost]
25         public IActionResult Post()
26         {
27             return Ok();
28         }
29     }
30 }

```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Projeto.Data.Contracts;

namespace Projeto.Services.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class EstoqueController : ControllerBase
    {
        //atributo
        private readonly IEstoqueRepository estoqueRepository;

        //construtor para injeção de dependência
        public EstoqueController(IEstoqueRepository estoqueRepository)
        {
            this.estoqueRepository = estoqueRepository;
        }

        [HttpPost]
        public IActionResult Post()
        {
            return Ok();
        }

        [HttpPut]
        public IActionResult Put()
        {
            return Ok();
        }

        [HttpDelete("{id}")]
        public IActionResult Delete(int id)
        {
            return Ok();
        }

        [HttpGet]
        public IActionResult GetAll()
        {
            return Ok();
        }

        [HttpGet("{id}")]
        public IActionResult GetById(int id)
        {
            return Ok();
        }
    }
}
```

/Controllers/ProdutoController.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Projeto.Data.Contracts;

namespace Projeto.Services.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class ProdutoController : ControllerBase
    {
        //atributo
        private readonly IProdutoRepository produtoRepository;

        //construtor para injeção de dependência
        public ProdutoController(IProdutoRepository produtoRepository)
        {
            this.produtoRepository = produtoRepository;
        }

        [HttpPost]
        public IActionResult Post()
        {
            return Ok();
        }

        [HttpPut]
        public IActionResult Put()
        {
            return Ok();
        }

        [HttpDelete("{id}")]
        public IActionResult Delete(int id)
        {
            return Ok();
        }

        [HttpGet]
        public IActionResult GetAll()
        {
            return Ok();
        }

        [HttpGet("{id}")]
        public IActionResult GetById(int id)
        {
            return Ok();
        }
    }
}
```

Desenvolvendo o controller: **EstoqueController.cs**
POST, PUT, DELETE, GET

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Projeto.Data.Contracts;
using Projeto.Data.Entities;
using Projeto.Services.Models;

namespace Projeto.Services.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class EstoqueController : ControllerBase
    {
        //atributo
        private readonly IEstoqueRepository estoqueRepository;

        //construtor para injeção de dependência
        public EstoqueController(IEstoqueRepository estoqueRepository)
        {
            this.estoqueRepository = estoqueRepository;
        }

        [HttpPost]
        public IActionResult Post(EstoqueCadastroModel model)
        {
            //verificando se os campos da model passaram nas validações
            if(ModelState.IsValid)
            {
                try
                {
                    var estoque = new Estoque();
                    estoque.Nome = model.Nome;

                    estoqueRepository.Inserir(estoque);

                    var result = new
                    {
                        message = "Estoque cadastrado com sucesso",
                        estoque
                    };

                    return Ok(result); //HTTP 200 (SUCESSO!)
                }
                catch(Exception e)
                {
                    return StatusCode(500, "Erro: " + e.Message);
                }
            }
            else
            {
                //Erro HTTP 400 (BAD REQUEST)
                return BadRequest("Ocorreram erros de validação.");
            }
        }
    }
}
```

```
[HttpPut]
public IActionResult Put(EstoqueEdicaoModel model)
{
    //verificando se os campos da model passaram nas validações
    if (ModelState.IsValid)
    {
        try
        {
            var estoque = new Estoque();
            estoque.IdEstoque = model.IdEstoque;
            estoque.Nome = model.Nome;

            estoqueRepository.Alterar(estoque);

            var result = new
            {
                message = "Estoque atualizado com sucesso",
                estoque
            };

            return Ok(result); //HTTP 200 (SUCESSO!)
        }
        catch (Exception e)
        {
            return StatusCode(500, "Erro: " + e.Message);
        }
    }
    else
    {
        //Erro HTTP 400 (BAD REQUEST)
        return BadRequest("Ocorreram erros de validação.");
    }
}

[HttpDelete("{id}")]
public IActionResult Delete(int id)
{
    try
    {
        //buscar o estoque referente ao id informado..
        var estoque = estoqueRepository.ObterPorId(id);

        //verificar se o estoque foi encontrado..
        if(estoque != null)
        {
            //excluindo o estoque
            estoqueRepository.Excluir(estoque);

            var result = new
            {
                message = "Estoque excluído com sucesso.",
                estoque
            };

            return Ok(result);
        }
        else
        {
            return BadRequest("Estoque não encontrado.");
        }
    }
}
```

```
        catch(Exception e)
        {
            return StatusCode(500, "Erro: " + e.Message);
        }
    }

    [HttpGet]
    public IActionResult GetAll()
    {
        try
        {
            var result = estoqueRepository.Consultar();
            return Ok(result);
        }
        catch(Exception e)
        {
            return StatusCode(500, "Erro: " + e.Message);
        }
    }

    [HttpGet("{id}")]
    public IActionResult GetById(int id)
    {
        try
        {
            var result = estoqueRepository.ObterPorId(id);

            if(result != null) //se o estoque foi encontrado..
            {
                return Ok(result);
            }
            else
            {
                return NoContent(); //HTTP 204 (SUCESSO -> Vazio)
            }
        }
        catch (Exception e)
        {
            return StatusCode(500, "Erro: " + e.Message);
        }
    }
}
```

Desenvolvendo o controller: **ProdutoController.cs**
POST, PUT, DELETE, GET

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Projeto.Data.Contracts;
using Projeto.Data.Entities;
using Projeto.Services.Models;

namespace Projeto.Services.Controllers
{
```



```
[Route("api/[controller]")]
[ApiController]
public class ProdutoController : ControllerBase
{
    //atributo
    private readonly IProdutoRepository produtoRepository;

    //construtor para injeção de dependência
    public ProdutoController(IProdutoRepository produtoRepository)
    {
        this.produtoRepository = produtoRepository;
    }

    [HttpPost]
    public IActionResult Post(ProdutoCadastroModel model)
    {
        if(ModelState.IsValid)
        {
            try
            {
                var produto = new Produto();
                produto.Nome = model.Nome;
                produto.Preco = model.Preco;
                produto.Quantidade = model.Quantidade;
                produto.IdEstoque = model.IdEstoque;

                produtoRepository.Inserir(produto);

                var result = new
                {
                    message = "Produto cadastrado com sucesso.",
                    produto
                };

                return Ok(result);
            }
            catch(Exception e)
            {
                return StatusCode(500, "Erro: " + e.Message);
            }
        }
        else
        {
            return BadRequest("Ocorreram erros de validação.");
        }
    }

    [HttpPut]
    public IActionResult Put(ProdutoEdicaoModel model)
    {
        if (ModelState.IsValid)
        {
            try
            {
                var produto = new Produto();
                produto.IdProduto = model.IdProduto;
                produto.Nome = model.Nome;
                produto.Preco = model.Preco;
                produto.Quantidade = model.Quantidade;
                produto.IdEstoque = model.IdEstoque;
            }
        }
    }
}
```

```
        produtoRepository.Alterar(produto);

        var result = new
        {
            message = "Produto atualizado com sucesso.",
            produto
        };

        return Ok(result);
    }
    catch (Exception e)
    {
        return StatusCode(500, "Erro: " + e.Message);
    }
}
else
{
    return BadRequest("Ocorreram erros de validação.");
}
}

[HttpDelete("{id}")]
public IActionResult Delete(int id)
{
    try
    {
        var produto = produtoRepository.ObterPorId(id);

        if(produto != null)
        {
            produtoRepository.Excluir(produto);

            var result = new
            {
                message = "Produto excluído com sucesso.",
                produto
            };

            return Ok(result);
        }
        else
        {
            return BadRequest("Produto não encontrado.");
        }
    }
    catch (Exception e)
    {
        return StatusCode(500, "Erro: " + e.Message);
    }
}

[HttpGet]
public IActionResult GetAll()
{
    try
    {
        var result = produtoRepository.Consultar();
        return Ok(result);
    }
    catch (Exception e)
    {

```

```

        return StatusCode(500, "Erro: " + e.Message);
    }
}

[HttpGet("{id}")]
public IActionResult GetById(int id)
{
    try
    {
        var result = produtoRepository.ObterPorId(id);

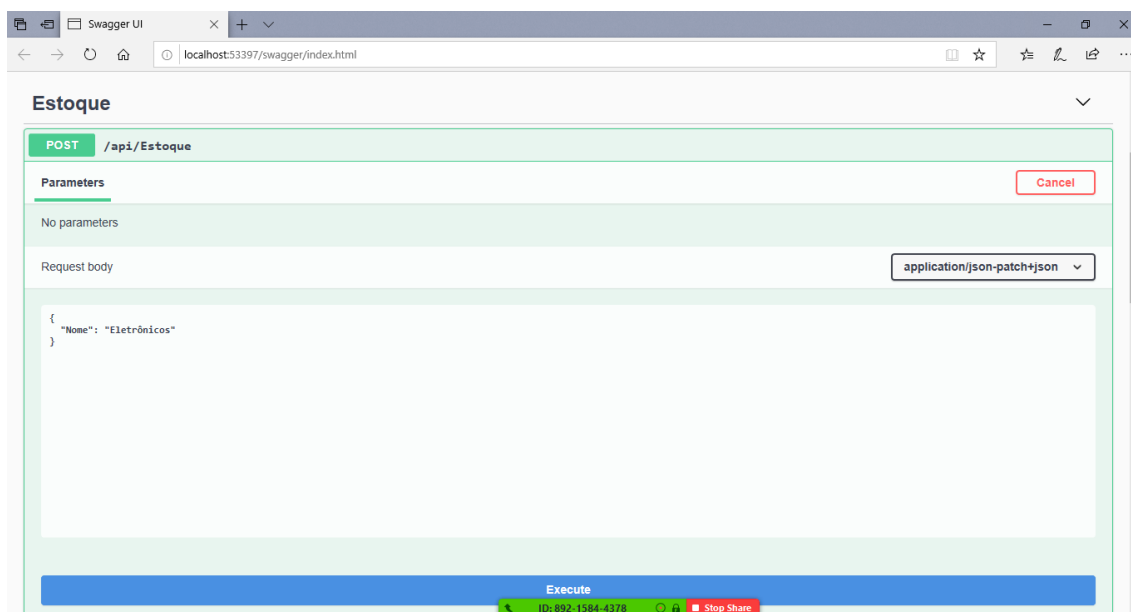
        if(result != null)
        {
            return Ok(result);
        }
        else
        {
            return NoContent();
        }
    }
    catch (Exception e)
    {
        return StatusCode(500, "Erro: " + e.Message);
    }
}
}
}
}

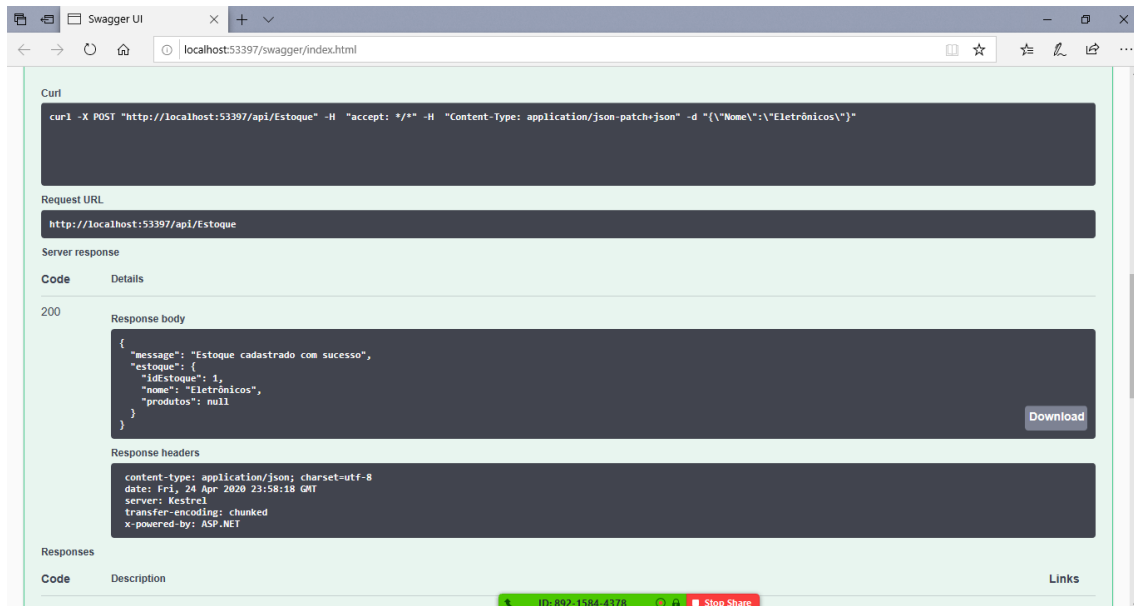
```

Executando:

POST /api/Estoque

Cadastro de estoque





Swagger UI interface showing a successful POST request to `http://localhost:53397/api/Estoque`. The response body is:

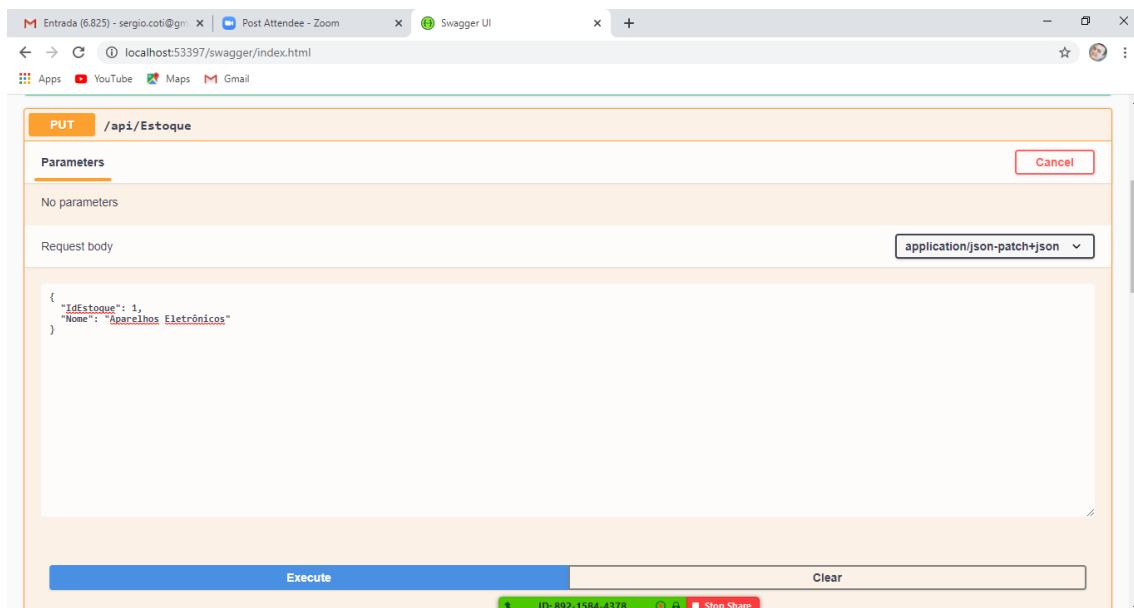
```
{
  "message": "Estoque cadastrado com sucesso",
  "estoque": {
    "idEstoque": 1,
    "nome": "Eletrônicos",
    "produtos": null
  }
}
```

The response headers include:

```
content-type: application/json; charset=utf-8
date: Fri, 24 Apr 2020 23:58:18 GMT
server: Kestrel
transfer-encoding: chunked
x-powered-by: ASP.NET
```

PUT /api/Estoque

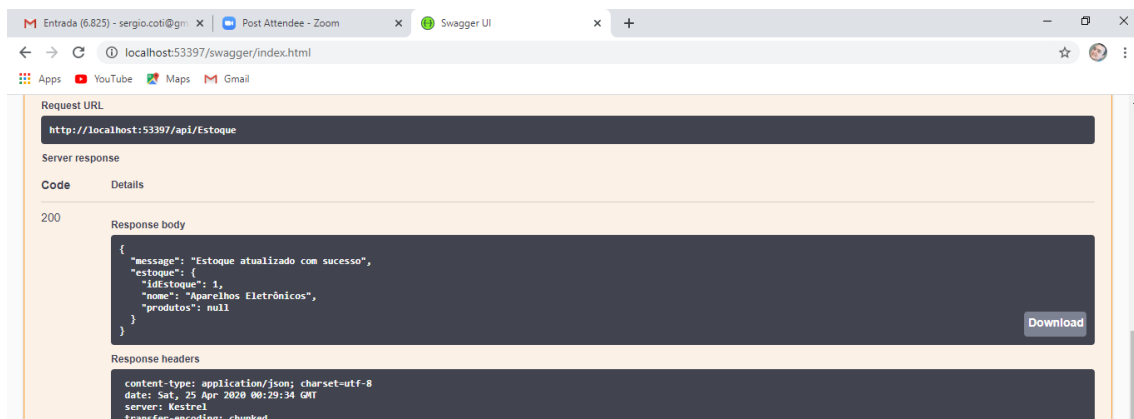
Edição de estoque



Swagger UI interface showing the configuration for the PUT `/api/Estoque` endpoint. The request body is:

```
{
  "idEstoque": 1,
  "nome": "Aparelhos Eletrônicos"
}
```

The interface includes an "Execute" button to test the endpoint.



Swagger UI interface showing the successful response of the PUT `/api/Estoque` endpoint. The response body is:

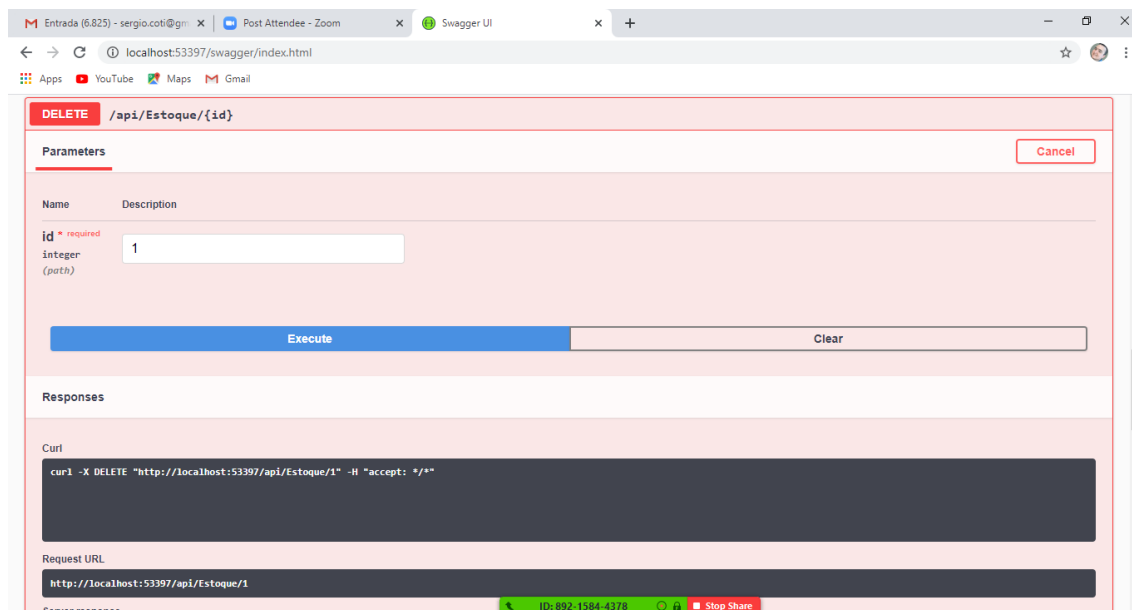
```
{
  "message": "Estoque atualizado com sucesso",
  "estoque": {
    "idEstoque": 1,
    "nome": "Aparelhos Eletrônicos",
    "produtos": null
  }
}
```

The response headers include:

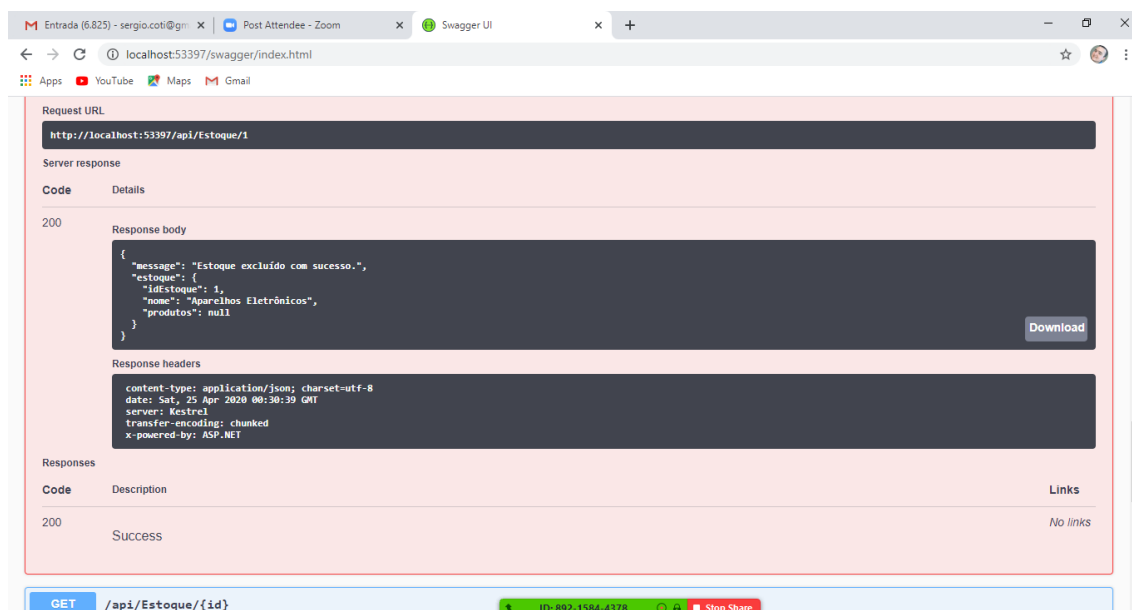
```
content-type: application/json; charset=utf-8
date: Sat, 25 Apr 2020 00:29:34 GMT
server: Kestrel
transfer-encoding: chunked
```

DELETE /api/Estoque/{id}

Exclusão de estoque



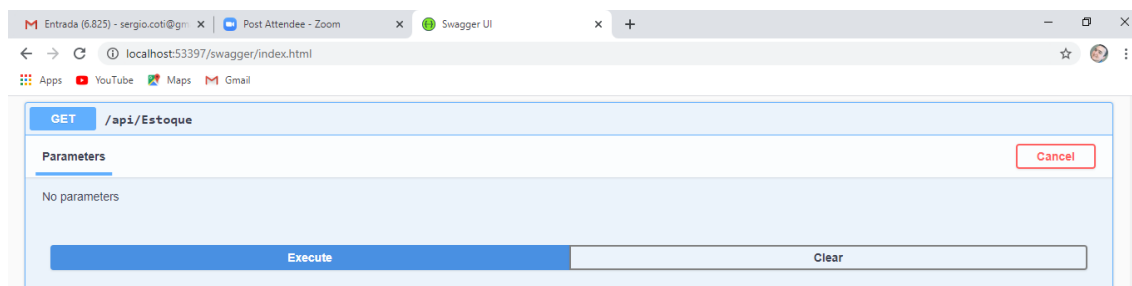
Swagger UI interface for the DELETE /api/Estoque/{id} endpoint. The interface shows the method DELETE, the path /api/Estoque/{id}, and a parameter id of type integer (path) with a value of 1. The Execute button is highlighted. The Responses section shows a 200 status code with a JSON response body: {"message": "Estoque excluído com sucesso.", "estoque": {"id": 1, "nome": "Aparelhos Eletrônicos", "produtos": null}}.



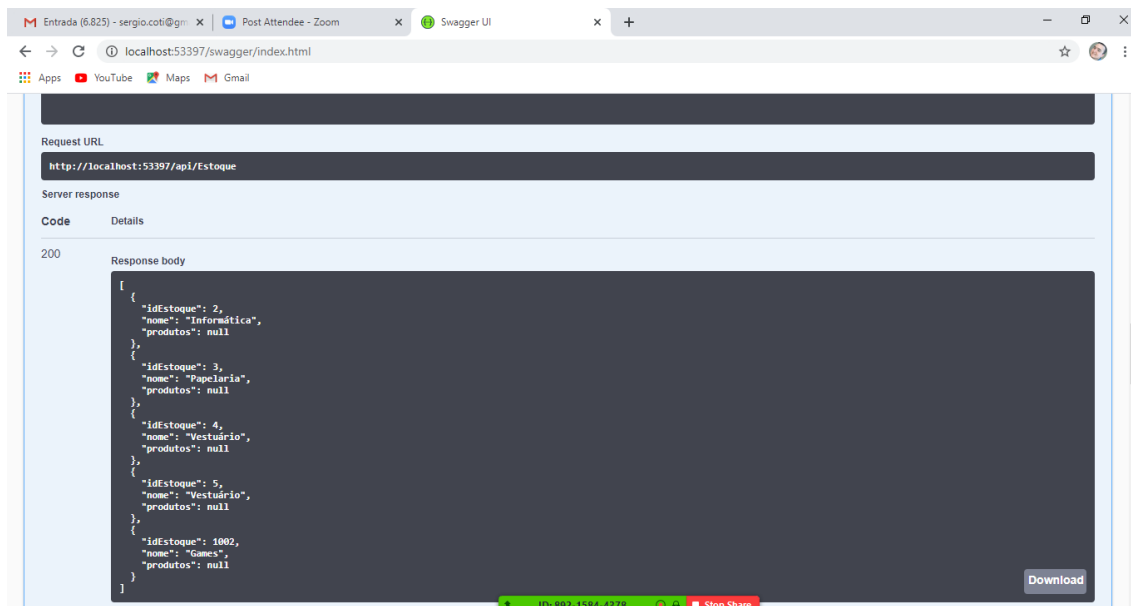
Swagger UI interface for the GET /api/Estoque/{id} endpoint. The interface shows the method GET, the path /api/Estoque/{id}, and no parameters. The Execute button is highlighted. The Responses section shows a 200 status code with a JSON response body: {"message": "Estoque excluído com sucesso.", "estoque": {"id": 1, "nome": "Aparelhos Eletrônicos", "produtos": null}}.

GET /api/Estoque/

Consultar todos os estoques

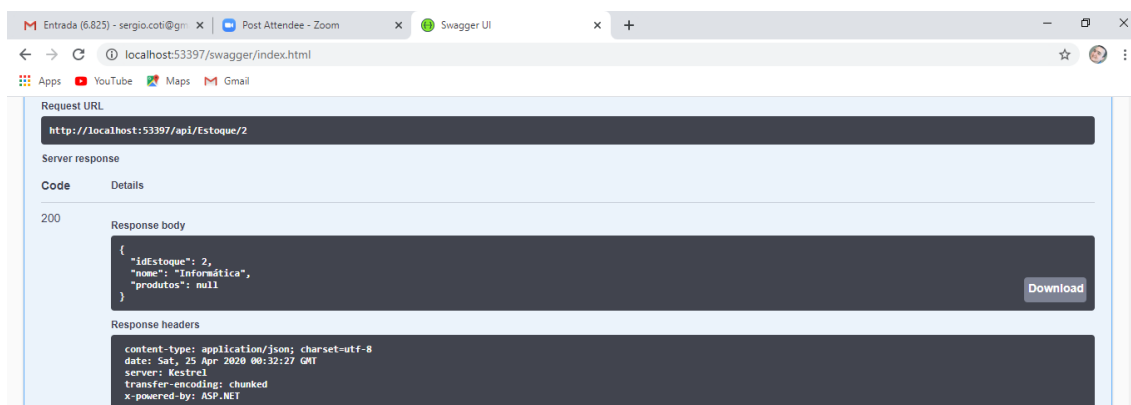
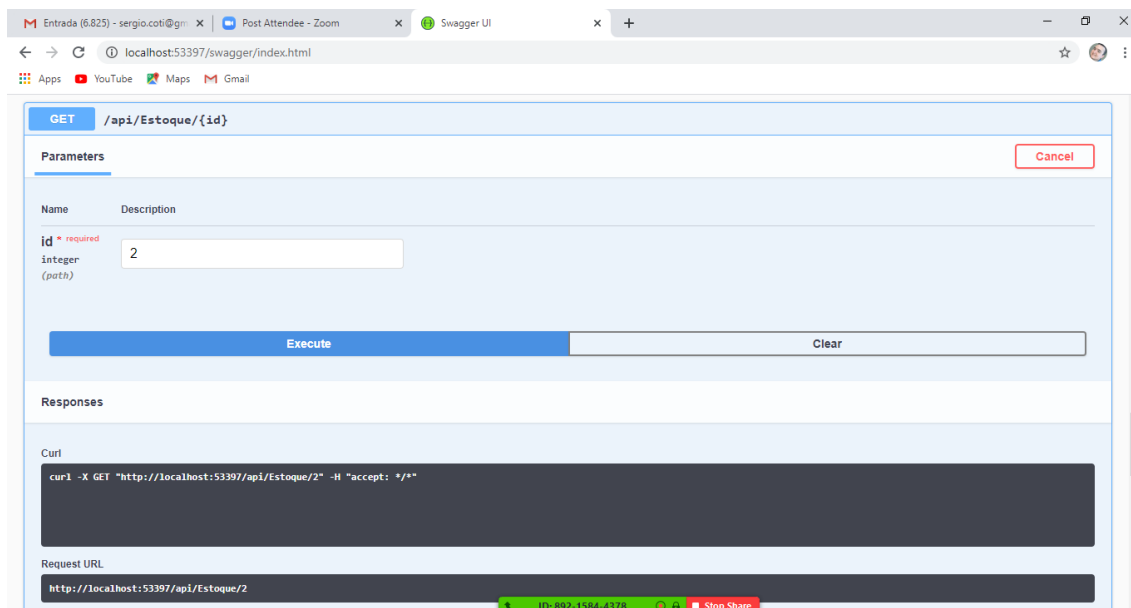


Swagger UI interface for the GET /api/Estoque/ endpoint. The interface shows the method GET, the path /api/Estoque/, and no parameters. The Execute button is highlighted.



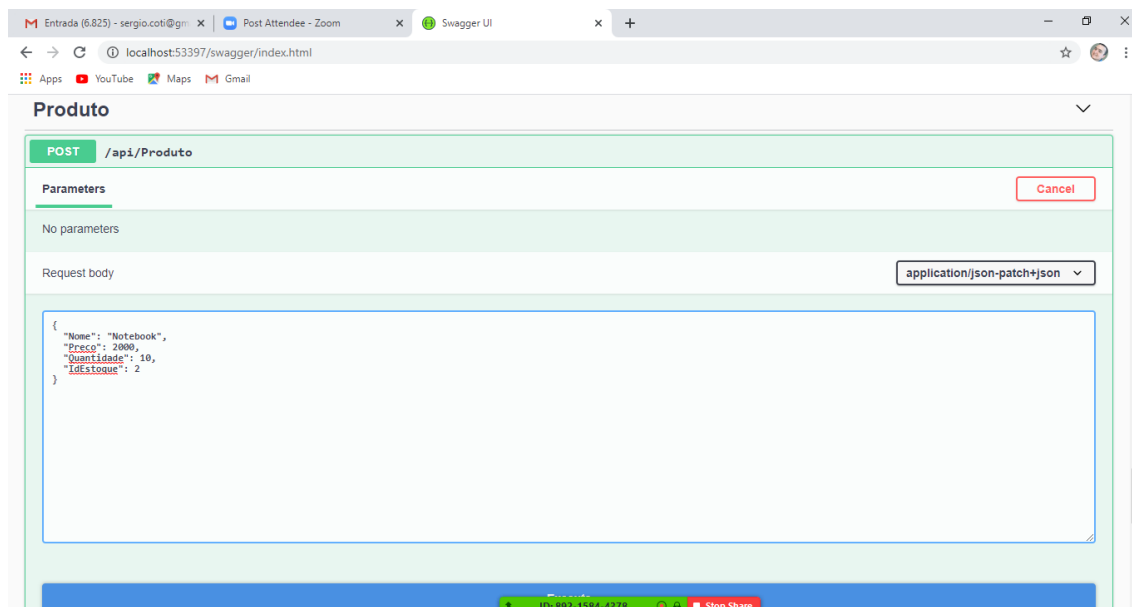
GET /api/Estoque/{id}

Consultar estoque pelo id



POST /api/Produto

Cadastro de produto



POST /api/Produto

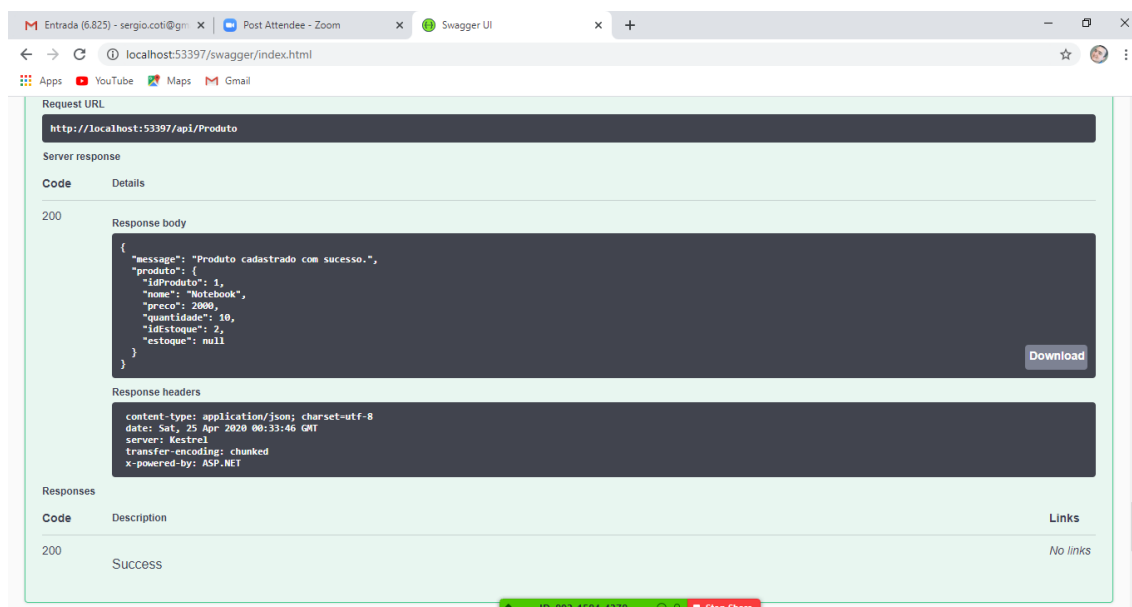
Parameters

No parameters

Request body

application/json-patch+json

```
{
  "Nome": "Notebook",
  "Preco": 2000,
  "Quantidade": 10,
  "Estoque": 2
}
```



Request URL

http://localhost:53397/api/Produto

Server response

Code

200

Response body

```
{
  "message": "Produto cadastrado com sucesso.",
  "produto": {
    "idProduto": 1,
    "nome": "Notebook",
    "preco": 2000,
    "quantidade": 10,
    "idEstoque": 2,
    "estoque": null
  }
}
```

Response headers

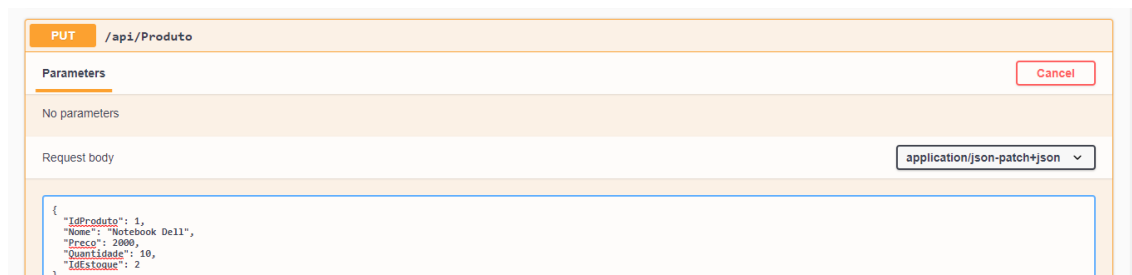
```
content-type: application/json; charset=utf-8
date: Sat, 25 Apr 2020 00:33:46 GMT
server: Kestrel
transfer-encoding: chunked
x-powered-by: ASP.NET
```

Responses

Code	Description	Links
200	Success	No links

PUT /api/Produto

Edição de produto



PUT /api/Produto

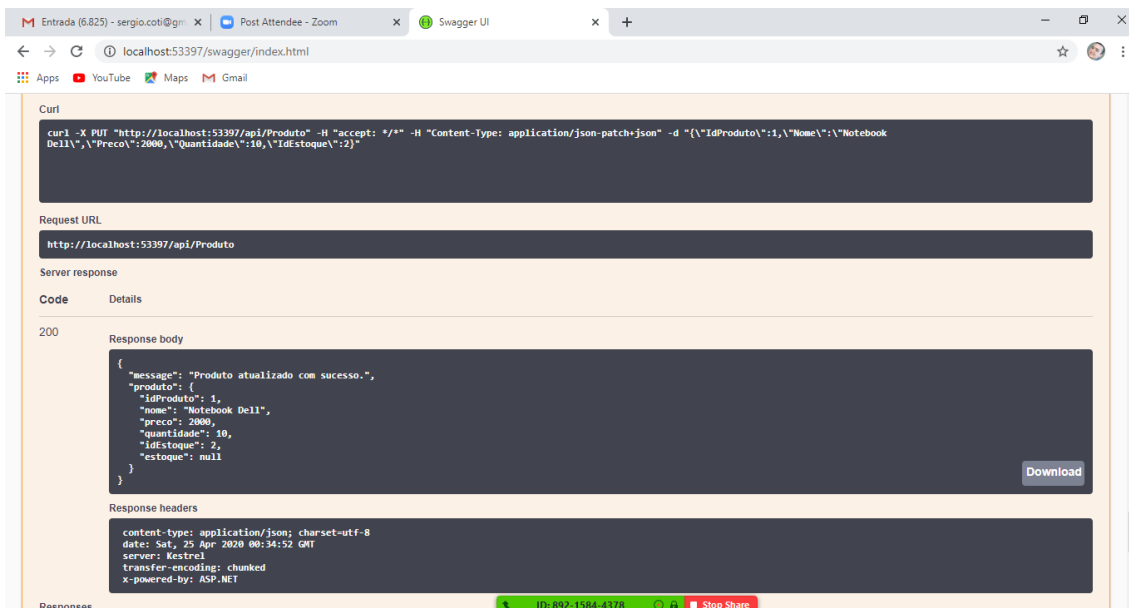
Parameters

No parameters

Request body

application/json-patch+json

```
{
  "idProduto": 1,
  "Nome": "Notebook Dell",
  "Preco": 2000,
  "Quantidade": 10,
  "Estoque": 2
}
```



Entrada (6.825) - sergio.coti@gmail.com x Post Attendee - Zoom x Swagger UI x +

localhost:53397/swagger/index.html

Apps YouTube Maps Gmail

Curl

```
curl -X PUT "http://localhost:53397/api/Produto" -H "accept: */*" -H "Content-Type: application/json-patch+json" -d '{"IdProduto":1,"Nome":"Notebook Dell","Preco":2000,"Quantidade":10,"IdEstoque":2}'
```

Request URL

http://localhost:53397/api/Produto

Server response

Code Details

200

Response body

```
{
  "message": "Produto atualizado com sucesso.",
  "produto": {
    "idProduto": 1,
    "nome": "Notebook Dell",
    "preco": 2000,
    "quantidade": 10,
    "idEstoque": 2,
    "estoque": null
  }
}
```

Download

Response headers

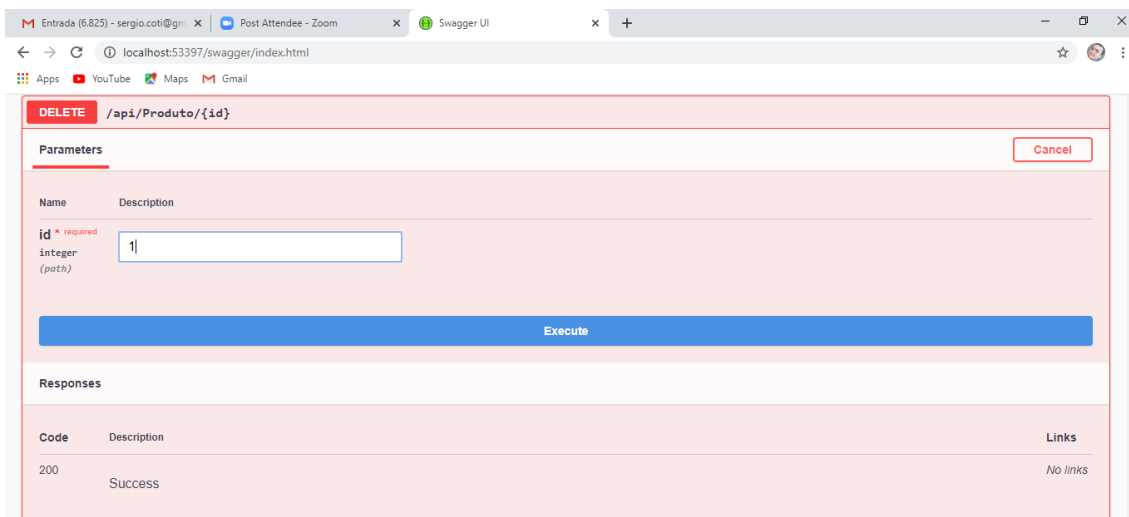
```
content-type: application/json; charset=utf-8
date: Sat, 25 Apr 2020 00:34:52 GMT
server: Kestrel
transfer-encoding: chunked
x-powered-by: ASP.NET
```

Responses

ID: 892-1584-4378 Stop Share

DELETE /api/Produto/{id}

Exclusão de produto



Entrada (6.825) - sergio.coti@gmail.com x Post Attendee - Zoom x Swagger UI x +

localhost:53397/swagger/index.html

Apps YouTube Maps Gmail

DELETE /api/Produto/{id}

Parameters

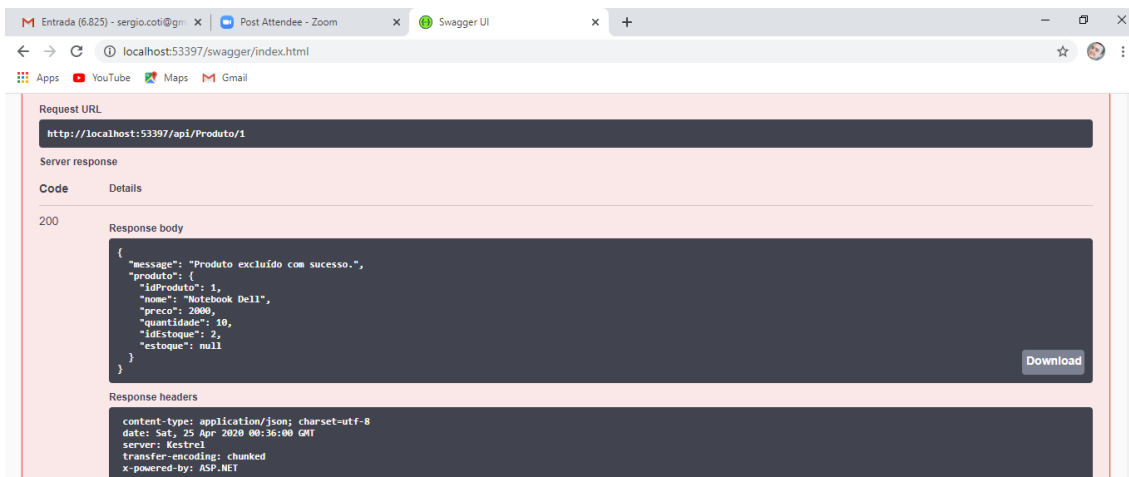
Cancel

Name	Description
id * required integer (path)	<input type="text" value="1"/>

Execute

Responses

Code	Description	Links
200	Success	No links



Entrada (6.825) - sergio.coti@gmail.com x Post Attendee - Zoom x Swagger UI x +

localhost:53397/swagger/index.html

Apps YouTube Maps Gmail

Request URL

http://localhost:53397/api/Produto/1

Server response

Code Details

200

Response body

```
{
  "message": "Produto excluído com sucesso.",
  "produto": {
    "idProduto": 1,
    "nome": "Notebook Dell",
    "preco": 2000,
    "quantidade": 10,
    "idEstoque": 2,
    "estoque": null
  }
}
```

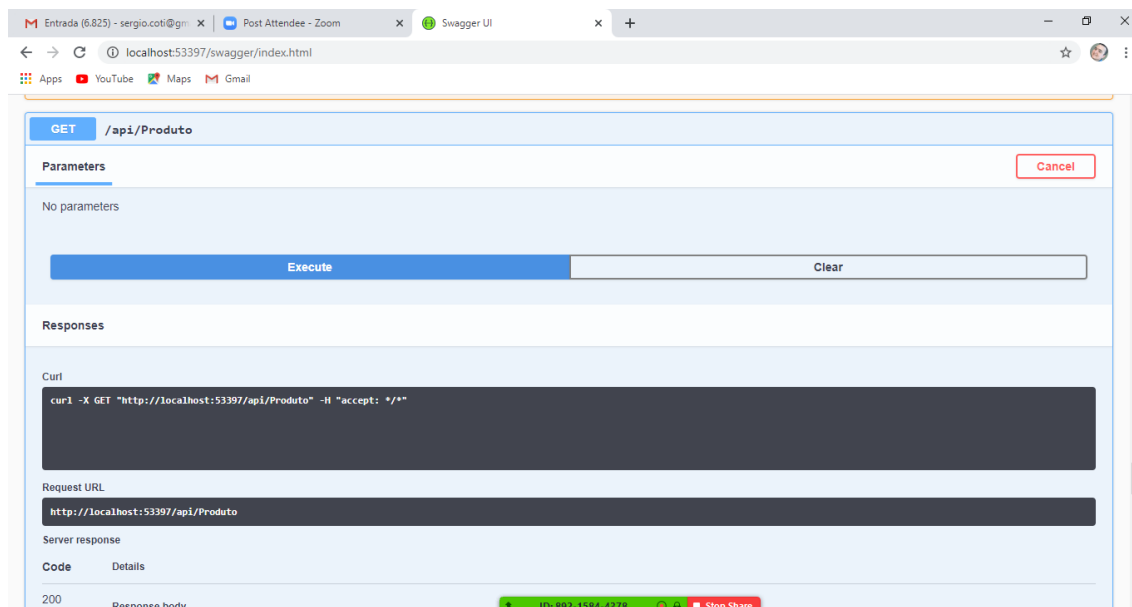
Download

Response headers

```
content-type: application/json; charset=utf-8
date: Sat, 25 Apr 2020 00:36:00 GMT
server: Kestrel
transfer-encoding: chunked
x-powered-by: ASP.NET
```


GET /api/Produto

Consulta de produto



GET /api/Produto

Parameters

No parameters

Execute Clear

Responses

Curl

```
curl -X GET "http://localhost:53397/api/Produto" -H "accept: */*"
```

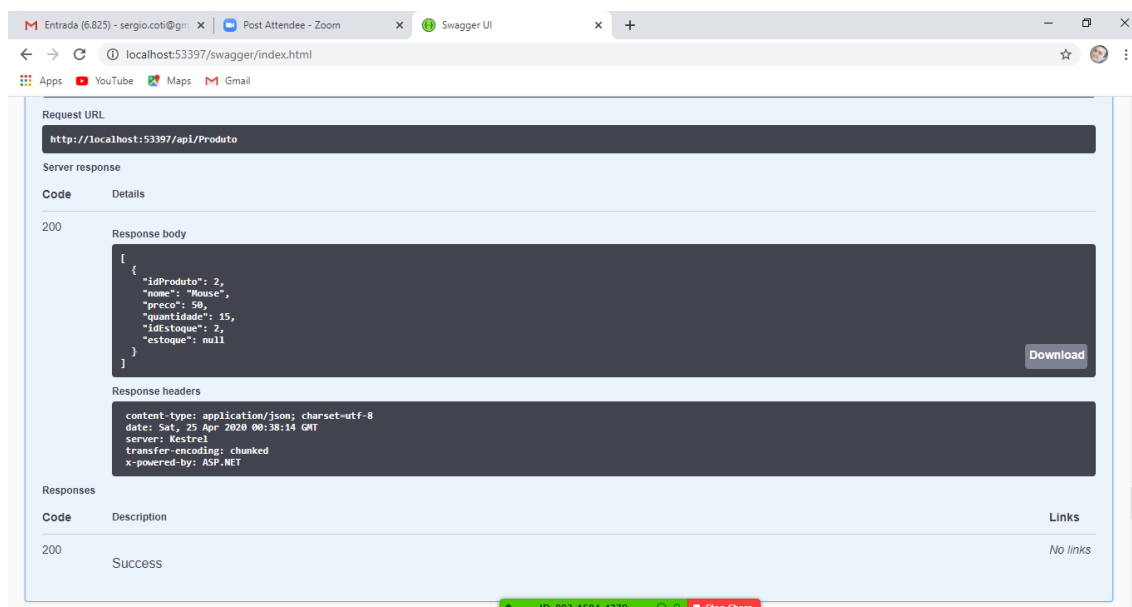
Request URL

http://localhost:53397/api/Produto

Server response

Code	Details
200	Response body

ID: 892-1584-4378 Stop Share



Request URL

http://localhost:53397/api/Produto

Server response

Code	Details
200	Response body

Response body

```
[
  {
    "idProduto": 2,
    "nome": "Mouse",
    "preco": 50,
    "quantidade": 15,
    "idEstoque": 2,
    "estoque": null
  }
]
```

Download

Response headers

```
content-type: application/json; charset=utf-8
date: Sat, 25 Apr 2020 00:38:14 GMT
server: Kestrel
transfer-encoding: chunked
x-powered-by: ASP.NET
```

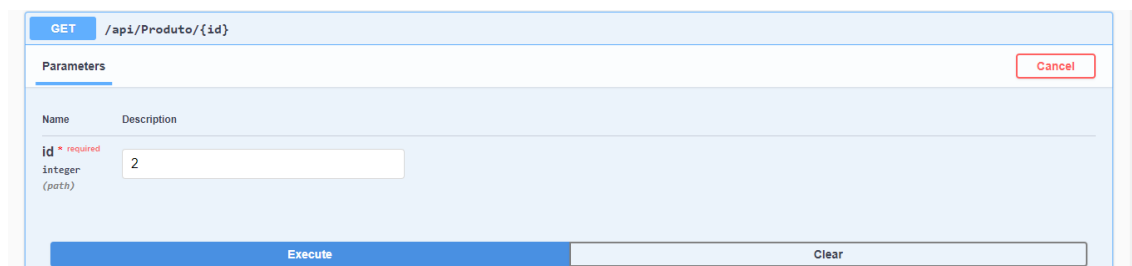
Responses

Code	Description	Links
200	Success	No links

ID: 892-1584-4378 Stop Share

GET /api/Produto/{id}

Consulta de produto por Id

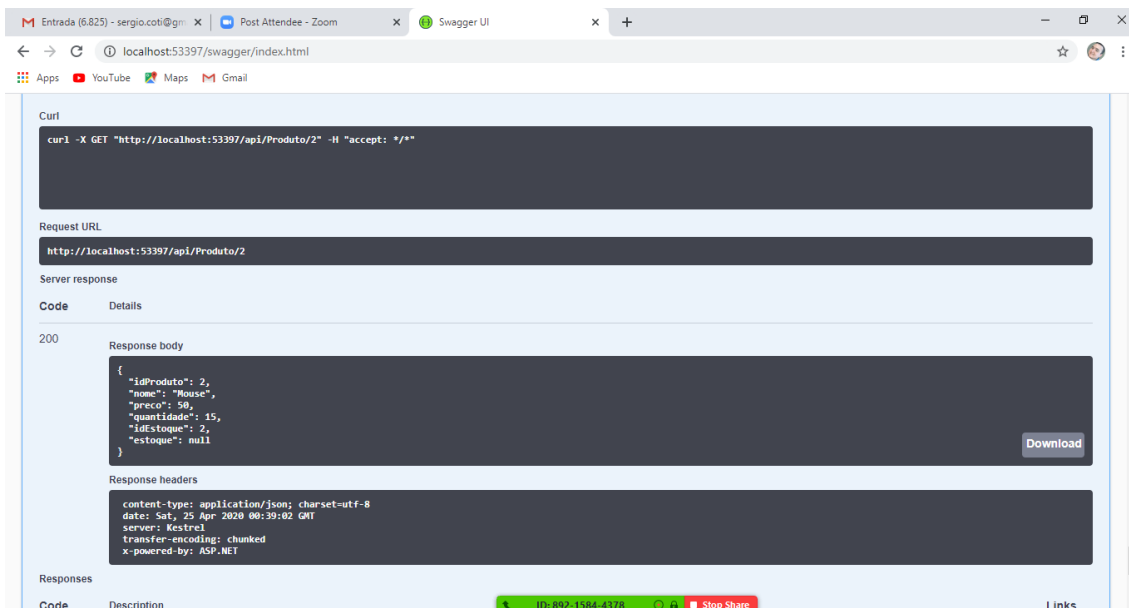


GET /api/Produto/{id}

Parameters

Name	Description
id * required integer (path)	2

Execute Clear



Pendências:

Gerenciamento de transações:

Trabalhar no EntityFramework com o conceito de Commit e Roolback

Solução: Implementar o padrão UnitOfWork

Consultas com JOIN:

Implementar consultas no EntityFramework que possam retornar dados de objetos relacionados a um objeto principal. Exemplo: Consulta de Produtos retornar tambem os dados do Estoque. Consulta de Estoque retornar tambem todos os Produtos relacionados ao Estoque.

Solução: utilizando consultas LAMBDA

Continua...