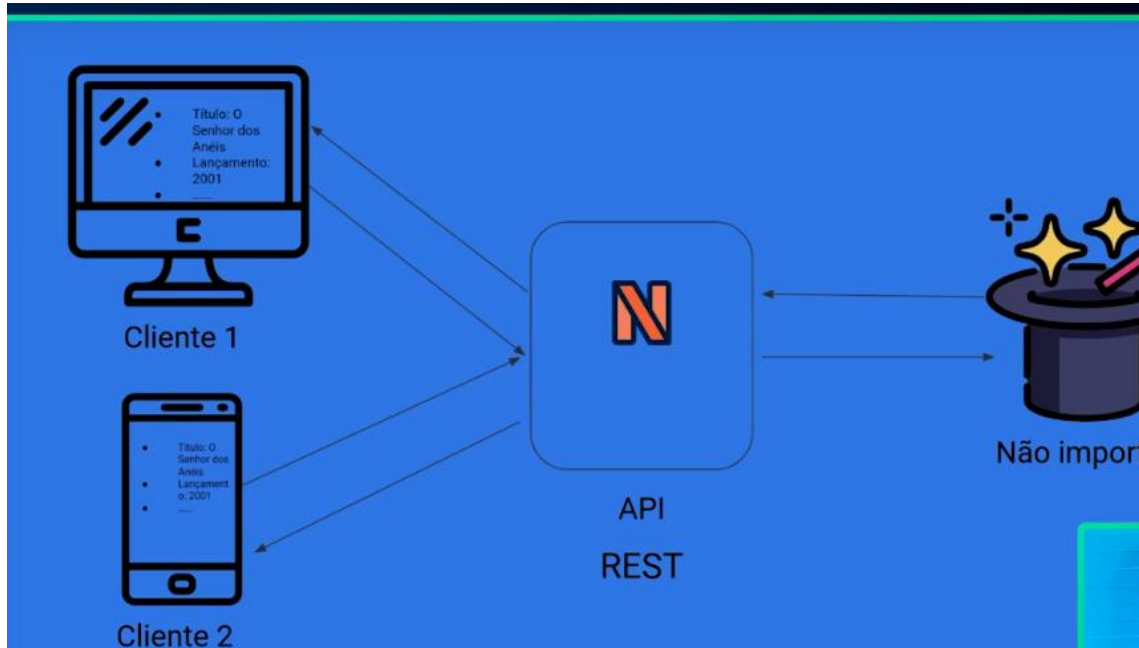
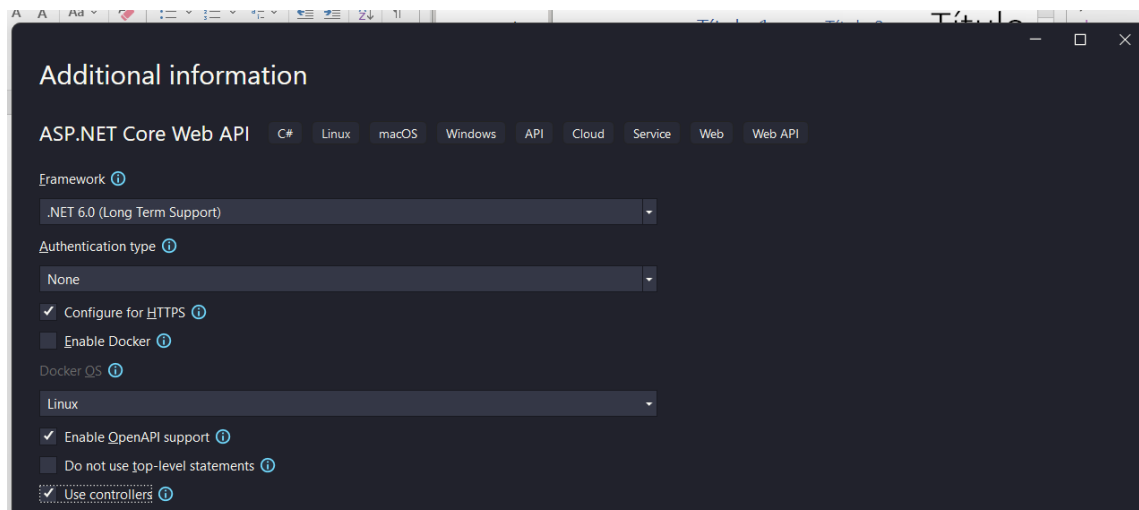


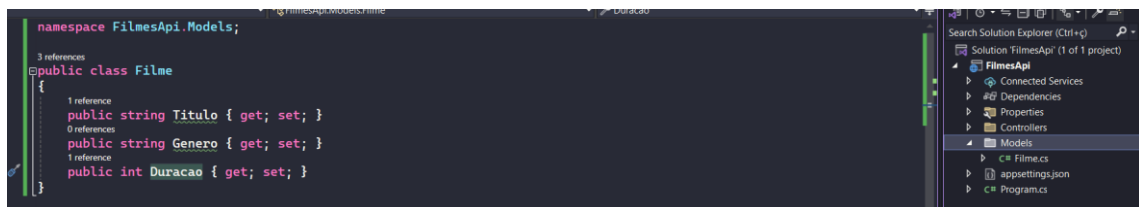
## #01 oque é uma API



## #02 criando um projeto no .net6



## #03 recebendo os dados de um filme



```
1 using FilmesApi.Models;
2 using Microsoft.AspNetCore.Mvc;
3
4 namespace FilmesApi.Controllers;
5
6 [ApiController]
7 [Route("[controller]")]
8 public class FilmeController : ControllerBase
9 {
10     private List<Filme> filmes = new List<Filme>();
11
12     [HttpPost] //tipo post
13     public void AdicioneFilme([FromBody] Filme filme) //vem do body
14     {
15         filmes.Add(filme);
16         Console.WriteLine(filme.Titulo);
17         Console.WriteLine(filme.Duracao);
18     }
19 }
```

#04 validando parâmetros recebido, os dataAnnotations são responsáveis pela validação

```
1 namespace FilmesApi.Models;
2
3 public class Filme
4 {
5     [Required(ErrorMessage = "O título do filme é obrigatório.")]
6     public string Titulo { get; set; }
7
8     [Required(ErrorMessage = "O gênero do filme é obrigatório.")]
9     [MaxLength(50, ErrorMessage = "Tamanho máximo 50 caracteres.")]
10    public string Genero { get; set; }
11
12    [Required]
13    [Range(70, 300, ErrorMessage = "Duração máxima entre 70 e 300 minutos.")] //intervalo de 70 até 300
14    public int Duracao { get; set; }
15 }
```

#05 Retornando filmes da api

```
1 [HttpGet] //tipo get
2 public IEnumerable<Filme> RecuperarFilmes()
3 {
4     return filmes;
5 }
```

#06 Recuperando filmes por id

```
1 public class Filme
2 {
3     public int Id { get; set; }
4
5     [Required(ErrorMessage = "O título do filme é obrigatório.")]
6     public string Titulo { get; set; }
7 }
8
9 [HttpGet("{int:id}")]
10 public Filme? RecuperarFilmePorId(int id)
11 {
12     return filmes.FirstOrDefault(filme => filme.Id == id);
13 }
```

#07 paginando resultados

```
1 [HttpGet] //tipo get
2 public IEnumerable<Filme> RecuperarFilmes([FromQuery] int take) //o take vem do query
3 {
4     return filmes.Take(take); //skip quantos quer pular, take quantos quer pegar
5 }
```

## #08 padronizando o retorno

```
[HttpGet("{id}")]
0 references
public IActionResult RecuperarFilmePorId(int id) //iActionResult é um interface do resultado do retorno de um ação
{
    var filme = filmes.FirstOrDefault(filme => filme.Id == id);

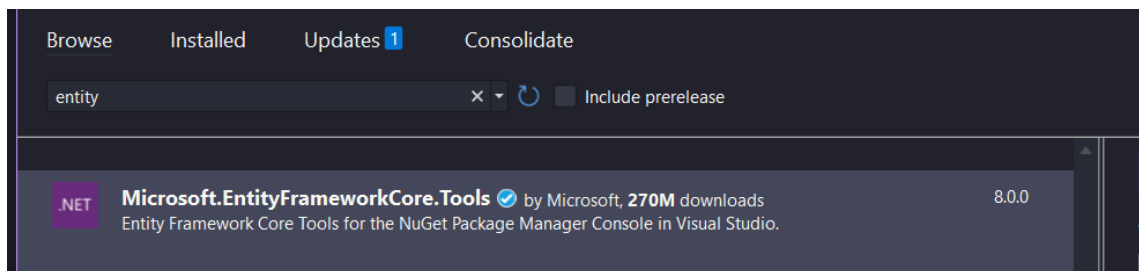
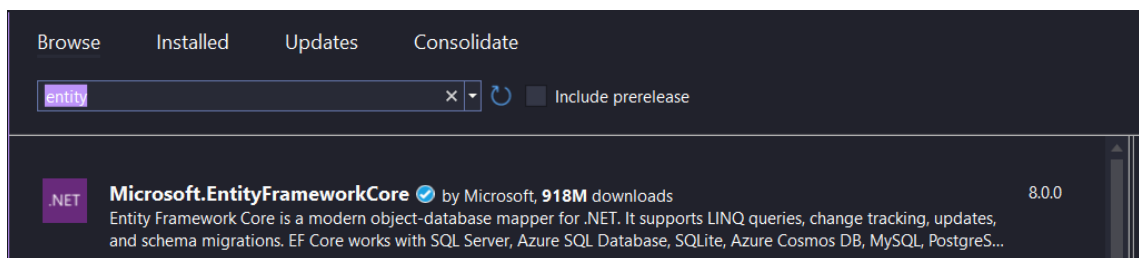
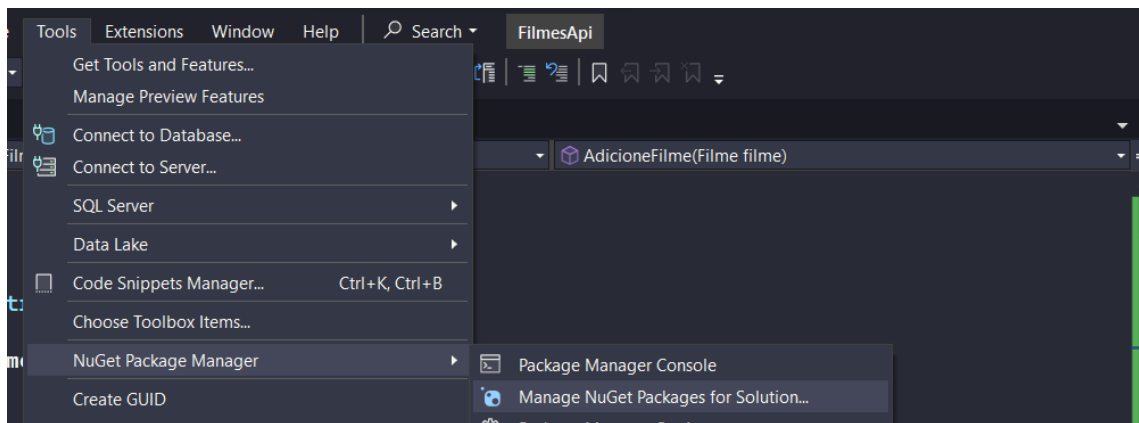
    if (filme == null)
    {
        return NotFound("Filme não encontrado");
    }

    return Ok(filme);
}
```

```
[HttpPost] //tipo post
0 references
public IActionResult AdicioneFilme([FromBody] Filme filme) //vem do body
{
    filme.Id = id++;
    filmes.Add(filme);

    return CreatedAtAction(nameof(RecuperarFilmePorId), new { id = filme.Id }, filme);
}
```

## #09 conectando ao banco de dados adicionar alguns pacotes para conectar com o banco



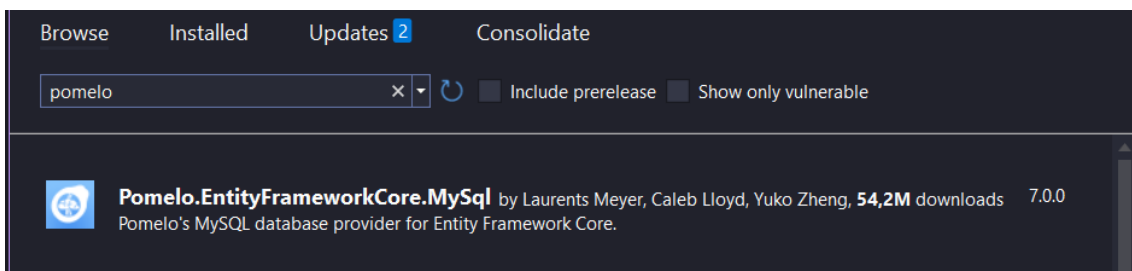
```
using FilmesApi.Models;
using Microsoft.EntityFrameworkCore;

namespace FilmesApi.Data;

2 references
public class FilmeContext : DbContext
{
    0 references
    public FilmeContext(DbContextOptions<FilmeContext> op) : base(op) {}

    0 references
    public DbSet<Filme> Filmes { get; set; }
}
```

```
appsettings.json  X FilmeContext.cs
Schema: https://json.schemastore.org/appsettings.json
1 {
2   "ConnectionStrings": {
3     "FilmeConnection": "server=localhost;database=filmeDb;user=root;password=1234"
4   },
5   "Logging": {
```



```
Program.cs  X appsettings.json  FilmeContext.cs
FilmesApi
1 using FilmesApi.Data;
2 using Microsoft.EntityFrameworkCore;
3
4 var builder = WebApplication.CreateBuilder(args);
5
6 var connectionString = builder.Configuration.GetConnectionString("FilmeConnection");
7
8 builder.Services.AddDbContext<FilmeContext>(op =>
9     op.UseMySQL(connectionString, ServerVersion.AutoDetect(connectionString)));
10
11 // Add services to the container.
```

## #10 gerando a primeira migration

```
4
5 references
5 public class Filme
6 {
7     [Key]
8     [Required]
9     3 references
10     public int Id { get; set; }
11 }
```

Abra o console para gerar as migrations fazer o comando dotnet ef migrations add NomeAqui

```
PS C:\Users\romar\Documentos\Estudos\Cursos\alura\net6CriandoWebApi\FilmesApi\FilmesApi> dotnet ef migrations add CriandoTabelaDeFilme
Build started...
```

Para gerar o banco de dados fazer o comando dotnet ef database update assim ele gera o banco com as tabelas

3 - Execute o comando de criação de migration:

```
dotnet ef migrations add FilmeMigration
```

4 - Aplique as mudanças no banco de dados:

```
dotnet ef database update
```

## #11 realizando operações no banco

```
1 using FilmesApi.Data;
2 using FilmesApi.Models;
3 using Microsoft.AspNetCore.Mvc;
4
5 namespace FilmesApi.Controllers;
6
7 [ApiController]
8 [Route("[controller]")]
9 public class FilmeController : ControllerBase
10 {
11     private FilmeContext _context;
12
13     public FilmeController(FilmeContext context)
14     {
15         _context = context;
16     }
17
18     [HttpPost] //tipo post
19     public IActionResult AdicioneFilme([FromBody] Filme filme) //vem do body
20     {
21         _context.Filmes.Add(filme);
22         _context.SaveChanges();
23
24         return CreatedAtAction(nameof(RecuperarFilmePorId), new { id = filme.Id }, filme);
25     }
26 }
```

```
[HttpGet] //tipo get
public IEnumerable<Filme> RecuperarFilmes([FromQuery] int take) //o take vem do query
{
    return _context.Filmes.Take(take); //skip quantos quer pular, take quantos quer pegar
}

[HttpGet("{id}")]
public IActionResult RecuperarFilmePorId(int id) //iActionResult é um interface do resultado do retorno de um ação
{
    var filme = _context.Filmes.FirstOrDefault(filme => filme.Id == id);

    if (filme == null)
    {
        return NotFound("Filme não encontrado");
    }


    return Ok(filme);
}
```


## #12 utilizando DTOS


```
1 using System.ComponentModel.DataAnnotations;
2
3 namespace FilmesApi.DTO;
4
5 public class CreateFilmeDTO
6 {
7     [Required(ErrorMessage = "O título do filme é obrigatório.")]
8     public string Titulo { get; set; }
9
10    [Required(ErrorMessage = "O gênero do filme é obrigatório.")]
11    [StringLength(50, ErrorMessage = "Tamanho máximo 50 caracteres.")]
12    public string Genero { get; set; }
13
14    [Required]
15    [Range(70, 300, ErrorMessage = "Duração máxima entre 70 e 300 minutos.")] //intervalo de 70 até 300
16    public int Duracao { get; set; }
17 }
```

Browse Installed Updates Consolidate

mapper x Include prerelease

 **AutoMapper** by Jimmy Bogard, 537M downloads  
A convention-based object-object mapper.

 **AutoMapper.Extensions.Microsoft.DependencyInjection** by Jimmy Bogard, 215M downloads 12.0.1  
AutoMapper extensions for ASP.NET Core

 **TinyMapper** by Sergey Morenko, 6,48M downloads 3.0.3

```
Program.cs x Filme.cs CreateFilmeDTO.cs FilmeController.cs
FilmesApi
10
11 builder.Services.AddAutoMapper(AppDomain.CurrentDomain.GetAssemblies());
12
```

```
FilmeProfile.cs x Program.cs Filme.cs CreateFilmeDTO.cs FilmeController.cs
FilmesApi
1 using AutoMapper;
2 using FilmesApi.DTO;
3 using FilmesApi.Models;
4
5 namespace FilmesApi.Profiles;
6
7 public class FilmeProfile : Profile
8 {
9     public FilmeProfile()
10    {
11        CreateMap<CreateFilmeDTO, Filme>();
12    }
13 }
```

## #13 atualizando dados do put

```
FilmeController.cs x FilmesApi.DTO.UpdateFilmeDTO
FilmesApi
1 using System.ComponentModel.DataAnnotations;
2
3 namespace FilmesApi.DTO;
4
5 public class UpdateFilmeDTO
6 {
7     [Required(ErrorMessage = "O título do filme é obrigatório.")]
8     public string Titulo { get; set; }
9
10    [Required(ErrorMessage = "O gênero do filme é obrigatório.")]
11    [StringLength(50, ErrorMessage = "Tamanho máximo 50 caracteres.")]
12    public string Genero { get; set; }
13
14    [Required]
15    [Range(70, 300, ErrorMessage = "Duração máxima entre 70 e 300 minutos.")] //intervalo de 70 até 300
16    public int Duracao { get; set; }
17 }
```

```
[HttpPut("{id}")]
0 references
public IActionResult AtualizarFilme(int id, [FromBody] UpdateFilmeDTO filmeDTO)
{
    var filme = _context.Filmes.FirstOrDefault(filme => filme.Id == id);

    if (filme == null)
    {
        return NotFound("Filme não encontrado.");
    }

    _mapper.Map(filmeDTO, filme);
    _context.SaveChanges();
    return NoContent();
}
```

```
FilmeProfile.cs x FilmeController.cs
FilmesApi
1 using AutoMapper;
2 using FilmesApi.DTO;
3 using FilmesApi.Models;
4
5 namespace FilmesApi.Profiles;
6
7 public class FilmeProfile : Profile
8 {
9     0 references
10     public FilmeProfile()
11     {
12         CreateMap<CreateFilmeDTO, Filme>();
13         CreateMap<UpdateFilmeDTO, Filme>();
14     }
15 }
```

#14 atualizando dados com patch alterar apenas um campo específico.

Newtonsoft.Json.Bson by James Newton-King, 604M downloads 1.0.2  
.json.NET BSON adds support for reading and writing BSON

Microsoft.AspNetCore.Mvc.NewtonsoftJson by Microsoft, 345M downloads 8.0.0  
ASP.NET Core MVC features that use Newtonsoft.Json. Includes input and output formatters for JSON and JSON PATCH.

Swashbuckle.AspNetCore.Newtonsoft by Swashbuckle.AspNetCore.Newtonsoft, 74,7M downloads 6.5.0  
Swagger Generator opt-in component to support Newtonsoft.Json serializer behaviors

Project	Version	Installed
FilmesApi		

Installed: not installed Uninstall

Version: 6.0.25 Install

```
Program.cs x NuGet - Solution FilmeProfile.cs FilmeController.cs
FilmesApi
19
20 builder.Services.AddControllers().AddNewtonsoftJson();
21
22 var app = builder.Build();
```

```
public class FilmeProfile : Profile
{
    0 references
    public FilmeProfile()
    {
        CreateMap<CreateFilmeDTO, Filme>();
        CreateMap<UpdateFilmeDTO, Filme>();
        CreateMap<Filme, UpdateFilmeDTO>();
    }
}
```

```

[HttpPatch("{id}")]
0 references
public IActionResult AtualizarFilmeParcial(int id, [FromBody] JsonPatchDocument<UpdateFilmeDTO> patch)
{
    var filme = _context.Filmes.FirstOrDefault(filme => filme.Id == id);

    if (filme == null)
    {
        return NotFound("Filme não encontrado");
    }

    var filmeParaAtualizar = _mapper.Map<UpdateFilmeDTO>(filme);

    patch.ApplyTo(filmeParaAtualizar, ModelState);

    if (!TryValidateModel(filmeParaAtualizar))
    {
        return ValidationProblem(ModelState);
    }

    _mapper.Map(filmeParaAtualizar, filme);
    _context.SaveChanges();
    return NoContent();
}

```

## #15 deletando filmes

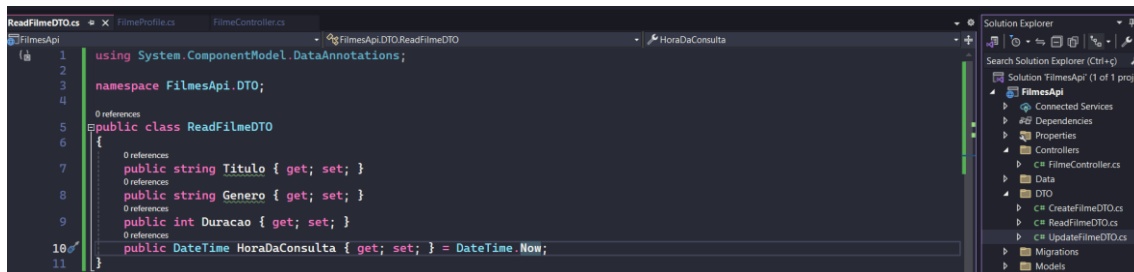
```

[HttpDelete("{id}")]
0 references
public IActionResult DeletarFilme(int id)
{
    var filme = _context.Filmes.FirstOrDefault(filme => filme.Id == id);

    if (filme == null)
    {
        return NotFound("Filme não encontrado.");
    }

    _context.Remove(filme);
    _context.SaveChanges();
    return NoContent();
}

```



```

[HttpGet] //tipo get
0 references
public IEnumerable<ReadFilmeDTO> RecuperarFilmes([FromQuery] int take) //o take vem do query
{
    return _mapper.Map<List<ReadFilmeDTO>>(_context.Filmes.Take(take)); //skip quantos quer pular, take quantos quer pular
}

[HttpGet("{id}")]
1 reference
public IActionResult RecuperarFilmePorId(int id) //iActionResult é um interface do resultado do retorno de um ação
{
    var filme = _context.Filmes.FirstOrDefault(filme => filme.Id == id);

    if (filme == null)
    {
        return NotFound("Filme não encontrado");
    }

    var filmeDTO = _mapper.Map<ReadFilmeDTO>(filme);

    return Ok(filmeDTO);
}

```



```
public class FilmeProfile : Profile
{
    0 references
    public FilmeProfile()
    {
        CreateMap<CreateFilmeDTO, Filme>();
        CreateMap<UpdateFilmeDTO, Filme>();
        CreateMap<Filme, UpdateFilmeDTO>();
        CreateMap<Filme, ReadFilmeDTO>();
    }
}
```

#16 documentando a api