

Documentação

Opflix

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Alameda Barão de Limeira, 539 – São Paulo/SP |  |  |  | sp.senai.br |

Sumário

1. [Resumo 2](#_Toc533767843)

[Objetivos](#_Toc533767844)

1. [Descrição do projeto 2](#_Toc533767845)

[Resumo do projeto](#_Toc533767846)

1. [Modelagem de Software 3](#_Toc533767847)

[Modelo Lógico](#_Toc533767848)

[Modelo Físico](#_Toc533767849)

[Modelo Conceitual](#_Toc533767850)

[Cronograma](#_Toc533767851)

1. [Funcionalidades 5](#_Toc533767852)

[Web](#_Toc533767853)

[Mobile](#_Toc533767854)

1. [Estrutura do projeto 6](#_Toc533767855)

Banco de dados

Swagger

Pacotes utilizados

Organização

Postman

Software utilizados

Roteiro para primeiro teste em máquina

Resumo

Objetivos

Este documento tem como finalidade descrever o projeto ‘Opflix’ bem como seu processo.

Descrição do projeto

Opflix é um sistema de dupla interface desenvolvido para influenciar uma maior integração de seus usuários com o que há de mais recente na área do audiovisual, tanto no cinema, televisão, plataformas de streaming e até mesmo em plataformas já fora de uso, com seus clássicos.

Resumo do projeto

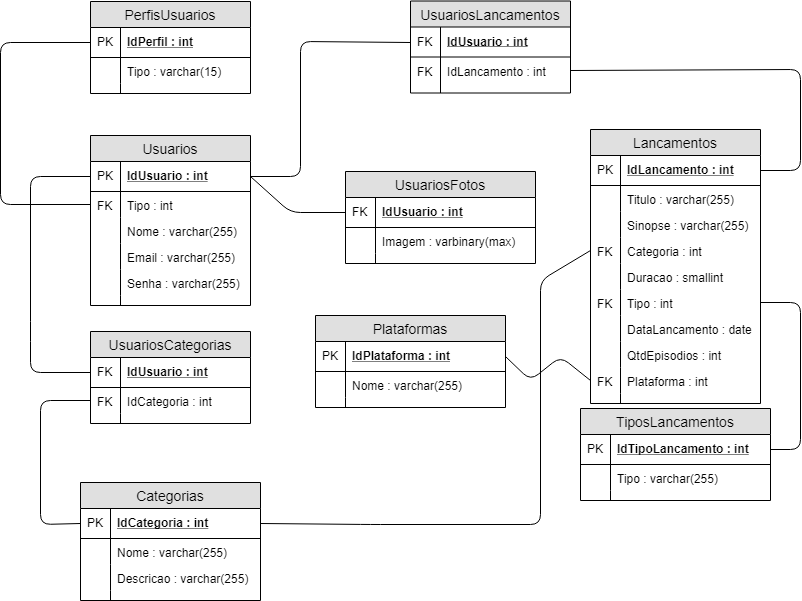
A proposta do software é, inicialmente, organizar, divulgar e orientar usuários através dos lançamentos cinematográficos e produções televisivas.

Modelagem de Software

A modelagem do Opflix demonstra bem como os dados serão armazenados os dados necessários para o bom funcionamento do projeto. Sendo que cada modelagem tem sua peculiaridade explicada e demonstrada nos itens abaixo.

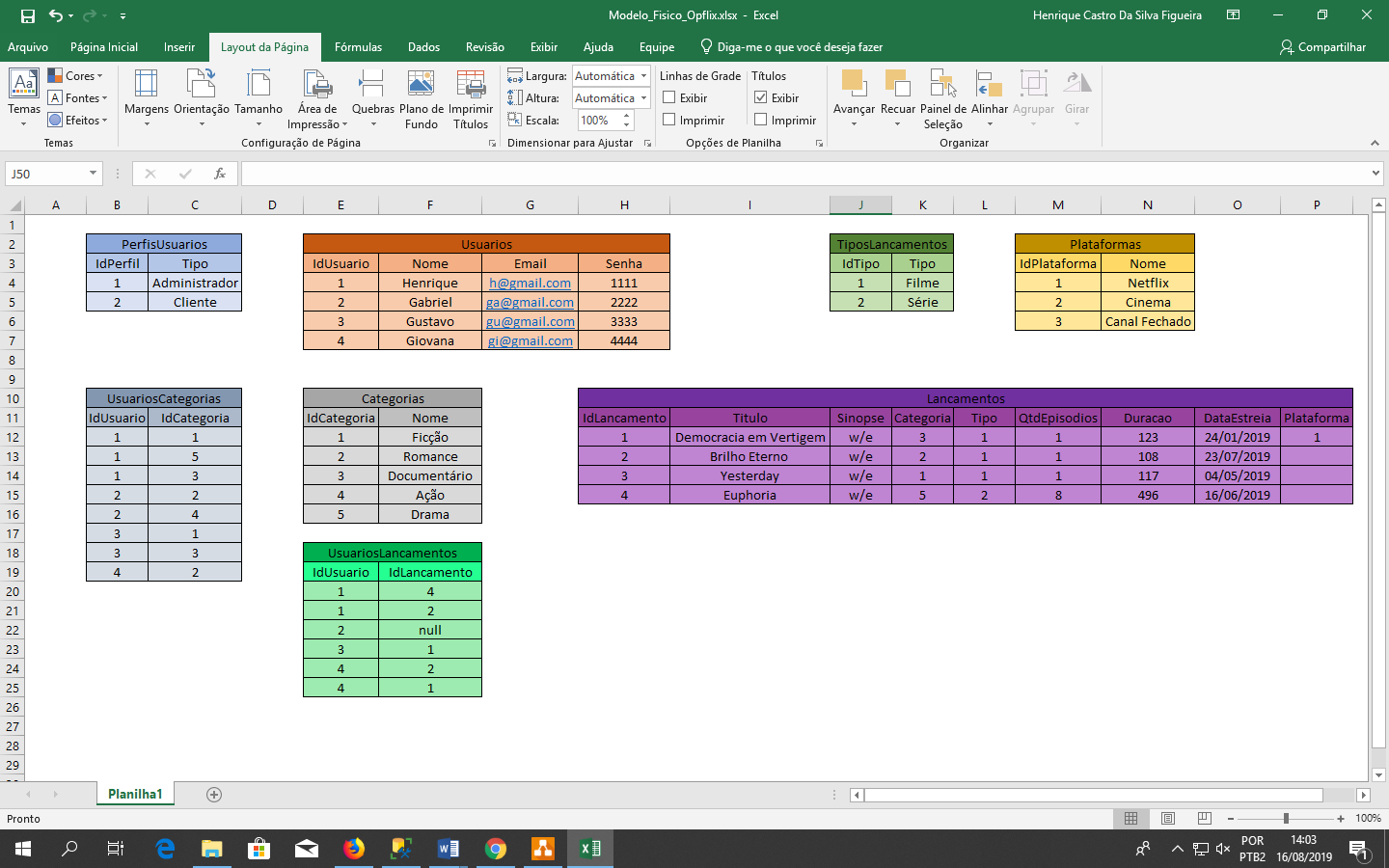
Modelo Lógico

Modelo representativo que mais se aproxima da estrutura de banco de dados a ser construída, definindo o padrão de nomenclatura utilizado para o desenvolvimento, as chaves primárias e estrangeiras, a normalização das informações entre outros fatores.



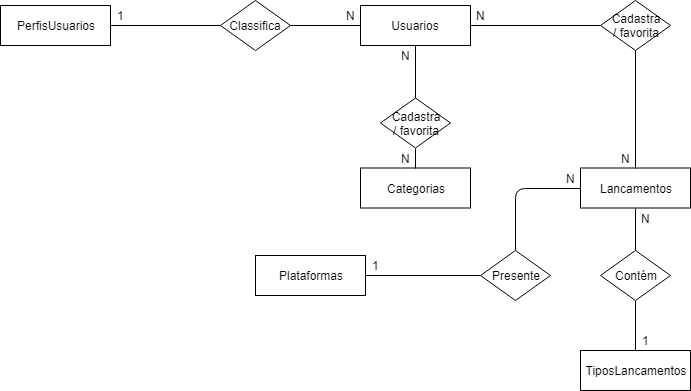
Modelo Físico

Este modelo tem como objetivo representar a forma como os dados serão armazenados na estrutura de banco de dados proposta, levando em conta as limitações do sistema de gerencialmente de banco de dados utilizado.



Modelo Conceitual

O modelo conceitual, diferentemente dos modelos anteriores, tem como foco ilustrar os relacionamentos das entidades existentes no projeto, demonstrando principalmente aspectos do negócio do cliente e não da tecnologia.



Cronograma

|  | Dia 1 | Dia 2 | Dia 3 | Dia 4 | Dia 5 | Dia 6 | Dia 7 | Dia 8 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Modelo Lógico** |  |  |  |  |  |  |  |  |
| **Modelo Físico** |  |  |  |  |  |  |  |  |
| Modelo Conceitual |  |  |  |  |  |  |  |  |

Funcionalidades

Web

No sistema web, usuários administradores poderão:

* Cadastrar usuários – sendo o novo usuário do tipo comum ou administrador;
* Cadastrar categorias – contendo apenas o nome;
* Cadastrar lançamentos – séries ou filmes contendo título, sinopse, categoria vinculada (apenas uma), tempo de duração (em caso de séries, adicionar a soma das durações de todos os episódios publicados), tipo de lançamento (filme ou série) e a data de estreia (no formato “ano-mês-dia”).

Usuários comuns poderão:

* Visualizar todos os lançamentos visíveis;
* Selecionar categorias favoritas;
* Selecionar lançamentos favoritos.

Mobile

No sistema mobile, usuários comuns poderão:

* Visualizar todos os lançamentos;
* Realizar a ordenação por data de lançamento e/ou categoria;
* Selecionar categorias favoritas;
* Selecionar lançamentos favoritos;
* Filtrar por plataforma/mídia/gênero.

Protótipos

Web

Mobile

Front-End

Mobile

Arquitetura do Projeto

Banco de dados

Como ferramenta de banco de dados, foi usada a linguagem SQL Server, sendo uma linguagem relacional, algumas tabelas existem com a ausência de chaves primárias, característica presente nas chamadas tabelas intermediárias. Estas tabelas são usadas neste projeto para relacionar os usuários com seus respectivos lançamentos e categorias favoritados. Porém, existe uma exceção, a tabela de usuários e suas fotos, que não é intermediária, já que sua relação com usuários é de 1 : 1.

Swagger

Esta é uma ferramenta instalada no projeto que permite a fácil ilustração de todas as entidades e seus métodos presentes no código do projeto através do link [hospedagem]/Swagger (sendo hospedagem localhost:5000 por padrão até que o projeto seja hospedado.

Pacotes utilizados

* Microsoft.EntityFrameworkCore.SqlServer – versão 2.1.11
* Microsoft.EntityFrameworkCore.SqlServer.Design – versão 1.1
* Microsoft.EntityFrameworkCore.Tools – versão 2.1.11
* SwashBuckle.AspNetCore – versão 4.0.1
* System.IdentityModel.Tokens.Jwt – versão 5.5.
* Microsoft.AspNetCore.Authentication.JwtBearer - versão 2.1.

Organização

O projeto usa o modelo de organização MVC, ou seja, todos os trechos de código estão contidos em pastas conforme a documentação do modelo supracitado. Este modelo faz com que o caminho seguido quando é requisitado alguma ação do projeto seja definido de uma maneira mais lógica e simples de entendimento.

Postman

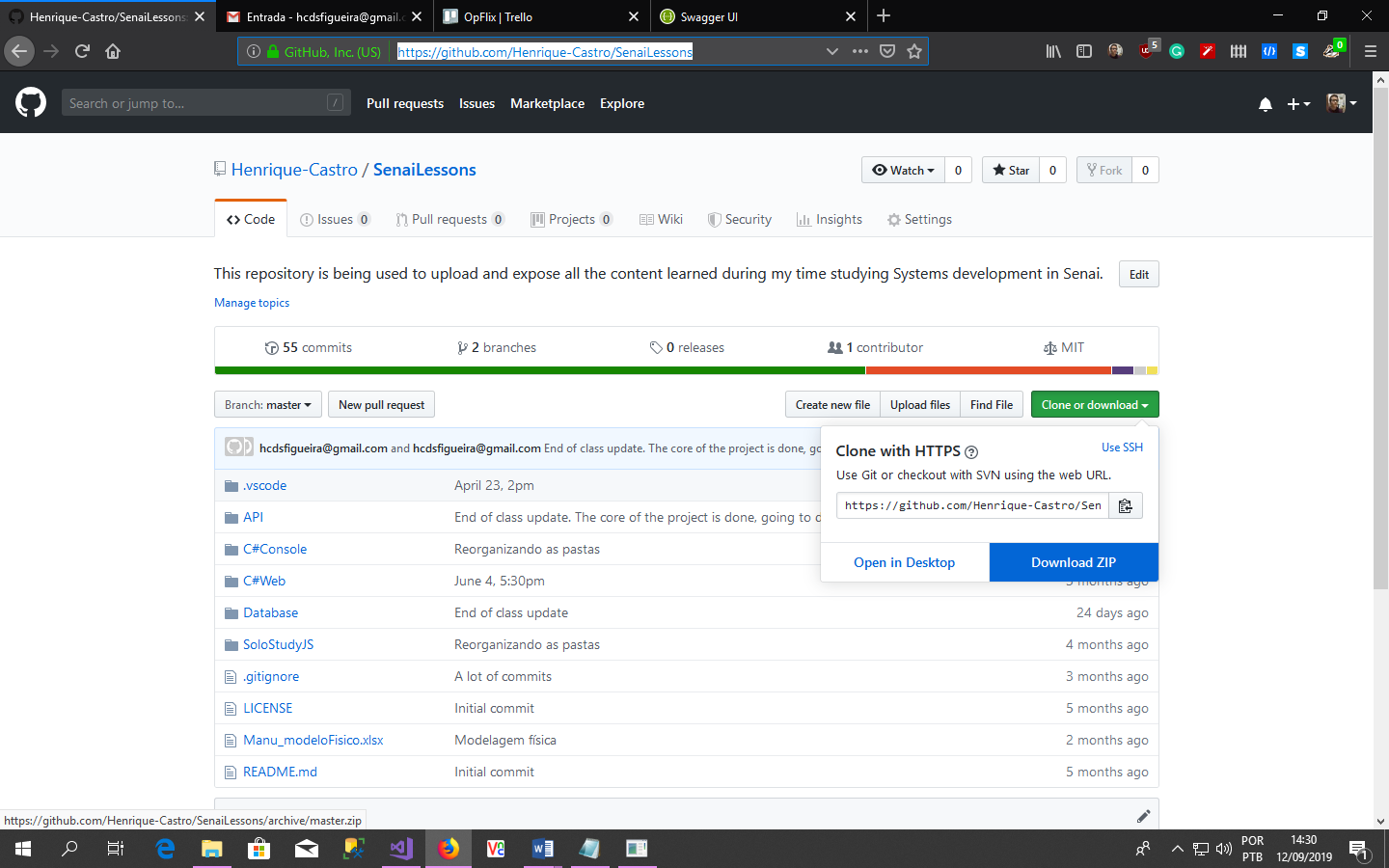
Esta ferramenta é utilizada no contexto deste projeto para o registro e execução de testes. Sendo armazenados em coleções e subpastas, o Postman permite que a coleção de testes seja exportada e importada de maneira dinâmica, permitindo sua avaliação em outras máquinas.

Softwares Utilizados

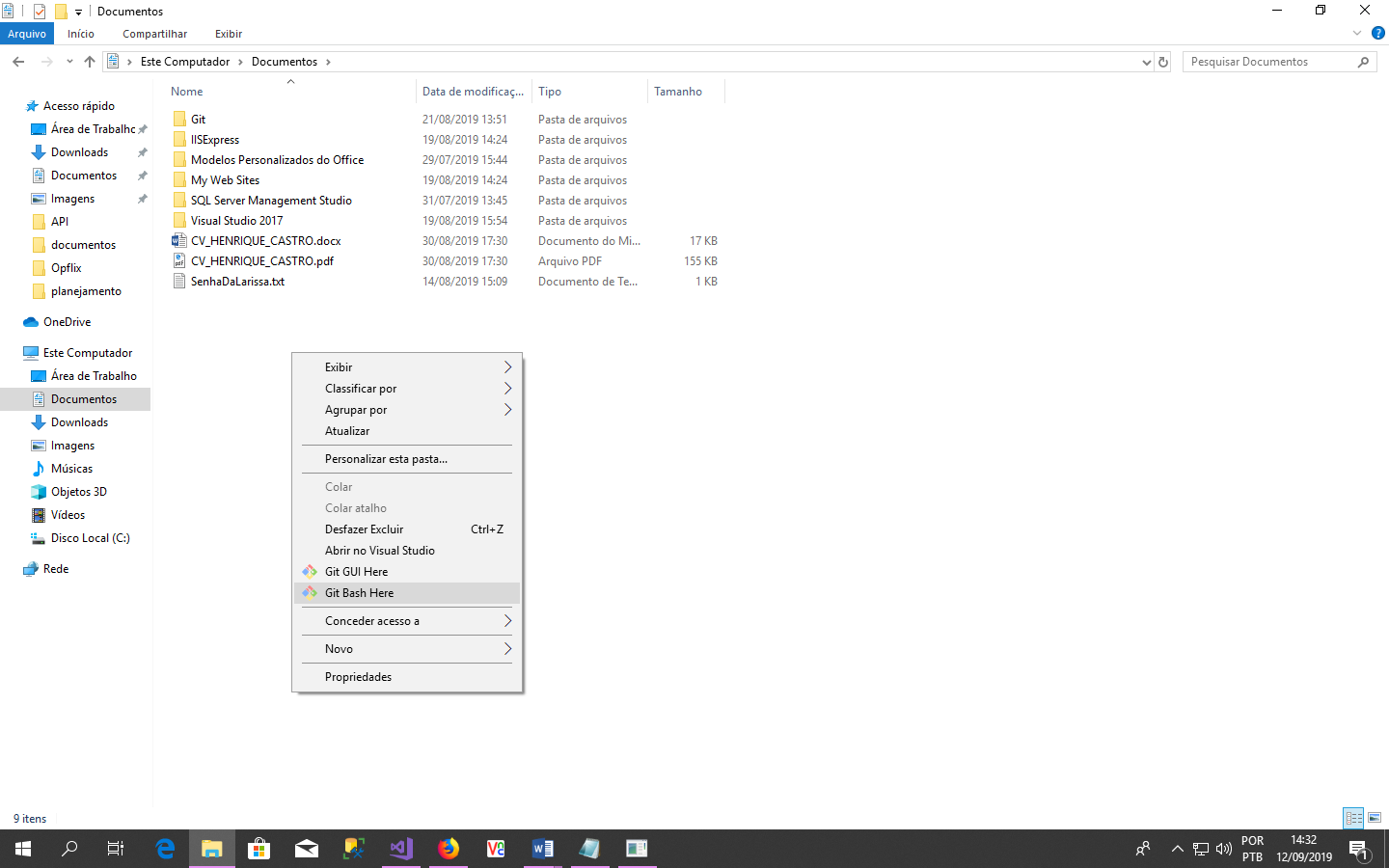
* Visual Studio 2017 – utilizado para a geração e desenvolvimento de todo o código.
* Postman – gerenciamento e armazenamento de testes.
* GitHub – usado para armazenar e versionar diferentes estados do projeto.
* Trello - gerenciamento de tarefas e organização de time.
* Microsoft SQL Server Management 2019 – usado para escrever todos os scrips referentes ao bando de dados que podem ser consultados nas pastas do projeto, presente no repositório Git.
* Draw.Io – ferramenta usada para realizar as modelagens conceitual e lógica do bando de dados.

Roteiro para primeiro teste em máquina

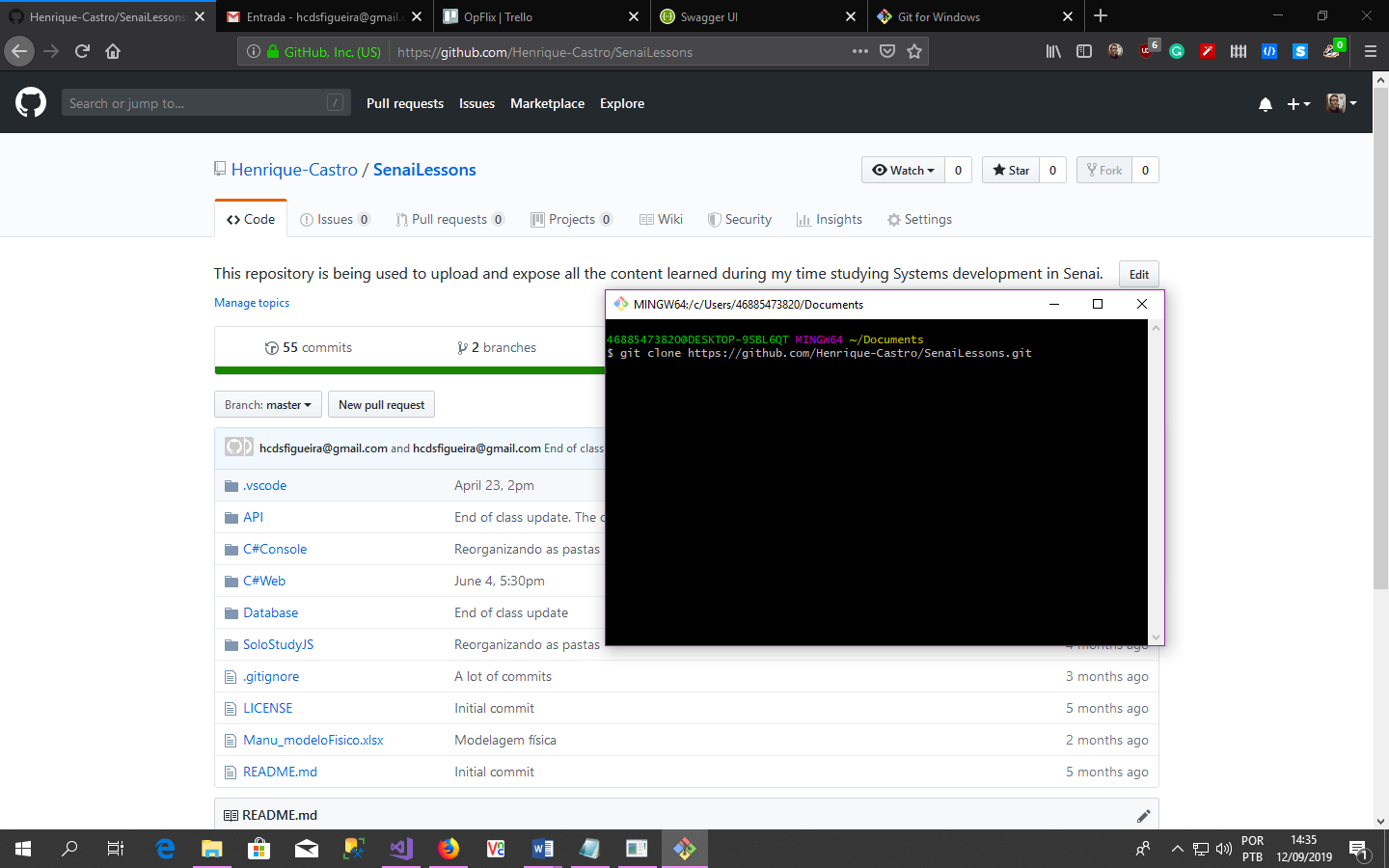
Acessar o link <https://github.com/Henrique-Castro/SenaiLessons> e escolher se deseja baixar o projeto ou cloná-lo na caixa demonstrada na imagem abaixo:



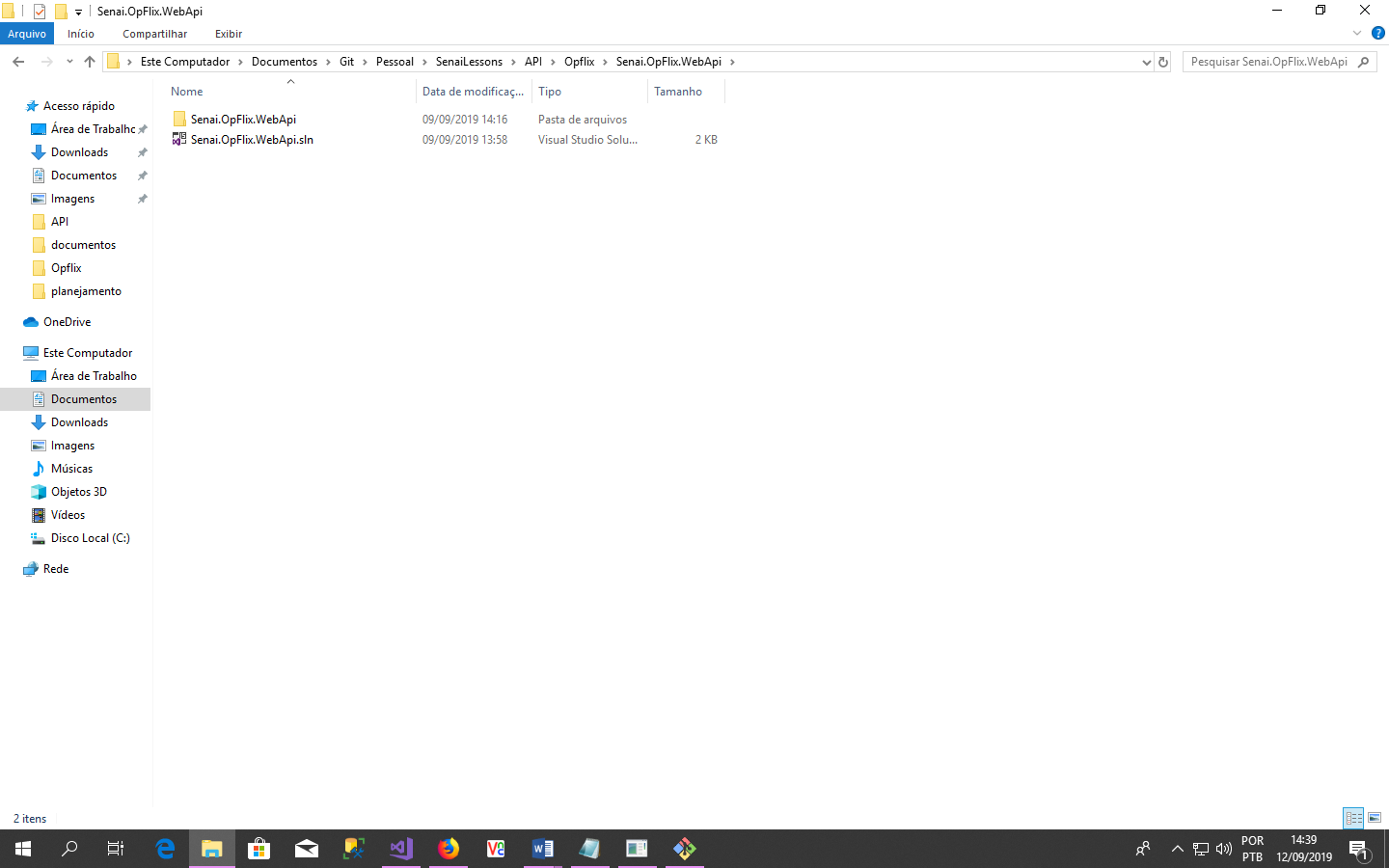
Caso escolha cloná-lo em seu computador basta copiar o link acima do botão azul da imagem, possuir o [GitForWindows](https://gitforwindows.org/) instalado em sua máquina e clicar com o botão direito do mouse onde deseja que a pasta esteja, e na caixa de opções escolher “GitBash Here” conforme a imagem:



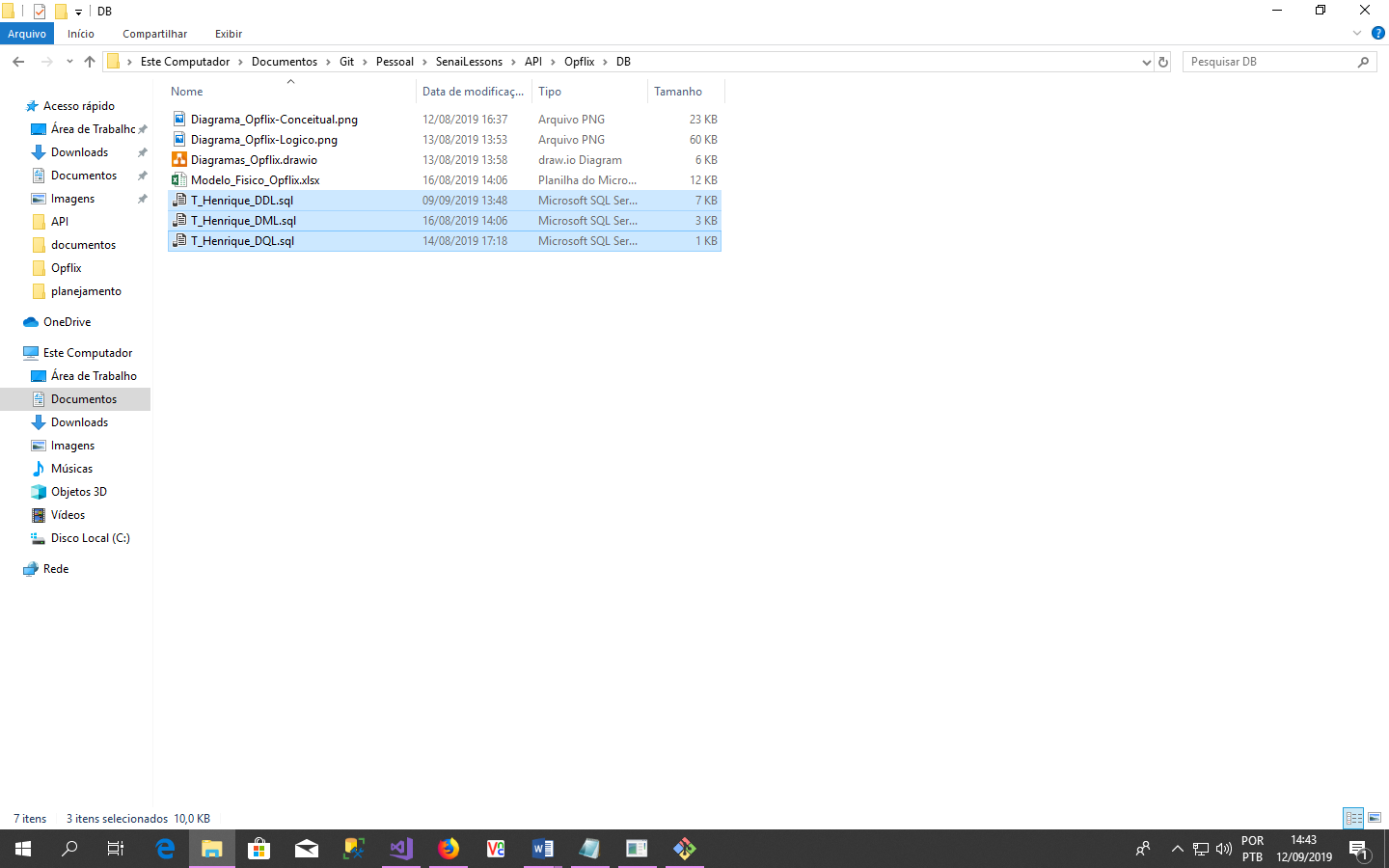
Um terminal de comando será aberto, nele, você deverá escrever “git clone “ e em seguida colar o link do repositório que foi copiado anteriormente:



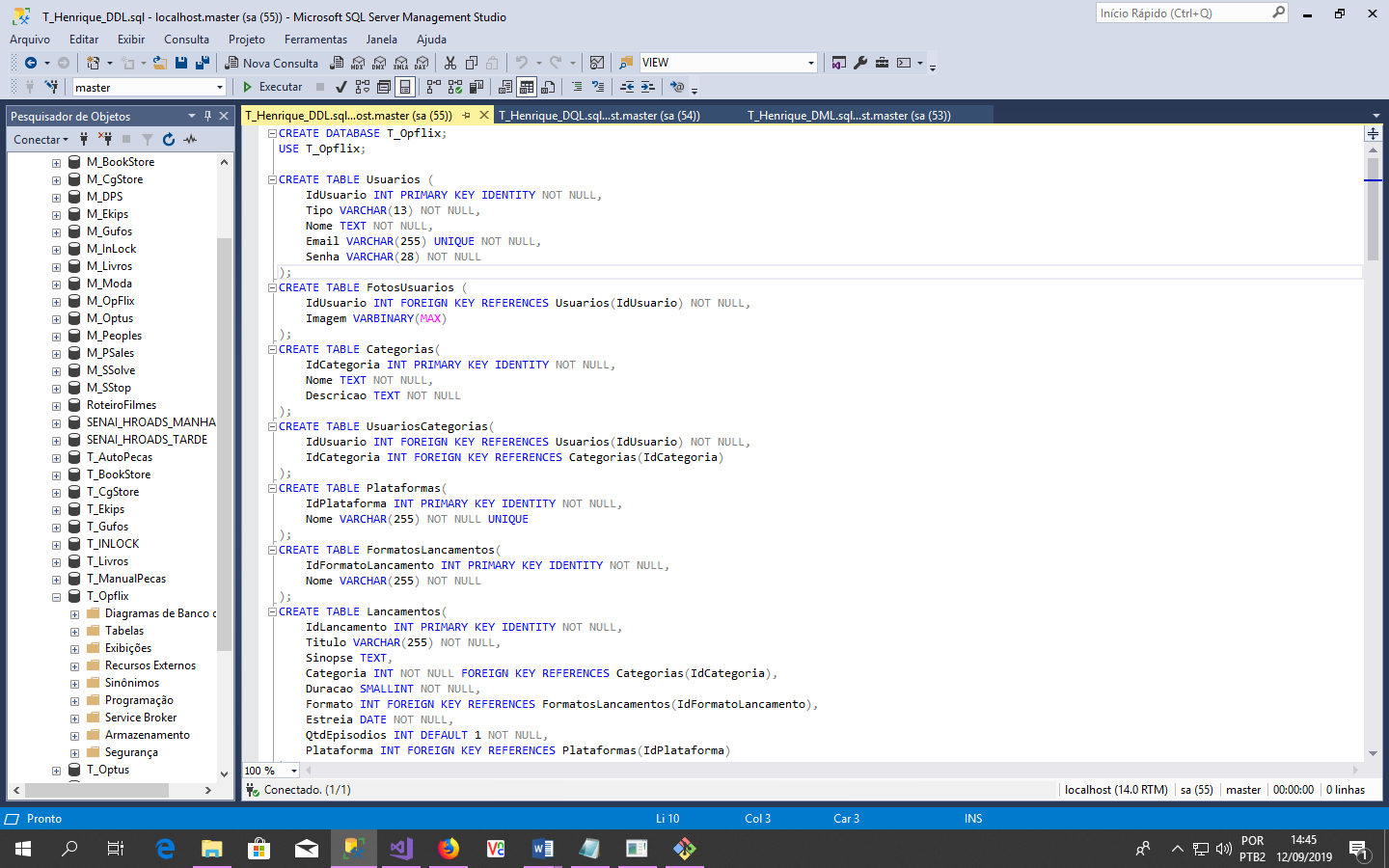
Pressione a tecla Enter. O repositório será clonado onde quer que você tenha escolhido. Siga este caminho para abrir a pasta do projeto “SenaiLessons>API>OpFlix>Senai.OpFlix.WebApi” e em seguida abra o arquivo de mesmo nome com o símbolo roxo:



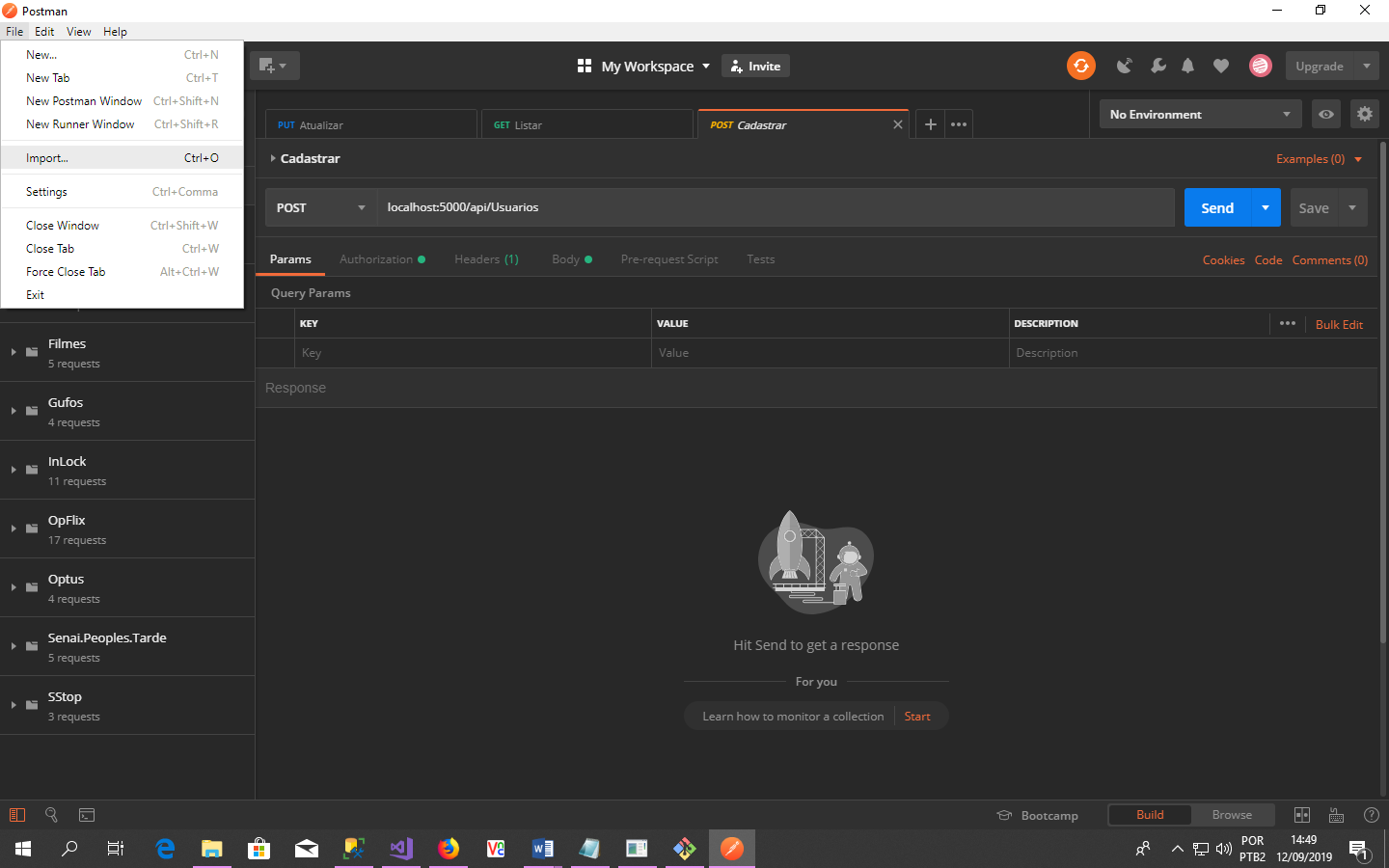
Desta forma, caso o programa Visual Studio esteja instalado em sua máquina, o projeto será aberto. Agora, siga o caminho para abrir os scripts de banco de dados do projeto “SenaiLessons>API>Opflix>DB”. Neste local, você abrirá os três arquivos selecionados na imagem e os executará segundo as imagens abaixo.



Execute primeiro o arquivo com a variação de nome DDL:



Logo após, abra o software Postman e importe a coleção de testes no caminho “SenaiLessons>API>OpFlix>Postman”.



E então basta arrastar a coleção ou selecioná-la no caminho descrito no item acima e realizar os testes.

Referências

Links

Livros