

**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. BackEnd ......................................................6
2. [Funcionalidades 8](#_Toc533767852)

[Web](#_Toc533767853)

[Mobile](#_Toc533767854)

1. [Protótipos 9](#_Toc533767855)

[Web](#_Toc533767856)

[Mobile](#_Toc533767857)

1. [Front-End 10](#_Toc533767858)
2. [Mobile 11](#_Toc533767859)
3. [Arquitetura do Projeto 12](#_Toc533767860)
4. [Referências 13](#_Toc533767861)

[Links](#_Toc533767862)

[Livros](#_Toc533767863)

Resumo

Objetivos

* Criar um banco de dados com as informações indicadas pelo cliente.

Descrição do projeto

* Criar/desenvolver uma companhia chamada Opflix, onde mostrará o catalogo geral de lançamentos de filmes e series com suas respectivas características.

Resumo do projeto

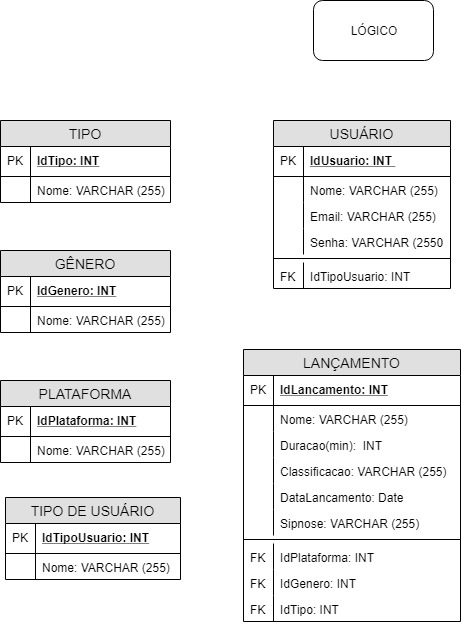
No projeto, foram necessários construir as modelagens exigidas. Após isso armazenamos e organizamos no banco de dados as informações da empresa.

Modelagem de Software

* Modelagem de software é uma representação simplificada de algo real. Por exemplo, uma planta de uma casa, ela apresenta todos os cômodos que o imóvel terá, suas medidas e as características da casa. Quando modelamos um software, também identificamos o que esse futuro sistema deverá fazer. Estaremos tratando as questões funcionais e seus fluxos de dados.

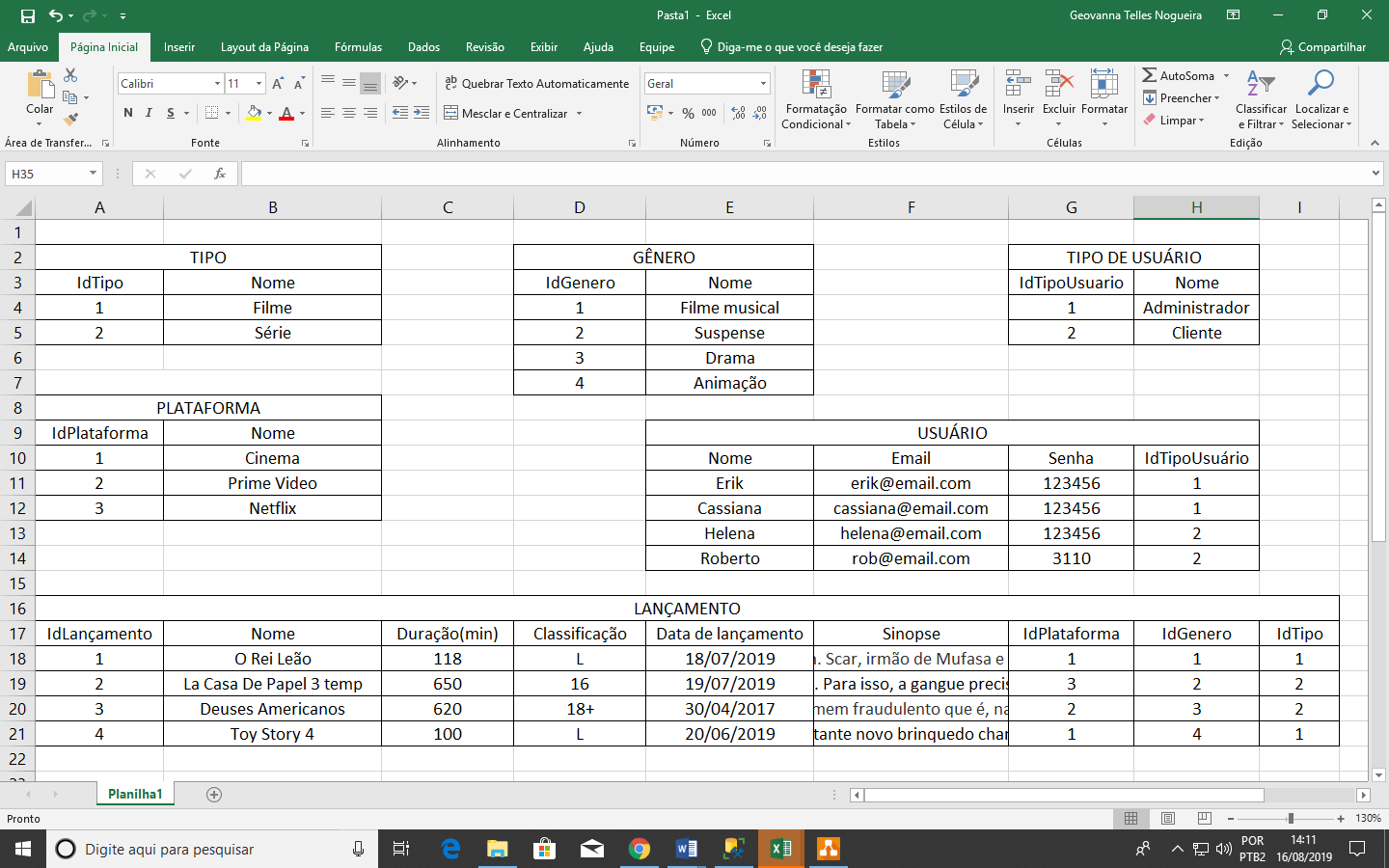
Modelo Lógico

* O modelo lógico já leva em conta algumas limitações e implementa recursos como adequação de padrão e nomenclatura, define as [chaves primárias e estrangeiras](https://www.luis.blog.br/chave%20primaria-chave-estrangeira-e-candidata.aspx), normalização, integridade referencial, entre outras. Para o modelo lógico deve ser criado levando em conta os exemplos de modelagem de dados criados no modelo conceitual.



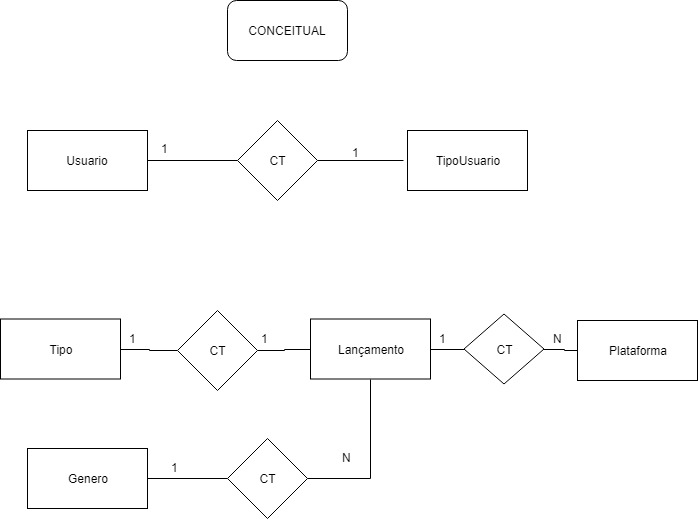
Modelo Físico

* No modelo físico fazemos a modelagem física do modelo de banco de dados. Neste caso leva-se em conta as limitações impostas pelo SGBD escolhido e deve ser criado sempre com base nos exemplos de modelagem de dados produzidos no item anterior, modelo lógico.



Modelo Conceitual

* É o modelo de mais alto nível, ou seja, que está mais próximo da realidade dos usuários. O nível conceitual é desenvolvido com alto nível de abstração, a partir dos requisitos do sistema, extraídos na fase de levantamento de requisitos.



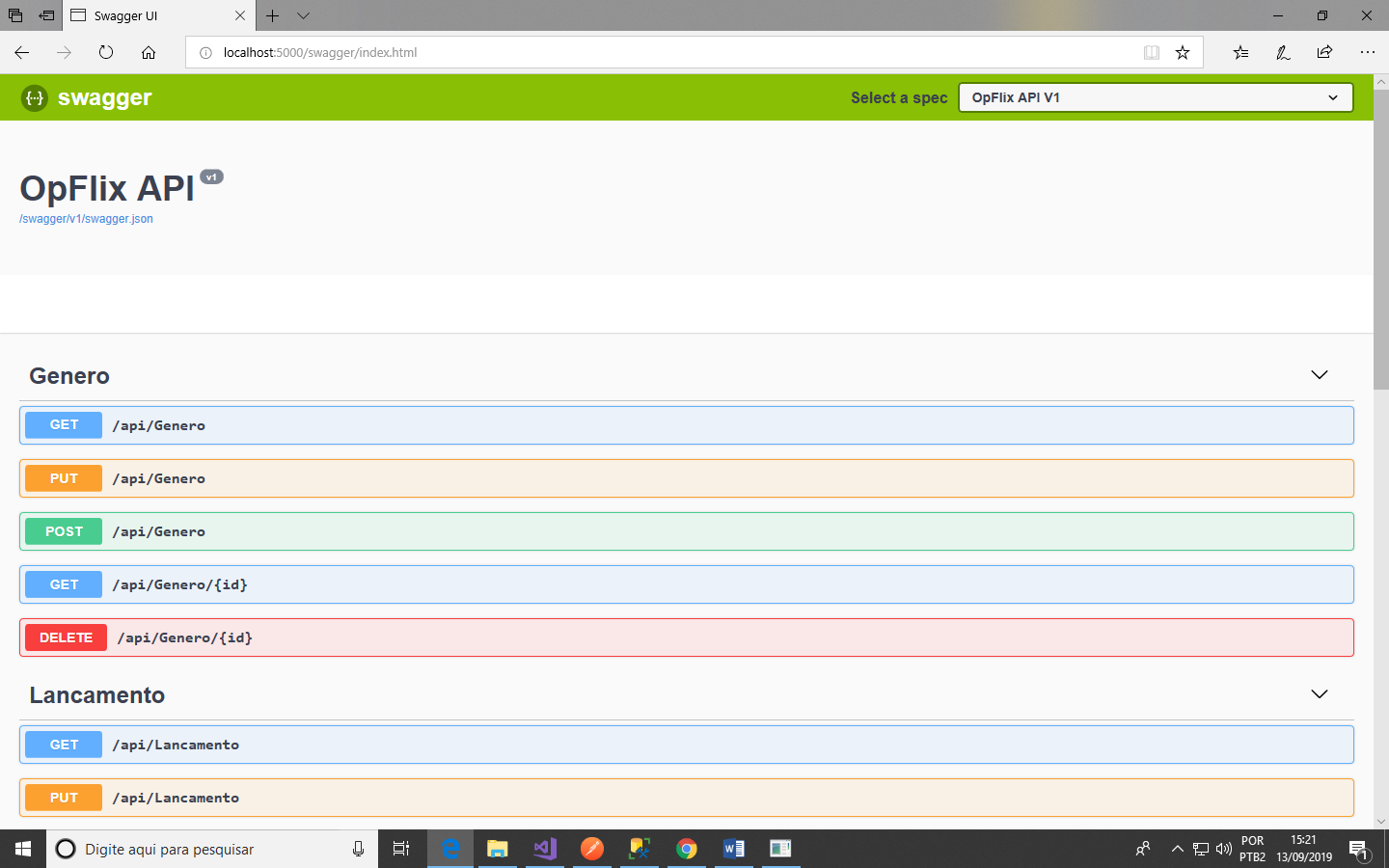
Cronograma

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

BackEnd

Como acessar o Swagger

* O Swagger é uma ferramenta que possibilita um fácil acesso para você fazer suas simulações, quando seu repositório e o seu controller estiverem terminados. Para acessar o Swagger basta você baixar o pacote dele no backend, colocar em uso, rodar o projeto e digitar na web a sua url: “<http://localhost:5000/swagger/index.html>”.



Fazer deploy do projeto

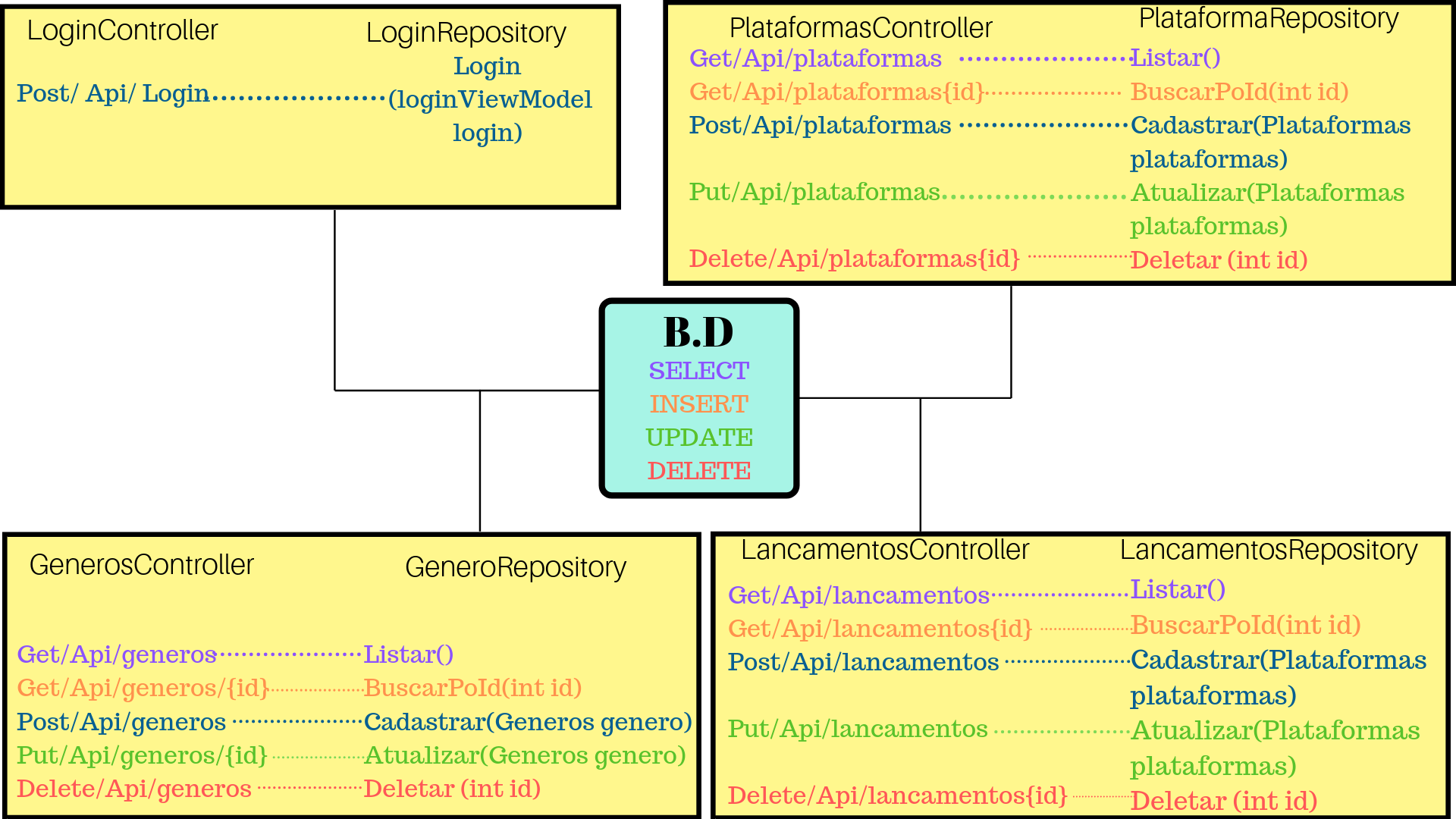
* Para fazer um deploy é necessário você ter o projeto salvo em alguma pasta do seu computador, usar um simples comando para copiar ou implantá-los com qualquer pacote de instalação que quiser. Isso serve para publicar componentes prontos dentro de um contexto de runtime.



Quais pacotes do NuGet foram utilizados

* Microsoft.EntityFrameworkCore.SqlServer - 2.1.11
* Microsoft.EntityFrameworkCore.SqlServer.Design - 1.1.6
* Microsoft.EntityFrameworkCore.Tools - 2.1.11
* System.IdentityModel.Tokens.Jwt - 5.5.0
* Microsoft.AspNetCore.Authentication.JwtBearer - 2.1.1
* Swashbuckle.AspNetCore - 4.0.1

Como está a arquitetura do projeto



Como exportar e importar o postman em outra máquina

* Para importar/exportar o projeto é fundamental que tenha o arquivo pronto para importação e a expotação.
* Basta você clicar no botão “Import”, selecionar o arquivo/caminho que você deseja.
* Para exportação, é necessário clicar nos 3 pontinhos que ficam ao lado da sua pasta e clicar na opção export. Em seguida, será aberta uma caixa com opções do tipo de arquivo que quer criar, logo será aberto uma caixa para você salvar seu arquivo.

Como realizar a criação de todo o banco e rodar o projeto no backend

* Para isso damos um Scaffold

Exibir -> Outras Janelas -> Console do Gerenciador de Pacotes

Por Exemplo : “Scaffold-DbContext "Data Source=.\SqlExpress;Initial Catalog=T\_OpFlix;User Id=sa;Pwd=132;" Microsoft.EntityFrameworkCore.SqlServer -OutputDir Domains -ContextDir Contexts -Context OpFlixContext”

Quais ferramentas foram utilizadas

* Visual Studio 2017
* Microsoft Sql Server Management Studio 18
* Postman
* Swagger Ul

Funcionalidades

Web

Mobile

Protótipos

Web

Mobile

Front-End

Mobile

* Página de Login

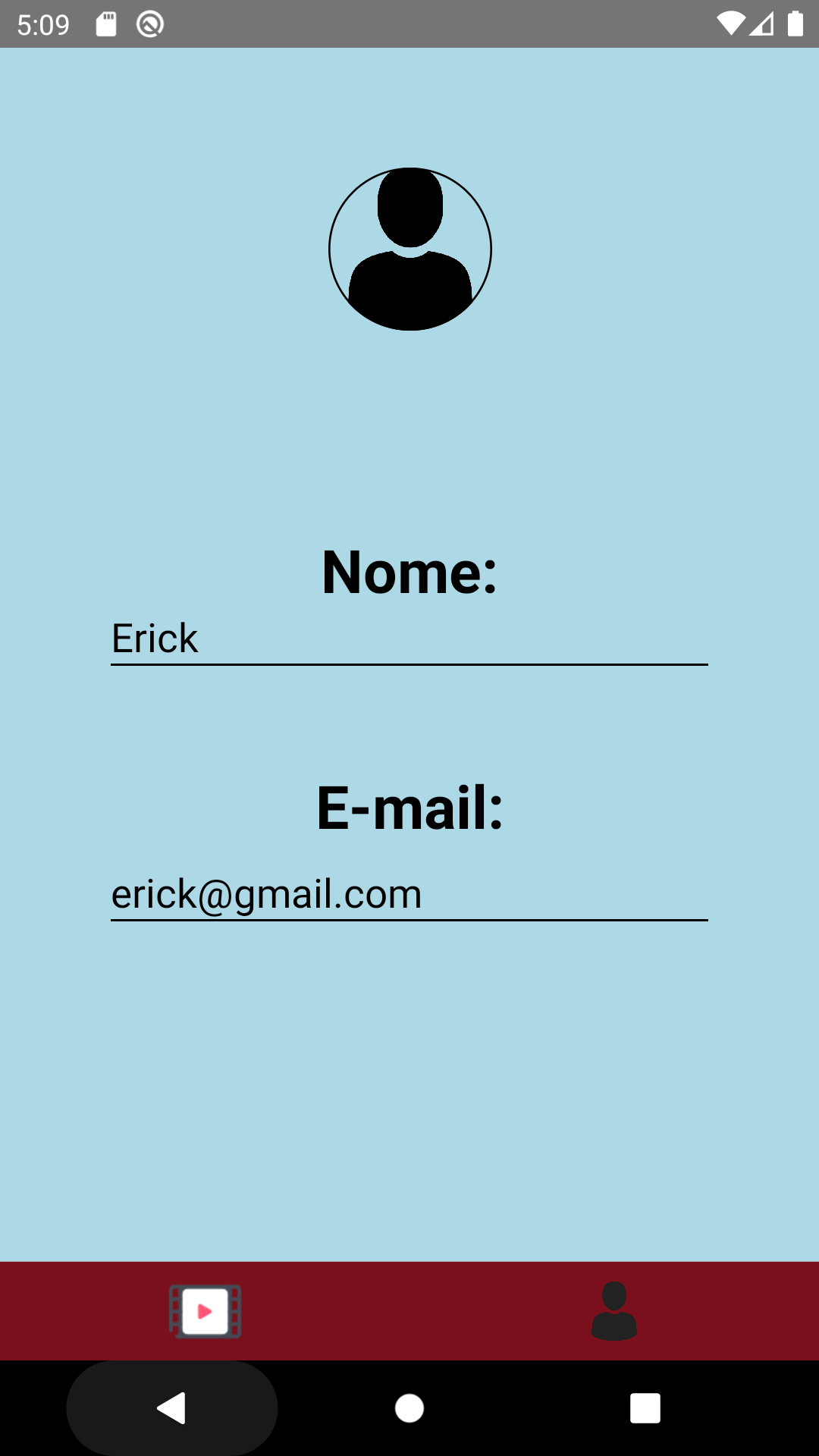


* Página de Listagem de Lançamentos



* Página de Cadastro de Usuário

* Dados do usuário



Arquitetura do Projeto

Referências

Links

Livros