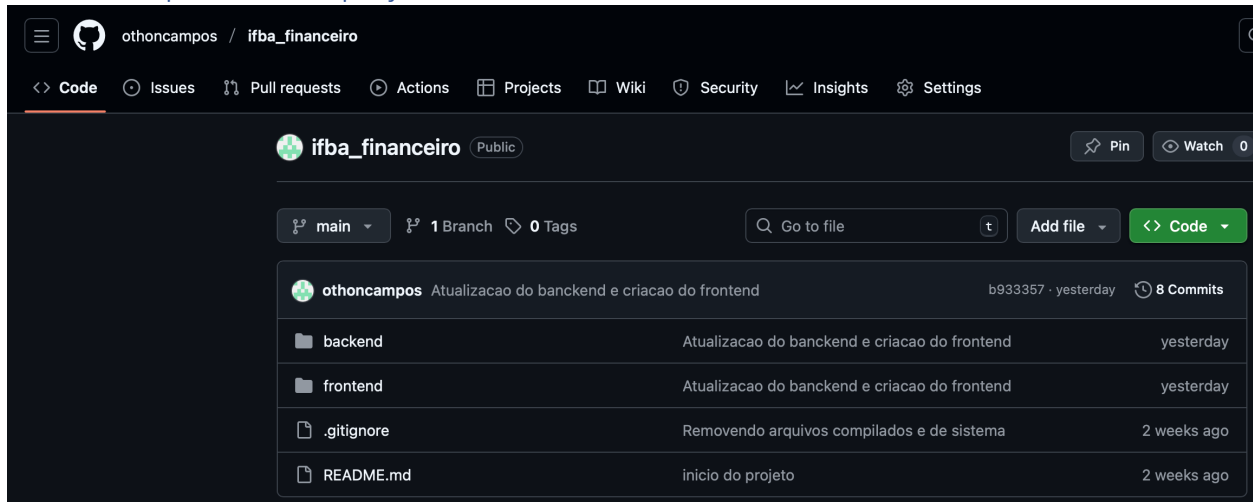


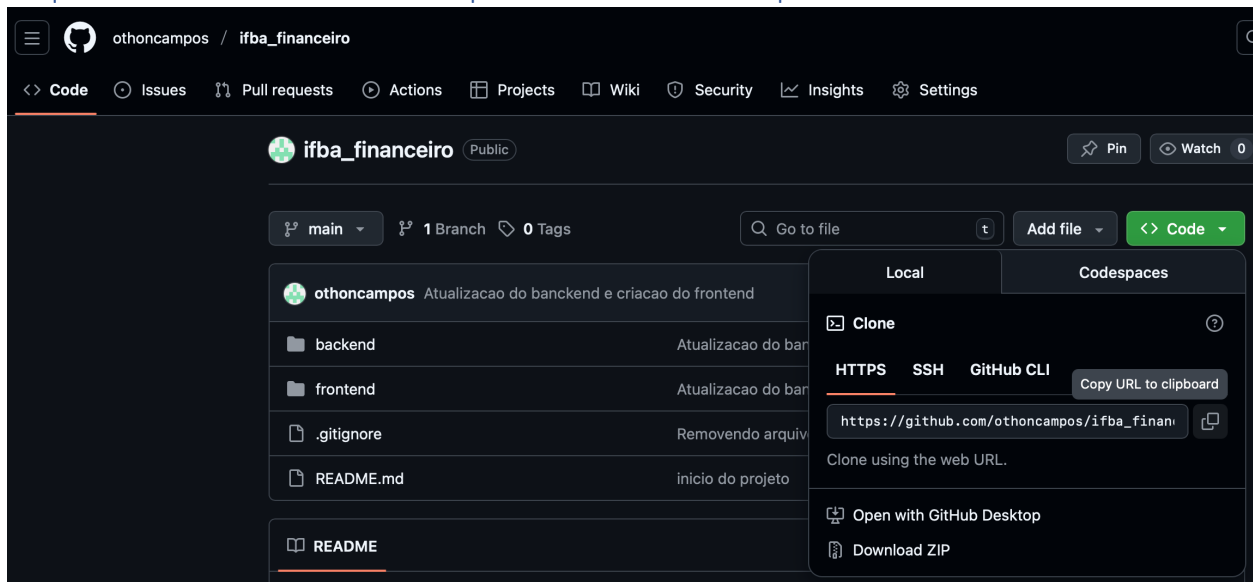
Roteiro de Instalação: IFBA Financeiro

1. Obter o projeto no GitHub

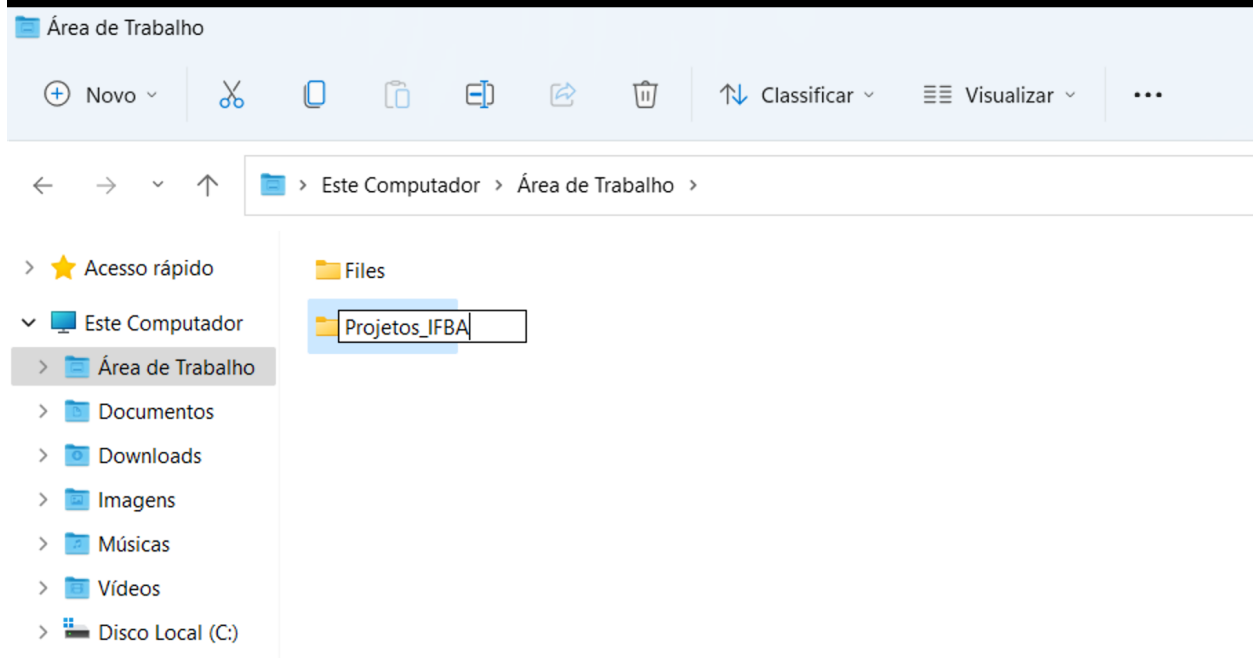
Acesse o repositório do projeto no GitHub



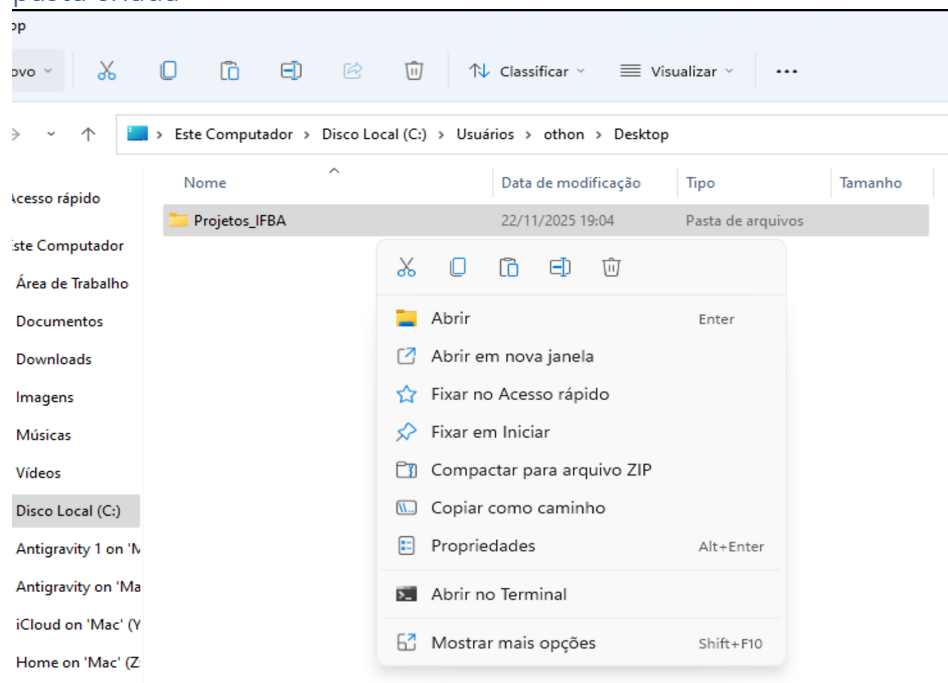
Clique no botão  e copie a URL HTTPS do repositório.



Crie uma pasta em seu computador, como por exemplo: projetos_ifba



Abra o **terminal** (Linux/macOS) ou **Command Prompt** (Windows) e navegue até essa pasta criada



No terminal, execute:

```
git clone https://github.com/othoncampos/ifba_financeiro.git
```

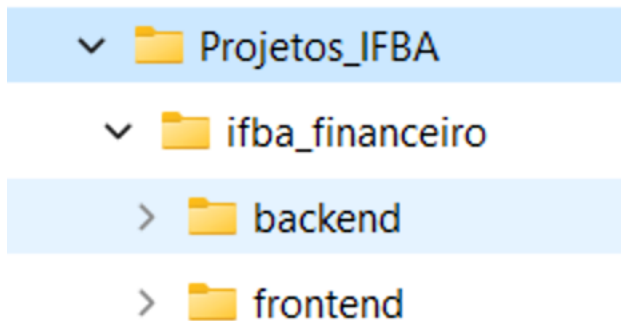
```
Desktop
Windows PowerShell
0 Windows PowerShell
Copyright (C) Microsoft Corporation. Todos os direitos reservados.

Instale o PowerShell mais recente para obter novos recursos e aprimoramentos! https://aka.ms/PSWindows

PS C:\Users\othon\Desktop\Projetos_IFBA> git clone https://github.com/othoncampos/ifba_financeiro.git
Cloning into 'ifba_financeiro'...
remote: Enumerating objects: 207, done.
remote: Counting objects: 100% (207/207), done.
remote: Compressing objects: 100% (144/144), done.
remote: Total 207 (delta 39), reused 189 (delta 24), pack-reused 0 (from 0)
Receiving objects: 100% (207/207), 142.94 KiB | 189.00 KiB/s, done.
Resolving deltas: 100% (39/39), done.
PS C:\Users\othon\Desktop\Projetos_IFBA> |
```

O Git fará o download do projeto, que terá a seguinte estrutura:

- **backend/** : código da API em **Java (Spring Boot)**
- **frontend/** : aplicação web feita em **Vue 3 + Vuetify**



2. Configuração do Banco de Dados (PostgreSQL)

- Crie um banco no PostgreSQL com o nome: **finanças**
- Execute o script SQL abaixo para criar as tabelas necessárias:

```
CREATE TABLE public.banco (  
    id_banco int4 GENERATED ALWAYS AS IDENTITY NOT NULL,  
    codigo int4 NULL,  
    nome varchar(50) NOT NULL,  
    url varchar(200) NOT NULL,  
    CONSTRAINT banco_pkey PRIMARY KEY (id_banco)  
);  
  
CREATE TABLE public.categoria (  
    id_categoria int4 GENERATED ALWAYS AS IDENTITY NOT NULL,  
    nome varchar(100) NOT NULL,  
    tipo_conta bpchar(1) NULL,  
    CONSTRAINT categoria_pkey PRIMARY KEY (id_categoria)  
);  
  
CREATE TABLE public.forma_pagamento (  
    id_forma_pagamento int4 GENERATED ALWAYS AS IDENTITY NOT NULL,  
    descricao varchar(50) NOT NULL,  
    CONSTRAINT forma_pagamento_pkey PRIMARY KEY (id_forma_pagamento)  
);  
  
CREATE TABLE public.usuario (  
    id serial4 NOT NULL,  
    cpf varchar(14) NOT NULL,  
    nome varchar(100) NOT NULL,  
    senha varchar(100) NULL,  
    dt_nasc date NULL,  
    CONSTRAINT usuario_pkey PRIMARY KEY (id)  
);  
  
INSERT INTO public.usuario (id, cpf, nome, senha, dt_nasc)  
VALUES (3, '1', 'nome_1', '123456', '1999-12-31');
```

3. Ajustes no Frontend

- Dentro da pasta frontend, crie um arquivo chamado: **.env**
- Com o seguinte conteúdo:
VITE_APP_URL_SERVER=http://localhost
VITE_APP_PORT_SERVER=8081

4. Configuração do Backend

- Abra o arquivo “backend/src/main/resources/application.properties” e verifique/ajuste:
spring.application.name=financeiro
server.port=8081
spring.datasource.url=jdbc:postgresql://localhost:5432/financas
spring.datasource.username=postgres
spring.datasource.password=postgres
spring.datasource.driver-class-name=org.postgresql.Driver

5. Instalar dependências e iniciar aplicações

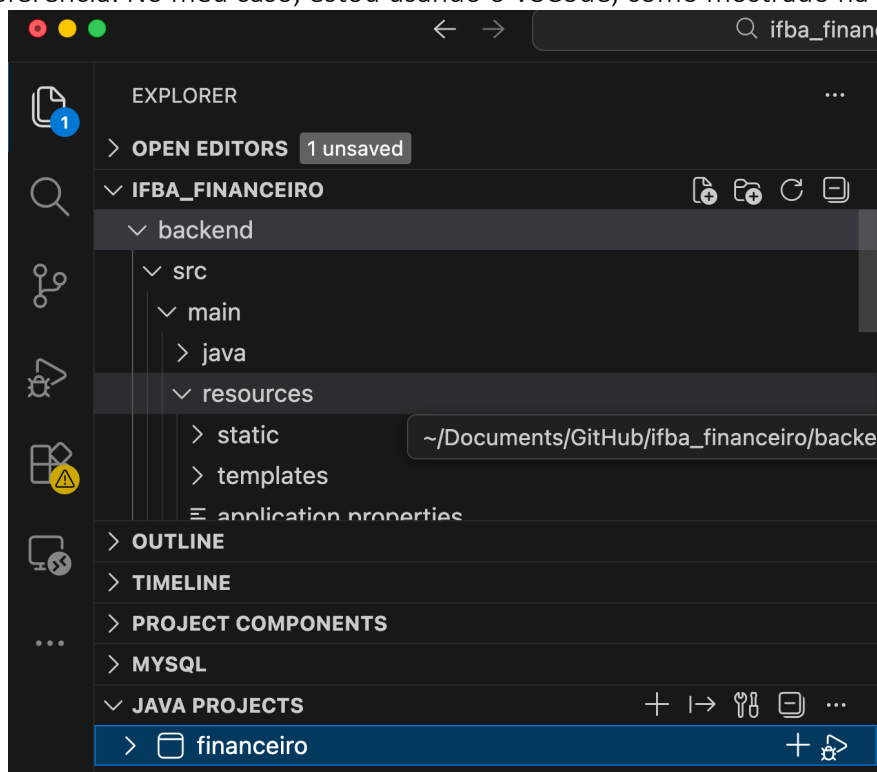
Frontend

No terminal, dentro da pasta frontend, execute:

```
npm install  
npm run dev
```

Backend

Inicie o projeto Java pela sua IDE (IntelliJ, Eclipse, VS Code) ou usando Maven/Gradle, conforme sua preferência. No meu caso, estou usando o VSCode, como mostrado na figura abaixo:



6. Acessar a aplicação

Se tudo estiver funcionando corretamente, abra no navegador a URL <http://localhost:3000>

A interface do sistema estará disponível.