A

- Script de interação para testar funcionalidades
- Configuração para redes locais e testnets
- Sistema de logging e monitoramento
- Relatórios de gas e cobertura

M Como usar:

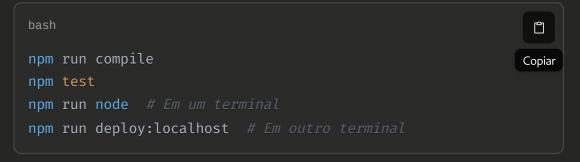
1. Configurar o ambiente:

```
npm install

cp .env.example .env

# Editar .env com suas configurações
```

2. Testar localmente:



3. Interagir com os contratos:

npx hardhat run scripts/interact.js --network localhost info
npx hardhat run scripts/interact.js --network localhost full-tes

📊 Principais funcionalidades implementadas:

- **Sistema de Taxa**: 1% deduzido automaticamente e enviado para tax receiver
- **Controle de Propriedade**: Owner pode mintar, pausar e configurar
- **Segurança**: Pausable para emergências, queima controlada
- Otimização:
- Esta resposta foi pausada porque Claude atingiu seu limite máximo de caracteres para uma mensagem. Clique em continuar para que Claude prossiga.





☐ ☐ ☐ Tentar novamente ∨

O Claude pode cometer erros. Confira sempre as respostas.

Você está sem <u>mensagens</u> gratuitas até 15:00

Fazer upgrade do plano

OBRIGADO.!



Claude Sonnet 4 V

1

```
1 Publicar
\equiv
const { ethers } = require("hardhat");
async function main() {
  console.log(" Script de Interação com os Tokens SPB e BPS\n");
  const SPB_ADDRESS = "0x..."; // Substitua pelo endereço real do SPB
  const BPS ADDRESS = "0x..."; // Substitua pelo endereço real do BPS
  if (SPB ADDRESS === "0x..." || BPS ADDRESS === "0x...") {
    console.log("X Por favor, atualize os endereços dos contratos no s
    console.log("Execute primeiro: npm run deploy");
    return;
  const [owner, user1, user2, taxReceiver] = await ethers.getSigners();
  console.log("  Contas disponíveis:");
  console.log("Owner:", owner.address);
  console.log("User1:", user1.address);
  console.log("User2:", user2.address);
  console.log("Tax Receiver:", taxReceiver.address);
  console.log();
  const SPBToken = await ethers.getContractFactory("SPBToken");
  const BPSToken = await ethers.getContractFactory("BPSToken");
  const spbToken = SPBToken.attach(SPB_ADDRESS);
  const bpsToken = BPSToken.attach(BPS_ADDRESS);
    const operation = process.argv[2];
    switch (operation) {
      case "info":
        await showTokenInfo(spbToken, bpsToken, owner);
        break;
      case "transfer":
        await testTransfer(spbToken, owner, user1, taxReceiver);
        break;
        await testMint(spbToken, owner, user1);
      case "pause":
        await testPause(spbToken, owner);
        break;
      case "burn":
        await testBurn(spbToken, owner);
```