Sperax Smart Contract Test Documentation

Project Overview

This project is a Solidity-based smart contract that allows minting, transferring tokens, and retrieving the balance of a given address. The project is developed using the Hardhat development environment.

Prerequisites

Before running or modifying this project, ensure you have the following installed:

- Node.js (v20.x or later)
- npm
- Hardhat

Installation

1. Clone the repository:
   ```
   git clone https://github.com/techpilot/sperax-test.git
   ```
2. Navigate to the project directory:
   ```
   cd sperax-test/smart-contract
   ```
3. Install the dependencies:
   ```
   npm install
   ```
4. Compile the smart contract:
   ```
   npx hardhat compile
   ```

Configuration

- **Solidity version**: 0.8.24
- **Networks**:
  - Sepolia
  - Hardhat (Localhost)

Smart Contract Overview

The main smart contract is `Token.sol`, which includes the following functionalities:

1. Mint Tokens

- **Function**: `mint(address to, uint256 amount)`
- **Description**: Mints `amount` of tokens to the specified address `to`.
- **Access Control**: Only the contract owner can mint new tokens.

2. Send Tokens

- **Function**: `sendToken(address to, uint256 amount) public`
- **Description**: Transfers `amount` of tokens from the caller's account to the specified address `to`.

3. Check Balance

- **Function**: `balanceOf(address account) public view returns (uint256)`
- **Description**: Returns the token balance of the provided address `account`.

4. Transfer Tokens

- **Function**: `tokenTransfer(address to, uint256 amount) public`
- **Description**: Transfers `amount` of tokens from the contract address that minted the tokens to the specified address `to`.

5. Reentrancy Guard

- **Modifier**
- **Description**: Prevents reentrancy during certain functions pausing the function once it is already running

6. Only Owner

- **Modifier**
- **Description**: Limits a function to be called only by the address that deployed the contract

Deployment

To deploy the contract on a local Hardhat network:

1. Start a local Hardhat node:
   `npx hardhat node`

2. Deploy the contract:
   `npx hardhat run scripts/deploy.ts --network localhost`

3. Deploy the contract on a testnet:
   `npx hardhat run scripts/deploy.ts --network sepolia`

Usage

After deployment, you can interact with the contract using frontend interface like React.

Example React Script

To token balance:

```
const handleGetBalance = async () => {
    try {
      // connects to the smart contract using WEb3.js
      const web3Instance = new Web3(window?.ethereum);
      const newContract = new web3Instance.eth.Contract(
      DesTokenContract.abi,
      DesTokenContract.address
      );
        // gets the address balance from blockchain
        const balance = await
newContract?.methods.getBalance(address).call();
        setBalance(balance);
         } catch (error: any) {
          console.log(error)
         }
  };
```

License

This project is licensed under the MIT License.