Bytus Token Admin Dashboard - Deployment Instructions

Overview

This admin dashboard provides a complete interface for interacting with the Bytus Token smart contract. It allows for testing all contract functions including transfers, approvals, burning tokens, and administrative controls like pausing/unpausing the contract.

Deployment Steps

1. Set Up a Local Blockchain Environment

To test locally, you'll need a local Ethereum development environment:

Option A: Hardhat

```bash
# Install Hardhat
npm install --save-dev hardhat

# Initialize a Hardhat project
npx hardhat init

# Start a local node
npx hardhat node
```

Option B: Ganache

```bash
# Install Ganache
npm install -g ganache-cli

# Start Ganache
ganache-cli
```

2. Deploy the Bytus Token Contract

1. Compile the updated BytusToken contract:

```bash
npx hardhat compile
```

2. Create a deployment script (e.g., `scripts/deploy.js`):

```javascript
async function main() {
  const [deployer] = await ethers.getSigners();
  console.log("Deploying contracts with the account:", deployer.address);

  const BytusToken = await ethers.getContractFactory("BytusToken");
  const token = await BytusToken.deploy(
    66000000, // 66 million initial supply (before decimals)
    "Bytus Token", // Name
    "BYTS" // Symbol
  );

  await token.deployed();
  console.log("BytusToken deployed to:", token.address);
}

main()
  .then(() => process.exit(0))
  .catch((error) => {
    console.error(error);
    process.exit(1);
  });
```

3. Run the deployment script:

```bash
npx hardhat run scripts/deploy.js --network localhost
```

4. Note the deployed contract address for use in the dashboard.

3. Set Up a Local Web Server

You can use any simple HTTP server to serve the dashboard. Here are a few options:

Option A: Node.js http-server

```bash
# Install http-server
npm install -g http-server

# Start server in the directory with the dashboard HTML file
http-server -p 8080
```

Option B: Python's built-in HTTP server

```bash
# For Python 3
python -m http.server 8080
```

4. Access the Dashboard

1. Open your browser and navigate to `http://localhost:8080`
2. Connect your wallet (MetaMask recommended)
3. Enter the contract address in the "Admin Controls" section
4. Now you can interact with all contract functions

Using the Dashboard

Main Features

1. **Overview Panel**
   - View token details (name, symbol, decimals, total supply)
   - Check your account balance and owner status
   - View recent transactions
2. **Transfer Panel**
   - Send tokens to any address
3. **Approve & Transfer From Panel**
   - Approve other addresses to spend your tokens
   - Use the approveAndCall function for contracts
   - Check existing allowances
   - Transfer tokens on behalf of others (if approved)
4. **Burn Panel**

    - Burn your own tokens
    - Burn tokens from other addresses (if approved)

5. **Admin Controls Panel**
    - Pause/unpause the contract (owner only)
    - View contract information

Connecting to Networks

The dashboard supports connecting to:

- Localhost (for development)
- Goerli Testnet
- Sepolia Testnet
- Ethereum Mainnet

To connect to public networks, you'll need to add your Infura or Alchemy API key in the `networkInfo` object in the JavaScript code.

Security Considerations

- This dashboard is intended for testing and administrative purposes only
- For production use, implement additional security measures such as:
    - Multi-signature capabilities for administrative functions
    - Rate limiting
    - IP restrictions
    - Additional authentication methods

Troubleshooting

1. **Connection Issues**
    - Ensure MetaMask is installed and unlocked
    - Make sure you're connected to the correct network
2. **Transaction Failures**
    - Check the JavaScript console for detailed error messages
    - Verify you have enough ETH for gas fees
    - For admin functions, ensure you're using the owner address
3. **Contract Interaction Failures**

- Verify the contract address is correct
- Check the browser console for specific error messages
- If the contract is paused, most operations will fail