Blockchain & Solidity Lab3 – Voting dApp Development

S2BC

Lab 3: Integrate Web App with Smart Contracts

- BUILD / TEST / **INTEGRATE** / RUN

Objective: The aim of this Lab3 is to integrate the smart contracts you developed in Lab1 and Lab2 with a Voting dApp for users to access the dApp using the web browser.

Setting Up the Frontend

In this section, we will guide you through setting up the frontend of your Voting dApp. Follow these steps to create the necessary folders and files:

1. Create a Frontend Folder

Begin by creating a folder named `frontend` within your project directory. This folder will house all the files related to the frontend of your dApp.

2. Create a Public Folder

Inside the `frontend` folder, create a subfolder named `public`. This folder will hold any publicly accessible files, such as HTML and images.

3. Create HTML and JavaScript Files

Within the `public` folder, create the following files:

- `index.html`: This is the main HTML file that will serve as the entry point of your web application. You can use this file to structure the layout of your dApp's user interface.

```
<!-- Add your HTML code here -->
```

- `script.js`: This JavaScript file will contain the client-side code that interacts with the smart contracts and updates the UI based on user actions.

```
// Add your JavaScript code here
```

4. Set Up a Server (Optional)

If your dApp requires server-side functionality, create a file named `server.js` in the `frontend` folder. This file will handle any backend logic your application may need.

5. Configure Environment Variables

To securely manage sensitive information, it’s best practice to use environment variables. Create a file named `.env.example` in the root directory of your project. This file will serve as an example template for your environment variables. Include any placeholders or default values that your application requires.
Here’s an example of how you can structure your `.env.example` file:

```
# This is the URL of the Ethereum RPC provider
RPC_URL="https://example.com/rpc"

# This is a private key for signing transactions
PRIVATE_KEY="your_private_key_here"

# This is an API key for accessing a specific service
API_KEY="your_api_key_here"

# This is the chain ID for the Ethereum network
CHAIN_ID=12345

# This are the addresses of smart-contracts
CONTRACT_ADDRESS='0x1234567890abcdef'
NFT_CONTRACT_ADDRESS='0x1234567890abcdef'
```

Diagram: Frontend Folder Structure

Frontend Folder Structure Diagram

Lab 3: Interacting with the Smart Contract

1. Connecting the Wallet

- Explain the purpose of this function and its significance in blockchain development.
- Emphasize the security aspect of connecting to a wallet and handling accounts.

```
// Add your code snippet for connecting the wallet here
```

2. Starting an Election

- Describe the process of starting an election, including providing candidate names and specifying the duration.
- Highlight any validations or checks that are in place to ensure a smooth election setup.

```
// Add your code snippet for starting an election here
```

3. Voting

- Walk through the process of casting a vote, emphasizing the importance of secure and transparent voting mechanisms.
- Explain any error handling or feedback mechanisms for incorrect inputs.

```
// Add your code snippet for the voting process here
```

4. Changing Election Duration

- Explain the purpose of this functionality and its potential use cases.
- Discuss any considerations regarding the duration of an election.

```
// Add your code snippet for changing election duration here
```

5. Resetting the Election

- Describe the purpose of resetting an election and any safeguards in place to prevent accidental resets.

```
// Add your code snippet for resetting an election here
```

6. Adding Candidates

- Walk through the process of adding candidates to an election, and discuss how this might be relevant in a real-world scenario.

```
// Add your code snippet for adding candidates here
```

7. Displaying Candidates

- Explain how the UI displays the list of candidates and the associated information.

```
// Add your code snippet for displaying candidates here
```

8. Showing the Timer

- Describe how the timer functionality works, and its significance in the context of an election.

```
// Add your code snippet for timer functionality here
```

9. Handling Events

- Discuss how events are used to track various actions within the smart contract and how they can be beneficial for auditing.

```
// Add your code snippet for handling events here
```

10. Minting NFTs

- Explain the concept of minting NFTs as a representation of election results.
- Highlight the potential use cases or benefits of issuing NFTs in this context.

```
// Add your code snippet for minting NFTs here
```

Diagram: Interaction Flow

Interaction Flow Diagram

General Tips:

- Provide clear explanations for each function or action, including its purpose and potential use cases.
- Include comments in the code to explain complex logic or any specific considerations.
- Encourage students to test each functionality with different inputs to gain a better understanding.
- Consider adding error handling mechanisms to gracefully handle unexpected situations.

Contact Information

S2BC