Frontend Development for DApps This assignment is exclusively for students in Abzal Kyzyrkanov's teaching groups.

Objective:

Develop a frontend for a decentralized application (DApp). Choose one of the two options: create a frontend for your final project's smart contract or for the smart contract provided in Week 6 practice.

Option 1: Frontend for Your Final Project's Smart Contract

Smart Contract Requirements:

1. **Struct Inclusion:** Must have at least one struct used in fields, function arguments, or return types.
2. **Events:** At least 5 events, with 3 having arguments.
3. **Functions:** A minimum of 5 external/public functions, 3 being state-changing and 1 requiring payment (msg.value).
4. **Function Restrictions:** At least 3 public functions with conditions (**require(condition, 'msg')**).
5. **Composed Type Variables:** At least 1 public function should take and return a composed type variable.
6. **Relevance:** All elements must be essential to the smart contract.
7. **Necessity of Types and Structs:** Types and structs should be created out of necessity, not just to fulfill the assignment requirements.

Team Submission Requirements:

1. **Individual Contributions:**
    - Each team member must develop a frontend for at least one event, one state-changing method, and one read-only function.
    - No overlapping of functions or events among team members.
    - All functions and events must be covered.
2. **Function Requirements Coverage:**
    - Each member's function must satisfy the following:
        - Has restrictions (**require()**).
        - Accepts composed type arguments.
        - Returns composed type values.
        - Is either state-changing or payable.
        - Is read-only (pure or view).
3. **Event Handling:**

- Each member must develop a frontend for at least one event with arguments, including dynamic updates based on emitted events.

4. **Diversity of Contributions:**
    - Different events and functions for each team member. If insufficient functions or events are available, Option 2 must be chosen.

Option 2: Frontend for Week 6 Practice Contract

Individual Task Requirements:

- Deploy the contract from Week 6 on a network (test or local).
- Develop an application covering the full functionality of the provided contract.

General Requirements for Both Options:

- **Web Frontend:** Acceptable in any form (React, raw HTML/JS), ensuring user-friendliness and aesthetic appeal.
- **Page Requirements:** At least one full-self-sufficient, easy-to-use page.
- **Functionality:** All elements must be functional; remove non-functional elements.

Frontend for Contract Events:

1. Display all historical events upon loading.
2. Display all emitted events immediately.

Frontend for Contract Functions:

1. Prevent multiple submissions (e.g., double-clicking).
2. Avoid blocking the entire interface during smart contract interactions.
3. Implement comprehensive error handling:
    - Errors from **require()**.
    - Connection loss.
    - User action rejection.
4. User input for most arguments, with suitable input types.
5. Auto-fill inputs upon page reload.
6. Allow users to add to list values.
7. Display payment amounts where applicable.
8. Reflect user restrictions in the interface (e.g., disable vote button after voting).

Coverage:

- All external/public functions and events must be integrated into the frontend.
- Include a button for network connection and switching.

Submission:

- **Report:** Submit a report detailing the frontend's functionality and design, including screenshots of the interface demonstrating the successful execution of smart contract functions.
- **Documentation:** Full documentation is not required; however, ensure the report is comprehensive and clearly explains the frontend components and their interactions with the smart contract.
- **Presentation:** Be prepared to run and demonstrate your project during Week 9 classes. Your demonstration should be a complete walkthrough of the frontend, showcasing its functionality and interaction with the smart contract.

Grading Criteria:

Grading will focus on the fulfillment of the general requirements, emphasizing the functionality, user experience, and robustness of the frontend application.

1. **User Interface and User Experience (UI/UX) - Total 30%:**
    - **Aesthetics (10%):**
        - Evaluate the visual appeal, professionalism, and cleanliness of the interface.
    - **Intuitiveness (10%):**
        - Assess how user-friendly and navigable the interface is, especially for those unfamiliar with DApps.
    - **Responsiveness (10%):**
        - Judge how well the interface adapts to different devices and screen sizes.
2. **Functionality and Feature Completeness - Total 25%:**
    - **Implementation of General Requirements (15%):**
        - Check the completeness and correctness of implemented features (input types, event handling, etc.).
    - **Coverage (10%):**
        - Evaluate how comprehensively the frontend covers all the external/public functions and events of the smart contract.
3. **Event Handling and Real-time Updates (15%):**
    - **Historic Events and Real-time Display:** Assess the accuracy and immediacy of event display and updates.

4. **Error Handling and Robustness (20%):**
    - **Comprehensiveness and User Guidance:** Evaluate the gracefulness of error handling and the clarity of feedback provided to users.
5. **Input Handling and Validation (10%):**
    - **Accuracy and Persistence:** Assess the correctness of input validation and the persistence of application state across sessions.

Penalties:

- **Reattempting/Retrying (10%):** If a project requires multiple attempts or retries during the presentation or does not function as expected on the first attempt, a penalty will be applied.
- **Not Meeting Requirements (10%):** Failure to meet the specified assignment requirements will result in a penalty.

Note: It's crucial to ensure that your project is well-prepared and thoroughly tested before the presentation. The ability to effectively demonstrate and explain your project's features and the rationale behind your design choices will be integral to the grading process. The penalties are in place to encourage thorough preparation and adherence to the assignment requirements.