Question 2: Implementing a ZK-Based Authentication System Using zkAuth Kit

Objective: Develop a front-end authentication system using the provided zkAuth kit to implement a zero-knowledge (ZK) proof-based login. Candidates will integrate the provided WebAssembly (WASM) compiled JavaScript files with the UI to simulate a registration and login process. Clone the given pkg folder onto the src folder where you want to integrate. Also, have a look at the given example on how to import wasm_js files.

Requirements:

1. Registration Flow:

- Build a registration page where users can input their desired username and password.
- Use the `get_pass_hash` function from the provided WASM JS file to generate a hashed password for the entered password.
- This hashed password should then be saved as part of the simulated backend process. For this test, saving the hash in any DB is sufficient to represent the backend storage.

2. Login Flow:

- Create a login page where users can input their username and password.
- When users attempt to log in, call the `generate_proof` function from the provided WASM JS file. This function should generate a ZK proof for the provided username and password in the front end.
- Send this ZK proof along with the username to the simulated backend (also stored in local storage).
- Use the `verify_proof` function in the WASM JS file to check the proof against the hash stored for the given user in DB in the backend. Display an alert indicating whether login is successful or failed based on verification.

Evaluation Criteria:

- **Functionality**: The registration and login flow should work seamlessly, accurately storing the password hash and verifying it with ZK proof.
- **Code Structure**: Code should be modular and well-organized, with clear separation between frontend and WASM calls.
- **Error Handling**: Implement validation and error handling (e.g., show errors if the proof fails or if inputs are invalid).