Frontend Documentation: Login/Register Component

Jarpula Bhanu Prasad
AI21BTECH11015

Kola Akshitha
AI21BTECH11017

1 Overview

The `LoginRegister` component is a React-based UI that handles both login and registration processes, including voice authentication. The component utilizes `styled-components` for styling and includes user authentication via voice input, registration forms, and state management for login failures, registration, and microphone handling.

2 Key Features

- **Login Form**: Allows users to log in using their username and password or via voice authentication.
- **Registration Form**: Collects user details (username, email, password) and a voice sample for registration.
- **Voice Authentication**: Users can authenticate via a voice sample for both login and registration processes.
- **Responsive UI**: The UI adjusts to show either the login or registration form depending on user interaction.

3 State Variables

- `action`: Determines whether the login or registration form is active.
- `loginFailedAttempts`: Tracks the number of failed login attempts.
- `agreeTnC`: Indicates whether the user agrees to the terms and conditions during registration.
- `recording`: Indicates whether the microphone is currently recording.
- `timer`: Tracks the countdown time for the recording (set to 10 seconds).

- `mediaRecorder`: Stores the instance of the `MediaRecorder` for handling voice recording.
- `audioChunks`: Stores the chunks of audio data captured during recording.

4 Key Methods

- `handleStartRecording`: Starts the voice recording by accessing the user's microphone and initializes a `MediaRecorder` instance. Records audio for 10 seconds and stores the audio chunks.
- `handleStopRecording`: Stops the voice recording and resets the timer. Invoked when the recording reaches 0 seconds.
- `sendAudioToBackend`: Sends the captured audio blob to the backend API for processing (e.g., voice authentication or registration). Uses `fetch` to send the audio data to the configured API endpoint.
- `handleVoiceLogin`: Starts a 5-second voice recording and sends the captured audio to the backend for voice authentication during login. Increments `loginFailedAttempts` if authentication fails.
- `sendRegistrationData`: Sends the registration details (username, email, password) along with the voice sample to the backend API for registration.
- `handleRegister`: Handles the registration form submission. Collects form data and audio chunks, then sends them to the backend for registration.
- `registerLink`: Switches the view to the registration form.
- `loginLink`: Switches the view to the login form.

5 User Flow

5.1 Login

- The user can either log in using the "Username" and "Password" fields or authenticate using their voice.
- If the user has failed to log in more than three times, they are prompted to enter their username and password instead of using voice authentication.
- Upon successful login via voice or credentials, the user is redirected to the profile page.

5.2 Registration

- The user fills out the registration form with their username, email, and password.
- The user records a voice sample for registration.
- Upon submission, the registration data (including voice sample) is sent to the backend for processing.
- If registration is successful, the user is notified; otherwise, an error message is displayed.

6 Backend Integration

The frontend interacts with various backend endpoints for login, registration, and voice authentication:

- **Login Endpoint**: Handles voice-based or standard login.
- **Register Endpoint**: Handles user registration with voice sample.
- **Upload Voice Endpoint**: Handles the upload of audio for backend processing.

7 Tools and Libraries

- **React**: For the component-based structure and state management.
- **Styled-components**: For styling the components.
- **React-icons**: For the icons used in the forms.
- **MediaRecorder API**: For handling voice recording.
- **Fetch API**: For interacting with the backend endpoints.

8 Notes

- The voice recording functionality uses the `MediaRecorder API`, which is dependent on browser support for audio recording.
- The component allows switching between login and registration forms without reloading the page, providing a seamless user experience.