**Assignment-6**

**JavaScript Validation**

The script validates form inputs for a user registration form. It ensures that the provided data meets specific criteria before allowing the form to be submitted. The validation is done in real-time, meaning the user receives feedback as they type.

**Key Elements and Functions**

1. **Element Selection**
   * The script begins by selecting important elements from the DOM using document.getElementById():
     + nameInput, emailInput, passwordInput, confirmPasswordInput, dateInput: These correspond to the input fields in the form.
     + submitButton: The form submission button.
     + signInButton, signInForm: Elements related to toggling between the registration and sign-in forms.
2. **Validation Functions**
   * Each validation function checks a specific input field against a set of rules. If the input is valid, the field is marked as valid; otherwise, it is marked as invalid.
   * **validateName()**:
     + **Rule**: The name must consist of at least 3 alphabetic characters (letters and spaces only).
     + **Regex**: /^[A-Za-z ]{3,}$/
     + **Feedback**: Adds is-valid or is-invalid class based on validity.
   * **validateEmail()**:
     + **Rule**: The email must follow the standard email format (e.g., user@example.com).
     + **Regex**: /^[^\s@]+@[^\s@]+\.[^\s@]+$/
     + **Feedback**: Similar to the name validation, the email field is marked valid or invalid.
   * **validatePassword()**:
     + **Rule**: The password must be at least 8 characters long, containing at least one lowercase letter, one uppercase letter, one digit, and one special character.
     + **Regex**: /^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*\d)(?=.\*[@$!%\*?&])[A-Za-z\d@$!%\*?&]{8,}$/
     + **Feedback**: Applies the appropriate class based on whether the password meets the criteria.
   * **validateConfirmPassword()**:
     + **Rule**: The confirmation password must match the password entered in the previous field.
     + **Feedback**: Ensures both passwords match and provides corresponding feedback.
   * **validateDate()**:
     + **Rule**: The user must be at least 18 years old.
     + **Logic**: Calculates the user's age based on the entered date of birth and compares it with the current date.
3. **Form Submission Handling**
   * **checkFormValidity()**:
     + This function calls each of the individual validation functions.
     + If all fields are valid, the submitButton is enabled; otherwise, it is disabled.
   * **form.addEventListener('submit', (event) => {...})**:
     + **Purpose**: This event listener prevents the default form submission behavior if the form is invalid. If valid, it resets the form and shows a success alert.
4. **Real-Time Validation**
   * Event listeners are added to each input field using .addEventListener('input', ...).
   * These listeners trigger the respective validation function and then recheck the entire form's validity.
5. **Sign-In Form Toggle**
   * **signInButton.addEventListener('click', () => {...})**:
     + This toggles between the registration and sign-in forms by showing or hiding the relevant sections of the page.