LAB 6

Self Learning : Tailwind CSS

Implementing a User Registration Form with Validation for "YOUR DOMAIN"

You're part of a development team responsible for creating a user registration form for a new website. The form will collect essential user information, ensuring all data is valid and meets specific criteria before submission. Accept only alphabetic characters and spaces, with a minimum of 3 characters. Validate for proper email format (e.g., "user@example.com"). Ensure the password is at least 8 characters long and contains both letters and numbers. This field must match the password entered. The date must be in "YYYY-MM-DD" format, and the user must be at least 18 years old. Implement form validation using JavaScript, focusing on delivering a user-friendly experience by appropriately handling events and providing clear feedback. Create the HTML structure for the registration form, including input fields for each of the above requirements, labels, and a submit button. Implement validation that checks each field upon form submission. If any field fails validation, display an error message next to it and prevent form submission until all fields are valid. As the user interacts with the form, provide real-time feedback. For example, show a green checkmark for valid inputs and a red "x" for invalid ones. This should update dynamically as the user types. Continuously check if the "Password" and "Confirm Password" fields match, and display a message if they don't. Calculate the user's age based on the "Date of Birth" field, and disable the submit button if the user is under 18. Attach event listeners to the form elements to trigger validation and provide immediate feedback as the user interacts with the form. Design the form to visually represent validation states. Use styles such as red borders for invalid inputs, green borders for valid inputs, and clear error messages to guide the user. You may use plain JavaScript or leverage libraries/frameworks of your choice to accomplish these tasks.

Document your code and explain the rationale behind your design and implementation decisions.

I've implemented a user registration form for an online Book store “Novel Nook” using bootstrap ,Tailwind CSS and JavaScript. The form includes input fields for essential user information, such as username, email, password, and date of birth, all of which are validated to ensure the data meets specific criteria.

The email field is validated for proper format using a regular expression. Passwords must be at least eight characters long and contain both letters and numbers, and the form checks if the "Password" and "Confirm Password" fields match.

The date of birth is validated to ensure the user is at least 18 years old, with the submit button disabled if the user is underage. Real-time feedback is provided through dynamic visual indicators, like green borders for valid inputs and red borders for invalid ones, giving users immediate feedback as they type. JavaScript event listeners are attached to the form elements to trigger validation and update the UI accordingly.

The form's design visually represents validation states with clear and responsive styling, making the process intuitive and user-friendly. Each piece of validation logic is documented to explain its purpose and the user experience design choices, ensuring the form is both functional and aligned with best practices for accessibility and user interaction.

**SIGN UP PAGE :**

This document provides an explanation of the HTML code used to create a user registration form for "NOVEL NOOK". The form is designed to collect user information, such as name, contact details, and other personal information.

**File Structure**

1. **DOCTYPE and HTML Declaration**
   * <!DOCTYPE html>: Declares the document type and version of HTML (HTML5).
   * <html lang="en">: The lang attribute specifies the language of the document as English.
2. **Head Section**
   * The <head> element contains meta information, stylesheets, and script references.
   * <meta charset="UTF-8">: Specifies the character encoding for the document.
   * <meta name="viewport" content="width=device-width, initial-scale=1.0">: Ensures proper scaling on different devices.
   * <title>User form</title>: Defines the title of the document displayed on the browser tab.
3. **Styles and Scripts**
   * <link rel="stylesheet" href="style.css">: Links to an external stylesheet (assumed to be in the same directory).
   * <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" rel="stylesheet">: Links to the Bootstrap 5.3 CSS for styling.
   * <link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-beta3/css/all.min.css" rel="stylesheet">: Links to Font Awesome for icon usage.
   * <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>: Includes jQuery for DOM manipulation.
   * <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.5.4/dist/umd/popper.min.js"></script>: Includes Popper.js for Bootstrap tooltips.
   * <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>: Links to Bootstrap 4.5 JavaScript for interactive components.

**Body Section**

1. **Header**
   * The header section contains the main title of the form, "NOVEL NOOK" and an instructional heading, "SIGN UP NOW!"
   * A decorative element with three circles connected by lines is included using <div class="draw">. The filled and unfilled classes likely represent different stages in a process (though styles are not detailed in this document).
2. **Form Structure**
   * The form is enclosed within a <div class="flex-container"> which is likely styled for a flexible box layout using CSS.
   * The form uses the GET method to submit data to "form2.html".
3. **Form Fields**
   * Each input field is wrapped in a <div class="flex-item"> for layout purposes. The fields include:
     + **Name** (<input type="text" name="f\_name" id="f\_name" required>): Text fields for user names.
     + **Contact Number** (<input type="number" name="contact\_no" id="contact\_no" required>): Number input for contact information.
     + **Email** (<input type="email" oninput="checkEmail()" name="email" id="email" required>): Email input with an oninput event to validate the format.
     + **Date of Birth** (<input type="date" name="dob" required oninput="checkAge()" id="dob" class="dob">): Date input with an oninput event to check if the user is 18 or older.
4. **Submit Button**
   * The submit button (<input type="submit" value="submit" id="submit" onclick="validateDob()" required>) triggers form submission and the validateDob function.

**JavaScript Functionality**

1. **validateDob()**
   * This function is intended to validate and retrieve form values but currently lacks implementation details for validation.
2. **checkEmail()**
   * This function checks if the email format is valid using a regular expression. If valid, the border color turns green; otherwise, it turns red.
3. **checkAge()**
   * This function calculates the user's age based on the date of birth and ensures that only users aged 18 or older can submit the form. The submit button is disabled and grayed out if the user is underage.

**Conclusion**

This HTML form is designed to collect user information with optional registration as a supplier. It uses Bootstrap for styling, Font Awesome for icons, and includes custom JavaScript functions for validation. The form is interactive, adapting to user inputs and providing visual feedback based on the data entered.

**INDEX.html**

This document is the home page which contains the nav bar implemented using HTML, Tailwind CSS, and JavaScript.