**Experiment 7**

**Aim:** Create a form and validate using JavaScript

**Learning Objectives:**

* Understand the role of JavaScript in client-side form validation to ensure user inputs meet required criteria before submission.
* Learn how to implement and customize validation rules using JavaScript to enhance the reliability and usability of web forms.

**Tools:** HTML, CSS, JavaScript, Text Editor/IDE

**Theory:**

Form validation is a crucial aspect of web development that ensures the data submitted by users is accurate, complete, and meets the required criteria before being processed by the server. JavaScript, a powerful client-side scripting language, is commonly used for this purpose due to its ability to provide immediate feedback to users without needing to refresh the page.

**Types of Validation**

There are two main types of form validation:

1. **Client-Side Validation:** This occurs in the user's browser, using JavaScript. It provides instant feedback by checking input values as soon as they are entered, such as ensuring an email address has the correct format or that a password meets complexity requirements.
2. **Server-Side Validation:** This validation happens on the server after the form has been submitted. Even with client-side validation, server-side validation is essential for security purposes, ensuring that only valid data is processed and stored.

**Benefits of Client-Side Validation**

* **Improved User Experience:** Users receive immediate feedback on their inputs, allowing them to correct errors without waiting for the form to be submitted.
* **Reduced Server Load:** By catching errors on the client side, unnecessary requests to the server are avoided, which can save resources and reduce processing time.
* **Enhanced Data Quality:** Ensuring that all required fields are filled out correctly and that inputs adhere to specific formats helps maintain the integrity and consistency of the data received.

**Implementing Validation with JavaScript**

JavaScript provides various methods and properties to validate form fields. Common validation tasks include:

* **Checking Required Fields:** Ensuring that fields such as names, emails, and passwords are not left blank.
* **Pattern Matching:** Using regular expressions to verify that inputs like email addresses, phone numbers, or ZIP codes match specific formats.
* **Length Validation:** Ensuring that inputs, such as usernames and passwords, meet minimum and maximum length requirements.
* **Numeric Validation:** Checking that number fields contain valid numerical values within a specified range.

JavaScript validation typically involves event handlers that trigger validation functions when the user interacts with form elements. For example, validation can occur when the user submits the form (onsubmit event) or as they type in an input field (oninput or onblur events).

**Example Scenario**

Consider a simple login form requiring an email address and a password. JavaScript validation would:

* Ensure the email field is not empty and follows a valid email format.
* Ensure the password field is not empty and meets a minimum length requirement, such as eight characters. If any of these conditions are not met, JavaScript can display an error message next to the respective field, preventing the form from being submitted until all inputs are valid.

**Enhancing Security with JavaScript Validation**

While client-side validation improves user experience, it is not foolproof against malicious attempts to bypass validation by manipulating the browser or the JavaScript code. Therefore, security practices must be observed:

* **Never Rely Solely on Client-Side Validation:** Client-side validation should always be accompanied by server-side validation. While JavaScript can catch most errors, server-side validation acts as a final line of defense against invalid or potentially harmful inputs.
* **Avoiding Common Vulnerabilities:** JavaScript validation can help prevent common security vulnerabilities like SQL injection or cross-site scripting (XSS) by ensuring that inputs do not contain harmful scripts or commands. However, it’s essential to sanitize and validate inputs on the server side as well.
* **Rate Limiting and CAPTCHA:** For forms that might be targeted by bots, such as login forms or contact forms, JavaScript can be used to integrate CAPTCHA systems or implement rate limiting, reducing the risk of automated attacks.

**Learning Outcomes:**

* Ability to implement JavaScript-based form validation to ensure that user inputs meet specific criteria before form submission.
* Understanding of how to enhance user experience and data accuracy by providing immediate feedback through client-side validation.

**Implementation Code:**

**function validateText(textid){**

**var textid = document.getElementById(textid);**

**if(textid.value == ""){**

**textid.style.border = '1px solid red';**

**textid.placeholder = "Enter value here!";**

**return false;**

**}**

**else{**

**return true;**

**}**

**}**

**function validatePassword(){**

**var password = document.getElementById('password');**

**var confirm = document.getElementById('confirm');**

**if(password.value != "" && password.value == confirm.value){**

**return true;**

**}**

**else{**

**var check = document.getElementById('check');**

**check.innerHTML = "The passwords do not match. Please try again.";**

**check.style.display = "block";**

**check.style.color = "red";**

**return false;**

**}**

**}**

**function validateEmail(){**

**var email = document.getElementById("email");**

**var invalid3 = document.getElementById("invalid3");**

**var val = email.value;**

**if(val != "" && val.substring(val.length-4) == ".com" && val.includes("@") && (val.length - val.indexOf("@") -1) - (val.length - val.lastIndexOf(".") -1) >= 3 && val.substring(0, val.indexOf('@')).length >= 1){**

**return true;**

**}**

**else{**

**invalid3.innerHTML = "Please enter a valid email id.";**

**invalid3.style.color = "red";**

**invalid3.style.display = "block"**

**return false;**

**}**

**}**

**function validateNumber(){**

**var number = document.getElementById("number");**

**var number\_value = number.value;**

**console.log(number\_value)**

**console.log(number\_value.length)**

**if(number\_value.length < 10 || number\_value.length > 10){**

**var invalid1 = document.getElementById("invalid1");**

**invalid1.style.display = "block";**

**invalid1.style.color = "red";**

**invalid1.innerHTML = "Invalid phone number.";**

**return false;**

**}**

**else{**

**return true;**

**}**

**}**

**function validateBldg(){**

**var bldg = document.getElementById("bldg")**

**var bldg\_val = bldg.value**

**if(bldg\_val < 1 || bldg\_val > 1999){**

**var invalid2 = document.getElementById("invalid2")**

**invalid2.style.display = "block";**

**invalid2.style.color = "red";**

**invalid2.innerHTML = "Please enter a value between 1 and 1999.";**

**return false;**

**}**

**else{**

**return true;**

**}**

**}**

**function validateCity(){**

**var city = document.getElementById("city");**

**var cities = document.getElementById("cities");**

**if(cities.selectedIndex == 0){**

**city.innerHTML = "Please select a city.";**

**city.style.display = "block";**

**city.style.color = "red";**

**return false;**

**}**

**else{**

**return true;**

**}**

**}**

**function validateRadio(){**

**var phone = document.getElementById("phone");**

**var emailid = document.getElementById("emailid");**

**var message = document.getElementById("message");**

**var radio = document.getElementById("radio");**

**if(phone.checked || emailid.checked || message.checked){**

**return true;**

**}**

**else{**

**radio.innerHTML = "Please select an option.";**

**radio.style.display = "block";**

**radio.style.color = "red";**

**return false;**

**}**

**}**

**function validateForm(){**

**var UserName = validateText('user\_name');**

**var StreetName = validateText('street');**

**var Password = validatePassword();**

**var Number1 = validateNumber();**

**var Building = validateBldg();**

**var City = validateCity();**

**var Contact = validateRadio();**

**var Email = validateEmail();**

**if(UserName && StreetName && Password && Number1 && Building && Contact && City && Email){**

**return true;**

**}**

**else{**

**return false;**

**}**

**}**

**Implentation Output:**

