**EXPERIMENT NO. 5**

**Aim:** To Implement JavaScript basics for handling events and validation.

**Theory:**

The change in the state of an object is known as an Event. In html, there are various events which represent that some activity is performed by the user or by the browser. When javascript code is included in HTML, js react over these events and allow the execution. This process of reacting over the events is called Event Handling.

Thus, js handles the HTML events via Event Handlers.Some of the HTML events and their event handlers are:

Mouse events:

|  |  |  |
| --- | --- | --- |
| Event Performed | Event Handler | Description |
| click | onclick | When mouse click on an element |
| mouseover | onmouseover | When the cursor of the mouse comes over the element |
| mouseout | onmouseout | When the cursor of the mouse leaves an element |
| mousedown | onmousedown | When the mouse button is pressed over the element |
| mouseup | onmouseup | When the mouse button is released over the element |
| mousemove | onmousemove | When the mouse movement takes place. |

**Form events:**

|  |  |  |
| --- | --- | --- |
| Event Performed | Event Handler | Description |
| focus | onfocus | When the user focuses on an element |
| submit | onsubmit | When the user submits the form |
| blur | onblur | When the focus is away from a form element |
| change | onchange | When the user modifies or changes the value of a form element |

**Window/Document events:**

|  |  |  |
| --- | --- | --- |
| Event Performed | Event Handler | Description |
| load | onload | When the browser finishes the loading of the page |
| unload | onunload | When the visitor leaves the current webpage, the browser unloads it |
| resize | onresize | When the visitor resizes the window of the browser |

It is important to validate the form submitted by the user because it can have inappropriate values. So, validation is a must to authenticate user.JavaScript provides a facility to validate the form on the client-side so data processing will be faster than server-side validation. Most of the web developers prefer JavaScript form validation.Through JavaScript, we can validate name, password, email, date, mobile numbers and more fields.

**Code:**

**HTML:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<link rel="stylesheet" href="style.css">

<title>Login Page</title>

</head>

<body>

<div class="container">

<form id="loginForm" onsubmit="return validateForm()">

<label for="username">Username:</label>

<input type="text" id="username" required>

<label for="password">Password:</label>

<input type="password" id="password" required>

<button type="submit">Login</button>

</form>

</div>

<script src="script.js"></script>

</body>

</html>

**CSS:**

body {

display: flex;

align-items: center;

justify-content: center;

height: 100vh;

margin: 0;

font-family: Arial, sans-serif;

background-color: #f0f0f0;

}

.container {

width: 300px;

background-color: #fff;

padding: 20px;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

}

form {

display: flex;

flex-direction: column;

}

label {

margin-bottom: 8px;

}

input {

padding: 10px;

margin-bottom: 16px;

border: 1px solid #ccc;

border-radius: 4px;

}

button {

padding: 12px;

background-color: #4caf50;

color: white;

border: none;

border-radius: 4px;

cursor: pointer;

}

button:hover {

background-color: #45a049;

}

**JAVASCRIPT:**

<script>

function validateForm() {

var username = document.getElementById("username").value;

var password = document.getElementById("password").value;

if (username === "" || password === "") {

alert("Please fill in all fields");

return false;

}

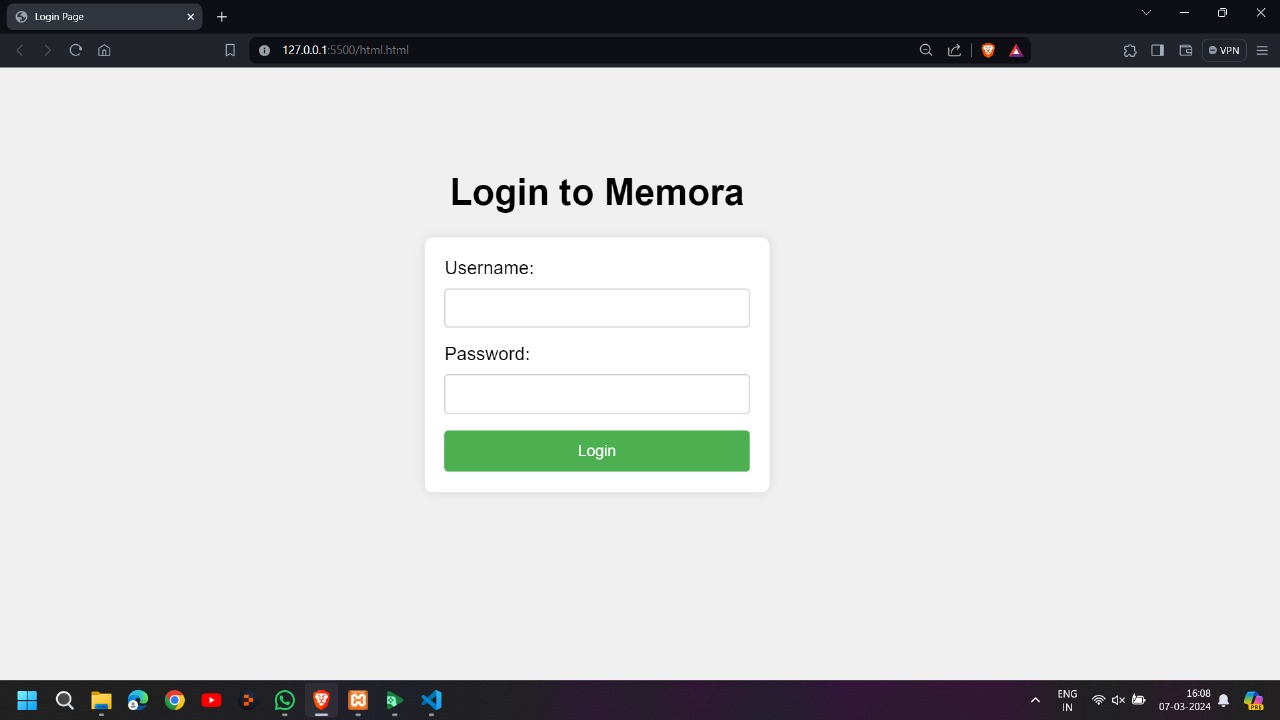
alert("Login successful!\nUsername: " + username + "\nPassword: " + password);

return true;

}

</script>

OUTPUT:



**Conclusion:** Thus, we have successfully implemented the javascript events such as onclickevent and also javascript validation on our web page. In conclusion, mastering JavaScript basics for event handling and validation is pivotal for building robust and user-friendly web applications. Understanding how to effectively handle user interactions through event listeners allows developers to create dynamic and responsive interfaces that react to user input in real-time. By harnessing the power of events such as clicks, keypresses, and form submissions, developers can design interactive web experiences that engage users and enhance usability.