

Exercise 9:

Design input forms that validate data (e.g., email, phone number) and display error messages using HTML/CSS,JavaScript (with Validator.js)

Aim:

The aim is to design input forms that validate data, such as email and phone number, and display error messages using HTML/CSS and JavaScript with Validator.js.

Procedure:

Step 1: Setting Up the HTML Form

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

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

  <title>Form Validation</title>

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

</head>

<body>

  <div class="container">

    <form id="myForm">

      <label for="email">Email:</label>

      <input type="email" id="email" name="email" required>
```

```
<span id="emailError" class="error"></span>
```

```
<label for="phone">Phone Number:</label>
```

```
<input type="text" id="phone" name="phone" required>
```

```
<span id="phoneError" class="error"></span>
```

```
<button type="submit">Submit</button>
```

```
</form>
```

```
</div>
```

```
<script  
src="https://cdnjs.cloudflare.com/ajax/libs/validator/13.6.0/validator.min.js"></scri  
pt>
```

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

```
</body>
```

```
</html>
```

Step 2: Styling the Form with CSS

```
/* style.css */
```

```
body {
```

```
  font-family: Arial,
```

```
  sans-serif;
```

```
  background-color: #f4f4f4;
```

```
  display: flex;
```

```
  justify-content: center;
```

```
  align-items: center;
```

```
height: 100vh;  
margin: 0;  
}
```

```
.container {  
  background-color: white;  
  padding: 20px;  
  border-radius: 5px;  
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);  
}
```

```
form {  
  display: flex;  
  flex-direction: column;  
}
```

```
label {  
  margin-bottom: 5px;  
}
```

```
input {  
  margin-bottom: 10px;  
  padding: 10px;  
  border: 1px solid #ccc;
```

```
border-radius: 3px;  
}
```

```
button {  
  padding: 10px;  
  background-color: #28a745;  
  color: white;  
  border: none;  
  border-radius: 3px;  
  cursor: pointer;  
}
```

```
button:hover {  
  background-color: #218838;  
}
```

```
.error {  
  color: red;  
  font-size: 0.875em;  
}
```

Step 3: Adding JavaScript for Validation

```
/* script.js */
```

```
document.getElementById('myForm').addEventListener('submit', function (e) {
```

```
e.preventDefault();
```

```
let email = document.getElementById('email').value;
```

```
let phone = document.getElementById('phone').value;
```

```
let emailError = document.getElementById('emailError');
```

```
let phoneError = document.getElementById('phoneError');
```

```
// Clear previous error messages
```

```
emailError.textContent = '';
```

```
phoneError.textContent = '';
```

```
// Validate email
```

```
if (!validator.isEmail(email)) {
```

```
    emailError.textContent = 'Please enter a valid email address.';
```

```
}
```

```
// Validate phone number
```

```
if (!validator.isMobilePhone(phone, 'any')) {
```

```
    phoneError.textContent = 'Please enter a valid phone number.';
```

```
}
```

```
// If no errors, submit the form (for demonstration purposes, we'll just log the values)
```

```
if (validator.isEmail(email) && validator.isMobilePhone(phone, 'any')) {
```

```
    console.log('Email:', email);  
    console.log('Phone:', phone);  
  }  
});
```

Output:

The image displays two side-by-side screenshots of a web form. The form has two input fields: 'Email:' and 'Phone Number:'. In the left screenshot, the 'Email' field contains '123@gmail.com' and the 'Phone Number' field contains '123456***'. Below the fields, a red error message reads 'Please enter a valid phone number.' and a green 'Submit' button is visible. In the right screenshot, the 'Email' field contains '123gmail.com' and the 'Phone Number' field contains '123456'. A dark grey tooltip with the text 'Please enter an email address.' is positioned over the 'Email' field. A green 'Submit' button is also present at the bottom.

Result:

The input form was successfully designed using HTML and styled with CSS. Validation of user inputs such as email and phone number were effectively implemented using JavaScript with the Validator.js library. The form displayed appropriate error messages for invalid inputs and allowed submission only when all data was valid, thereby ensuring accurate and user-friendly data entry.

SIDDARTH SAKTHI M 230701314