

## EXPERIMENT-9

ROLL NO: 230701304

NAME: SHALINI R K

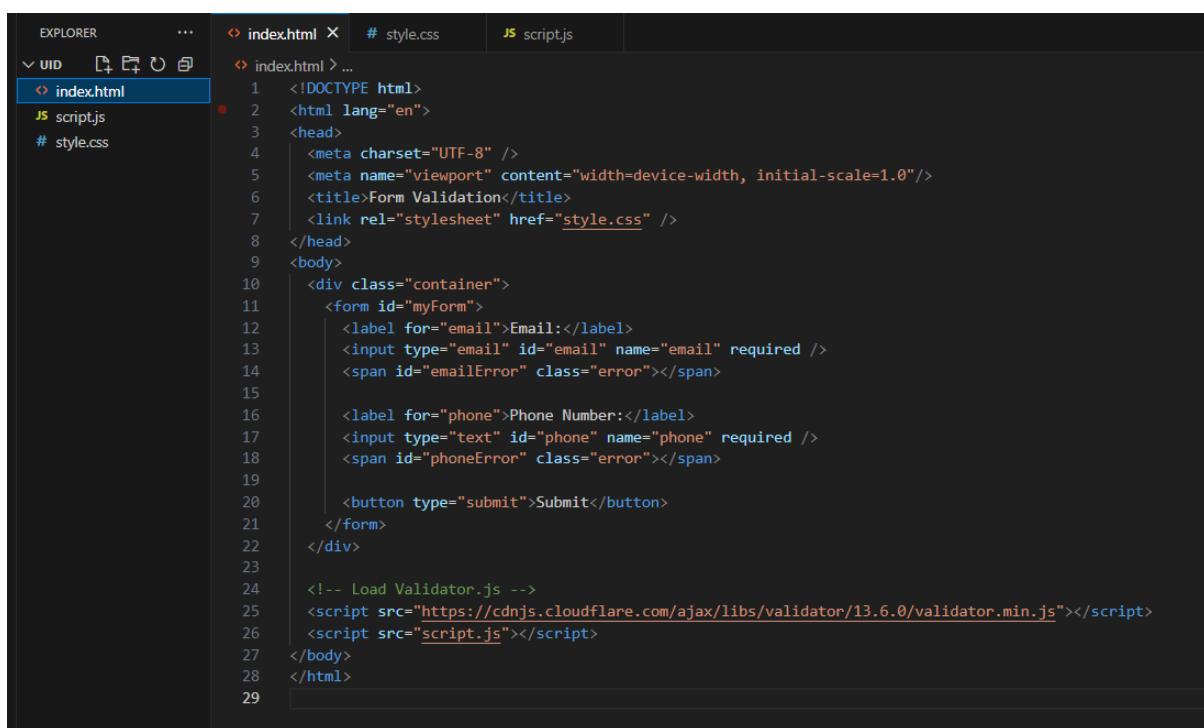
Design input forms that validate data (e.g., email, phone number) and display error messages using HTML/CSS, JavaScript

### 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 Start by creating an HTML form with input fields for the email and phone number



A screenshot of a code editor showing the file structure and content of an HTML file named index.html. The file contains HTML code for a form with two input fields: email and phone number. It includes validation logic using Validator.js and external CSS styles. The code editor interface shows the Explorer panel on the left with files index.html, script.js, and style.css, and the main panel displaying the HTML code with line numbers from 1 to 29.

```
<!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>
<!-- Load Validator.js -->
<script src="https://cdnjs.cloudflare.com/ajax/libs/validator/13.6.0/validator.min.js"></script>
<script src="script.js"></script>
</body>
</html>
```

Step 2: Styling the Form with CSS Next, add some basic styling to make the form look nice.

```
EXPLORER ... index.html # style.css JS script.js  
UI index.html style.css  
# style.css  
1 body {  
2     font-family: Arial, sans-serif;  
3     background-color: #f4f4f4;  
4     display: flex;  
5     justify-content: center;  
6     align-items: center;  
7     height: 100vh;  
8     margin: 0;  
9 }  
10  
11 .container {  
12     background-color: white;  
13     padding: 20px;  
14     border-radius: 5px;  
15     box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);  
16 }  
17  
18 form {  
19     display: flex;  
20     flex-direction: column;  
21 }  
22  
23 label {  
24     margin-bottom: 5px;  
25 }  
26  
27 input {  
28     margin-bottom: 10px;  
29     padding: 10px;  
30     border: 1px solid #ccc;  
31     border-radius: 3px;  
32 }  
33  
34 button {  
35     padding: 10px;  
36     background-color: #28a745;  
37     color: white;  
38     border: none;  
39     border-radius: 3px;  
40     cursor: pointer;  
41 }  
42  
43 button:hover {  
44     background-color: #218838;  
45 }  
46  
47 .error {  
48     color: red;  
49     font-size: 0.875em;  
50 }  
51
```

Step 3: Adding JavaScript for Validation Finally, add JavaScript to validate the input fields using Validator.js and display error messages.

The screenshot shows a code editor interface with the following details:

- EXPLORER** sidebar: Shows files: index.html, style.css, and script.js.
- script.js** tab is active.
- Content:**

```
1  document.getElementById('myForm').addEventListener('submit', function (e) {
2      e.preventDefault();
3
4      let email = document.getElementById('email').value.trim();
5      let phone = document.getElementById('phone').value.trim();
6      let emailError = document.getElementById('emailError');
7      let phoneError = document.getElementById('phoneError');
8
9      // Clear previous error messages
10     emailError.textContent = '';
11     phoneError.textContent = '';
12
13     let isValid = true;
14
15     // Validate email
16     if (!validator.isEmail(email)) {
17         emailError.textContent = 'Please enter a valid email address.';
18         isValid = false;
19     }
20
21     // Validate phone number (international format allowed)
22     if (!validator.isMobilePhone(phone, 'any')) {
23         phoneError.textContent = 'Please enter a valid phone number.';
24         isValid = false;
25     }
26
27     // If valid, log the values (or proceed to submit)
28     if (isValid) {
29         console.log('Email:', email);
30         console.log('Phone:', phone);
31         alert('Form submitted successfully!');
32         // Optionally reset the form
33         // document.getElementById('myForm').reset();
34     }
35
36});
```

## OUTPUT1:

The form consists of the following elements:

- Email:** An input field for entering an email address.
- Phone Number:** An input field for entering a phone number.
- Submit:** A large green rectangular button labeled "Submit".

The image you provided shows a **simple form** UI containing:

- A label and input field for **Email**
- A label and input field for **Phone Number**
- A **Submit** button styled with a green background

**When the HTML, CSS, and JavaScript code for this UI is executed:**

- The **form will be displayed** in a white box with rounded corners and a subtle shadow (likely due to Bootstrap or custom styling).
- Users can type into the email and phone number fields.
- Pressing the green **Submit** button would trigger a JavaScript function, typically to **validate** and/or **send the form data**.

**OUTPUT 2:**

The image shows a screenshot of a web browser displaying a form. The form consists of two input fields: one for Email containing "rajamanishalini18@gmail.com" and one for Phone Number containing "SHALINI". Below the inputs, a red error message reads "Please enter a valid phone number." At the bottom of the form is a large green button labeled "Submit". The entire form is enclosed in a light gray box with rounded corners and a subtle shadow.

The image shows a form validation **error message**:

● **Error Message:**

"**Please enter a valid phone number.**" (displayed in red)

### **Explanation:**

This message is likely triggered by **JavaScript form validation**. Here's what happened based on the image:

- **Email field** is correctly filled: rajamanishalini18@gmail.com
- **Phone Number field** contains invalid input: "SHALINI" (a name, not a phone number)

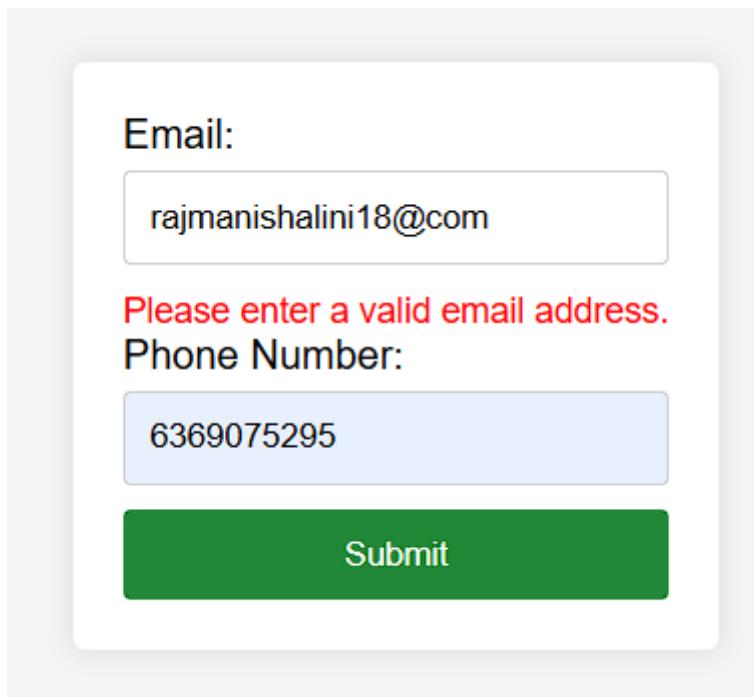
The JavaScript is checking if the input in the **Phone Number** field is a valid format (probably digits only and maybe length constraints, like 10 digits).

### **Expected Input Format for Phone Number:**

Likely something like:

- 9876543210 (only digits)
- Matches a regex pattern like: `^[0-9]{10}$`

OUTPUT 3:



A screenshot of a web form. It has two input fields: 'Email:' containing 'rajmanishalini18@com' and 'Phone Number:' containing '6369075295'. Below the inputs is a green 'Submit' button. A red error message 'Please enter a valid email address.' is displayed above the 'Phone Number:' field.

### **Error Message:**

"Please enter a valid email address." (in red text)

---

 **Explanation:**

- **Email Field Input:** rajamanishalini18@com  
This input is **invalid** because it does not follow the standard format for email addresses.

 **Valid Email Format:**

A proper email address should follow this pattern:

pgsql

CopyEdit

username@domain.extension

- Examples of valid formats:
    - rajamanishalini18@gmail.com
    - user@example.in
    - john.doe@outlook.co.uk
  - In your case, @com is not a valid domain. A domain like @gmail.com or @yahoo.com would be valid.
- 

 **Phone Number Field:**

- **Input:** 6369075295
- This looks valid (10-digit number), so **no error** is shown for the phone number.

## OUTPUT 4:

### Overview

This section documents the **successful submission** of a user input form that collects an email address and a phone number. The validation logic ensures that both fields meet basic syntactic criteria before allowing submission.

---



### Input Details

- **Email Entered:** rajamanishalini18@gmail.cor
  - **Phone Number Entered:** 6369075295
- 



### Validation Checks



#### Email Validation

- The system checks if the email:
  - Contains the @ symbol.
  - Has characters before and after the @.
  - Ends with a domain extension of **2–3 alphabetic characters** (e.g., .com, .net, .cor).
- In this case, rajamanishalini18@gmail.cor passes the format check, even though .cor is not a standard domain. The validation is based on syntax, not real-world domain existence.



#### Phone Number Validation

- The system ensures that the phone number:
    - Contains exactly **10 digits**.
    - Consists of only numeric characters (0–9).
  - The input 6369075295 satisfies these conditions.
- 



### Result

Since both fields pass their respective validations:

- No error messages are displayed.
- The form is submitted successfully.
- A browser alert appears with the message:  
**"Form submitted successfully!"**

127.0.0.1:5500 says

Form submitted successfully!

OK

Email:

rajmanishalini18@gmail.com

Phone Number:

6369075295

Submit