Exercise 9

Form Validation System – Documentation
Overview
The Form Validation System is designed to validate user inputs for email addresses and phone numbers. It provides real-time feedback by displaying error messages if the entered data is invalid.
The system enhances user experience by ensuring only correct information is submitted.
Objective
> To validate form inputs (email and phone number) before submission.
>
> To display appropriate error messages using JavaScript and Validator.js.
>
> To improve form usability and data accuracy.

Technologies Used
> HTML5 — Form structure
>
> CSS3 — Styling and layout
>
> JavaScript — Input validation logic
>
> Validator.js — External JavaScript library for robust validation

Project Structure
> `index.html` — Contains the form elements and loads required scripts and styles.
>
> `style.css` — Styles the form layout and error messages.
>
> `script.js` — JavaScript file handling validation logic.
>
> `Validator.js` — External library included via CDN.

Features
> Email validation: Ensures the email entered follows a standard email format.
>
> Phone number validation: Checks if the input is a valid phone number.
>
> Inline error messages: Displays specific error text immediately below invalid fields.
>
> Responsive and clean form design.

How It Works
> User inputs email and phone number into the form.
>
> Upon clicking Submit:
>> JavaScript checks:
>>> If the email is in a valid format.
>>>
>>> If the phone number is a valid mobile number.
>
> If validation fails:
>> Corresponding error messages are shown in red below each field.
>
> If validation succeeds:
>> Form values are logged to the console (demo purpose).

Code Snippets
HTML (**index.html**) html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Form Validation</title>
  <link rel="stylesheet" href="style.css">
</head>
```

```html
<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"></script>
  <script src="script.js"></script>
</body>
</html>
```
CSS

(style.css) css

```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;
}
```

JavaScript **(script.js)** javascript

```javascript
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, log values   if (validator.isEmail(email) &&
validator.isMobilePhone(phone, 'any')) {      console.log('Email:', email);
console.log('Phone:', phone);
  }
});
```
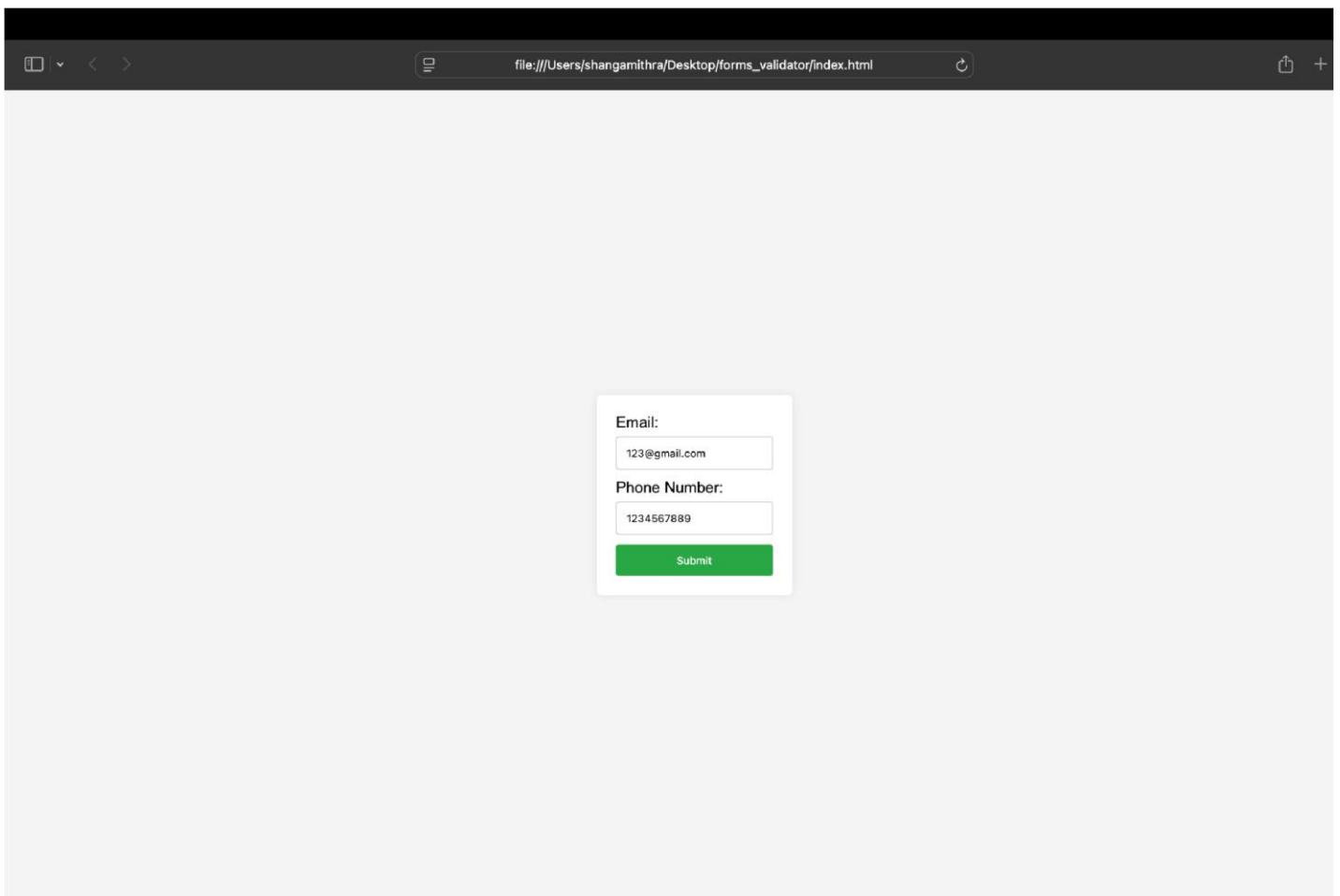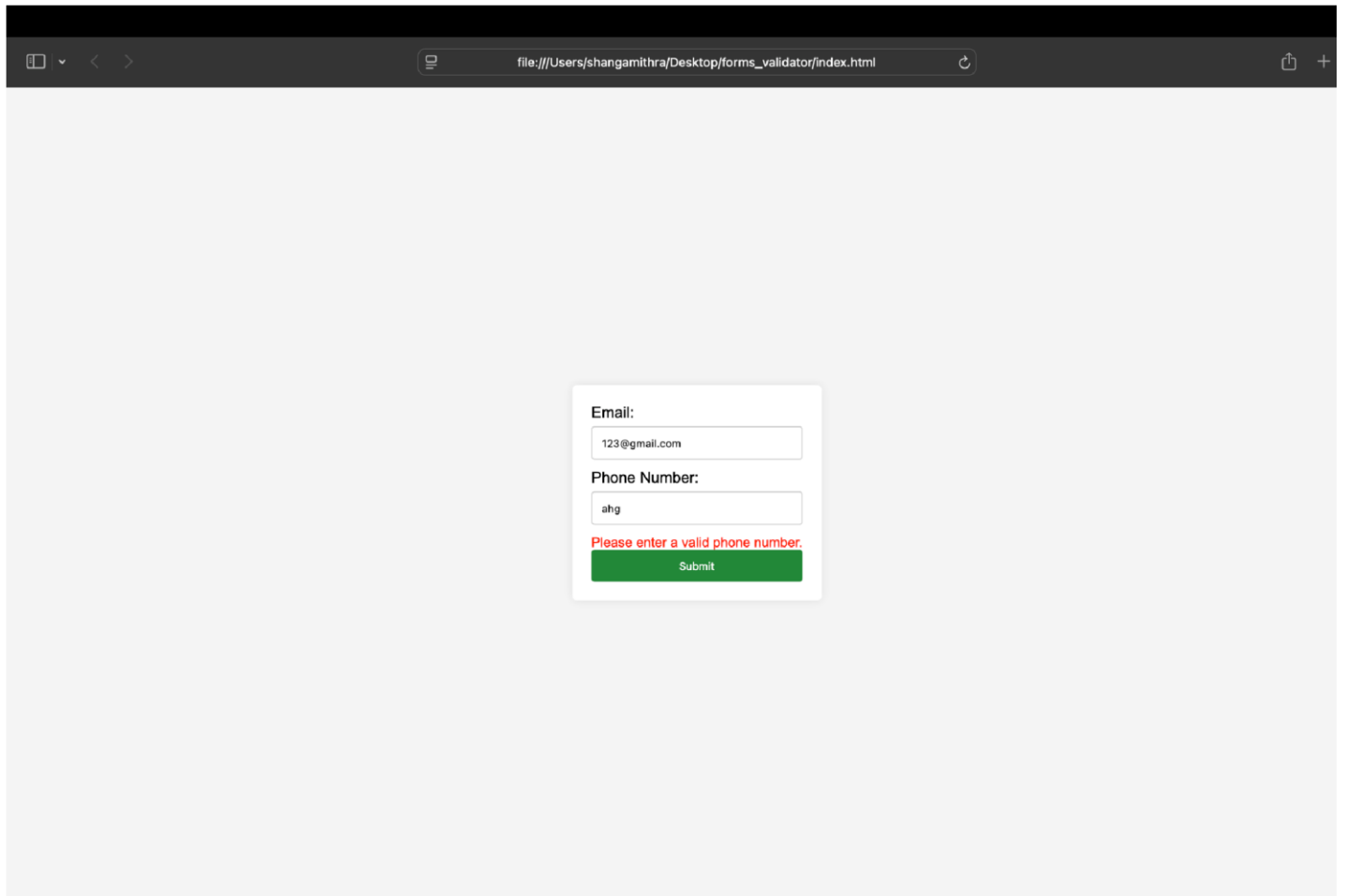
Screenshots

Valid Input Form — shows fields ready to submit.

Invalid Input Form — shows error message when incorrect phone number is entered.

Email:

123@gmail.com

Phone Number:

ahg

Please enter a valid phone number.

Submit

Conclusion

This project demonstrates real-time form validation using JavaScript and Validator.js, ensuring that data collected from users is accurate and reliable.

It offers a strong foundation for building more complex forms in larger web applications.