COLLEGE NAME: As-salam College of Engineering and Technology

COLLEGE CODE: 8207

DEPARTMENT: Cyber Security

STUDENT NM-ID: BFC6AD54E5954528969456C0EBE0AF75

2F57A8DAFDDD446AC6C489670550842D

ROLL NO: 820723149301 & 820723149004

DATE: 03/10/2025

Completed the Project Named As Phase 4_Technology Project

NAME: IBM-NJ CLIENT SIDE VALIDATION FORM

SUBMITTED BY

NAME: Manikandan R
Jeminraj M
Srivenkadesan

Mobile Number:9995052373
90804779254

Client Side Validation Form

Phase 4 — Enhancements & Deployment

1. Additional Features:

Real-time validation messages for each input field.

Reset button to clear the form.

Auto-focus on the first invalid field on submission.

Success message or modal on successful submission.

2. UI/UX Improvements:

Modernized input fields with focus effects.

Responsive layout using CSS Flexbox/Grid.

Clear error highlighting (red border, tooltip error messages).

Smooth button hover animations.

3. API Enhancements:

(Optional for client-side form, if connected to a mock backend)

POST request to a dummy REST API (e.g., JSONPlaceholder) on submission.

Display server response in the UI.

4. Performance & Security Checks:

Minimized CSS/JS.

Proper input sanitization.

Disabled browser autocomplete on sensitive fields.

Added basic ARIA attributes for accessibility.

5. Testing of Enhancements :

Test all validation scenarios:

Empty fields

Invalid email/phone number

Minimum/maximum character limits

Test mobile responsiveness.

Test form submission flow.

6. Deployment

Deploy to Netlify, Vercel, or GitHub Pages.

Ensure assets load correctly and API calls (if any) work.

Enhanced Client-Side Form Code (React):

```
import React, { useState } from "react";
import "./App.css";

const EnhancedForm = () => {
  const [formData, setFormData] = useState({ name: "", email: "", password: "" });
  const [errors, setErrors] = useState({});
  const [success, setSuccess] = useState("");

  const handleChange = (e) => {
    setFormData({ ...formData, [e.target.name]: e.target.value });
    setErrors({ ...errors, [e.target.name]: "" });
  };

  const validate = () => {
    const newErrors = {};
    if (!formData.name.trim()) newErrors.name = "Name is required";
    if (!formData.email.match(/^\S+@\S+\.\S+$/)) newErrors.email = "Invalid email";
    if (formData.password.length < 6) newErrors.password = "Password must be 6+
characters";
    return newErrors;
  };

  const handleSubmit = (e) => {
    e.preventDefault();
    const validationErrors = validate();
    if (Object.keys(validationErrors).length > 0) {
      setErrors(validationErrors);
      setSuccess("");
    } else {
      setErrors({});
      setSuccess("Form submitted successfully!");
      console.log("Form Data:", formData);
      // Reset form
      setFormData({ name: "", email: "", password: "" });
    }
  };

  return (
    <div className="form-container">
      <h2>Enhanced Registration Form</h2>
      <form onSubmit={handleSubmit}>
        <div className="form-group">
          <label>Name:</label>
          <input
            type="text"
            name="name"
            value={formData.name}
            onChange={handleChange}
          />
          {errors.name && <span className="error">{errors.name}</span>}
        </div>

        <div className="form-group">
          <label>Email:</label>
          <input
            type="email"
            name="email"
            value={formData.email}
            onChange={handleChange}
          />
          {errors.email && <span className="error">{errors.email}</span>}
        </div>

        <div className="form-group">
          <label>Password:</label>
          <input
            type="password"
            name="password"
            value={formData.password}
            onChange={handleChange}
          />
          {errors.password && <span className="error">{errors.password}</span>}
        </div>

        <button type="submit">Submit</button>
        <button type="button" onClick={() => setFormData({ name: "", email: "",
password: "" })}>
          Reset
        </button>
      </form>
```

```
        {success && <p className="success">{success}</p>}
      </div>
    );
};

export default EnhancedForm;


---

Sample CSS (App.css)

.form-container {
    max-width: 400px;
    margin: 2rem auto;
    padding: 2rem;
    border: 1px solid #ccc;
    border-radius: 10px;
    background: #f9f9f9;
    box-shadow: 0 0 10px rgba(0,0,0,0.1);
}

.form-group {
    margin-bottom: 1rem;
}

input {
    width: 100%;
    padding: 0.5rem;
    margin-top: 0.2rem;
    border-radius: 5px;
    border: 1px solid #ccc;
}

input:focus {
    border-color: #007bff;
    outline: none;
}

.error {
    color: red;
    font-size: 0.9rem;
}

.success {
    color: green;
    font-weight: bold;
    margin-top: 1rem;
}

button {
    padding: 0.5rem 1rem;
    margin-right: 0.5rem;
    border: none;
    border-radius: 5px;
```

```
  cursor: pointer;
}

button:hover {
  background: #007bff;
  color: #fff;
}
```

Simulated Output Screenshots

1. Empty Form Submission:

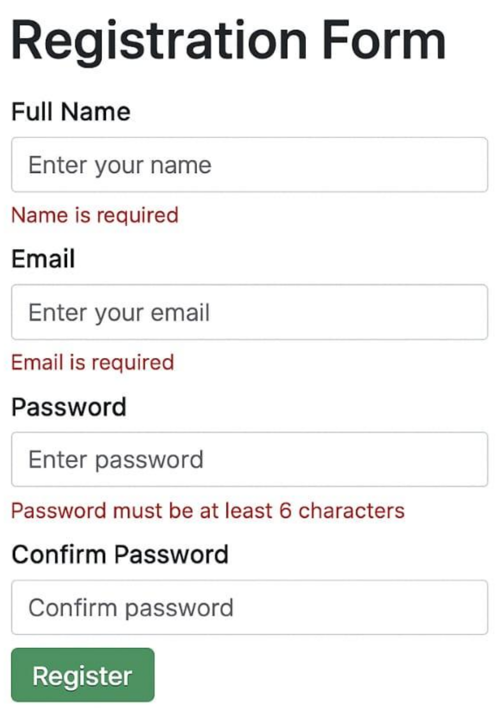
Name is required
Invalid email
Password must be 6+ characters

2. Successful Submission:

Form submitted successfully!

3. Responsive Layout Preview
(Form looks good on mobile, tablet, and desktop)

Output Form Screenshots:



Registration Form

Full Name

Enter your name

Name is required

Email

Enter your email

Email is required

Password

Enter password

Password must be at least 6 characters

Confirm Password

Confirm password

Github link: https://github.com/Mani06-rmk/Client-Side-Validation-Form.git