**Unit 6**

**Q.1) Create an application in ReactJS form and add client and server side validation.**

**We need to install axios**

**In command prompt type command npm install axios.**

**Create componets folder in src and add file**

**FormValidation.js**

import React, { useState } from 'react';

import axios from 'axios';

const FormValidation = () => {

  const [formData, setFormData] = useState({

    name: '',

    email: '',

    password: '',

  });

  const [errors, setErrors] = useState({});

  const [serverMessage, setServerMessage] = useState('');

  const handleChange = (e) => {

    const { name, value } = e.target;

    setFormData({ ...formData, [name]: value });

  };

  const validate = () => {

    const newErrors = {};

    if (!formData.name.trim()) newErrors.name = 'Name is required.';

    if (!formData.email.trim()) {

      newErrors.email = 'Email is required.';

    } else if (!/\S+@\S+\.\S+/.test(formData.email)) {

      newErrors.email = 'Email format is invalid.';

    }

    if (!formData.password) {

      newErrors.password = 'Password is required.';

    } else if (formData.password.length < 8) {

      newErrors.password = 'Password must be at least 8 characters.';

    } else if (!/[A-Z]/.test(formData.password)) {

      newErrors.password = 'Password must contain at least one uppercase letter.';

    }

    setErrors(newErrors);

    return Object.keys(newErrors).length === 0;

  };

  const mockServerValidation = async (data) => {

    try {

      const response = await axios.post('/api/validate', data);

      return response.data;

    } catch (error) {

      return error.response.data;

    }

  };

  const handleSubmit = async (e) => {

    e.preventDefault();

    setServerMessage('');

    if (validate()) {

      const serverResponse = await mockServerValidation(formData);

      if (serverResponse.success) {

        setServerMessage('Form submitted successfully!');

        setFormData({ name: '', email: '', password: '' }); // Reset form

      } else {

        setErrors(serverResponse.errors);

      }

    }

  };

  return (

    <div style={styles.container}>

      <h2 style={styles.header}>React Form with Validation</h2>

      <form onSubmit={handleSubmit} style={styles.form}>

        <div style={styles.field}>

          <label style={styles.label}>Name:</label>

          <input

            type="text"

            name="name"

            value={formData.name}

            onChange={handleChange}

            style={styles.input}

          />

          {errors.name && <p style={styles.error}>{errors.name}</p>}

        </div>

        <div style={styles.field}>

          <label style={styles.label}>Email:</label>

          <input

            type="email"

            name="email"

            value={formData.email}

            onChange={handleChange}

            style={styles.input}

          />

          {errors.email && <p style={styles.error}>{errors.email}</p>}

        </div>

        <div style={styles.field}>

          <label style={styles.label}>Password:</label>

          <input

            type="password"

            name="password"

            value={formData.password}

            onChange={handleChange}

            style={styles.input}

          />

          {errors.password && <p style={styles.error}>{errors.password}</p>}

        </div>

        <button type="submit" style={styles.button}>Submit</button>

      </form>

      {serverMessage && <p style={styles.success}>{serverMessage}</p>}

    </div>

  );

};

// Mock server-side validation API

axios.post = (url, data) => {

  return new Promise((resolve, reject) => {

    setTimeout(() => {

      if (data.email === 'taken@example.com') {

        reject({

          response: {

            data: {

              success: false,

              errors: { email: 'Email is already taken.' },

            },

          },

        });

      } else {

        resolve({ data: { success: true } });

      }

    }, 1000);

  });

};

const styles = {

  container: {

    maxWidth: '400px',

    margin: '20px auto',

    padding: '20px',

    background: '#fff',

    boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',

    borderRadius: '8px',

  },

  header: { textAlign: 'center', marginBottom: '20px', fontSize: '1.5rem' },

  form: { display: 'flex', flexDirection: 'column' },

  field: { marginBottom: '15px' },

  label: { marginBottom: '5px', fontWeight: 'bold' },

  input: {

    padding: '10px',

    border: '1px solid #ccc',

    borderRadius: '4px',

    width: '100%',

  },

  button: {

    padding: '10px',

    backgroundColor: '#4CAF50',

    color: '#fff',

    border: 'none',

    borderRadius: '4px',

    cursor: 'pointer',

    fontWeight: 'bold',

  },

  error: { color: 'red', fontSize: '0.9rem' },

  success: { color: 'green', fontSize: '1rem', marginTop: '15px' },

};

export default FormValidation;

**App.js**

import React from 'react';

import './App.css';

import FormValidation from './components/FormValidation';

function App() {

  return (

    <div className="App">

      <h1>Form Validation Demo</h1>

      <FormValidation />

    </div>

  );

}

export default App;

**index.js**

import React from "react";

import ReactDOM from "react-dom/client";

import "./index.css";

import App from "./App";

import reportWebVitals from "./reportWebVitals";

const root = ReactDOM.createRoot(document.getElementById("root"));

root.render(

  <React.StrictMode>

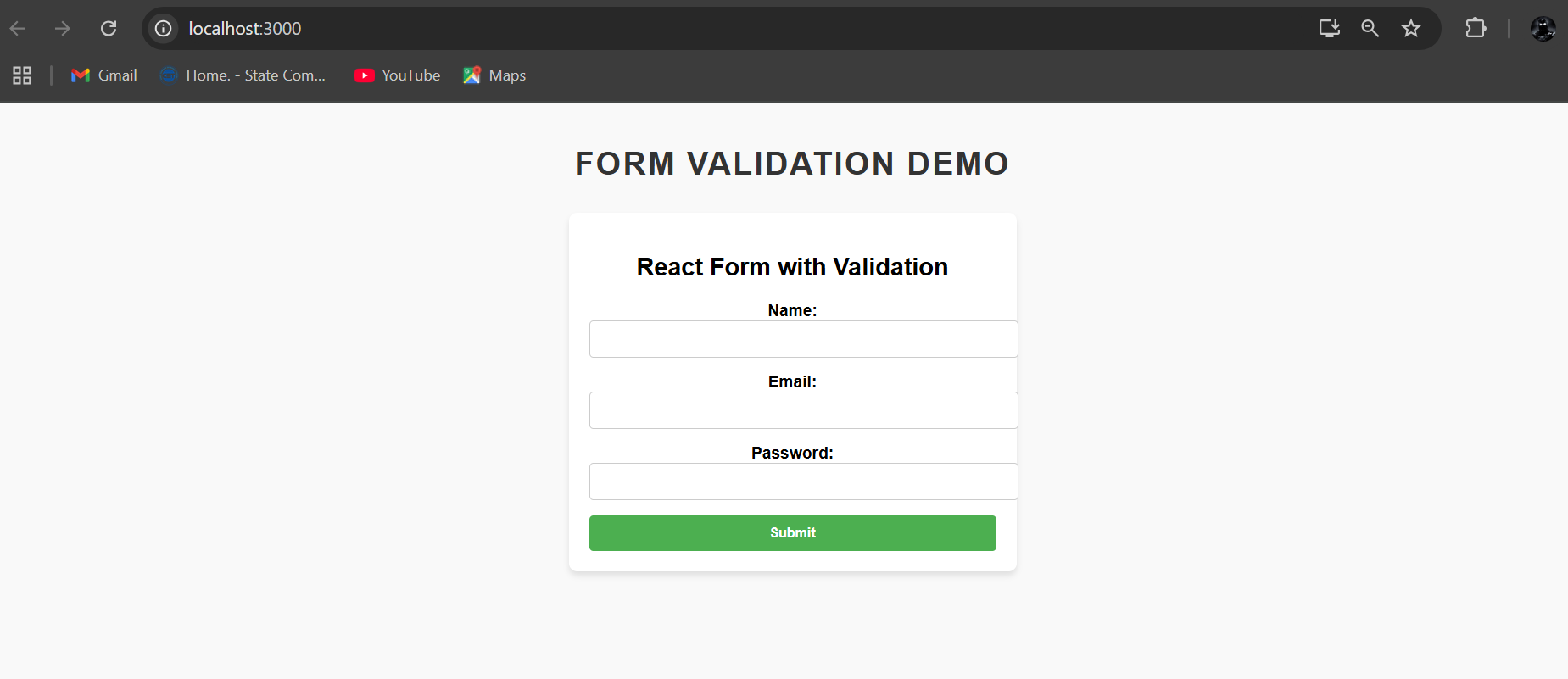
    <App />

  </React.StrictMode>

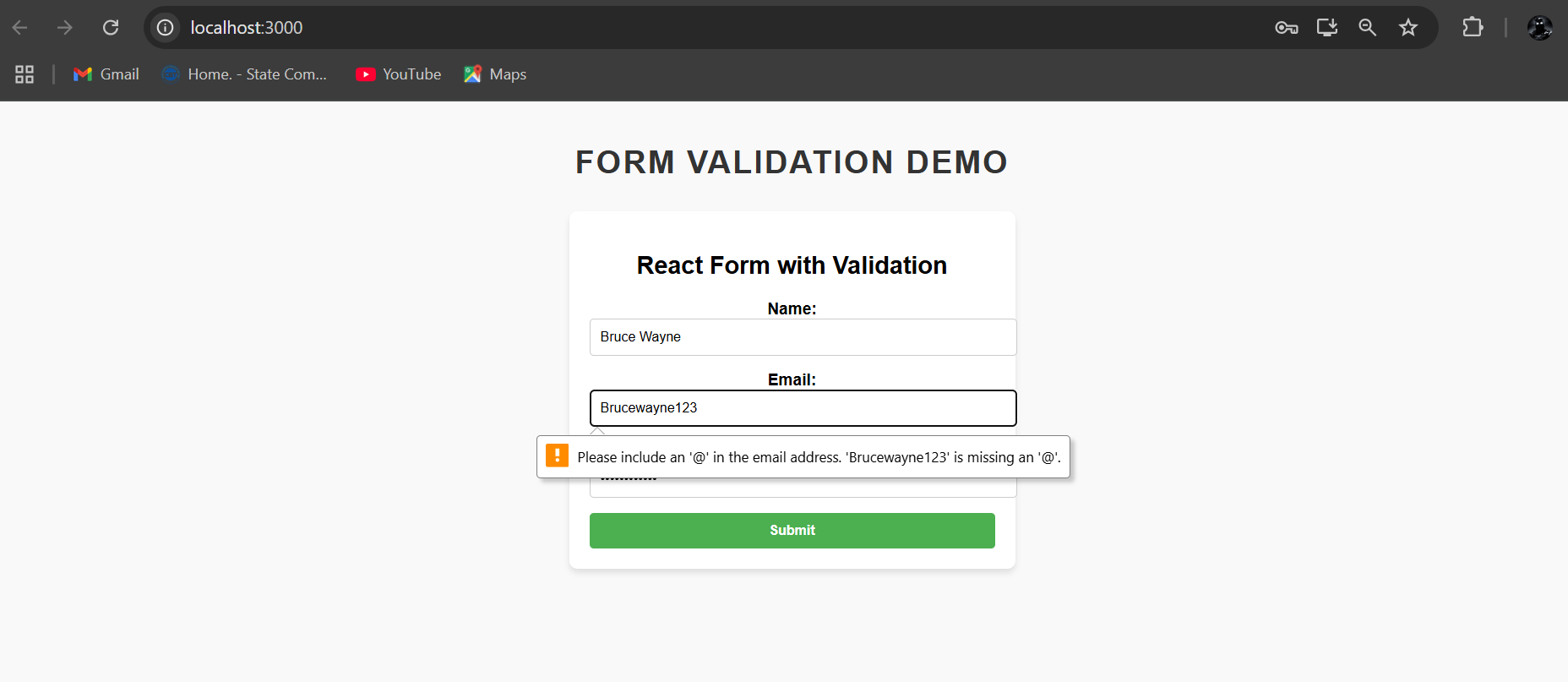
);

reportWebVitals();

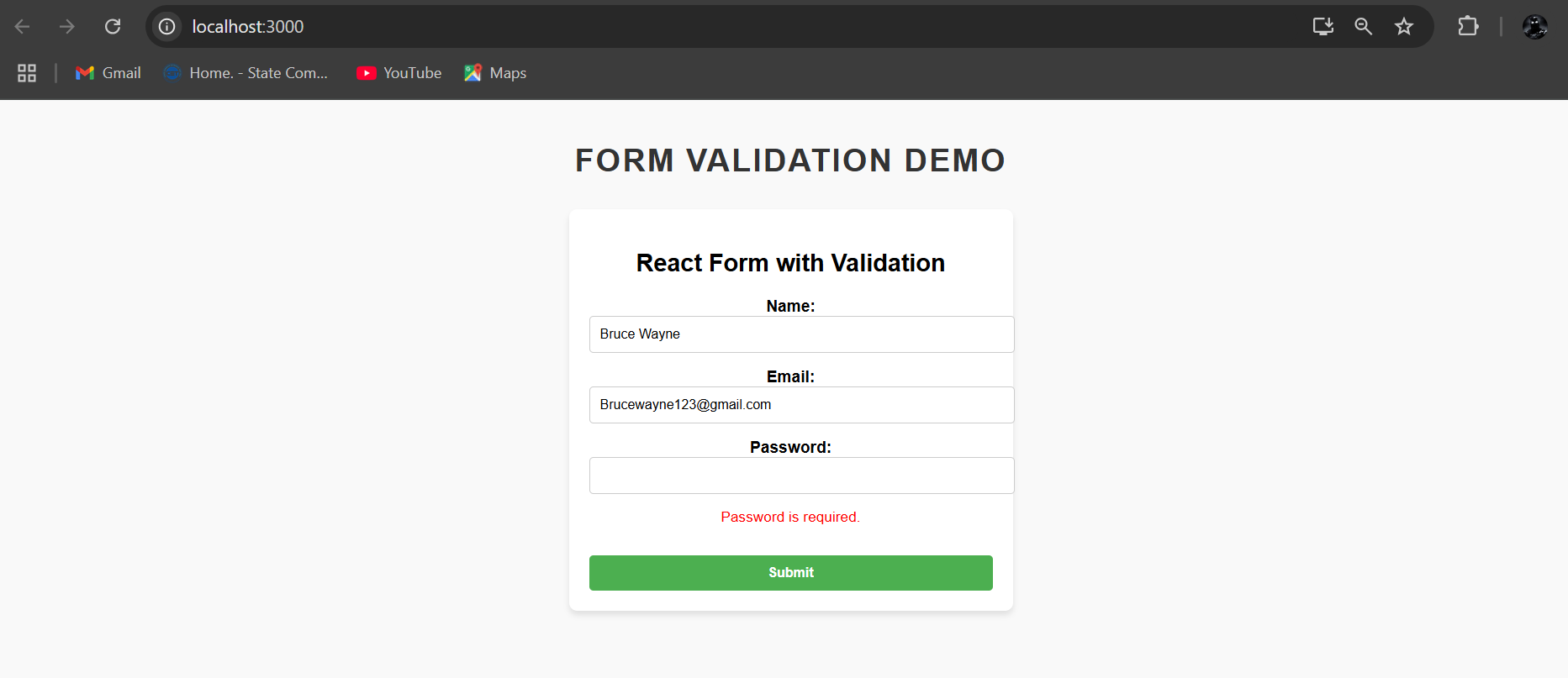
Output :

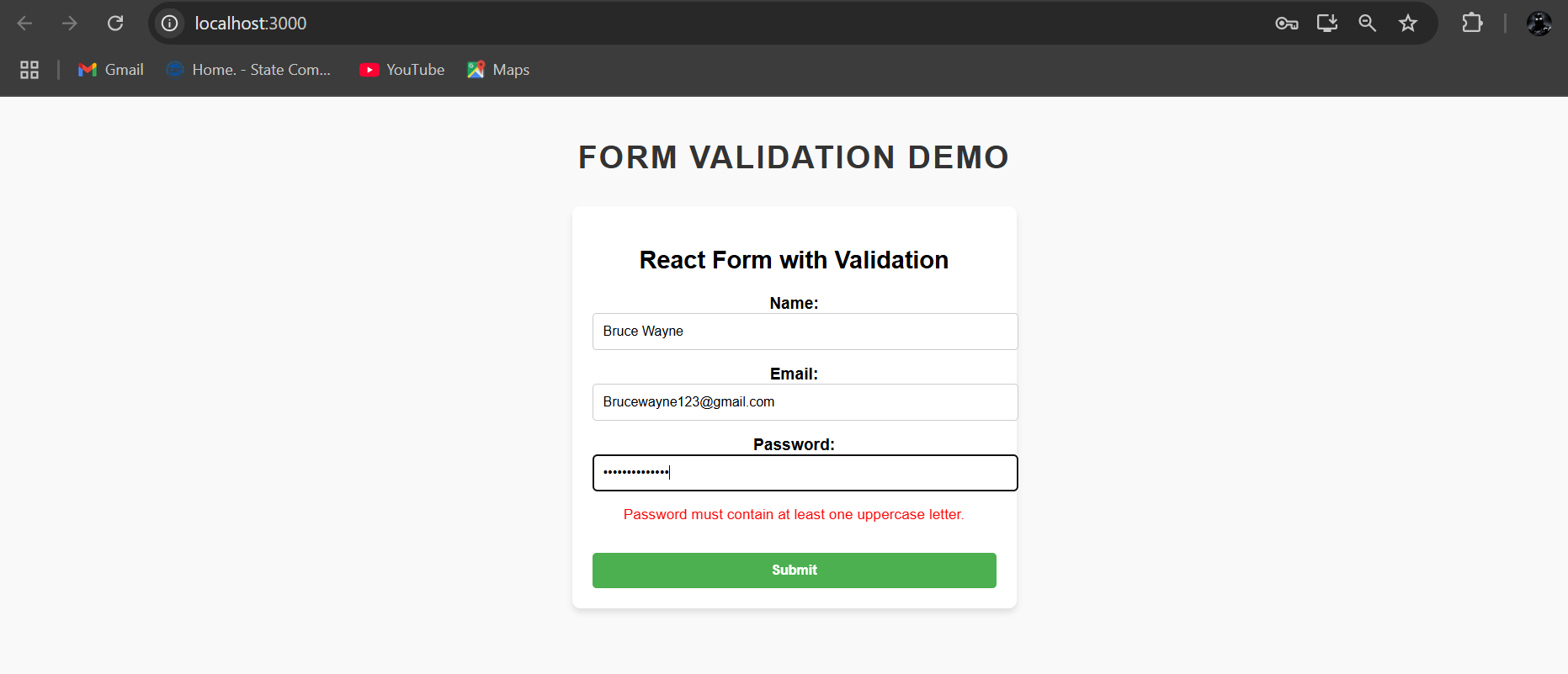


Email validation

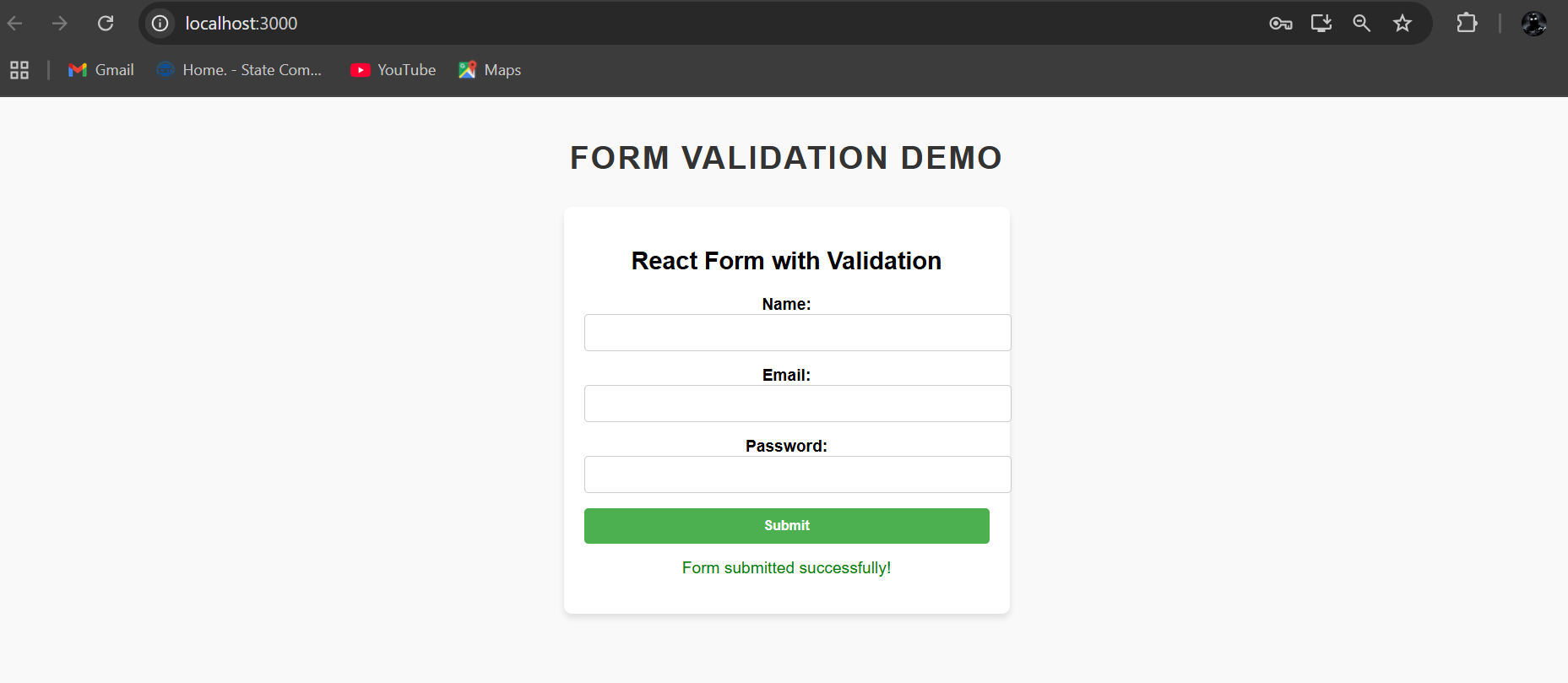


Password requirement





If everything is correct.



**Q.2) Create an application to implement react hooks.**

**In the src folder create component folder and add following files.**

**AppContext.js**

import React, { createContext } from 'react';

export const AppContext = createContext();

export const AppProvider = ({ children }) => {

  const user = { name: 'John Doe', age: 30 };

  return <AppContext.Provider value={user}>{children}</AppContext.Provider>;

};

**ContextDemo.js**

import React, { useContext } from 'react';

import { AppContext } from './AppContext';

const ContextDemo = () => {

const user = useContext(AppContext);

  return (

    <div style={styles.container}>

      <h3>useContext Demo</h3>

      <p>Name: {user.name}</p>

      <p>Age: {user.age}</p>

    </div>

  );

};

const styles = {

  container: { margin: '10px 0', padding: '10px', border: '1px solid #ccc', borderRadius: '5px' },

};

export default ContextDemo;

**EffectDemo.js**

import React, { useState } from 'react';

const StateDemo = () => {

 const [count, setCount] = useState(0);

  return (

    <div style={styles.container}>

      <h3>useState Demo</h3>

      <p>Count: {count}</p>

      <button onClick={() => setCount(count + 1)} style={styles.button}>Increment</button>

      <button onClick={() => setCount(count - 1)} style={styles.button}>Decrement</button>

    </div>

  );

};

const styles = {

  container: { margin: '10px 0', padding: '10px', border: '1px solid #ccc', borderRadius: '5px' },

  button: { margin: '5px', padding: '5px 10px', backgroundColor: '#4caf50', color: 'white', border: 'none', borderRadius: '3px' },

};

export default StateDemo;

**ReducerDemo.js**

import React, { useReducer } from 'react';

const initialState = { count: 0 };

function reducer(state, action) {

  switch (action.type) {

    case 'increment':

      return { count: state.count + 1 };

    case 'decrement':

      return { count: state.count - 1 };

    case 'reset':

      return { count: 0 };

    default:

      return state;

  }

}

const ReducerDemo = () => {

const [state, dispatch] = useReducer(reducer, initialState);

  return (

    <div style={styles.container}>

      <h3>useReducer Demo</h3>

      <p>Count: {state.count}</p>

      <button onClick={() => dispatch({ type: 'increment' })} style={styles.button}>Increment</button>

      <button onClick={() => dispatch({ type: 'decrement' })} style={styles.button}>Decrement</button>

      <button onClick={() => dispatch({ type: 'reset' })} style={styles.button}>Reset</button>

    </div>

  );

};

const styles = {

  container: { margin: '10px 0', padding: '10px', border: '1px solid #ccc', borderRadius: '5px' },

  button: { margin: '5px', padding: '5px 10px', backgroundColor: '#008cba', color: 'white', border: 'none', borderRadius: '3px' },

};

export default ReducerDemo;

**RefDemo.js**

import React, { useRef } from 'react';

const RefDemo = () => {

 const inputRef = useRef();

  const focusInput = () => {

    inputRef.current.focus();

  };

  return (

    <div style={styles.container}>

      <h3>useRef Demo</h3>

      <input type="text" ref={inputRef} placeholder="Focus me!" style={styles.input} />

      <button onClick={focusInput} style={styles.button}>Focus Input</button>

    </div>

  );

};

const styles = {

  container: { margin: '10px 0', padding: '10px', border: '1px solid #ccc', borderRadius: '5px' },

  input: { width: '100%', padding: '5px', margin: '10px 0', borderRadius: '3px', border: '1px solid #ccc' },

  button: { margin: '5px', padding: '5px 10px', backgroundColor: '#f0ad4e', color: 'white', border: 'none', borderRadius: '3px' },

};

export default RefDemo;

**StateDemo.js**

import React, { useState } from 'react';

const StateDemo = () => {

const [count, setCount] = useState(0);

  return (

    <div style={styles.container}>

      <h3>useState Demo</h3>

      <p>Count: {count}</p>

      <button onClick={() => setCount(count + 1)} style={styles.button}>Increment</button>

      <button onClick={() => setCount(count - 1)} style={styles.button}>Decrement</button>

    </div>

  );

};

const styles = {

  container: { margin: '10px 0', padding: '10px', border: '1px solid #ccc', borderRadius: '5px' },

  button: { margin: '5px', padding: '5px 10px', backgroundColor: '#4caf50', color: 'white', border: 'none', borderRadius: '3px' },

};

export default StateDemo;

**App.js**

import React from 'react';

import './App.css';

import StateDemo from './components/StateDemo';

import EffectDemo from './components/EffectDemo';

import ReducerDemo from './components/ReducerDemo';

import RefDemo from './components/RefDemo';

import ContextDemo from './components/ContextDemo';

import { AppProvider } from './components/AppContext';

function App() {

  return (

    <AppProvider>

      <div className="App">

        <h1>React Hooks Demo</h1>

        <StateDemo />

        <EffectDemo />

        <ReducerDemo />

        <RefDemo />

        <ContextDemo />

      </div>

    </AppProvider>

  );

}

export default App;

**index.js**

import React from "react";

import ReactDOM from "react-dom/client";

import "./index.css";

import App from "./App";

import reportWebVitals from "./reportWebVitals";

const root = ReactDOM.createRoot(document.getElementById("root"));

root.render(

  <React.StrictMode>

    <App />

  </React.StrictMode>

);

reportWebVitals();

Output :

