**Explain about React form**

In **React**, forms are used to capture user input, similar to HTML forms. However, instead of the browser handling the form elements directly, React gives you control over them through **state**. This makes forms more predictable and easier to debug.

Key ideas:

React tracks input field values in the **component state**.

You can respond to user input immediately using event handlers (e.g., onChange).

Forms are often used for authentication, search, filtering, etc.

**Define controlled components**

A controlled component in React is an input element whose value is controlled by React state.

function NameForm() {

const [name, setName] = useState('');

return (

<input

type="text"

value={name}

onChange={(e) => setName(e.target.value)}

/>

);

}

**Explain about various input controls**

| **Control Type** | **React Code Example** |
| --- | --- |
| **Text input** | <input type="text" value={state} onChange={handleChange} /> |
| **Textarea** | <textarea value={state} onChange={handleChange} /> |
| **Checkbox** | <input type="checkbox" checked={state} onChange={handleChange} /> |
| **Radio buttons** | <input type="radio" value="A" checked={state === 'A'} onChange={handleChange} /> |
| **Select dropdown** | <select value={state} onChange={handleChange}>...</select> |

**Explain about handling forms**

Handling forms in React usually involves:

1. Storing form values in state.
2. Updating state on input changes.
3. Preventing default form submission.
4. Handling validation or transformation if needed.

function LoginForm() {

const [email, setEmail] = useState('');

const [password, setPassword] = useState('');

const handleSubmit = (e) => {

e.preventDefault(); // Prevent default page reload

console.log(email, password); // You can send to API here

};

return (

<form onSubmit={handleSubmit}>

<input type="email" value={email} onChange={(e) => setEmail(e.target.value)} />

<input type="password" value={password} onChange={(e) => setPassword(e.target.value)} />

<button type="submit">Login</button>

</form>

);

}

**Explain about submitting forms**

Form submission is handled using the onSubmit event in React. You usually:

* Attach onSubmit={handleSubmit} to the <form> element.
* Use e.preventDefault() to stop the browser's default behavior.
* Do something with the form data (like call an API).

**Form submission process:**

1. User clicks Submit.
2. handleSubmit function is triggered.
3. The function can validate, sanitize, or send the data.

**Create a component named “ComplaintRegister” with a form containing a textbox to enter the employee name and a textarea to enter the complaint. Use “handleSubmit” event of the button to submit the complaint and generate a Reference number for further follow ups in the alert box.**

**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();

**App.js :-**

import React from 'react';

import ComplaintRegister from './ComplaintRegister';

function App() {

  return (

    <div className="App">

      <ComplaintRegister />

    </div>

  );

}

export default App;

**ComplaintRegister.js :-**

// src/ComplaintRegister.js

import React, { useState } from 'react';

function ComplaintRegister() {

  const [name, setName] = useState('');

  const [complaint, setComplaint] = useState('');

  const handleSubmit = (e) => {

    e.preventDefault();

    // Generate a random transaction ID

    const transactionId = Math.floor(Math.random() \* 1000) + 1;

    alert(

      `Thanks ${name}\nYour Complaint was Submitted.\nTransaction ID is: ${transactionId}`

    );

    // Clear form

    setName('');

    setComplaint('');

  };

  return (

    <div style={{ textAlign: 'center', marginTop: '50px' }}>

      <h2 style={{ color: 'red' }}>Register your complaints here!!!</h2>

      <form onSubmit={handleSubmit}>

        <div>

          <label>Name: </label>

          <input

            type="text"

            value={name}

            onChange={(e) => setName(e.target.value)}

            required

          />

        </div>

        <br />

        <div>

          <label>Complaint: </label>

          <textarea

            value={complaint}

            onChange={(e) => setComplaint(e.target.value)}

            required

          />

        </div>

        <br />

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

      </form>

    </div>

  );

}

export default ComplaintRegister;

**Output :-**





