Ex.No:11
Date:02-09-2025

PROGRAMS ON CONTROLLED AND UNCONTROLLED COMPONENTS

Aim:

To write and execute the programs on controlled and uncontrolled components and to verify the output.

Program:

a) Controlled components:

Job Application Form

An HR portal needs a Job Application Form built with controlled components.
Requirements:

- Inputs (controlled): Name (text), Email (email), Phone Number (text)
  Select Dropdown → Job Role (Developer, Designer, Manager)
  Radio Buttons → Gender (Male, Female, Other)
  Checkboxes → Skills (React, Node.js, SQL, Python)
  Textarea → Description
- Validation: All fields required
  Email must contain @
  Phone must be 10 digits
  At least one skill must be selected
  On Submit → Show the collected data in an alert (or console).

App.js:

```
import React from "react";
import JobApplicationForm from "./components/JobApplicationForm";

const App = () => {
  return (
    <div style={{ padding: "20px", fontFamily: "Arial" }}>
      <h1>HR Portal</h1>
      <JobApplicationForm />
    </div>
  );
};

export default App;
```

JobApplicationForm.js:

```
import React, { useState } from "react";
```

```
const JobApplicationForm = () => {
  const [formData, setFormData] = useState({
    name: "",
    email: "",
    phone: "",
    role: "Developer",
    gender: "",
    skills: [],
    description: ""
  });

  const handleChange = (e) => {
    const { name, value, type, checked } = e.target;

    if (type === "checkbox") {
      let updatedSkills = [...formData.skills];
      if (checked) {
      updatedSkills.push(value);
      } else {
        updatedSkills = updatedSkills.filter((skill) => skill !== value);
      }
      setFormData({ ...formData, skills: updatedSkills });
    } else {
      setFormData({ ...formData, [name]: value });
    }
  };

  const handleSubmit = (e) => {
    e.preventDefault();

    // Validation
    if (
      !formData.name ||
      !formData.email ||
      !formData.phone ||
      !formData.gender ||
      !formData.description ||
      formData.skills.length === 0
    ) {
      alert("All fields are required.");
      return;
    }
    if (!formData.email.includes("@")) {
      alert("Email must contain '@'.");
      return;
    }
    if (!/^\d{10}$/.test(formData.phone)) {
```

```
    alert("Phone number must be 10 digits.");
    return;
  }

  // Show collected data
  alert(JSON.stringify(formData, null, 2));
};

return (
  <div>
    <h2>Job Application Form</h2>
    <form onSubmit={handleSubmit}>
      <div>
        <input
          type="text"
          name="name"
          placeholder="Name"
          value={formData.name}
          onChange={handleChange}
        />
      </div>

      <div>
        <input
          type="email"
          name="email"
          placeholder="Email"
          value={formData.email}
          onChange={handleChange}
        />
      </div>

      <div>
        <input
          type="text"
          name="phone"
          placeholder="Phone Number"
          value={formData.phone}
          onChange={handleChange}
        />
      </div>

      <div>
        <select name="role" value={formData.role} onChange={handleChange}>
          <option value="Developer">Developer</option>
          <option value="Designer">Designer</option>
          <option value="Manager">Manager</option>
```

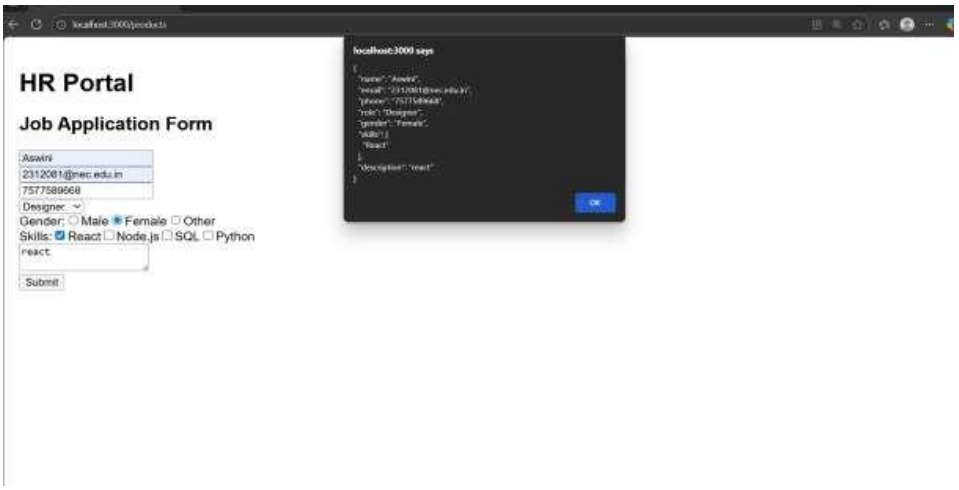
```
  </select>
</div>

<div>
  Gender:
  <label>
    <input
      type="radio"
      name="gender"
      value="Male"
      checked={formData.gender === "Male"}
      onChange={handleChange}
    />
    Male
  </label>
  <label>
    <input
      type="radio"
      name="gender"
      value="Female"
      checked={formData.gender === "Female"}
      onChange={handleChange}
    />
    Female
  </label>
  <label>
    <input
      type="radio"
      name="gender"
      value="Other"
      checked={formData.gender === "Other"}
      onChange={handleChange}
    />
    Other
  </label>
</div>

<div>
  Skills:
  <label>
    <input
      type="checkbox"
      name="skills"
      value="React"
      checked={formData.skills.includes("React")}
      onChange={handleChange}
    />
```

```
          React
        </label>
        <label>
          <input
            type="checkbox"
            name="skills"
            value="Node.js"
            checked={formData.skills.includes("Node.js")}
            onChange={handleChange}
          />
          Node.js
        </label>
        <label>
          <input
            type="checkbox"
            name="skills"
            value="SQL"
            checked={formData.skills.includes("SQL")}
            onChange={handleChange}
          />
          SQL
        </label>
        <label>
          <input
            type="checkbox"
            name="skills"
            value="Python"
            checked={formData.skills.includes("Python")}
            onChange={handleChange}
          />
          Python
        </label>
      </div>

      <div>
        <textarea
          name="description"
          placeholder="Description"
          value={formData.description}
          onChange={handleChange}
        ></textarea>
      </div>

      <button type="submit">Submit</button>
    </form>
  </div>
);
```

};

export default JobApplicationForm;

Output:



b) Uncontrolled Component:

Employee Feedback Form (Uncontrolled Components)
The company wants a feedback form where employees can submit feedback without saving input values in React state (i.e., uncontrolled components).
Fields: Employee Name , Email, Feedback (textarea), Rating (1–5 select dropdown)
Use useRef to read values on submit instead of useState.
On submit → Show entered details in an alert/console.

App.js:

```
import React from "react";
import EmployeeFeedbackForm from "./components/EmployeeFeedbackForm";

const App = () => {
  return (
    <div style={{ padding: "20px", fontFamily: "Arial" }}>
      <h1>HR Portal</h1>
      <EmployeeFeedbackForm />
    </div>
  );
};
```

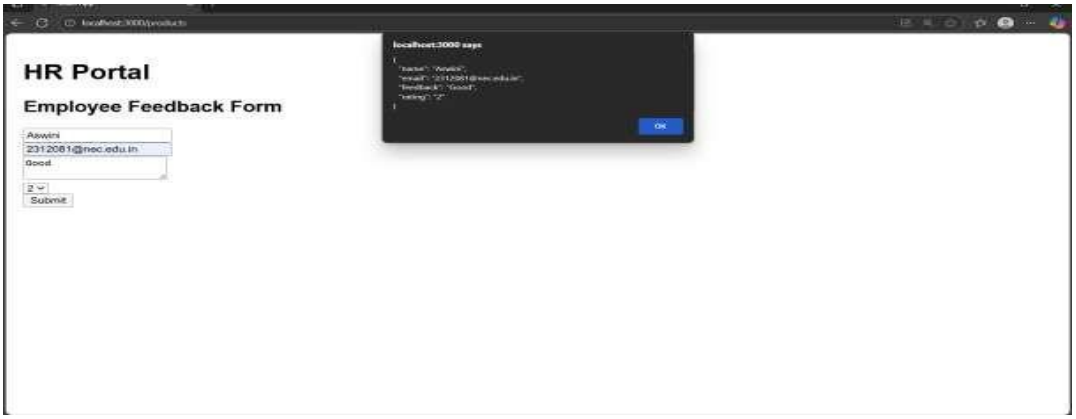
```
export default App;
```

EmployeeFeedbackForm:

```
import React, { useRef } from "react";

const EmployeeFeedbackForm = () => {
  const nameRef = useRef();
  const emailRef = useRef();
  const feedbackRef = useRef();
  const ratingRef = useRef();

  const handleSubmit = (e) => {
    e.preventDefault();

    const data = {
      name: nameRef.current.value,
      email: emailRef.current.value,
      feedback: feedbackRef.current.value,
      rating: ratingRef.current.value,
    };

    alert(JSON.stringify(data, null, 2));
  };

  return (
    <div>
      <h2>Employee Feedback Form</h2>
      <form onSubmit={handleSubmit}>
        <div>
          <input type="text" placeholder="Employee Name" ref={nameRef} />
        </div>

        <div>
          <input type="email" placeholder="Email" ref={emailRef} />
        </div>

        <div>
          <textarea placeholder="Feedback" ref={feedbackRef}></textarea>
        </div>

        <div>
          <select ref={ratingRef}>
            <option value="1">1</option>
            <option value="2">2</option>
            <option value="3">3</option>
            <option value="4">4</option>
            <option value="5">5</option>
          </select>
        </div>
```

```
        <button type="submit">Submit</button>
      </form>
    </div>
  );
};

export default EmployeeFeedbackForm;
```

Output:



Problem Analysis (10)	Coding & Implementation (10)	Time Management (10)	Viva (10)	Total (10)

Result:

Thus the programs on controlled and uncontrolled components was executed successfully and the output was verified.