

## **Don Bosco Institute of Technology**

Kurla (W), Mumbai 400 070

### **Internet Programming**

Name: Parth Pravin SHikhare

Class: TE-IT

Roll No.: 53

Date: 17/09/25

### **Experiment No.: 7**

#### **Title:**

To Create a web pages using React Form Validations

#### **Problem Definition:**

To create web pages that demonstrate form validations using React along with JavaScript concepts such as Validations, Arrays, String manipulations, and Date functions.

#### **Pre-requisites:**

1. Basic knowledge of HTML5, CSS3, and JavaScript
2. Understanding of React.js framework and JSX syntax
3. Familiarity with React components, props, and state
4. A text editor or IDE (like VS Code) with Node.js installed
5. Knowledge of JavaScript arrays, strings, and date handling

#### **Theory:**

React is a JavaScript library for building user interfaces, particularly single-page applications. React components allow for modular, reusable, and efficient UI design. Form validation in React can be achieved using controlled components, event handling, and JavaScript functions.

Key Concepts covered in this experiment:

- JavaScript Validations: Ensuring input fields follow specific criteria (e.g., email format, password length).
- Arrays: Storing and processing collections of data using array methods like push, pop, map, filter.
- String Functions: Manipulating user input using methods like length, toUpperCase,

toLowerCase, substring.

- Date Functions: Working with JavaScript Date object for validation (e.g., age calculation, current date checks).
- React Form Handling: Using state hooks (useState) to manage input values and error messages.
- Event Handling: Validating on form submission or onChange events.

### Procedure:

1. Set up a new React project using Create React App (CRA) or Vite.
2. Create a form component with input fields such as Name, Email, Password, and Date of Birth.
3. Use useState hooks to manage input values and errors.
4. Implement JavaScript validation functions to check email format, password rules, and date validation.
5. Use array methods to process multiple entries or form data lists.
6. Apply string methods to validate and transform input values (e.g., trimming spaces, converting cases).
7. Use JavaScript Date functions to validate date fields (e.g., ensuring user age is above 18).
8. Attach event handlers (onChange, onSubmit) to trigger validation functions.
9. Display validation error messages dynamically using conditional rendering.
10. Test the React app in a web browser and refine code as necessary.

### Program:

#### Form.jsx

```
import React, { useState } from "react";  
import "../Form.css"; // External CSS file (styles below)
```

```
const Form = () => {  
  const [formData, setFormData] = useState({  
    name: "",  
    password: "",  
    tel: "",  
    address: "",  
    age: "",  
    gender: "",  
    hobbies: [],  
    technology: "Java",  
    info: "",  
  });  
  
  const handleChange = (e) => {  
    const { name, value, type, checked } = e.target;  
  
    if (type === "checkbox") {
```

```

setFormData((prev) => {
  const hobbies = checked
    ? [...prev.hobbies, value]
    : prev.hobbies.filter((hobby) => hobby !== value);
  return { ...prev, hobbies };
});
} else if (type === "radio") {
  setFormData((prev) => ({ ...prev, [name]: value }));
} else {
  setFormData((prev) => ({ ...prev, [name]: value }));
}
};

const handleSubmit = (e) => {
  e.preventDefault();
  const { name, password, tel, age } = formData;

  const nameRegex = /^[A-Za-z]+$;/;
  if (!nameRegex.test(name)) {
    alert("Username should not contain numbers");
    return;
  }

  if (password.length < 8) {
    alert("Password should be at least 8 characters long");
    return;
  }

  if (age < 8 || age > 80) {
    alert("Age should be between 8 and 80");
    return;
  }

  const phoneRegex = /^[0-9]{10}$/;
  if (!phoneRegex.test(tel)) {
    alert("Phone number should contain exactly 10 digits");
    return;
  }

  const pname = prompt("Enter your name");
  if (pname !== null) {
    const conf = confirm("Do you want to submit?");
    if (conf) {
      alert("Confirm Successful");
    }
  }
};

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

```

```

const pname = prompt("Enter your name");
if (pname !== null) {
  const conf = confirm("Do you want to cancel?");
  if (conf) {
    alert("Cancel Successful");
    setFormData({
      name: "",
      password: "",
      tel: "",
      address: "",
      age: "",
      gender: "",
      hobbies: [],
      technology: "Java",
      info: "",
    });
  }
}
};

return (
  <div>
    <h1>Don Bosco Institute of Technology</h1>
    <h3>Department of Information Technology</h3>
    <h2>TE Form</h2>

    <form onSubmit={handleSubmit}>
      <label>Username :</label>
      <input
        type="text"
        name="name"
        placeholder="Enter your username"
        value={formData.name}
        onChange={handleChange}
        required
      />

      <label>Password :</label>
      <input
        type="password"
        name="password"
        placeholder="Enter your password"
        value={formData.password}
        onChange={handleChange}
        required
      />

      <label>Phone No. :</label>
      <input
        type="tel"

```

```
name="tel"
placeholder="Enter your phone number"
value={formData.tel}
onChange={handleChange}
required
/>
```

```
<label>Address :</label>
<textarea
  name="address"
  placeholder="Enter your address"
  value={formData.address}
  onChange={handleChange}
/>
```

```
<label>Age :</label>
<input
  type="number"
  name="age"
  value={formData.age}
  onChange={handleChange}
  required
/>
```

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

```
<div className="inline-options">
  <label>Hobbies :</label>
  <input
    type="checkbox"
    name="hobby"
    value="Sports"
```

```

        checked={formData.hobbies.includes("Sports")}
        onChange={handleChange}
      />{" "}
      Sports
      <input
        type="checkbox"
        name="hobby"
        value="Reading"
        checked={formData.hobbies.includes("Reading")}
        onChange={handleChange}
      />{" "}
      Reading
      <input
        type="checkbox"
        name="hobby"
        value="Music"
        checked={formData.hobbies.includes("Music")}
        onChange={handleChange}
      />{" "}
      Music
    </div>

    <label>Technology :</label>
    <select
      name="technology"
      value={formData.technology}
      onChange={handleChange}
    >
      <option value="Java">Java</option>
      <option value="C">C</option>
      <option value="Python">Python</option>
    </select>

    <label>Personal Information :</label>
    <textarea
      name="info"
      placeholder="Enter your personal information"
      value={formData.info}
      onChange={handleChange}
    />

    <button type="submit">Submit</button>
    <button type="reset" onClick={handleCancel}>
      Cancel
    </button>
  </form>
</div>
);
};

```

```
export default Form;
```

## Form.css

```
/* Make the entire page a centered flex container */
body {
  font-family: "Segoe UI", Arial, sans-serif;
  background: linear-gradient(135deg, #e3f2fd, #90caf9);
  margin: 0;
  height: 100vh;
  display: flex;
  justify-content: center;
  align-items: center;
}

/* Wrapper to hold headings and form together */
.container {
  text-align: center;
  width: 100%;
  max-width: 600px;
}

/* Headings styling */
h1,
h2,
h3 {
  margin: 5px 0;
  color: #0d47a1;
}

h1 {
  font-size: 28px;
  font-weight: 700;
}

h2 {
  font-size: 22px;
  font-weight: 600;
}

h3 {
  font-size: 18px;
  font-weight: 500;
}

/* Form container */
form {
  margin-top: 15px;
}
```

```
padding: 30px;
background: #ffffff;
border-radius: 16px;
box-shadow: 0 6px 20px rgba(0, 0, 0, 0.1);
text-align: left;
transition: all 0.3s ease;
}
```

```
form:hover {
  transform: translateY(-3px);
}
```

```
/* Labels */
```

```
label {
  display: block;
  margin-bottom: 6px;
  font-weight: 600;
  color: #0d47a1;
}
```

```
/* Inputs, Textareas, Selects */
```

```
input[type="text"],
input[type="email"],
input[type="number"],
input[type="tel"],
input[type="password"],
textarea,
select {
  width: 100%;
  padding: 10px 12px;
  margin-bottom: 18px;
  border: 1px solid #cfd8dc;
  border-radius: 8px;
  font-size: 15px;
  color: #333;
  background-color: #fafafa;
  transition: 0.3s;
}
```

```
input:focus,
textarea:focus,
select:focus {
  outline: none;
  border-color: #2196f3;
  box-shadow: 0 0 6px rgba(33, 150, 243, 0.3);
  background-color: #fff;
}
```

```
/* Inline radio and checkbox groups */
```

```
.inline-options {
```



```
margin-bottom: 18px;
}
```

```
.inline-options label {
margin-right: 15px;
font-weight: normal;
color: #333;
}
```

```
.inline-options input {
margin-right: 6px;
}
```

```
/* Buttons */
button {
padding: 10px 22px;
margin: 10px 10px 0 0;
border: none;
border-radius: 8px;
font-size: 15px;
cursor: pointer;
transition: all 0.3s ease;
}
```

```
/* Submit button */
button[type="submit"] {
background: #2196f3;
color: white;
font-weight: 600;
}
```

```
button[type="submit"]:hover {
background: #1976d2;
box-shadow: 0 3px 8px rgba(25, 118, 210, 0.3);
}
```

```
/* Reset button */
button[type="reset"] {
background: #f44336;
color: white;
font-weight: 600;
}
```

```
button[type="reset"]:hover {
background: #d32f2f;
box-shadow: 0 3px 8px rgba(211, 47, 47, 0.3);
}
```

```
/* Textareas */
textarea {
```

```
resize: vertical;
min-height: 80px;
}

/* Responsive design */
@media (max-width: 600px) {
  .container {
    padding: 0 15px;
  }

  form {
    padding: 20px;
  }

  button {
    width: 100%;
    margin-bottom: 10px;
  }
}
```

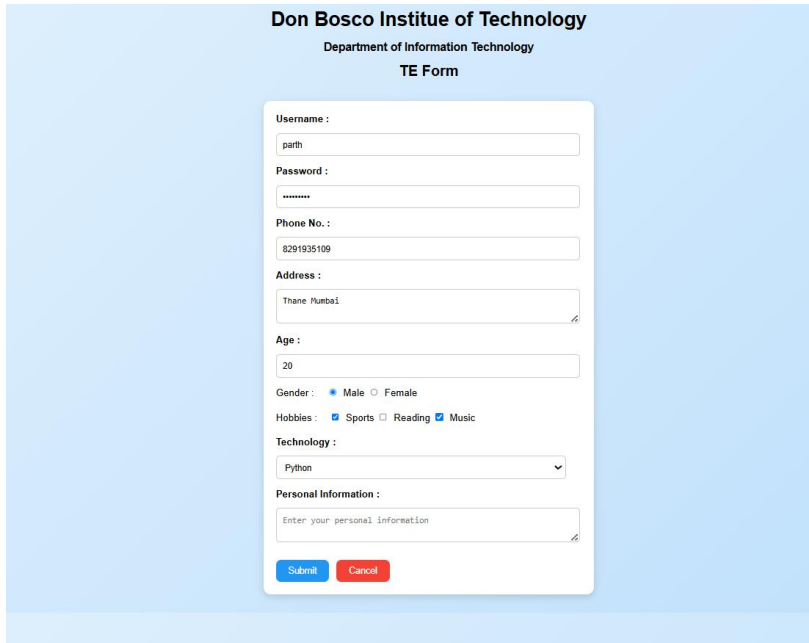
### **App.jsx**

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

const App = () => {
  return <Form />;
};

export default App;
```

## Output:



The screenshot displays a web form for 'Don Bosco Institute of Technology', specifically for the 'Department of Information Technology' and 'TE Form'. The form is centered on a light blue background. It contains several input fields: 'Username' (filled with 'parth'), 'Password' (masked with dots), 'Phone No.' (filled with '8291935109'), 'Address' (filled with 'Thane Mumbai'), 'Age' (filled with '20'), 'Gender' (radio buttons for 'Male' and 'Female', with 'Male' selected), 'Hobbies' (checkboxes for 'Sports', 'Reading', and 'Music', with 'Sports' and 'Music' selected), 'Technology' (a dropdown menu showing 'Python'), and 'Personal Information' (a text area with the placeholder 'Enter your personal information'). At the bottom of the form are two buttons: 'Submit' (blue) and 'Cancel' (red).

## Results:

Successfully created a React-based web page implementing form validations using JavaScript validations, arrays, string methods, and date functions.

## References:

1. <https://react.dev/>
2. <https://developer.mozilla.org/en-US/docs/Web/JavaScript>
3. <https://www.w3schools.com/react/>