

# TASK

## User Registration Form

### Create a user registration form with validation.

You are building a user registration form for a new web application. The form should collect the following information from the user:

First Name

Last Name

Email

Password

Confirm Password

### Create a form with the necessary input fields.

1. Add basic validation for required fields and email format.
2. Ensure that the password and confirm password fields match.
3. Display error messages for invalid inputs.
4. Implement a success message upon successful form submission.

## Booking Form

Create a booking form for scheduling appointments.

You are developing a booking form for scheduling appointments at a clinic. The form should collect the following information:

Full Name

Email

Phone Number

Appointment Date

Appointment Time

1. Create a form with the necessary input fields.
2. Add basic validation for required fields and email format.
3. Display a confirmation message upon successful submission.
4. Add date and time pickers for selecting the appointment date and time.

## **Solution –**

### **App.js**

```
import React from 'react';  
import './App.css'; // Import the CSS file  
import UserRegistration from './UserRegistration';  
import BookingForm from './BookingForm';
```

```
const App = () => {  
  return (  
    <div>  
      <h1>Form Examples</h1>  
      <UserRegistration />  
      <BookingForm />  
    </div>  
  );  
};
```

```
export default App;
```

```

my-forms-app > src > JS App.js > ...
1  import React from 'react';
2  import './App.css'; // Import the CSS file
3  import UserRegistration from './UserRegistration';
4  import BookingForm from './BookingForm';
5
6  const App = () => {
7    return (
8      <div>
9        <h1>Form Examples</h1>
10       <UserRegistration />
11       <BookingForm />
12     </div>
13   );
14 };
15
16 export default App;
17

```

## UserRegistration.js

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

```

const UserRegistration = () => {
  const [formData, setFormData] = useState({
    firstName: "",
    lastName: "",
    email: "",
    password: "",
    confirmPassword: ""
  });

```

```

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

```

```

const handleChange = (e) => {
  const { name, value } = e.target;
  setFormData((prevData) => ({

```

```
...prevData,  
[name]: value  
}));  
};
```

```
const validate = () => {  
  const newErrors = {};  
  
  const { firstName, lastName, email, password, confirmPassword } =  
    formData;  
  
  // Required fields validation  
  if (!firstName) newErrors.firstName = 'First name is required';  
  if (!lastName) newErrors.lastName = 'Last name is required';  
  if (!email) newErrors.email = 'Email is required';  
  if (!password) newErrors.password = 'Password is required';  
  if (!confirmPassword) newErrors.confirmPassword = 'Confirm password is  
required';  
  
  // Email format validation  
  if (email && !/^S+@\S+\.\S+/.test(email)) {  
    newErrors.email = 'Email is invalid';  
  }  
  
  // Password and confirm password match validation  
  if (password && confirmPassword && password !== confirmPassword) {  
    newErrors.confirmPassword = 'Passwords do not match';  
  }  
}
```

```
    return newErrors;
};
```

```
const handleSubmit = (e) => {
    e.preventDefault();
    const validationErrors = validate();
    if (Object.keys(validationErrors).length === 0) {
        setSuccessMessage('Registration successful!');
        setErrors({});
    } else {
        setErrors(validationErrors);
        setSuccessMessage("");
    }
};
```

```
return (
    <div>
        <h2>User Registration</h2>
        <form onSubmit={handleSubmit}>
            <div>
                <label>First Name</label>
                <input
                    type="text"
                    name="firstName"
                    value={formData.firstName}
                    onChange={handleChange}
                />
            </div>
        </form>
    </div>
);
```

```
    {errors.firstName} && <span>{errors.firstName}</span>
  </div>
```

```
<div>
  <label>Last Name</label>
  <input
    type="text"
    name="lastName"
    value={formData.lastName}
    onChange={handleChange}
  />
  {errors.lastName} && <span>{errors.lastName}</span>
</div>
```

```
<div>
  <label>Email</label>
  <input
    type="email"
    name="email"
    value={formData.email}
    onChange={handleChange}
  />
  {errors.email} && <span>{errors.email}</span>
</div>
```

```
<div>
  <label>Password</label>
```

```

    <input
      type="password"
      name="password"
      value={formData.password}
      onChange={handleChange}
    />

    {errors.password && <span>{errors.password}</span>}
  </div>

  <div>
    <label>Confirm Password</label>
    <input
      type="password"
      name="confirmPassword"
      value={formData.confirmPassword}
      onChange={handleChange}
    />

    {errors.confirmPassword && <span>{errors.confirmPassword}</span>}
  </div>

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

  {successMessage && <p>{successMessage}</p>}
</form>
</div>

);
};

```

export default UserRegistration;

my-forms-app > src > JS UserRegistration.js > UserRegistration > handleChange > setFormData() callback > [name]

```
1  import React, { useState } from 'react';
2
3  const UserRegistration = () => {
4    const [formData, setFormData] = useState({
5      firstName: '',
6      lastName: '',
7      email: '',
8      password: '',
9      confirmPassword: ''
10   });
11
12   const [errors, setErrors] = useState({});
13   const [successMessage, setSuccessMessage] = useState('');
14
15   const handleChange = (e) => {
16     const { name, value } = e.target;
17     setFormData((prevData) => ({
18       ...prevData,
19       [name]: value
20     }));
21   };
22
23   const validate = () => {
24     const newErrors = {};
25     const { firstName, lastName, email, password, confirmPassword } = formData;
26
27     // Required fields validation
28     if (!firstName) newErrors.firstName = 'First name is required';
```

```
27   // Required fields validation
28   if (!firstName) newErrors.firstName = 'First name is required';
29   if (!lastName) newErrors.lastName = 'Last name is required';
30   if (!email) newErrors.email = 'Email is required';
31   if (!password) newErrors.password = 'Password is required';
32   if (!confirmPassword) newErrors.confirmPassword = 'Confirm password is required';
33
34   // Email format validation
35   if (email && !/\S+@\S+\.\S+/.test(email)) {
36     newErrors.email = 'Email is invalid';
37   }
38
39   // Password and confirm password match validation
40   if (password && confirmPassword && password !== confirmPassword) {
41     newErrors.confirmPassword = 'Passwords do not match';
42   }
43
44   return newErrors;
45   };
46
47   const handleSubmit = (e) => {
48     e.preventDefault();
49     const validationErrors = validate();
50     if (Object.keys(validationErrors).length === 0) {
51       setSuccessMessage('Registration successful!');
```



```

51     setSuccessMessage('Registration successful!');
52     setErrors({});
53   } else {
54     setErrors(validationErrors);
55     setSuccessMessage('');
56   }
57   };
58
59   return (
60     <div>
61       <h2>User Registration</h2>
62       <form onSubmit={handleSubmit}>
63         <div>
64           <label>First Name</label>
65           <input
66             type="text"
67             name="firstName"
68             value={formData.firstName}
69             onChange={handleChange}
70           />
71           {errors.firstName && <span>{errors.firstName}</span>}
72         </div>
73
74         <div>
75           <label>Last Name</label>

```

```

74         <div>
75           <label>Last Name</label>
76           <input
77             type="text"
78             name="lastName"
79             value={formData.lastName}
80             onChange={handleChange}
81           />
82           {errors.lastName && <span>{errors.lastName}</span>}
83         </div>
84
85         <div>
86           <label>Email</label>
87           <input
88             type="email"
89             name="email"
90             value={formData.email}
91             onChange={handleChange}
92           />
93           {errors.email && <span>{errors.email}</span>}
94         </div>
95
96         <div>
97           <label>Password</label>
98           <input
99             type="password"

```

```

my-forms-app > src > JS UserRegistration.js > [9] UserRegistration
3   const UserRegistration = () => {
99       type="password"
100      name="password"
101      value={formData.password}
102      onChange={handleChange}
103    />
104    {errors.password && <span>{errors.password}</span>}
105  </div>
106
107    <div>
108      <label>Confirm Password</label>
109      <input
110        type="password"
111        name="confirmPassword"
112        value={formData.confirmPassword}
113        onChange={handleChange}
114      />
115      {errors.confirmPassword && <span>{errors.confirmPassword}</span>}
116    </div>
117
118    <button type="submit">Register</button>
119    {successMessage && <p>{successMessage}</p>}
120  </form>
121 </div>
122 ];
123 };
124
125 export default UserRegistration;

```

## BookingForm.js

import React, { useState } from 'react';

```

const BookingForm = () => {
  const [formData, setFormData] = useState({
    fullName: "",
    email: "",
    phoneNumber: "",
    appointmentDate: "",
    appointmentTime: ""
  });

```

```

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

```

```

const [confirmationMessage, setConfirmationMessage] = useState("");

```

```
const handleChange = (e) => {  
  const { name, value } = e.target;  
  setFormData((prevData) => ({  
    ...prevData,  
    [name]: value  
  }));  
};
```

```
const validate = () => {  
  const newErrors = {};  
  const { fullName, email, phoneNumber, appointmentDate, appointmentTime  
} = formData;
```

```
  // Required fields validation  
  if (!fullName) newErrors.fullName = 'Full name is required';  
  if (!email) newErrors.email = 'Email is required';  
  if (!phoneNumber) newErrors.phoneNumber = 'Phone number is required';  
  if (!appointmentDate) newErrors.appointmentDate = 'Appointment date is  
required';  
  if (!appointmentTime) newErrors.appointmentTime = 'Appointment time is  
required';
```

```
  // Email format validation  
  if (email && !/^[S+@S+\.S+/.test(email)) {  
    newErrors.email = 'Email is invalid';  
  }
```

```
  return newErrors;
```

```
};
```

```
const handleSubmit = (e) => {  
  e.preventDefault();  
  const validationErrors = validate();  
  if (Object.keys(validationErrors).length === 0) {  
    setConfirmationMessage('Appointment booked successfully!');  
    setErrors({});  
  } else {  
    setErrors(validationErrors);  
    setConfirmationMessage("");  
  }  
};
```

```
return (  
  <div>  
    <h2>Booking Form</h2>  
    <form onSubmit={handleSubmit}>  
      <div>  
        <label>Full Name</label>  
        <input  
          type="text"  
          name="fullName"  
          value={formData.fullName}  
          onChange={handleChange}  
        />  
        {errors.fullName && <span>{errors.fullName}</span>}
```

```
</div>
```

```
<div>
```

```
  <label>Email</label>
```

```
  <input
```

```
    type="email"
```

```
    name="email"
```

```
    value={formData.email}
```

```
    onChange={handleChange}
```

```
  />
```

```
  {errors.email && <span>{errors.email}</span>}
```

```
</div>
```

```
<div>
```

```
  <label>Phone Number</label>
```

```
  <input
```

```
    type="text"
```

```
    name="phoneNumber"
```

```
    value={formData.phoneNumber}
```

```
    onChange={handleChange}
```

```
  />
```

```
  {errors.phoneNumber && <span>{errors.phoneNumber}</span>}
```

```
</div>
```

```
<div>
```

```
  <label>Appointment Date</label>
```

```
  <input
```

```

        type="date"
        name="appointmentDate"
        value={formData.appointmentDate}
        onChange={handleChange}
      />
      {errors.appointmentDate && <span>{errors.appointmentDate}</span>}
    </div>

    <div>
      <label>Appointment Time</label>
      <input
        type="time"
        name="appointmentTime"
        value={formData.appointmentTime}
        onChange={handleChange}
      />
      {errors.appointmentTime && <span>{errors.appointmentTime}</span>}
    </div>

    <button type="submit">Book Appointment</button>
    {confirmationMessage && <p>{confirmationMessage}</p>}
  </form>
</div>
);
};

export default BookingForm;

```

my-forms-app > src > JS BookingForm.js > [⌘] BookingForm

```
1  import React, { useState } from 'react';
2
3  const BookingForm = () => {
4    const [formData, setFormData] = useState({
5      fullName: '',
6      email: '',
7      phoneNumber: '',
8      appointmentDate: '',
9      appointmentTime: ''
10   });
11
12   const [errors, setErrors] = useState({});
13   const [confirmationMessage, setConfirmationMessage] = useState('');
14
15   const handleChange = (e) => {
16     const { name, value } = e.target;
17     setFormData((prevData) => ({
18       ...prevData,
19       [name]: value
20     }));
21   };
22
23   const validate = () => {
24     const newErrors = {};
25     const { fullName, email, phoneNumber, appointmentDate, appointmentTime } = formData;
26
27     // Required fields validation
28     if (!fullName) newErrors.fullName = 'Full name is required';
```

my-forms-app > src > JS BookingForm.js > ...

```
3    const BookingForm = () => {
23     const validate = () => {
29       if (!email) newErrors.email = 'Email is required';
30       if (!phoneNumber) newErrors.phoneNumber = 'Phone number is required';
31       if (!appointmentDate) newErrors.appointmentDate = 'Appointment date is required';
32       if (!appointmentTime) newErrors.appointmentTime = 'Appointment time is required';
33
34       // Email format validation
35       if (email && !/\S+@\S+\.\S+/.test(email)) {
36         newErrors.email = 'Email is invalid';
37       }
38
39       return newErrors;
40     };
41
42     const handleSubmit = (e) => {
43       e.preventDefault();
44       const validationErrors = validate();
45       if (Object.keys(validationErrors).length === 0) {
46         setConfirmationMessage('Appointment booked successfully!');
47         setErrors({});
48       } else {
49         setErrors(validationErrors);
50         setConfirmationMessage('');
51       }
52     };
53   };
54 }
```

```

54   return (
55     <div>
56       <h2>Booking Form</h2>
57       <form onSubmit={handleSubmit}>
58         <div>
59           <label>Full Name</label>
60           <input
61             type="text"
62             name="fullName"
63             value={formData.fullName}
64             onChange={handleChange}
65           />
66           {errors.fullName && <span>{errors.fullName}</span>}
67         </div>
68
69         <div>
70           <label>Email</label>
71           <input
72             type="email"
73             name="email"
74             value={formData.email}
75             onChange={handleChange}
76           />
77           {errors.email && <span>{errors.email}</span>}
78         </div>
79

```

my-forms-app > src > JS BookingForm.js > ...

```

3   const BookingForm = () => {
80     <div>
81       <label>Phone Number</label>
82       <input
83         type="text"
84         name="phoneNumber"
85         value={formData.phoneNumber}
86         onChange={handleChange}
87       />
88       {errors.phoneNumber && <span>{errors.phoneNumber}</span>}
89     </div>
90
91     <div>
92       <label>Appointment Date</label>
93       <input
94         type="date"
95         name="appointmentDate"
96         value={formData.appointmentDate}
97         onChange={handleChange}
98       />
99       {errors.appointmentDate && <span>{errors.appointmentDate}</span>}
100    </div>
101
102    <div>
103      <label>Appointment Time</label>
104      <input
105        type="time"

```



```

105         type="time"
106         name="appointmentTime"
107         value={formData.appointmentTime}
108         onChange={handleChange}
109     />
110     {errors.appointmentTime && <span>{errors.appointmentTime}</span>}
111 </div>
112
113     <button type="submit">Book Appointment</button>
114     {confirmationMessage && <p>{confirmationMessage}</p>}
115 </form>
116 </div>
117 );
118 };
119
120 export default BookingForm;
121

```

## OUTPUT-

The screenshot displays a web application with two main sections: "Form Examples" and "User Registration". Below these, there are two forms: "User Registration" and "Booking Form".

**User Registration Form:**

- First Name:
- Last Name:
- Email:
- Password:
- Confirm Password:
- 
- Registration successful!

**Booking Form:**

- Full Name:
- Email:
- Phone Number:
- Appointment Date:
- Appointment Time:
- 
- Appointment booked successfully!