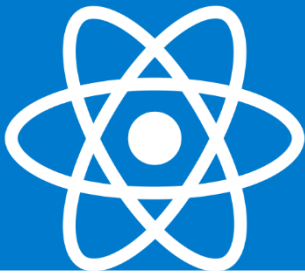




# React JS

#React Notes

# | Validation



# Create Component

- Create Class Component

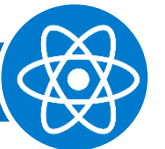
```
JS App.js M X
src > JS App.js > [default]
1  import React from 'react';
2  class App extends React.Component {
3    constructor(props) {
4      super(props);
5      this.state = {  };
6    }
7    render() {
8      return (
9        <h2>React Validation</h2>
10      );
11    }
12  }
13  export default App;
```



# Design State

- State Value will be Display in Textbox

```
App.js M x
src > App.js > default
1  import React from 'react';
2  class App extends React.Component {
3    constructor(props) {
4      super(props);
5      this.state = {
6        username : "",
7        password : "",};
8    }
9    render() {
10     const { username, password} = this.state;
11     return (
12       <React.Fragment>
13         <h2>React Validation</h2>
14         <form onSubmit={this.onSubmit}>
15           Name :<input type="text" name="username" value={username} />
16           <br/>
17           Password :<input type="text" name="password" value={password} />
18           <br/>
19           <input type="submit" value="Login"/>
20         </form>
21       </React.Fragment>
22     );
23   }
24 }
25 export default App;
```



# OnChange and OnSubmit

JS App.js M X

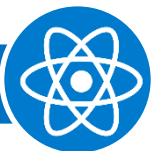
src > JS App.js > App

```
1 import React from 'react';
2 class App extends React.Component {
3   constructor(props) {
4     super(props);
5     this.state = {
6       username : "",
7       password : "",};
8   }
9   onChange= (e) => {
10    this.setState({
11      [e.target.name]: e.target.value
12    });
13  }
14  onSubmit = (e) =>{
15    e.preventDefault();
16    console.log("onSubmit",this.state);
17  }
```

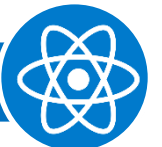
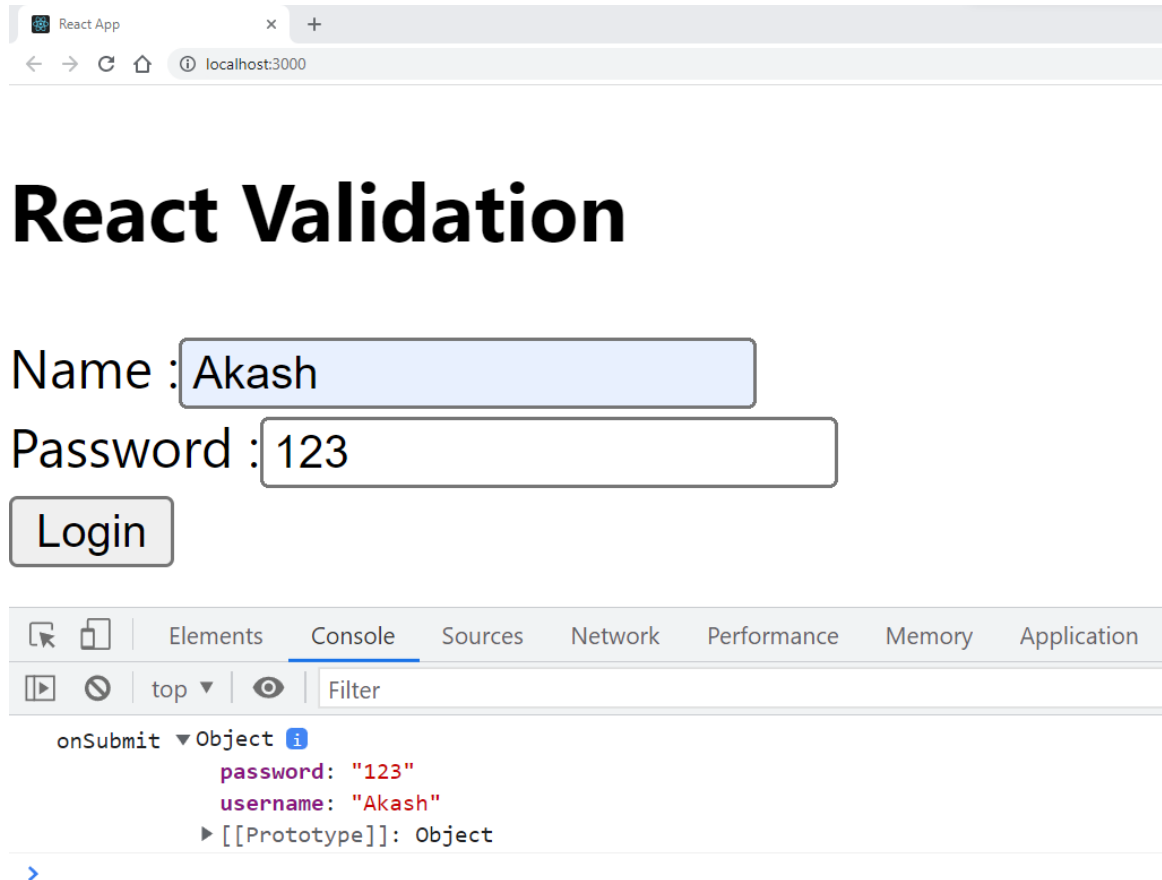
JS App.js M X

src > JS App.js > App

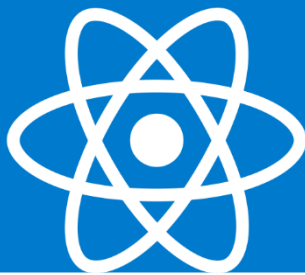
```
19   render() {
20     const { username, password} = this.state;
21     return (
22       <React.Fragment>
23       <h2>React Validation</h2>
24       <form onSubmit={this.onSubmit}>
25         Name :<input type="text" name="username" value={username} onChange={this.onChange.bind(this)} />
26         <br/>
27         Password :<input type="text" name="password" value={password} onChange={this.onChange.bind(this)} />
28         <br/>
29         <input type="submit" value="Login"/>
30       </form>
31     </React.Fragment>
32   );
33 }
34 }
35 export default App;
```



# Basic Form is Ready with OnChange and Submit 😊



# | Validation Process



# Define Error Object in State

- Create Blank Error Object which will stores all error in Object

```
this.state = {  
  username: "",  
  password: "",  
  errors: {}  
};
```

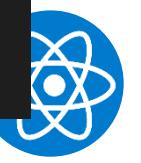
```
formValidation = () => {  
  const { username, password } = this.state; //Desctucting Assignment  
  let isValid = true;  
  const errors = {}; //Blank Object
```

```
  if (!username) {  
    errors.username = "Enter Username";  
    isValid = false;  
  }
```

```
  if (!password) {  
    errors.password = "Enter Password";  
    isValid = false;  
  }
```

```
  this.setState({ errors }); //Append Error Object in State  
  return isValid; //Function Return  
}
```

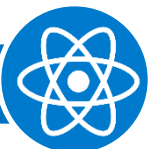
```
App.js  M  X  
src > App.js > App > constructor  
1  import React from 'react';  
2  class App extends React.Component {  
3    constructor(props) {  
4      super(props);  
5      this.state = {  
6        username: "",  
7        password: "",  
8        errors: {}  
9      };  
10   }  
11   onChange= (e) => {  
12     this.setState({  
13       [e.target.name]: e.target.value  
14     });  
15   }  
16   //Validation  
17   formValidation = () => {  
18     const { username, password } = this.state; //Desctucting Assignment  
19     let isValid = true;  
20     const errors = {}; //Blank Object  
21  
22     if (!username) {  
23       errors.username = "Enter Username";  
24       isValid = false;  
25     }  
26     if (!password) {  
27       errors.password = "Enter Password";  
28       isValid = false;  
29     }  
30     this.setState({ errors }); //Append Error Object in State  
31     return isValid; //Function Return  
32   }  
33 }
```



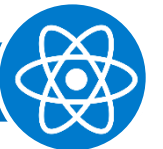


# Statement

- First we will create blank object in State
  - errors: {}
- Create One Validation Functions which will return Boolean value.(True/False)
- First Define Boolean variable for validating Data.
- Extract /assign state data in variable
- Write if Condition to check state value is present or not
  - If state value is not present then return false in variable and Store Error message in Error Object.
  - So on.....
- At last Error Object data assign to Error State and Return Boolean value.



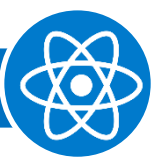
- If Data is validation Boolean value will return True and Error Object will return blank.
- Create Submit Event on Submit Print Error Value in Console



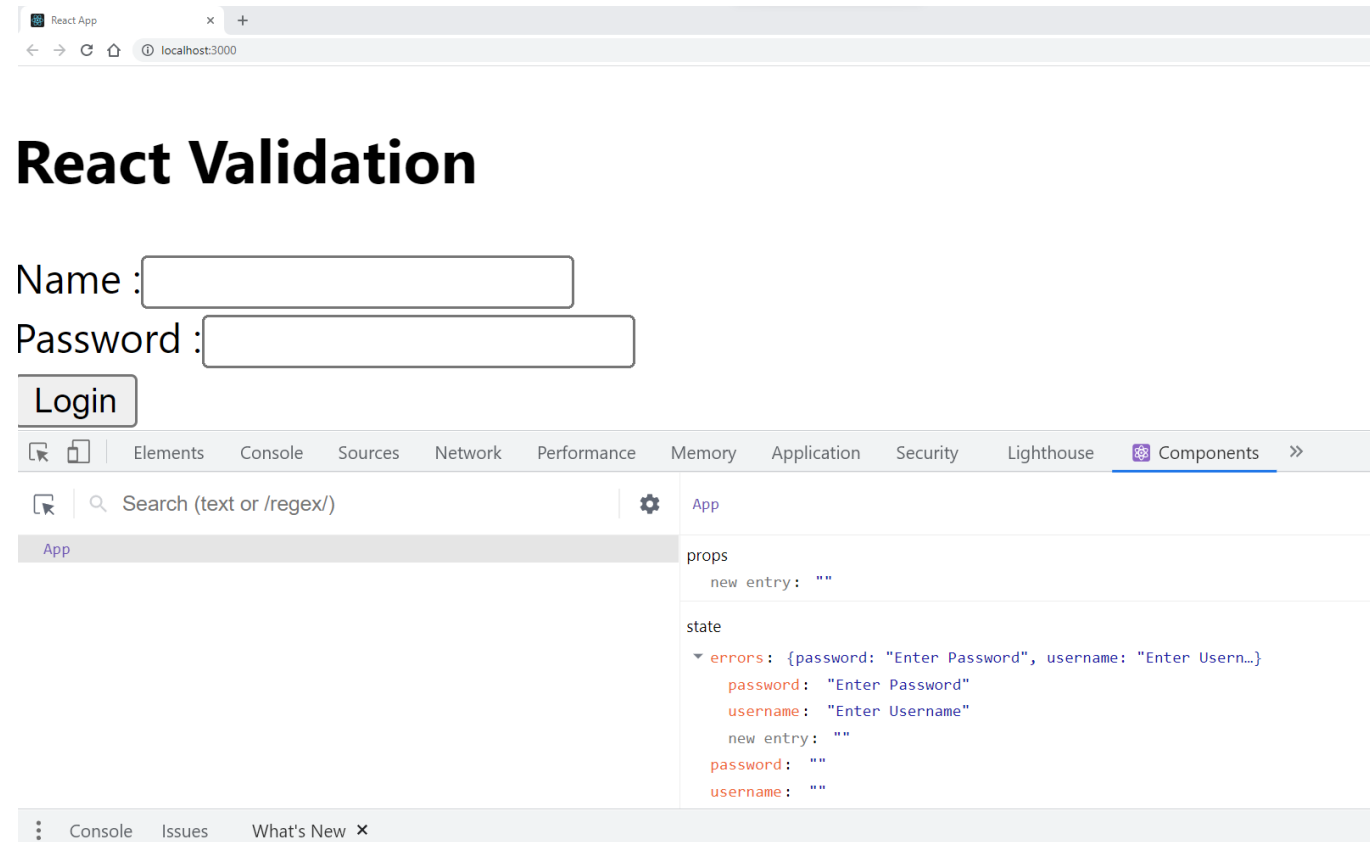
# OnSubmit Call Validation

```
onSubmit = (e) =>{  
  e.preventDefault();  
  const isValid = this.formValidation(); //get Validation  
  console.log("onSubmit",this.state);  
}
```

```
App.js 1, M x  
src > App.js > App > onSubmit  
15 }  
16 //Validation  
17 formValidation = () => {  
18   const { username, password } = this.state; //Destructuring Assignment  
19   let isValid = true;  
20   const errors = {}; //Blank Object  
21  
22   if (!username) {  
23     errors.username = "Enter Username";  
24     isValid = false;  
25   }  
26   if (!password) {  
27     errors.password = "Enter Password";  
28     isValid = false;  
29   }  
30   this.setState({ errors }); //Append Error Object in State  
31   return isValid; //Function Return  
32 }  
33  
34 onSubmit = (e) =>{  
35   e.preventDefault();  
36   const isValid = this.formValidation(); //get Validation  
37   console.log("onSubmit",this.state);  
38 }  
39
```

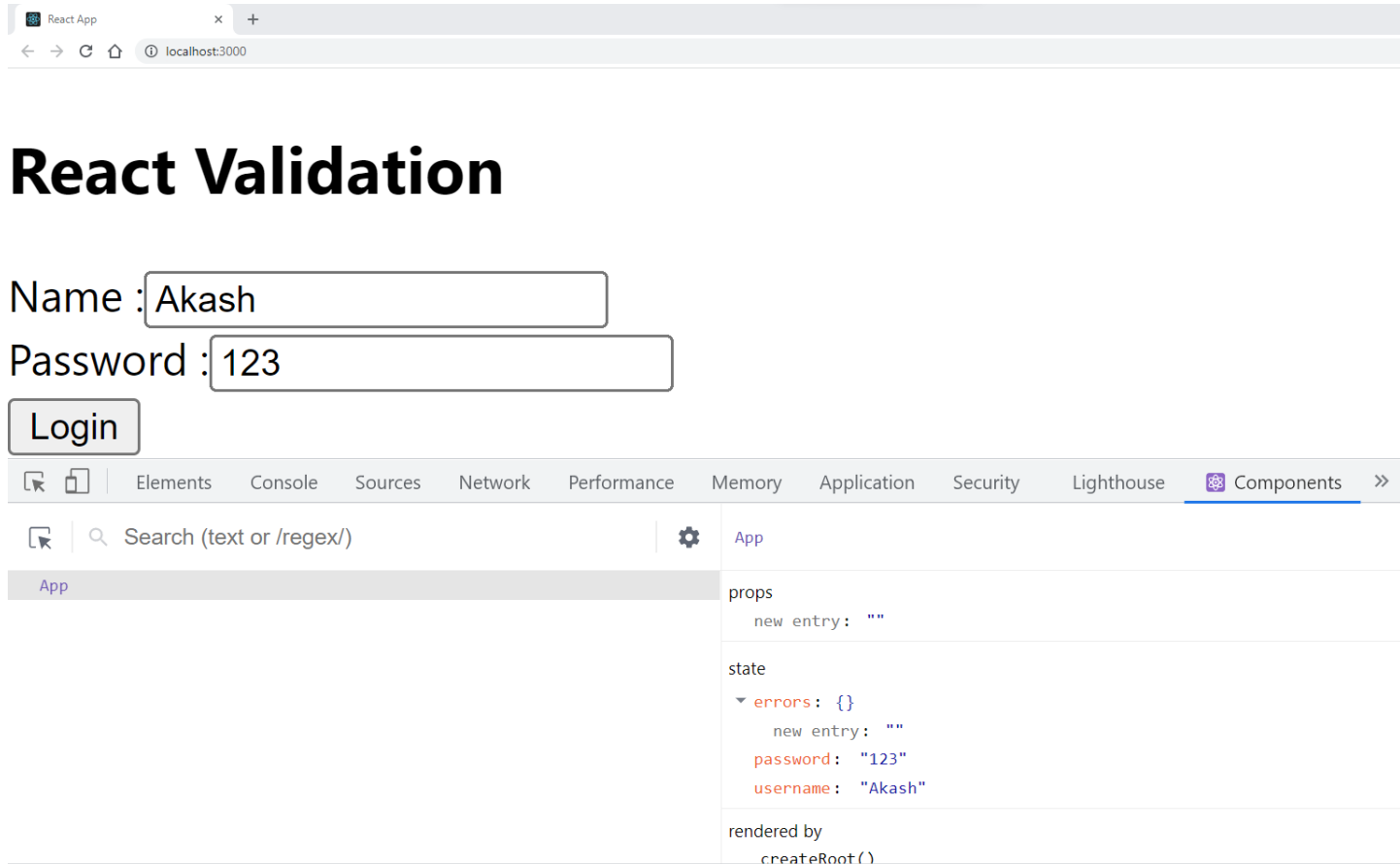


- OnSubmit Error Message will Append in Errors Object and we can access from React Developer Tool



# Validate Data

- On Valid Details Error Message will Hide

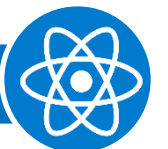


The screenshot shows a web browser window with a tab titled "React App" and the address bar displaying "localhost:3000". The page content includes a form with the following elements:

- A label "Name :" followed by a text input field containing the value "Akash".
- A label "Password :" followed by a text input field containing the value "123".
- A "Login" button.

Below the form, the React DevTools "Components" panel is open, showing the component tree for the "App" component. The state of the component is displayed as follows:

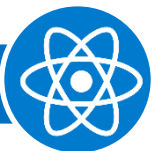
```
App
├── props
│   └── new entry: ""
├── state
│   ├── errors: {}
│   ├── new entry: ""
│   ├── password: "123"
│   └── username: "Akash"
└── rendered by
    └── createRoot()
```



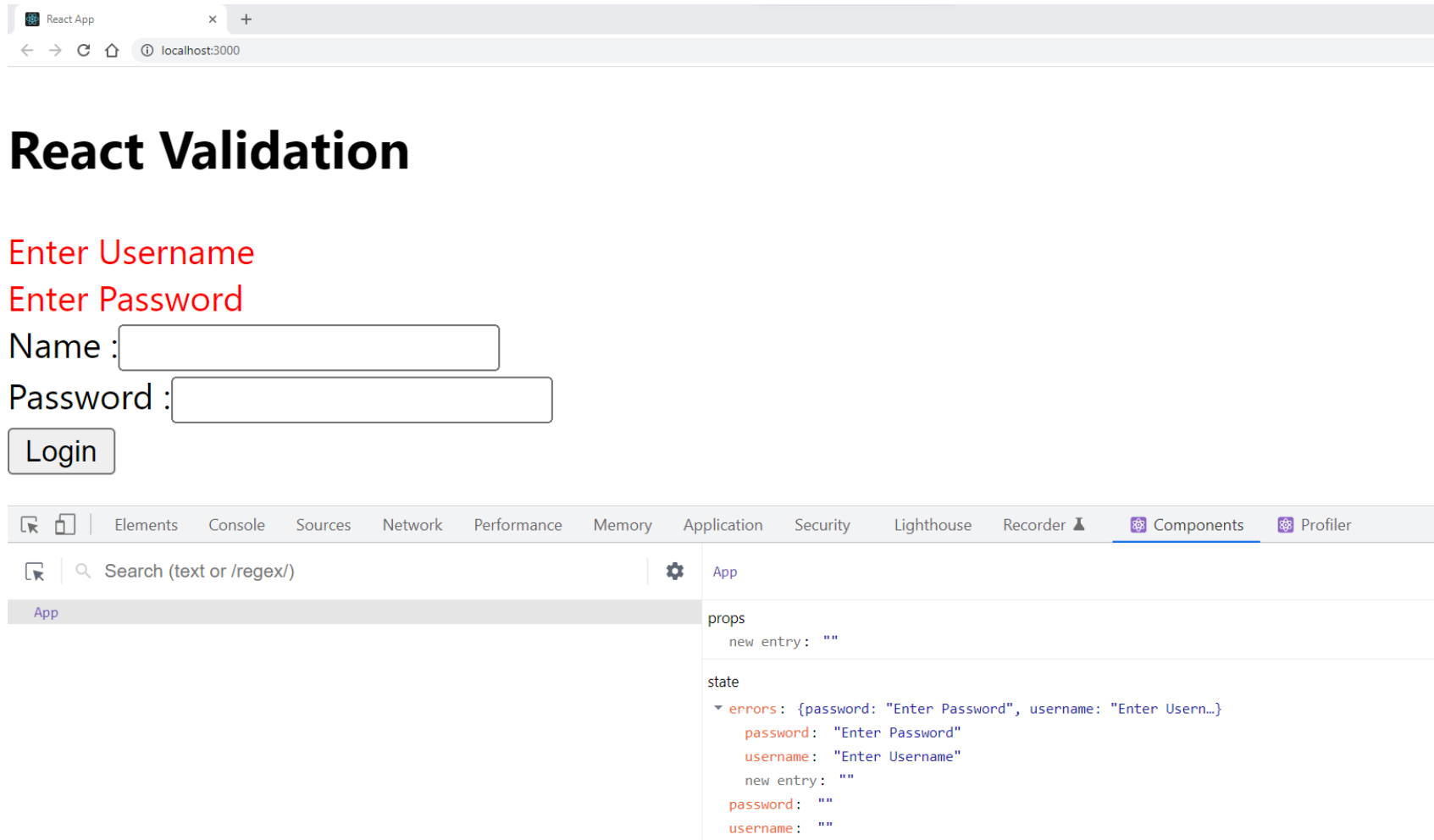
# Print Error Using Map

```
App.js 1, M X
src > App.js > App > render

40 render() {
41   const { username, password, errors } = this.state;
42   return (
43     <React.Fragment>
44       <h2>React Validation</h2>
45       <form onSubmit={this.onSubmit}>
46
47         {Object.keys(errors).map((key) => {
48           return <div style={{ color: 'red' }} key={key}>{errors[key]}</div>
49         })}
50
51       Name :<input type="text" name="username" value={username} onChange={this.onChange.bind(this)} />
52       <br/>
53       Password :<input type="text" name="password" value={password} onChange={this.onChange.bind(this)} />
54       <br/>
55       <input type="submit" value="Login"/>
56     </form>
57   </React.Fragment>
58 );
59 }
60 }
61 export default App;
```



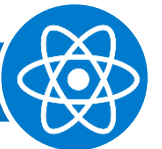
# Error Display



The screenshot shows a web browser window with a tab titled "React App" and the address bar displaying "localhost:3000". The page content includes the heading "React Validation", followed by two red error messages: "Enter Username" and "Enter Password". Below these are two input fields labeled "Name :" and "Password :", and a "Login" button.

The bottom of the image shows the React DevTools interface. The "Components" panel is active, displaying the component tree for "App". The state of the "App" component is shown, including the "errors" object:

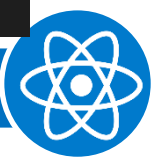
```
App
└─ props
   └─ new entry: ""
state
  └─ errors: {password: "Enter Password", username: "Enter Usern...}
      password: "Enter Password"
      username: "Enter Username"
      new entry: ""
      password: ""
      username: ""
```



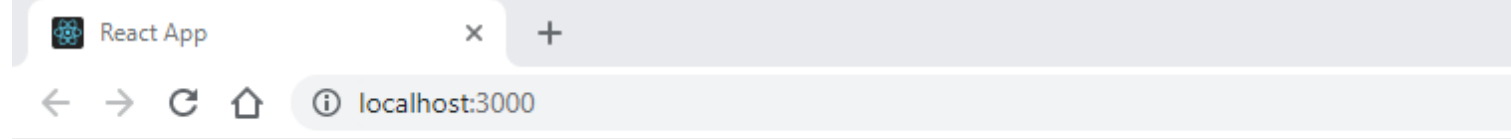
# OnSubmit Print Data and Clear State

```
JS App.js M X
src > JS App.js > App

33
34 onSubmit = (e) =>{
35   e.preventDefault();
36   const isValid = this.formValidation(); //get Validation
37   if (isValid) {
38     console.warn("Form Data " + this.state.username + this.state.password); // Print State Value
39     this.setState({ username: "", password: "", errors: {} }); // Remove State Value and Assign Blank
40   }
41
42 }
43
44 render() {
45   const { username, password, errors } = this.state;
46   return (
47     <React.Fragment>
48     <h2>React Validation</h2>
49     <form onSubmit={this.onSubmit}>
50
51     {Object.keys(errors).map((key) => {
52       return <div style={{ color: 'red' }} key={key}>{errors[key]}</div>
53     })}
54
55     Name :<input type="text" name="username" value={username} onChange={this.onChange.bind(this)} />
56     <br/>
```





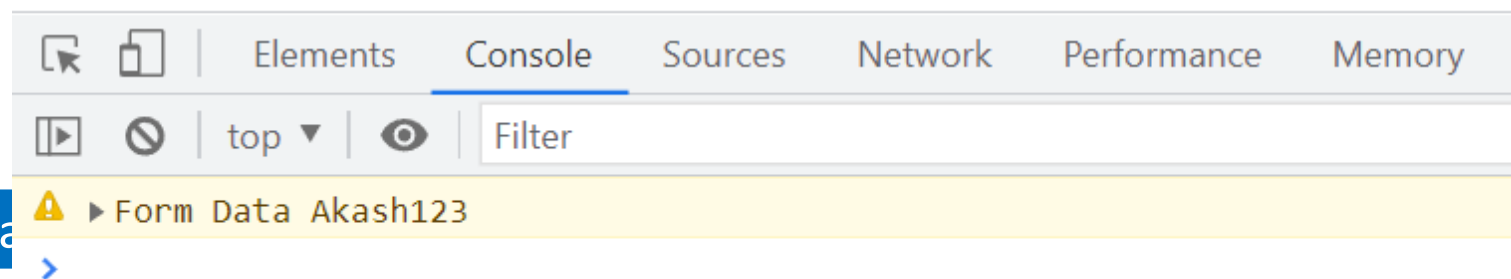


# React Validation

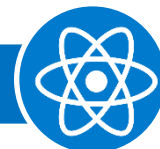
Name :

Password :

Login



Aka



# Example

- `<span style={{color: "red"}} >{this.state.errors.username}</span>`

```
App.js 1 X
src > App.js > default
45 render() {
46   const { username, password, errors } = this.state;
47   return (
48     <React.Fragment>
49       <h2>React Validation</h2>
50       <form onSubmit={this.onSubmit}>
51
52         Name :<input type="text" name="username" value={username} onChange={this.onChange.bind(this)} />
53         <span style={{color: "red"}} >{this.state.errors.username}</span>
54
55       <br />
56       Password :<input type="text" name="password" value={password} onChange={this.onChange.bind(this)} />
57       <span style={{color: "red"}} >{this.state.errors.password}</span>
58       <br />
59       <input type="submit" value="Login" />
60     </form>
61   </React.Fragment>
62 );
63 }
64 }
65 export default App;
66
```

React App

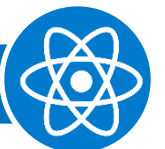
localhost:3000

## React Validation

Name :  Enter Username

Password :  Enter Password

Login



```

import React from 'react';
class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      username : "",
      password : "",
      errors: {}
    };
  }
  onChange= (e) => {
    this.setState({
      [e.target.name]: e.target.value
    });
  }
  //Validation
  formValidation = () => {
    const { username, password } = this.state; //Desctucting Assignment
    let isValid = true;
    const errors = {}; //Blank Object

    if (!username) {
      errors.username = "Enter Username";
      isValid = false;
    }
    if (!password) {
      errors.password = "Enter Password";
      isValid = false;
    }
    this.setState({ errors }); //Append Error Object in State
    return isValid; //Function Return
  }
}

```

```

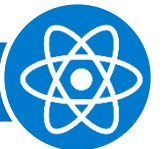
onSubmit = (e) =>{
  e.preventDefault();
  const isValid = this.formValidation(); //get Validation
  if (isValid) {
    console.warn("Form Data " + this.state.username + this.state.password); // Print State Value
    this.setState({ username: "", password: "", errors: {} }); // Remove State Value and Assign Blank
  }
}

render() {
  const { username, password, errors } = this.state;
  return (
    <React.Fragment>
    <h2>React Validation</h2>
    <form onSubmit={this.onSubmit}>

    {Object.keys(errors).map((key) => {
      return <div style={{ color: 'red' }} key={key}>{errors[key]}</div>
    })}

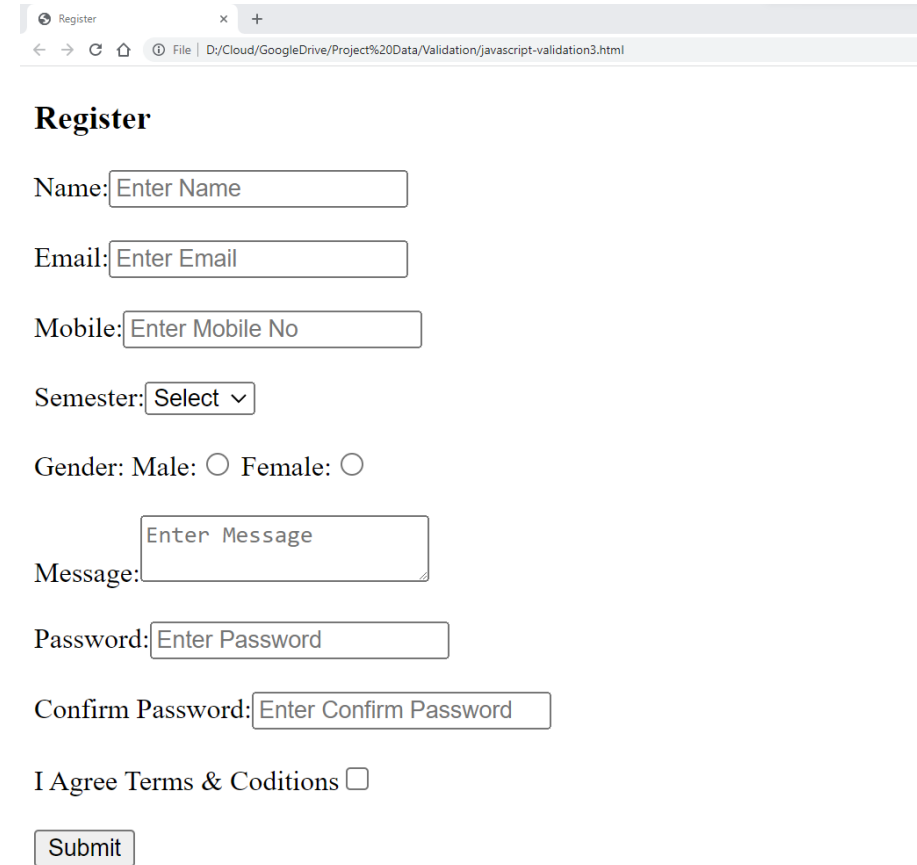
    Name :<input type="text" name="username" value={username} onChange={this.onChange.bind(this)} />
    <br/>
    Password :<input type="text" name="password" value={password} onChange={this.onChange.bind(this)} />
    <br/>
    <input type="submit" value="Login"/>
    </form>
    </React.Fragment>
  );
}
export default App;

```



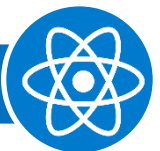
# Task

- Make Same Form in Class with Proper Validation
- Create Same program using Hooks



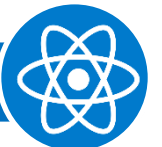
The screenshot shows a web browser window with a single tab titled 'Register'. The address bar displays the file path: 'D:/Cloud/GoogleDrive/Project%20Data/Validation/javascript-validation3.html'. The form itself is titled 'Register' and contains the following elements:

- Name:** A text input field with the placeholder text 'Enter Name'.
- Email:** A text input field with the placeholder text 'Enter Email'.
- Mobile:** A text input field with the placeholder text 'Enter Mobile No'.
- Semester:** A dropdown menu with 'Select' and a downward arrow.
- Gender:** Two radio buttons labeled 'Male' and 'Female'.
- Message:** A text area with the placeholder text 'Enter Message'.
- Password:** A text input field with the placeholder text 'Enter Password'.
- Confirm Password:** A text input field with the placeholder text 'Enter Confirm Password'.
- I Agree Terms & Coditions:** A checkbox.
- Submit:** A button.



```
if (!values.name) {  
    errors.name = 'Name is required';  
} else if (values.name.length > 15) {  
    errors.name = 'Must be 15 characters or less';  
}
```

```
if (!values.email) {  
    errors.email = 'Email is required';  
} else if (!/^([A-Z0-9._%+-]+@[A-Z0-9.-]+\.[A-Z]{2,4})$/i.test(values.email)) {  
    errors.email = 'Invalid email address';  
}
```



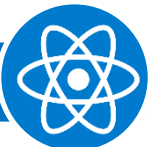
```
const validateEmployee = empData => {
  const errors = {};

  if (!empData.Name) {
    errors.Name = 'Please Enter Employee Name';
  } else if (empData.Name.length > 20) {
    errors.Name = 'Name cannot exceed 20 characters';
  }

  if (!empData.Location) {
    errors.Location = 'Please Enter Employee Location';
  }

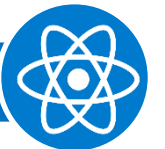
  if (!empData.EmailId) {
    errors.EmailId = 'Please Enter Email ID';
  } else if (!/^[A-Z0-9._%+-]+@[A-Z0-9.-]+\.[A-Z]{2,4}$/i.test(empData.EmailId)) {
    errors.EmailId = 'Invalid email address';
  }

  return errors;
};
```



# Validation Rules

- All Fields must be required
- Name must be Character ,
- Minimum 2 Character required in Name and Maximum 10 Characters.
- Email must be Valid Format
- Mobile Number must be == 10 Digits only
- Password and Confirm Password Must be Same.
- Take a Radio,CheckBox and Dropdown Field in Form with Proper Validation.

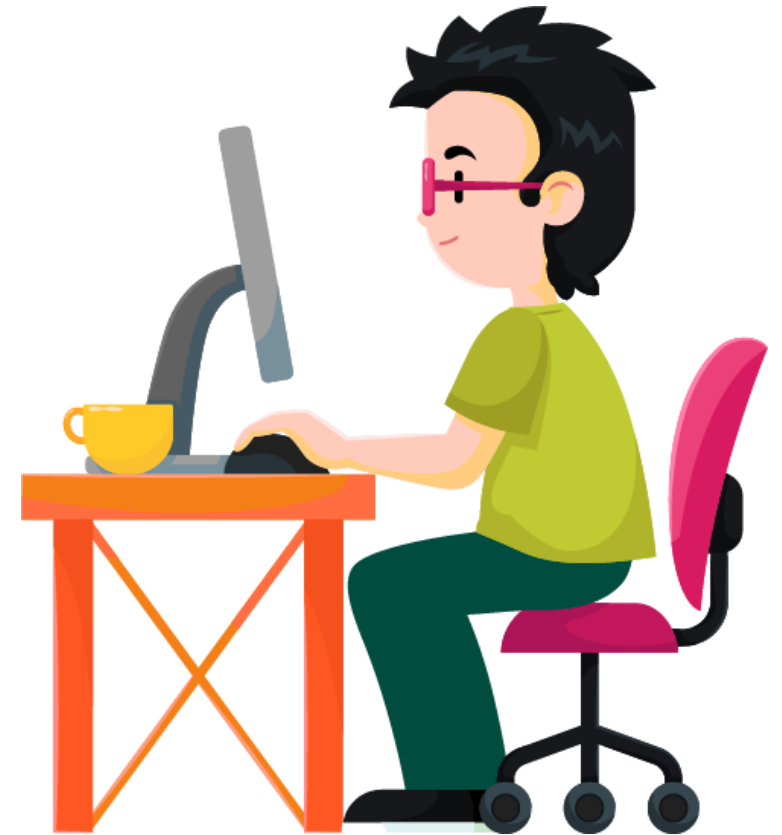


# Get Exclusive Video Tutorials



[www.apptutorials.com](http://www.apptutorials.com)

<https://www.youtube.com/user/Akashtips>







Get More Details

[www.akashsir.com](http://www.akashsir.com)



# If You Liked It !

## Rating Us Now



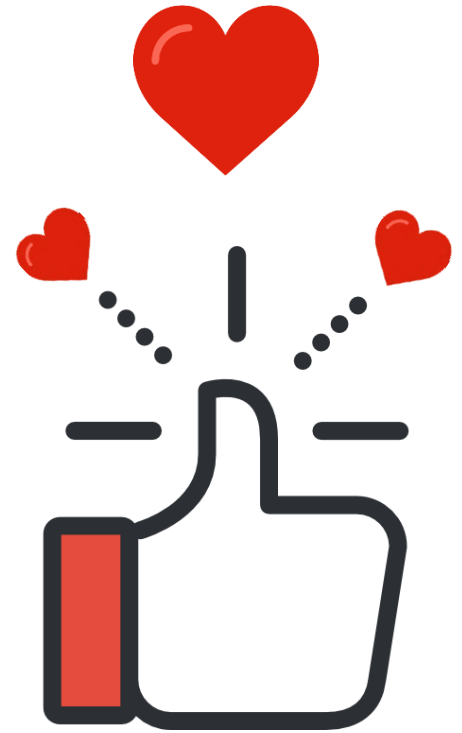
**Just Dial**

[https://www.justdial.com/Ahmedabad/Akash-Technolabs-Navrangpura-Bus-Stop-Navrangpura/079PXX79-XX79-170615221520-S5C4\\_BZDET](https://www.justdial.com/Ahmedabad/Akash-Technolabs-Navrangpura-Bus-Stop-Navrangpura/079PXX79-XX79-170615221520-S5C4_BZDET)



**Sulekha**

<https://www.sulekha.com/akash-technolabs-navrangpura-ahmedabad-contact-address/ahmedabad>



# Connect With Me



Akash Padhiyar  
#AkashSir

[www.akashsir.com](http://www.akashsir.com)

[www.akashtechlabs.com](http://www.akashtechlabs.com)

[www.akashpadhiyar.com](http://www.akashpadhiyar.com)

[www.apptutorials.com](http://www.apptutorials.com)

## # Social Info



Akash.padhiyar



Akashpadhiyar



Akash\_padhiyar



+91 99786-21654



#Akashpadhiyar  
#apptutorials