Forms in React

The standard way with Forms is to use Controlled Components. So we make React state to be the "single source of truth".

In HTML, form elements such as `<input>`, `<textarea>`, and `<select>` typically maintain their own state and update it based on user input. In React, mutable state is typically kept in the state property of components, and only updated with `setState()`.

We can combine the two by making the React state be the "single source of truth". Then the React component that renders a form also controls what happens in that form on subsequent user input. An input form element whose value is controlled by React in this way is called a "controlled component".

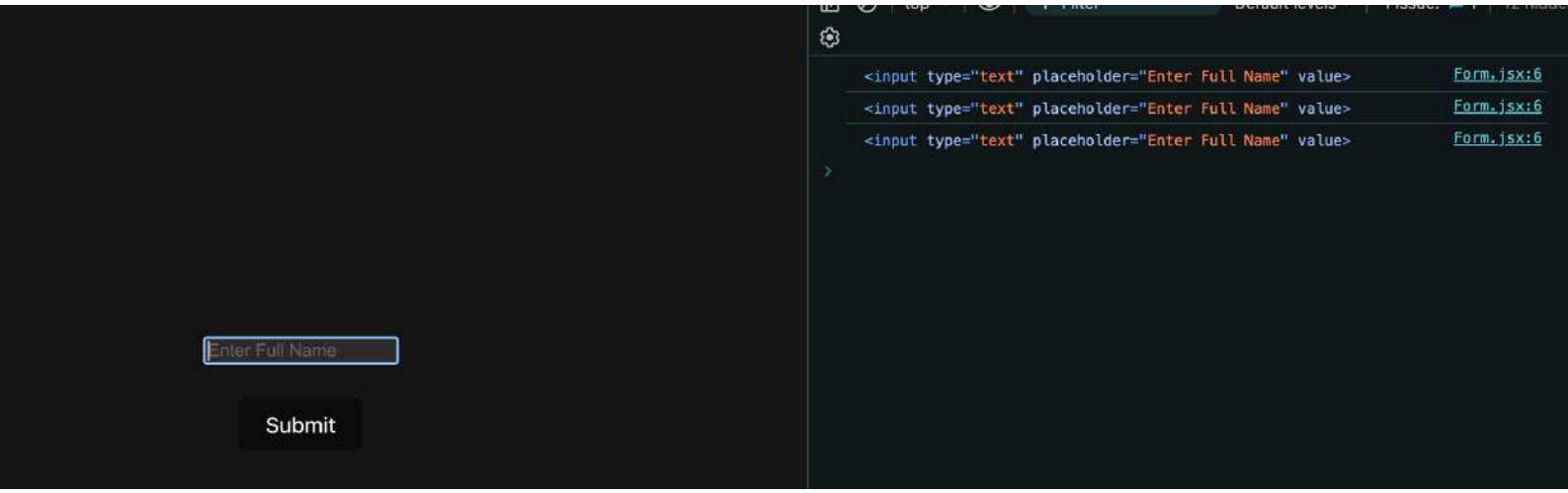
react old docs

App.jsx U Form.jsx U

Learning > FRONTEND > Apna College > React > React Lecture-6 > Forms_In_react > src > Form.jsx > Form > handleNameCha

```jsx
import {useState} from "react"
Complexity is 8 It's time to do something...
export default function Form(){
    let[fullname,setFullname]=useState("")

    let handleNameChange= (event) => {
        console.log(event.target);
        // console.log(event.target.value);
        // setFullname(event.target.value);
    };
    return(<form>
        <input type="text" placeholder="Enter Full Name" value={fullname} onChange={handleNameChange}/>
        <br /><br />
        <button>Submit</button>
    </form>)
}
```


Enter Full Name
Submit
<input type="text" placeholder="Enter Full Name" value>
Form.jsx:6
<input type="text" placeholder="Enter Full Name" value>
Form.jsx:6
<input type="text" placeholder="Enter Full Name" value>
Form.jsx:6

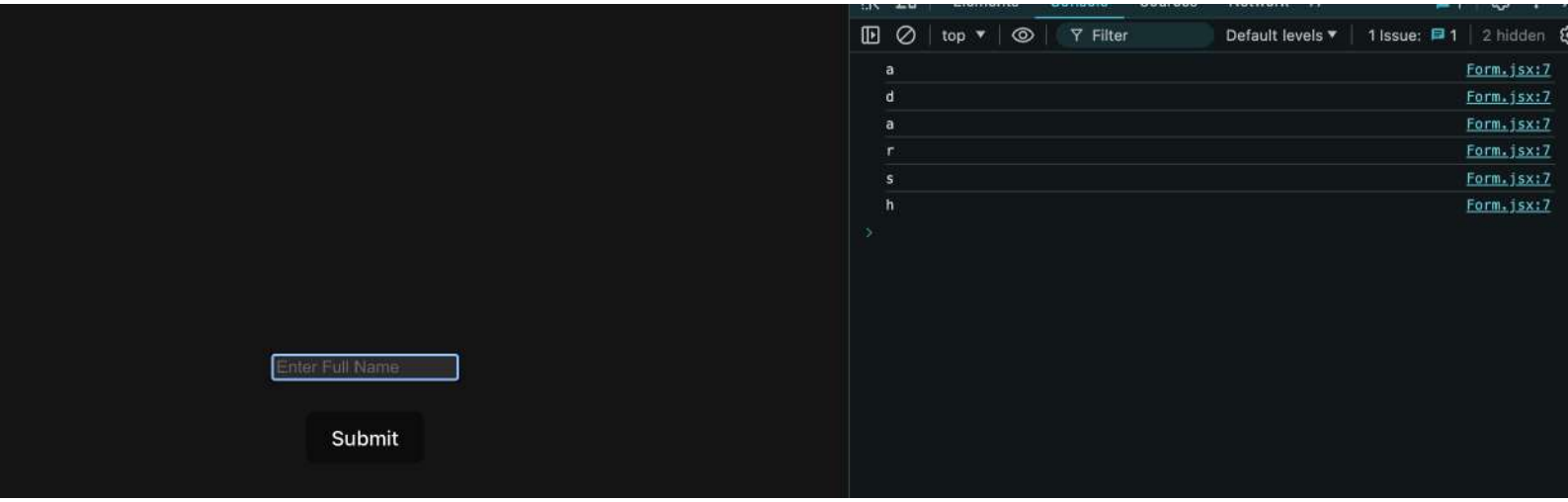
App.jsx U Form.jsx U

Learning > FRONTEND > Apna College > React > React Lecture-6 > Forms_In_react > src > Form.jsx > Form > handleNameC

```jsx
import {useState} from "react"
Complexity is 8 It's time to do something...
export default function Form(){
    let[fullname,setFullname]=useState("")

    let handleNameChange= (event) => {
        // console.log(event.target);
        console.log(event.target.value);
        // setFullname(event.target.value);
    };
    return(<form>
        <input type="text" placeholder="Enter Full Name" value={fullname} onChange={handleNameChange}/>
        <br /><br />
        <button>Submit</button>
    </form>)
}

```


Enter Full Name
Submit
top
Filter
Default levels
1 Issue: 1
2 hidden
a Form.jsx:7
d Form.jsx:7
a Form.jsx:7
r Form.jsx:7
s Form.jsx:7
h Form.jsx:7

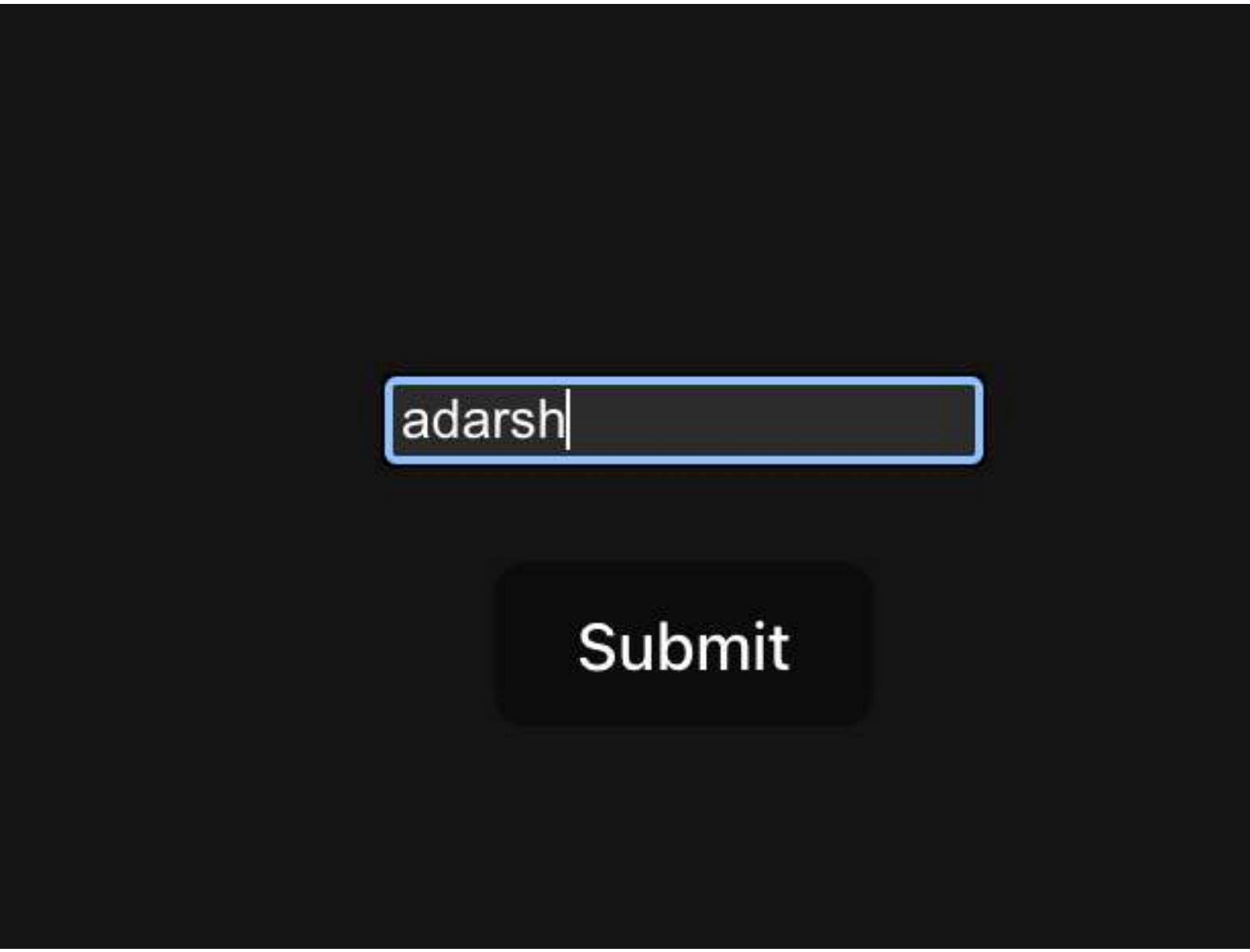
App.jsx U Form.jsx U

Learning > FRONTEND > Apna College > React > React Lecture-6 > Forms_In_react > src > Form.jsx > Form > handleName

```jsx
import {useState} from "react"
Complexity is 8 It's time to do something...
export default function Form(){
    let[fullname,setFullname]=useState("")

    let handleNameChange= (event) => {
        // console.log(event.target);
        // console.log(event.target.value);
        setFullname(event.target.value);
    };
    return(<form>
        <input type="text" placeholder="Enter Full Name" value={fullname} onChange={handleNameChange}/>
        <br /><br />
        <button>Submit</button>
    </form>)
}

```


adarsh
Submit

1) input change

↓

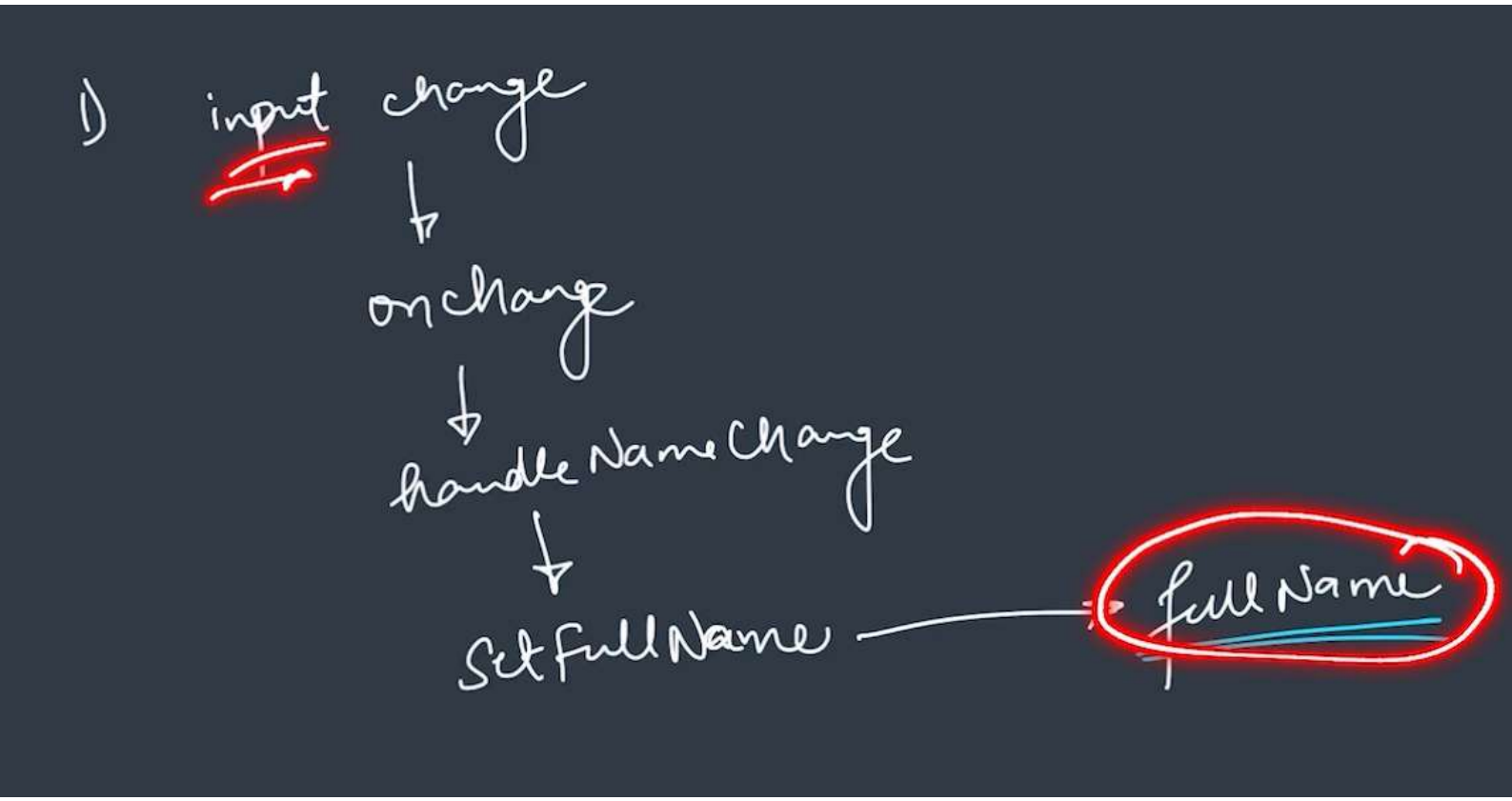
onChange

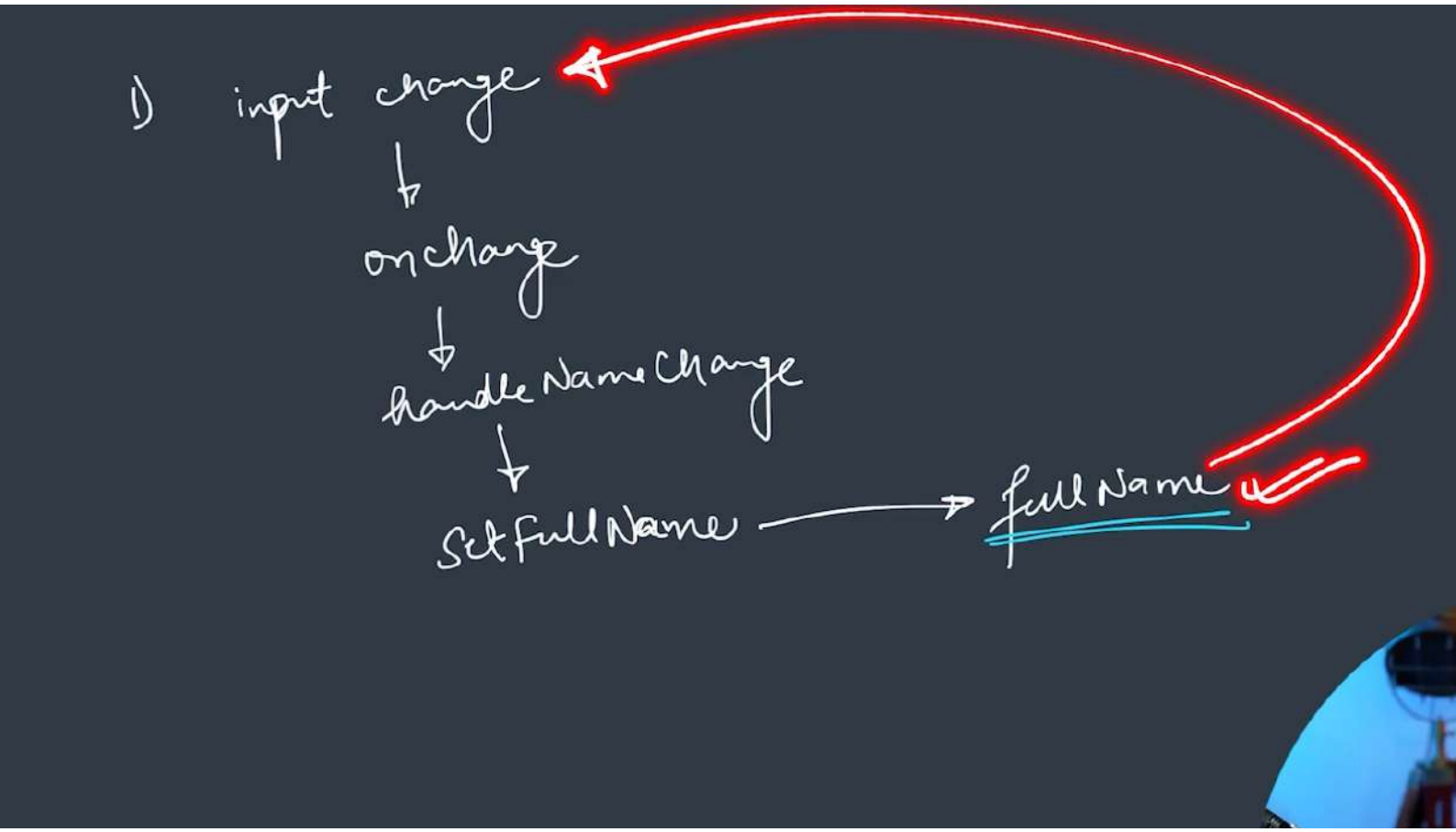
↓

handleNameChange

↓

SetFullName ⟶ fullName


1) input change
onChange
handleNameChange
SetFullName
fullName


1) input change
onChange
handleNameChange
SetFullName
fullName

Form Labels in React

Form Labels in React

```
return (
  <form>
    <label htmlFor="username">Full Name</label>
    <input
      placeholder="enter full name"
      type="text"
      value={fullName}
      onChange={handleNameChange}
      id="username"
    />
    <button>Submit</button>
  </form>
);
}
```

App.jsx U Form.jsx U

Learning > FRONTEND > Apna College > React > React Lecture-6 > Forms_In_react > src > Form.jsx > Form

```jsx
import {useState} from "react"
Complexity is 17 You must be kidding
export default function Form(){
    let[fullname,setFullname]=useState("")
    let[username,setUsername]=useState("")

    let handleNameChange= (event) => {
        // console.log(event.target);
        // console.log(event.target.value);
        setFullname(event.target.value);
    };
    let handleUserName= (event) => {
        // console.log(event.target);
        // console.log(event.target.value);
        setUsername(event.target.value);
    };
    return(
        <>
    <form>
        <label htmlFor="Full_Name">Full Name  </label>
        <input type="text" placeholder="Enter Full Name" value={fullname} onChange={handleNameChange} id="Full_Name"/>
        <br /><br />
        <button>Submit</button>
    </form>
    <br />
    <form>
        <label htmlFor="User_Name">User Name  </label>
        <input type="text" placeholder="Enter UserName" value={username} onChange={handleUserName} id="User_Name"/>
        <br /><br />
        <button>Submit</button>
    </form>
    </>
    )
```


Full Name
Enter Full Name
Submit
User Name
Enter UserName
Submit

Handling Multiple Inputs

We create a common handleInputChange( ) for all elements

event.target.name	changed field
event.target.value	new value of the field

Handling Multiple Inputs

We create a common handleInputChange( ) for all elements

event.target.name changed field

event.target.value new value of the field

objects

{ fullName: "" "" username: "" "" password: "" "" }

[variable]

computed property name

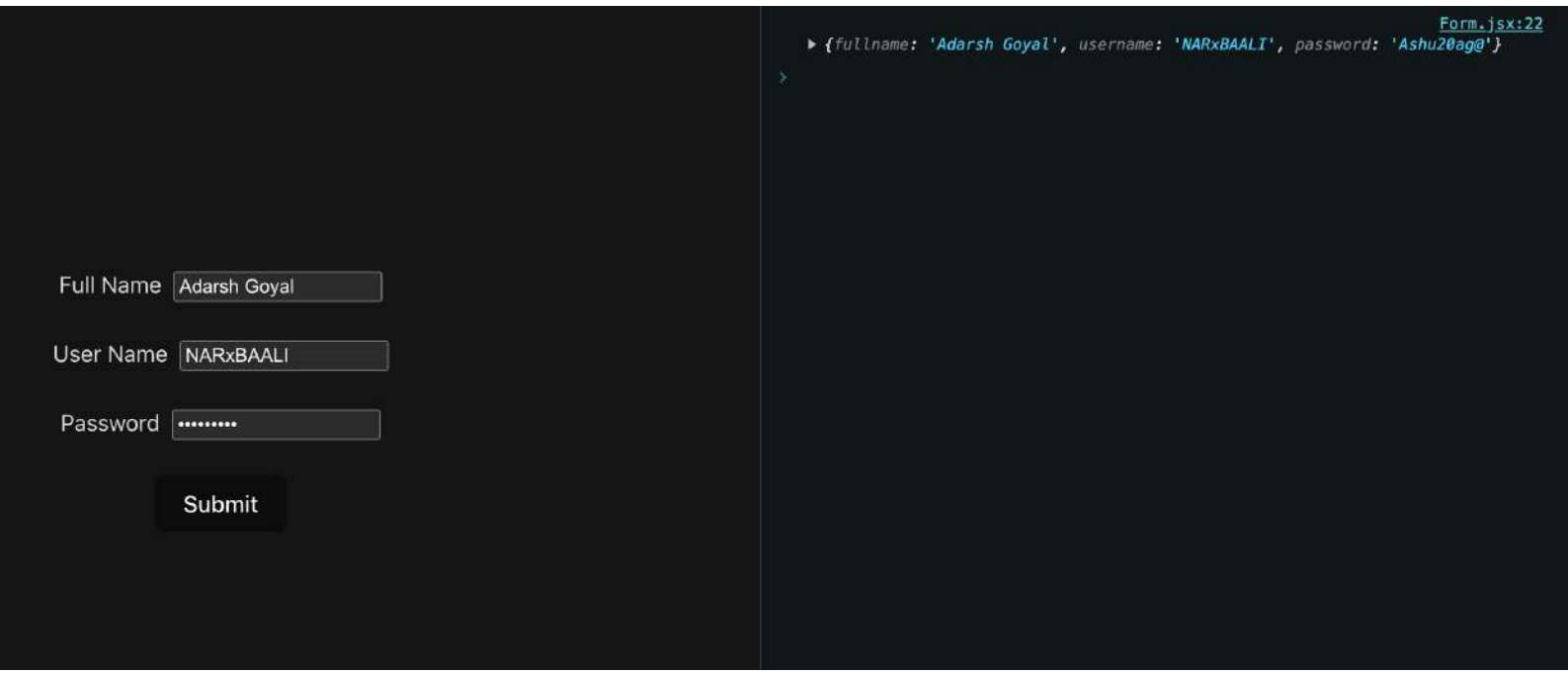
App.jsx U Form.jsx U

Learning > FRONTEND > Apna College > React > React Lecture-6 > Forms_In_react > src > Form.jsx > Form > handleSubmit

```jsx
import {useState} from "react"
Complexity is 20 You must be kidding
export default function Form(){
    let [formdata,setFormdata]=useState({
        fullname:"",
        username:"",
        password:""
    })

    Complexity is 3 Everything is cool!
    let handleformChange=(event)=>{
        // let fieldName=event.target.name;
        // let fieldValue=event.target.value;
        // setFormdata((curData)=>{
        //     return{...curData,[fieldName]:fieldValue};
        // })
        setFormdata((curData)=>{
            return{...curData,[event.target.name]:event.target.value};
        })
    }

    let handleSubmit=(event)=>{
        event.preventDefault()
        console.log(formdata)
        setFormdata({
            fullname:"",
            username:"",
            password:""
        })
    }
    return(
        <>
    <form onSubmit={handleSubmit}>
        <label htmlFor="fullname">Full Name  </label>
        <input type="text" placeholder="Enter Full Name" value={formdata.fullname} onChange={handleformChange} id="fullname" name="fullname"/>
    </form>
    <br />
    <form onSubmit={handleSubmit}>
        <label htmlFor="username">User Name  </label>
        <input type="text" placeholder="Enter UserName" value={formdata.username} onChange={handleformChange} id="username" name="username"/>
    </form>
    <br />
    <form onSubmit={handleSubmit}>
        <label htmlFor="password">Password  </label>
        <input type="password" placeholder="Enter password" value={formdata.password} onChange={handleformChange} id="password" name="password"/>
        <br /><br />
        <button>Submit</button>
    </form>
    </>
    )
}
```


Full Name
Adarsh Goyal
User Name
NARxBAALI
Password
Submit
Form.jsx:22
{fullname: 'Adarsh Goyal', username: 'NARxBAALI', password: 'Ashu20ag@'}

Comments Form

username

remark

rating 1 - 5

App.jsx U Commentform.jsx U Form.jsx U

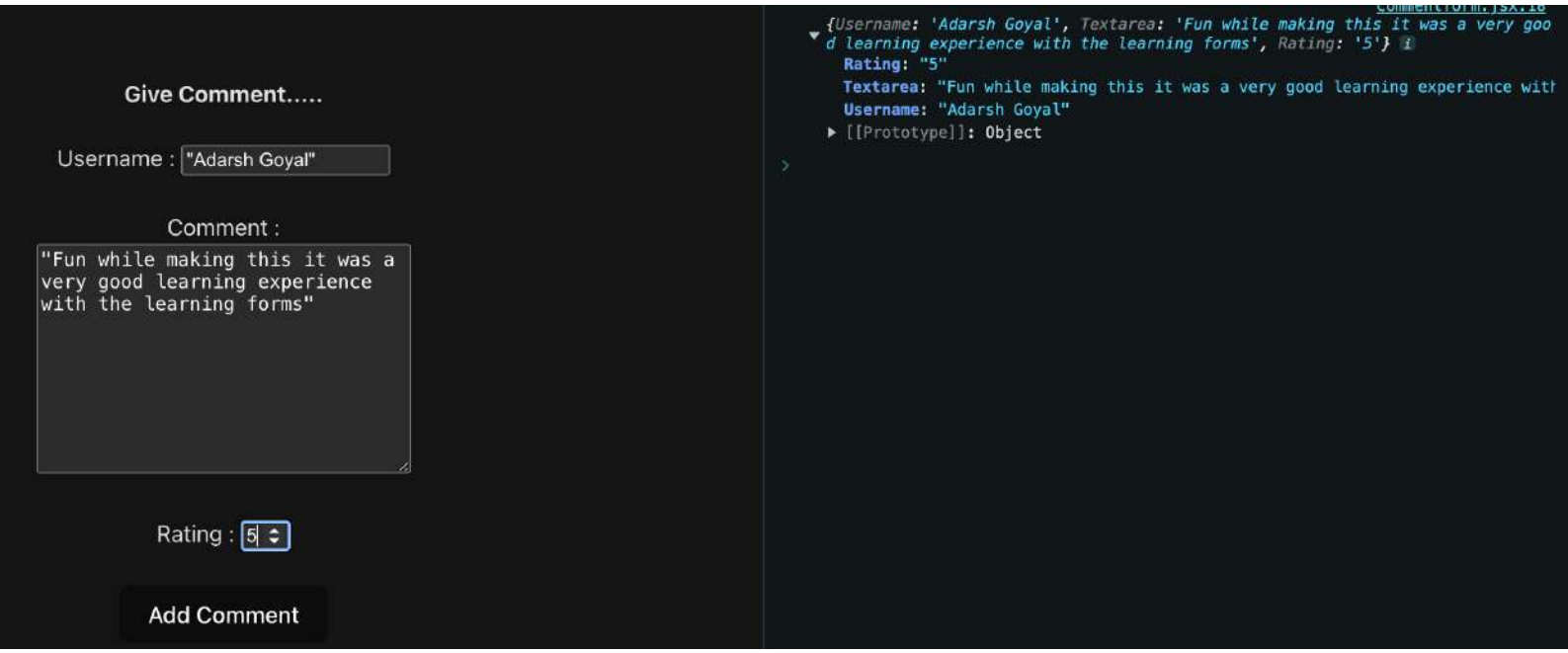
Learning > FRONTEND > Apna College > React > React Lecture-6 > Forms_In_react > src > Commentform.jsx > Commentform

```jsx
import { useState } from "react";

Complexity is 23 You must be kidding
export default function Commentform() {
  let [formdata, setFormdata] = useState({
    Username: "",
    Textarea: "",
    Rating: "",
  });

  Complexity is 3 Everything is cool!
  let handleformChange = (event) => {
    setFormdata((curdata)=>{
        return{...curdata,[event.target.name]:event.target.value};
    });
  };

  let handleSubmit = (event) => {
    event.preventDefault();
    console.log(formdata);
    setFormdata({
      Username: "",
      Textarea: "",
      Rating: "",
    });
  };
  return (
    <div>
      <h4>Give Comment.....</h4>
      <form onSubmit={handleSubmit}>
        <label htmlFor="Username">Username : </label>
        <input type="text" placeholder="Username" name="Username" value={formdata.Username} onChange={handleformChange}/>
        <br />
        <br />
        <label htmlFor="Comment">Comment : </label>
        <br />
        <textarea name="Textarea" id="Comment" cols="30" rows="10" placeholder="Enter Your Comment" value={formdata.Textarea} onChange={handleformChange}>
        </textarea>
        <br />
        <br />
        <label htmlFor="Rating">Rating : </label>
        <input type="number" placeholder="" min={1} max={5} name="Rating" value={formdata.Rating} onChange={handleformChange}/>
        <br />
        <br />
        <button>Add Comment</button>
      </form>
    </div>
  );
}
```


Give Comment.....
Username : "Adarsh Goyal"
Comment :
"Fun while making this it was a very good learning experience with the learning forms"
Rating : 5
Add Comment
CommentForm.jsx:16
{Username: 'Adarsh Goyal', Textarea: 'Fun while making this it was a very good learning experience with the learning forms', Rating: '5'}
Rating: "5"
Textarea: "Fun while making this it was a very good learning experience with
Username: "Adarsh Goyal"
[[Prototype]]: Object

useEffect( )

The Effect Hook lets you perform side effects in function components

Data fetching, setting up a subscription, and manually changing the DOM in React components are all examples of side effects.

useEffect( )

The Effect Hook lets you perform side effects in function components

Data fetching, setting up a subscription, and manually changing the DOM in React components are all examples of side effects.

render

render

useEffect( )

useEffect(func)

```
import { useState, useEffect } from "react";
export default function Counter() {
  let [count, setCount] = useState(0);
  const incCount = () => {
    setCount((currCount) => currCount + 1);
  };

  useEffect(function sideEffect() {
    console.log("this is the side effect");
  });

  return (
    <div>
      <h1>Count = {count}</h1>
      <button onClick={incCount}>+1</button>
    </div>
  );
```

App.jsx U Counter.jsx U

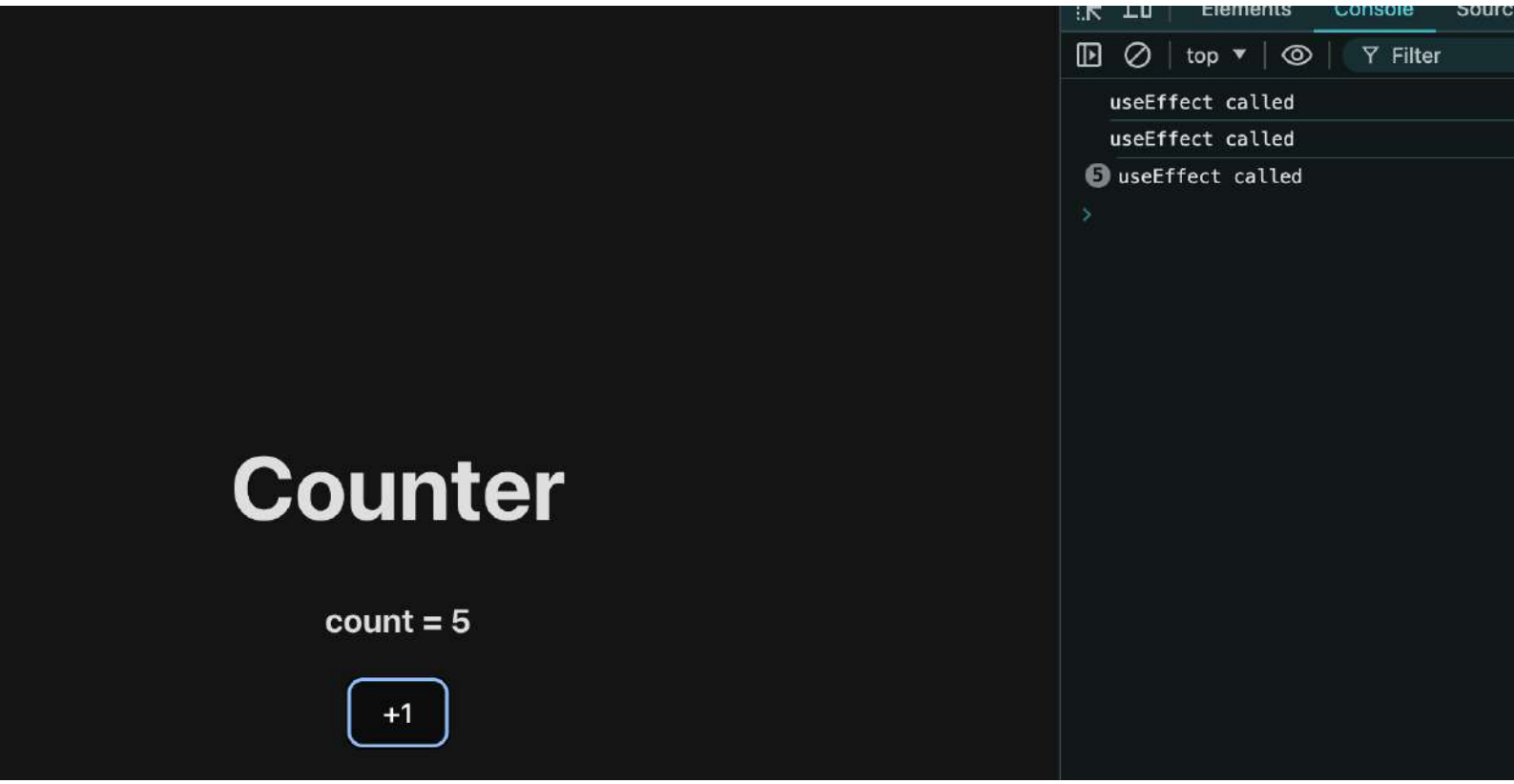
Learning > FRONTEND > Apna College > React > React Lecture-6 > Fo

```jsx
import { useState,useEffect } from "react";

Complexity is 9 It's time to do something...
export default function Counter(){
    let [count,setCount]=useState(0);

    let incCount= ()=>{
        setCount(currCount=>currCount+1)
    }

    useEffect(function() {
        console.log("useEffect called");
    });
    return(
    <div>
        <h1>Counter</h1>
        <h3>count = {count}</h3>
        <button onClick={incCount}>+1</button>
    </div>
    );
}
```


Elements
Console
top
Filter
useEffect called
useEffect called
5
useEffect called
Counter
count = 5
+1

useEffect( )

useEffect(setup, dependencies?)

```
useEffect(
  function sideEffect() {
    console.log("this is the side effect");
  },
  [countx]
);
```

App.jsx U Counter.jsx U

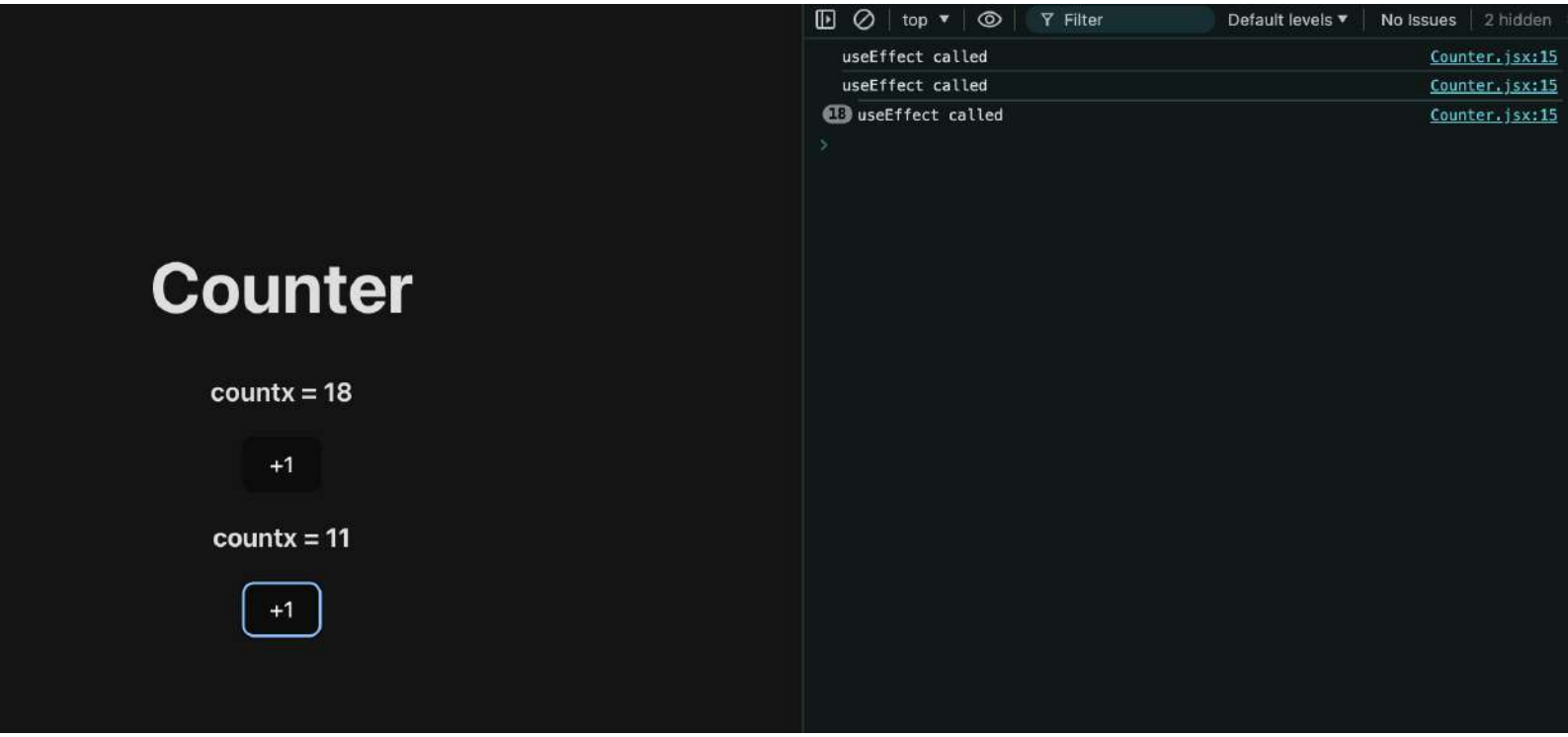
Learning > FRONTEND > Apna College > React > React Lecture-6 > Form

```jsx
import { useState,useEffect } from "react";

Complexity is 13 You must be kidding
export default function Counter(){
    let [countx,setCountx]=useState(0);
    let [county,setCounty]=useState(0);

    let incCountx= ()=>{
        setCountx(currCount=>currCount+1)
    }
    let incCounty= ()=>{
        setCounty(currCount=>currCount+1)
    }

    useEffect(function() {
        console.log("useEffect called");
    },[countx]);
    return(
    <div>
        <h1>Counter</h1>
        <h3>countx = {countx}</h3>
        <button onClick={incCountx}>+1</button>
        <h3>countx = {county}</h3>
        <button onClick={incCounty}>+1</button>
    </div>
    );
}
```


Counter
countx = 18
+1
countx = 11
+1
top
Filter
Default levels
No Issues
2 hidden
useEffect called
Counter.jsx:15
useEffect called
Counter.jsx:15
18
useEffect called
Counter.jsx:15

useEffect( )

useEffect(setup, dependencies?)

state vars

```
useEffect(
  function sideEffect() {
    console.log("this is the side effect");
  },
  [countx]
);
```

1) ~ [s1, s2, s3]

2) [ ] 1st render ✓ re-render ✗

App.jsx U × Counter.jsx U ×

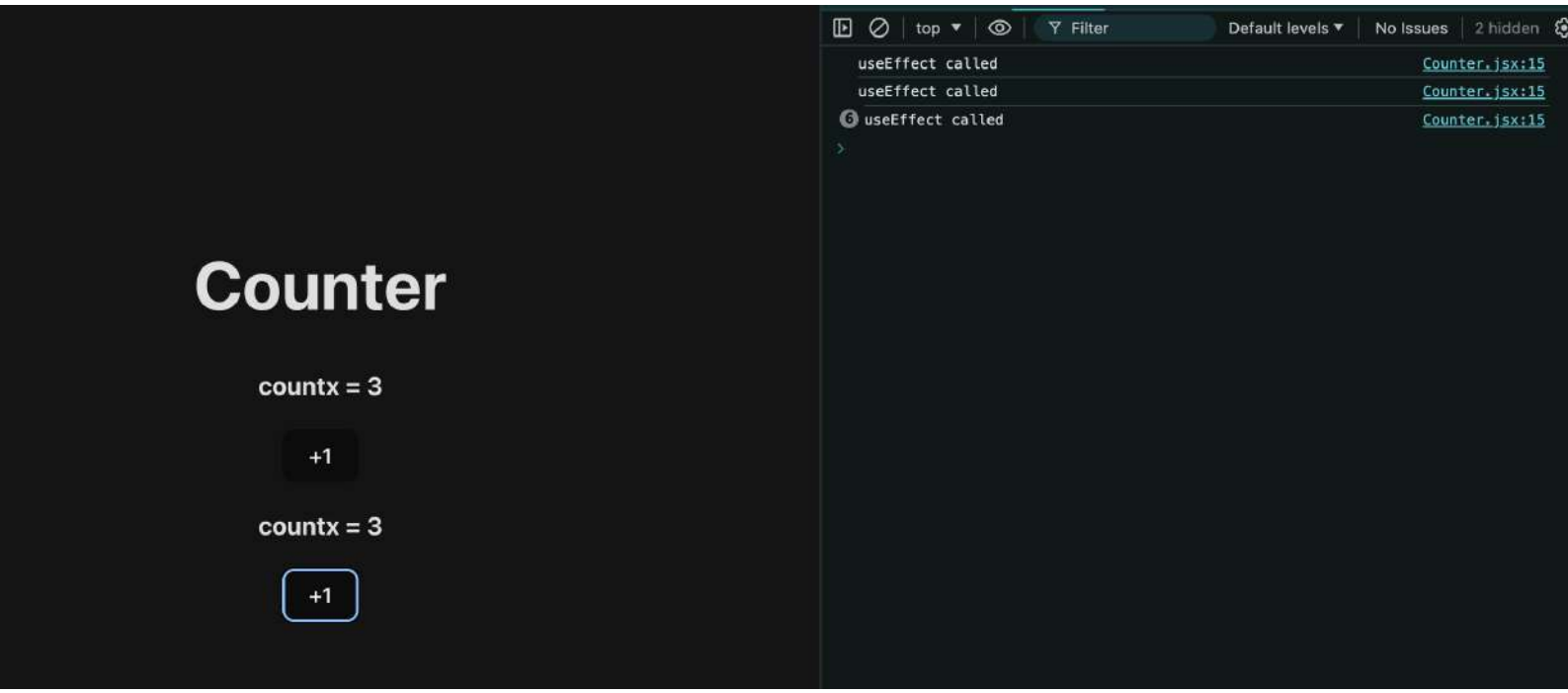
Learning > FRONTEND > Apna College > React > React Lecture-6 > Forms_In_re

```jsx
import { useState,useEffect } from "react";

Complexity is 13 You must be kidding
export default function Counter(){
    let [countx,setCountx]=useState(0);
    let [county,setCounty]=useState(0);

    let incCountx= ()=>{
        setCountx(currCount=>currCount+1)
    }
    let incCounty= ()=>{
        setCounty(currCount=>currCount+1)
    }

    useEffect(function() {
        console.log("useEffect called");
    },[countx,county]);
    return(
    <div>
        <h1>Counter</h1>
        <h3>countx = {countx}</h3>
        <button onClick={incCountx}>+1</button>
        <h3>countx = {county}</h3>
        <button onClick={incCounty}>+1</button>
    </div>
    );
}
```


Counter
countx = 3
+1
countx = 3
+1
top
Filter
Default levels
No Issues
2 hidden
useEffect called
Counter.jsx:15
useEffect called
Counter.jsx:15
6
useEffect called
Counter.jsx:15

App.jsx U Counter.jsx U

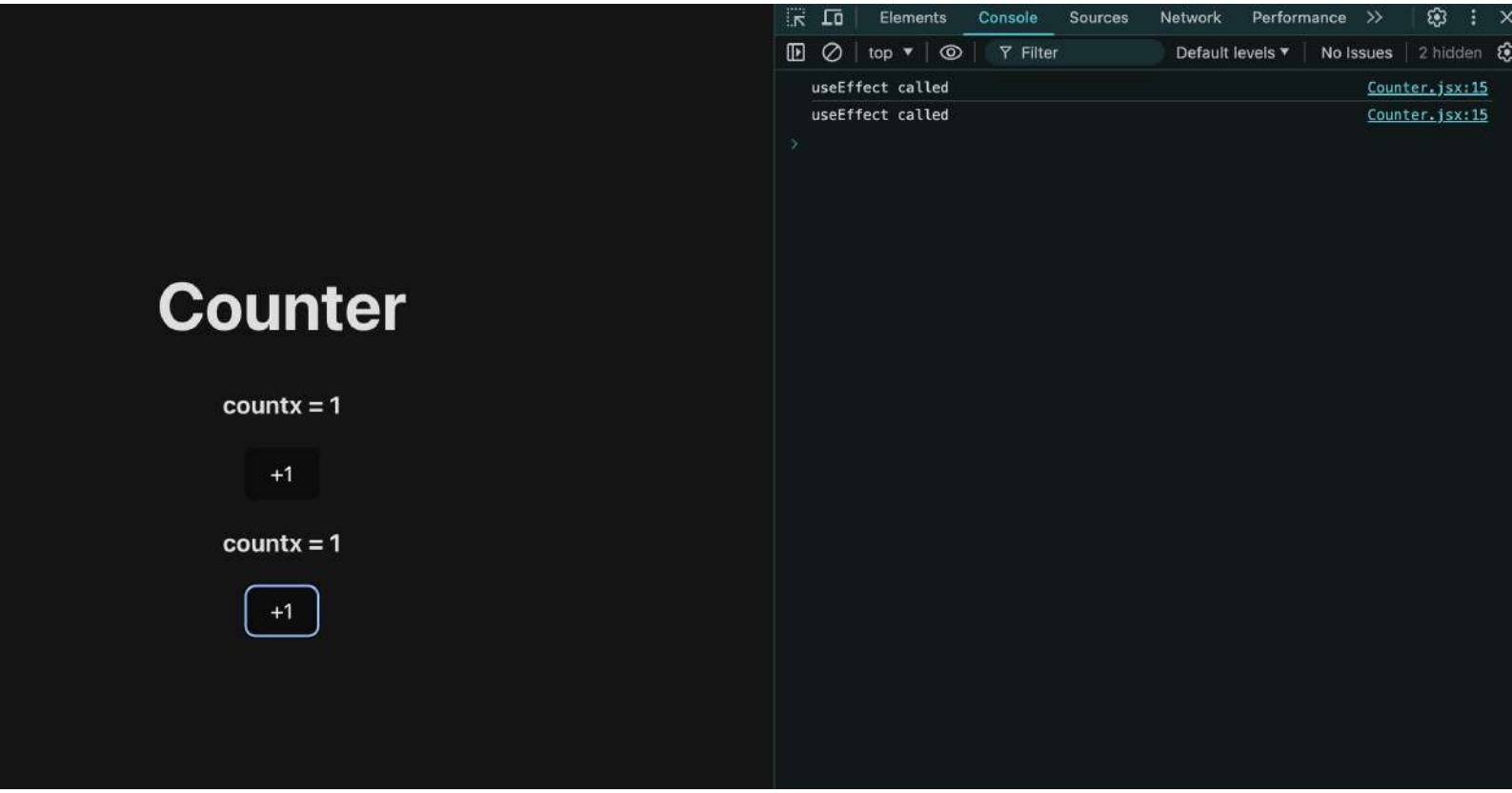
Learning > FRONTEND > Apna College > React > React Lecture-6 > Forms_In_react > src

```jsx
import { useState,useEffect } from "react";

Complexity is 13 You must be kidding
export default function Counter(){
    let [countx,setCountx]=useState(0);
    let [county,setCounty]=useState(0);

    let incCountx= ()=>{
        setCountx(currCount=>currCount+1)
    }
    let incCounty= ()=>{
        setCounty(currCount=>currCount+1)
    }

    useEffect(function() {
        console.log("useEffect called");
    },[]);
    return(
    <div>
        <h1>Counter</h1>
        <h3>countx = {countx}</h3>
        <button onClick={incCountx}>+1</button>
        <h3>countx = {county}</h3>
        <button onClick={incCounty}>+1</button>
    </div>
    );
}
```


Counter
countx = 1
+1
countx = 1
+1
Elements
Console
Sources
Network
Performance
top
Filter
Default levels
No Issues
2 hidden
useEffect called
Counter.jsx:15
useEffect called
Counter.jsx:15

Use Cases

API Calls - Asynchronous Operations

Use Cases

API Calls - Asynchronous Operations

App.jsx M joker.jsx U

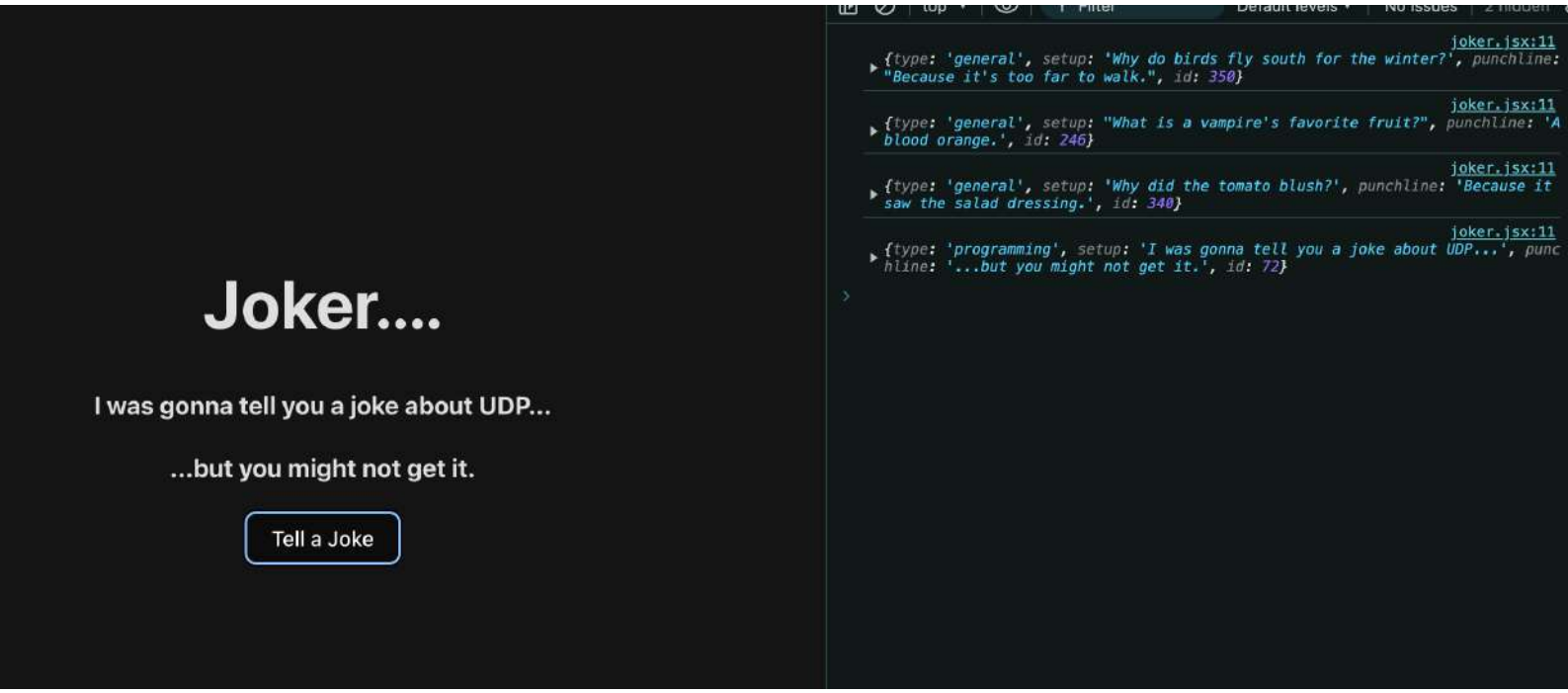
Learning > FRONTEND > Apna College > React > React Lecture-6 > Forms_In_react > src > joke

```jsx
import { useState } from "react";

Complexity is 8 It's time to do something...
export default function joker(){
    const url ="https://official-joke-api.appspot.com/random_joke";

    let [joke,setJoke]=useState({});

    const getJoke = async ()=>{
        let res=await fetch(url);
        let jsonresp = await res.json();
        console.log(jsonresp);
        setJoke({setup:jsonresp.setup,punchline:jsonresp.punchline})
    }
    return(
        <div>
            <h1>Joker....</h1>
            <h3>{joke.setup}</h3>
            <h3>{joke.punchline}</h3>
            <button onClick={getJoke}>Tell a Joke</button>
        </div>
    )
}
```


Joker....
I was gonna tell you a joke about UDP...
...but you might not get it.
Tell a Joke
top
Filter
Default levels
No issues
joker.jsx:11
{type: 'general', setup: 'Why do birds fly south for the winter?', punchline: "Because it's too far to walk.", id: 350}
joker.jsx:11
{type: 'general', setup: "What is a vampire's favorite fruit?", punchline: 'A blood orange.', id: 246}
joker.jsx:11
{type: 'general', setup: 'Why did the tomato blush?', punchline: 'Because it saw the salad dressing.', id: 340}
joker.jsx:11
{type: 'programming', setup: 'I was gonna tell you a joke about UDP...', punchline: '...but you might not get it.', id: 72}

App.jsx M joker.jsx U

Learning > FRONTEND > Apna College > React > React Lecture-6 > Forms_In_react > src > joker.jsx > joker

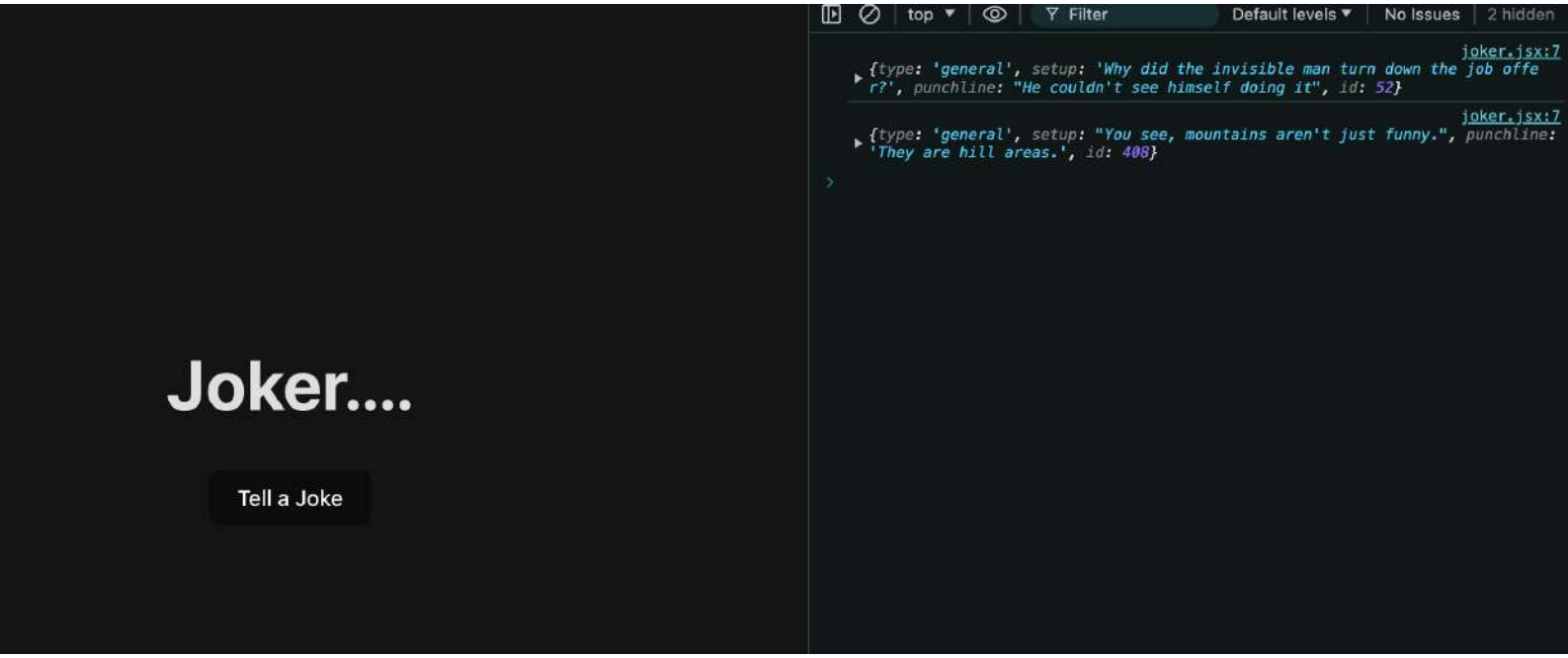
```jsx
import { useState } from "react";

Complexity is 8 It's time to do something...
export default function joker(){
    const getJoke = async ()=>{
        let res=await fetch(url);
        let jsonresp = await res.json();
        console.log(jsonresp);
        // setJoke({setup:jsonresp.setup,punchline:jsonresp.punchline})
    }

    const url ="https://official-joke-api.appspot.com/random_joke";

    let [joke,setJoke]=useState(getJoke);


    return(
        <div>
            <h1>Joker....</h1>
            <h3>{joke.setup}</h3>
            <h3>{joke.punchline}</h3>
            <button onClick={getJoke}>Tell a Joke</button>
        </div>
    )
}
```


Joker....
Tell a Joke
top
Filter
Default levels
No Issues
2 hidden
joker.jsx:7
{type: 'general', setup: 'Why did the invisible man turn down the job offer?', punchline: "He couldn't see himself doing it", id: 52}
joker.jsx:7
{type: 'general', setup: "You see, mountains aren't just funny.", punchline: 'They are hill areas.', id: 408}

App.jsx M joker.jsx U

Learning > FRONTEND > Apna College > React > React Lecture-6 > Forms_In_react > src > joker.jsx > jok

```jsx
import { useEffect, useState } from "react";

Complexity is 10 It's time to do something...
export default function joker(){
    const url ="https://official-joke-api.appspot.com/random_joke";
    let [joke,setJoke]=useState({});
    const getJoke = async ()=>{
        let res=await fetch(url);
        let jsonresp = await res.json();
        setJoke({setup:jsonresp.setup,punchline:jsonresp.punchline})
    }
    useEffect(()=>{
        async function refreshjoke(){
        let res=await fetch(url);
        let jsonresp = await res.json();
        setJoke({setup:jsonresp.setup,punchline:jsonresp.punchline})
        }
        refreshjoke();
    },[]);
    return(
        <div>
            <h1>Joker....</h1>
            <h3>{joke.setup}</h3>
            <h3>{joke.punchline}</h3>
            <button onClick={getJoke}>Tell a Joke</button>
        </div>
    )
}
```

Joker....

What do you get when you cross a bee and a sheep?

A bah-humbug.

Tell a Joke