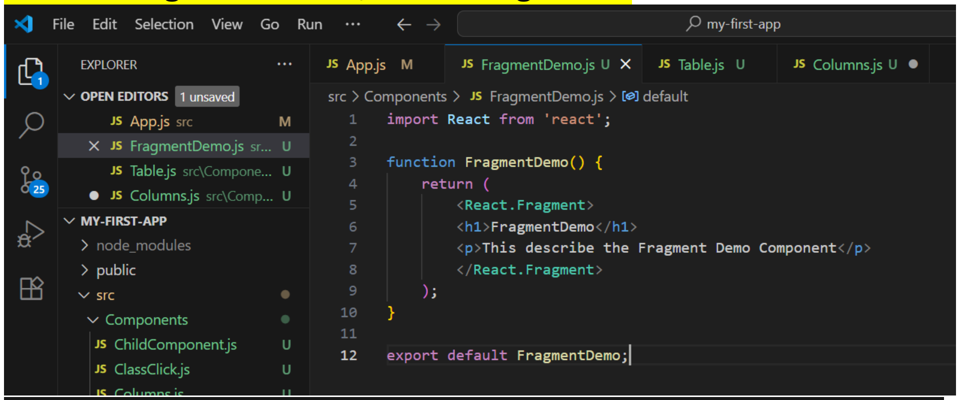
## Fragments:(25)

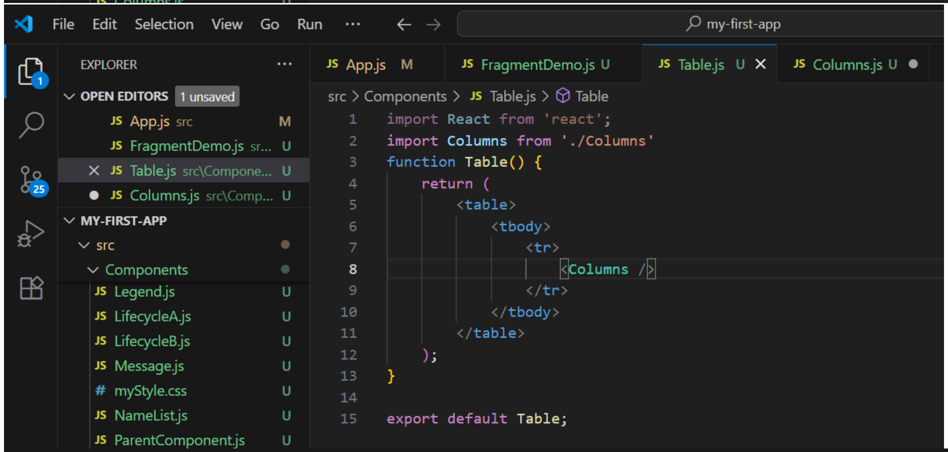
# <React.Fragment>…….</React.Fragment>

#<>……</> => same as <React.Fragment>…….</React.Fragment> But can not pass in key attributes.

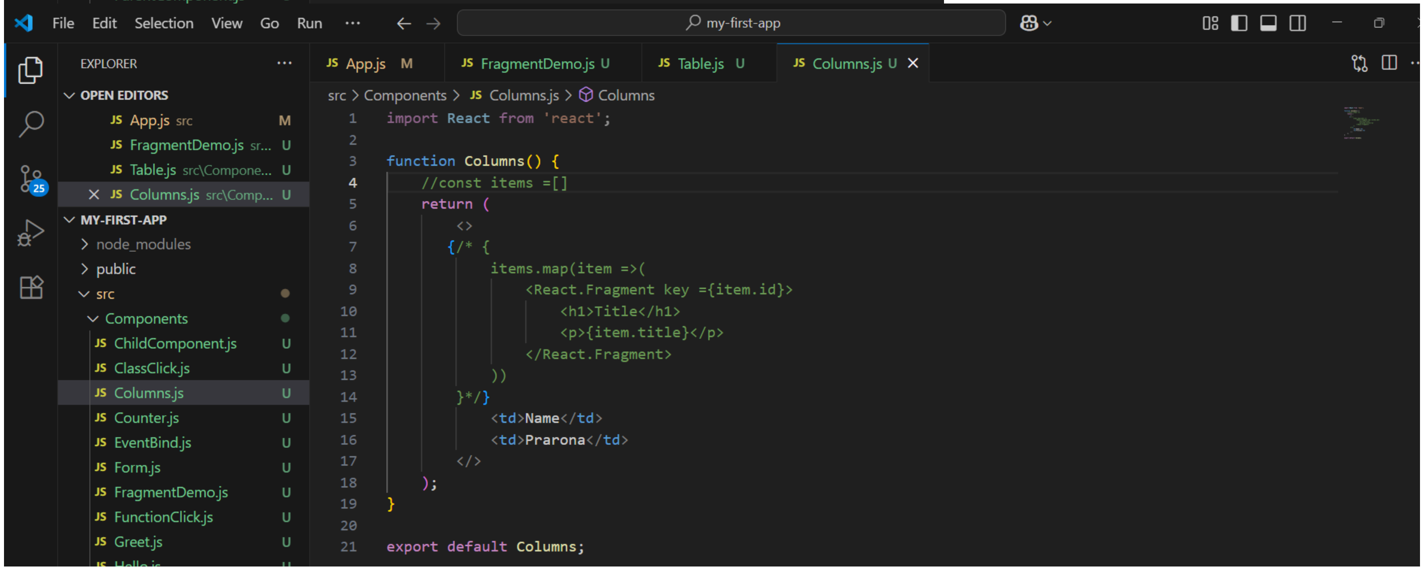
#key ={item.id} =>is the only attribute that call possible pass in <React.Fragment>…….</React.Fragment>



```
import React from 'react';

function FragmentDemo() {
    return (
        <React.Fragment>
        <h1>FragmentDemo</h1>
        <p>This describe the Fragment Demo Component</p>
        </React.Fragment>
    );
}

export default FragmentDemo;
```



```
import React from 'react';
import Columns from './Columns'
function Table() {
    return (
        <table>
            <tbody>
                <tr>
                    <Columns />
                </tr>
            </tbody>
        </table>
    );
}

export default Table;
```



```
import React from 'react';

function Columns() {
    //const items =[]
    return (
        <>
       {/* {
            items.map(item =>(
                <React.Fragment key ={item.id}>
                    <h1>Title</h1>
                    <p>{item.title}</p>
                </React.Fragment>
            ))
        }*/}
            <td>Name</td>
            <td>Prarona</td>
        </>
    );
}

export default Columns;
```

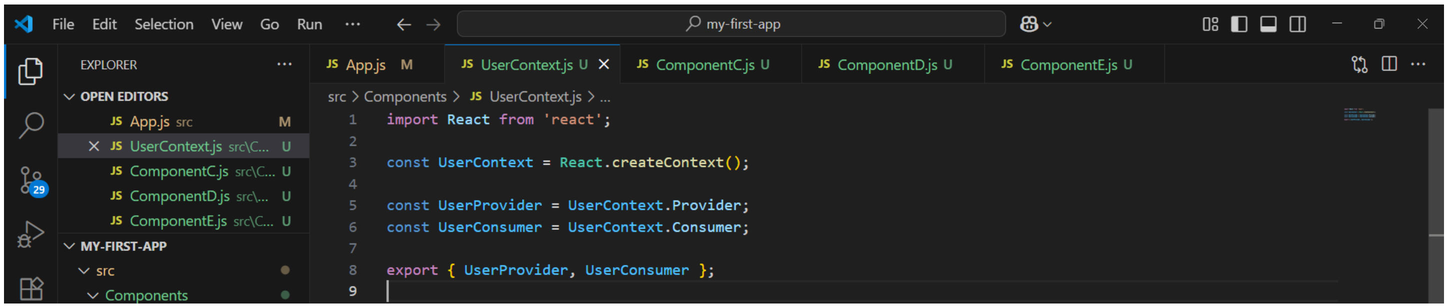
## Context:(38):

**Context provides a way to pass data through the component tree without having to pass props down manually at every level.**

# Context-2:(39):

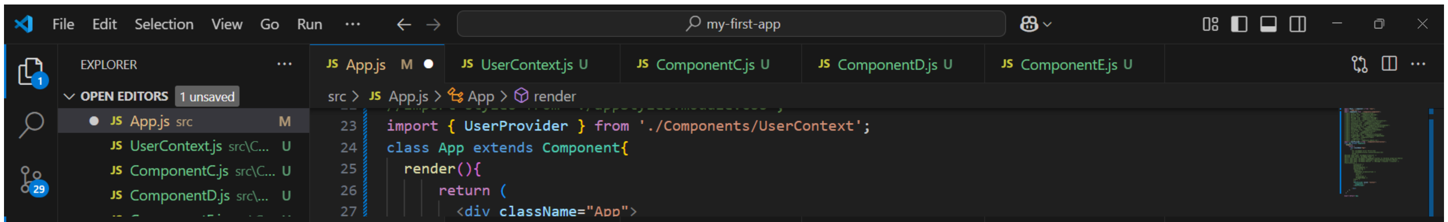
## Steps:

## 1.Create the Context.

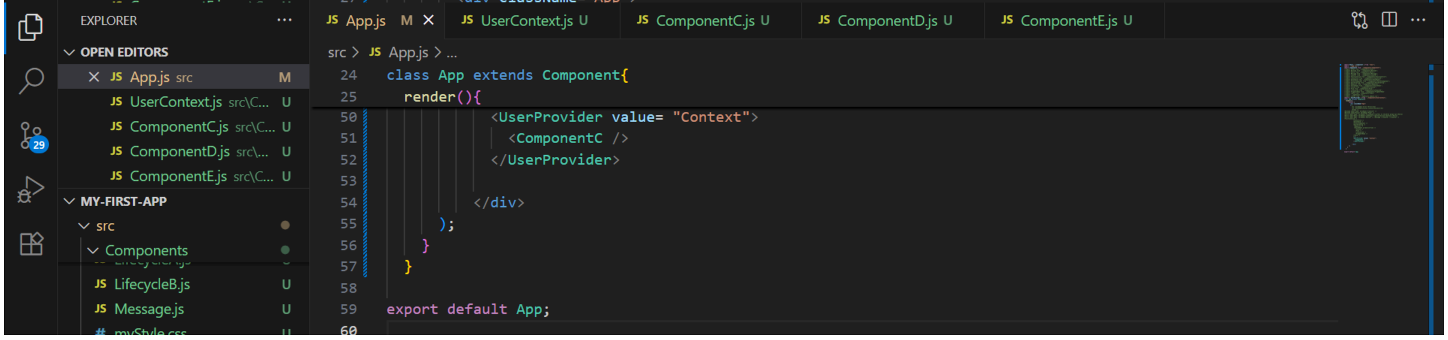


```
import React from 'react';

const UserContext = React.createContext();

const UserProvider = UserContext.Provider;
const UserConsumer = UserContext.Consumer;

export { UserProvider, UserConsumer };
```

## 2.Provide a context value.

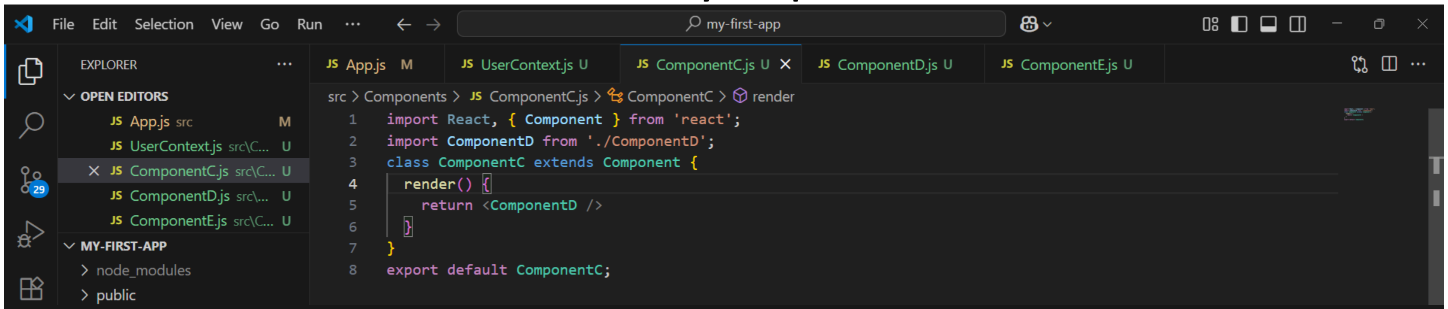


```
import { UserProvider } from './Components/UserContext';
class App extends Component{
  render(){
    return (
      <div className="App">
```

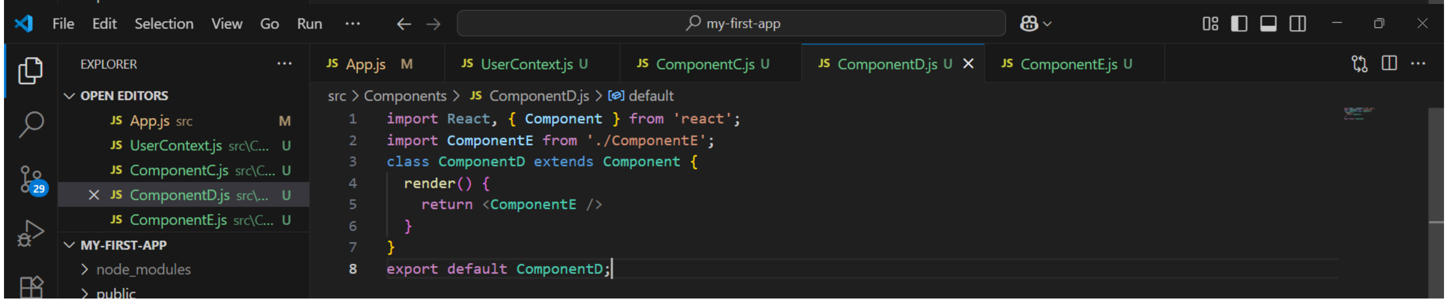


```
class App extends Component{
  render(){
          <UserProvider value= "Context">
            <ComponentC />
          </UserProvider>

        </div>
      );
    }
  }

export default App;
```

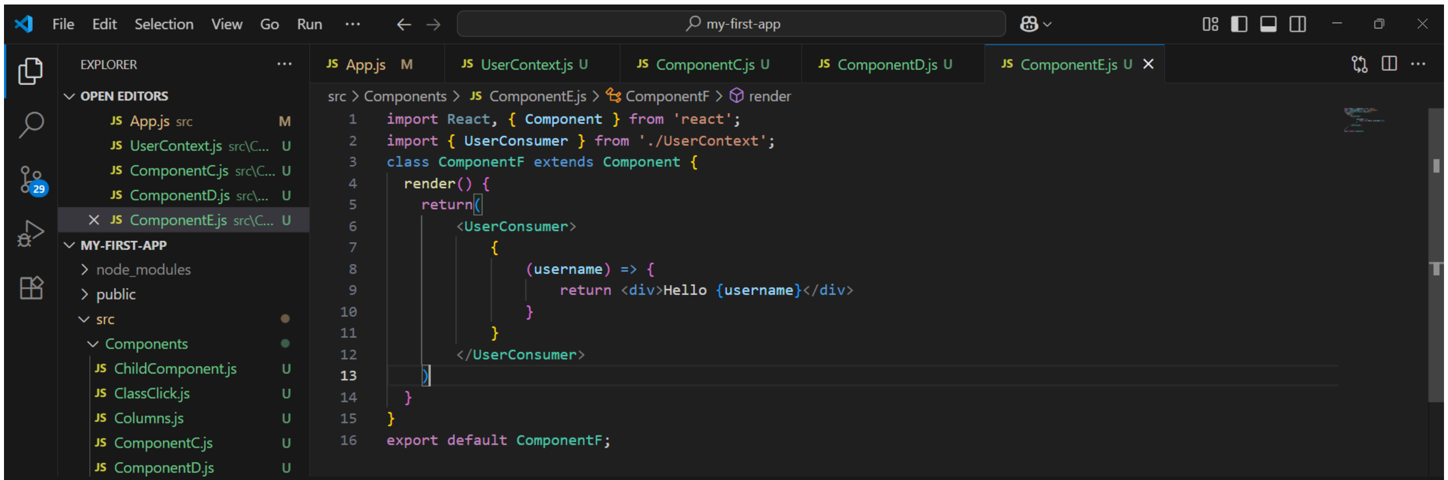
## 3.Consume the Context value in the necessary component.



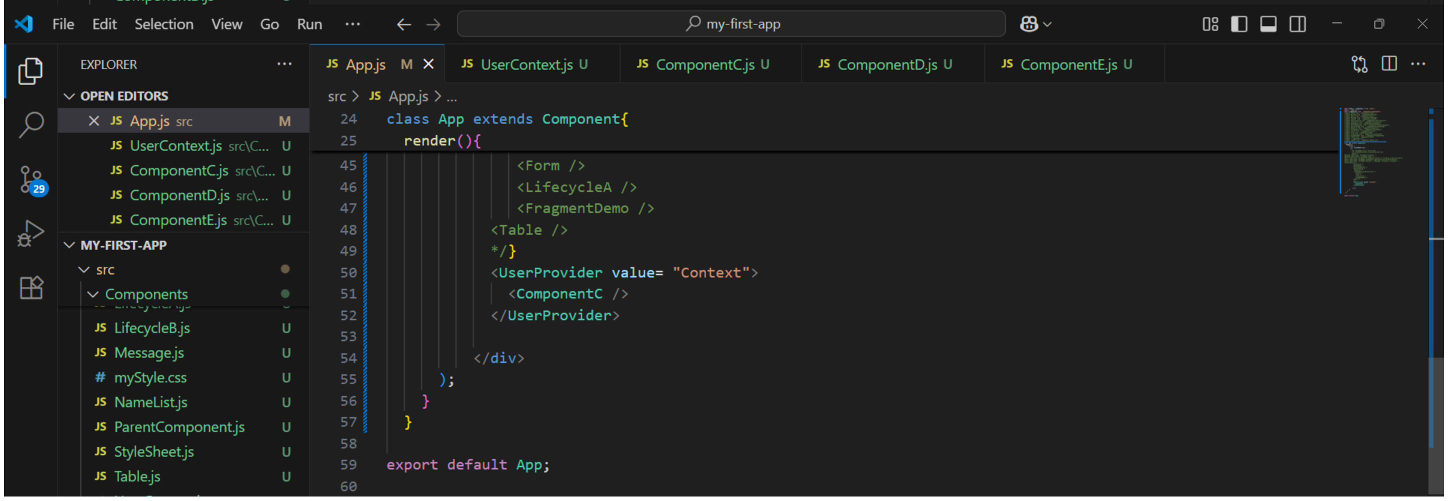
```
import React, { Component } from 'react';
import ComponentD from './ComponentD';
class ComponentC extends Component {
  render() {
    return <ComponentD />
  }
}
export default ComponentC;
```



```
import React, { Component } from 'react';
import ComponentE from './ComponentE';
class ComponentD extends Component {
  render() {
    return <ComponentE />
  }
}
export default ComponentD;
```



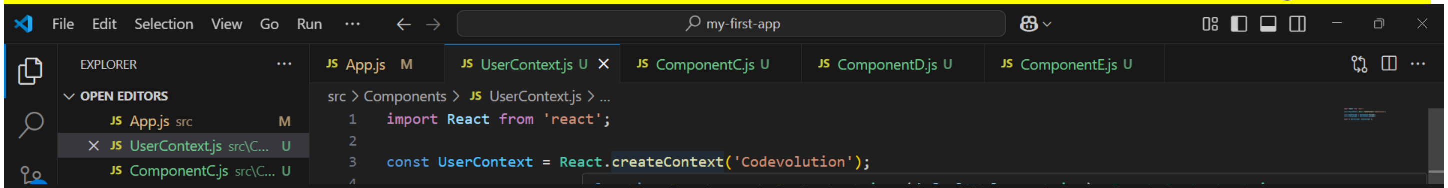
```
import React, { Component } from 'react';
import { UserConsumer } from './UserContext';
class ComponentF extends Component {
  render() {
    return(
        <UserConsumer>
            {
                (username) => {
                    return <div>Hello {username}</div>
                }
            }
        </UserConsumer>
    )
  }
}
export default ComponentF;
```



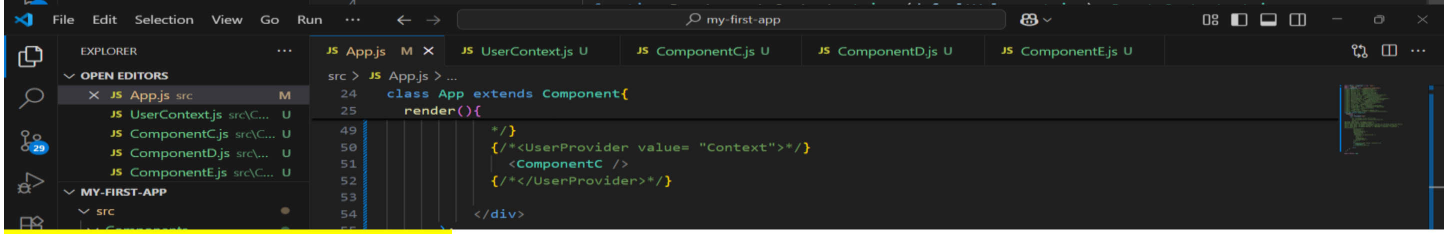
```
class App extends Component{
  render(){
              <Form />
              <LifecycleA />
              <FragmentDemo />
           <Table />
           */}
           <UserProvider value= "Context">
             <ComponentC />
           </UserProvider>

         </div>
     );
   }
 }

export default App;
```

# Context-3:(40):

## We can set a default context value while we create context as argument:

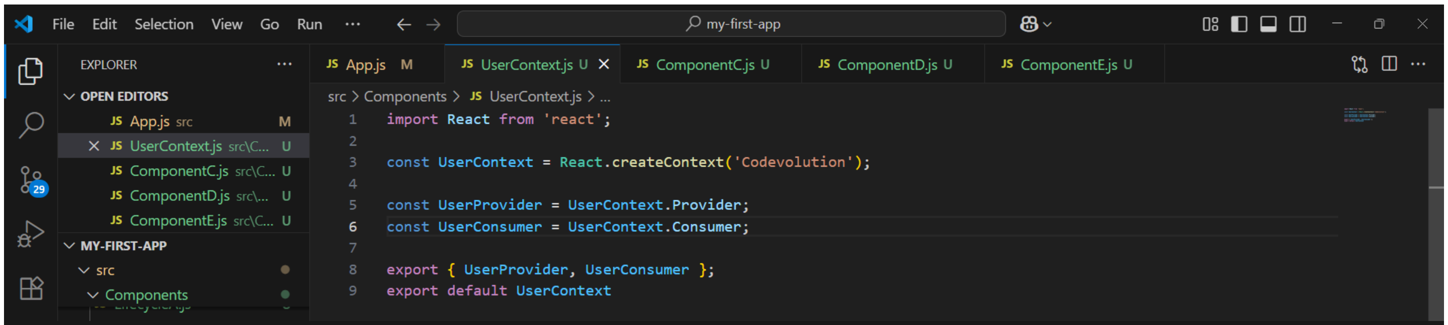


```
import React from 'react';

const UserContext = React.createContext('Codevolution');
```

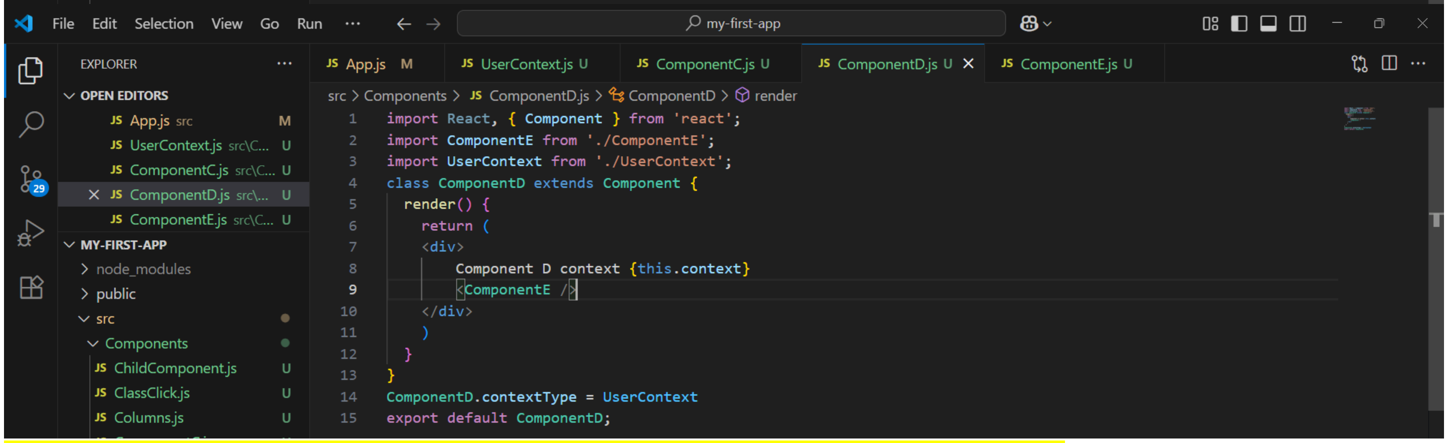


```
class App extends Component{
  render(){
           */}
           {/*<UserProvider value= "Context">*/}
             <ComponentC />
           {/*</UserProvider>*/}

         </div>
```

## ContextType Property:



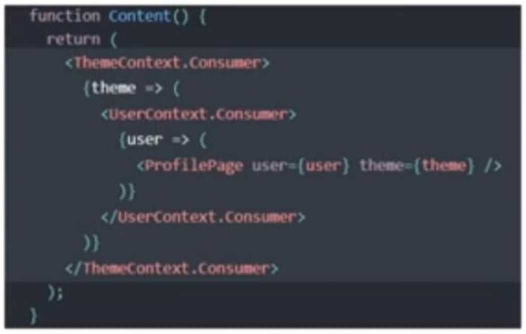
```
import React from 'react';

const UserContext = React.createContext('Codevolution');

const UserProvider = UserContext.Provider;
const UserConsumer = UserContext.Consumer;

export { UserProvider, UserConsumer };
export default UserContext
```



```
import React, { Component } from 'react';
import ComponentE from './ComponentE';
import UserContext from './UserContext';
class ComponentD extends Component {
  render() {
    return (
      <div>
        Component D context {this.context}
        <ComponentE />
      </div>
    )
  }
}
ComponentD.contextType = UserContext
export default ComponentD;
```

Can also be **{/* can also be static contextType = UserContext*/}** in ComponentD. But it only works with class component, only can subscribe in one context.

## Consuming Multiple Contexts



```
function Content() {
  return (
    <ThemeContext.Consumer>
      {theme => (
        <UserContext.Consumer>
          {user => (
            <ProfilePage user={user} theme={theme} />
          )}
        </UserContext.Consumer>
      )}
    </ThemeContext.Consumer>
  );
}
```