

Create a React Component Using React Features:

Start by setting up a new React project using create-react-app or your preferred setup. Within this project, create your React component. Make sure to import `useState` and `useEffect` from React, and utilize these features within your component.

Cmd : `npx create-react-app sample`

Run : `npm start`

App.js

```
import Content from './Components/Content';
import './App.css';
import Header from './Components/Header';
function App() {
  return (
    <>
      <Header />
      <Content />
    </>
  );
}
export default App;
```

Header.js

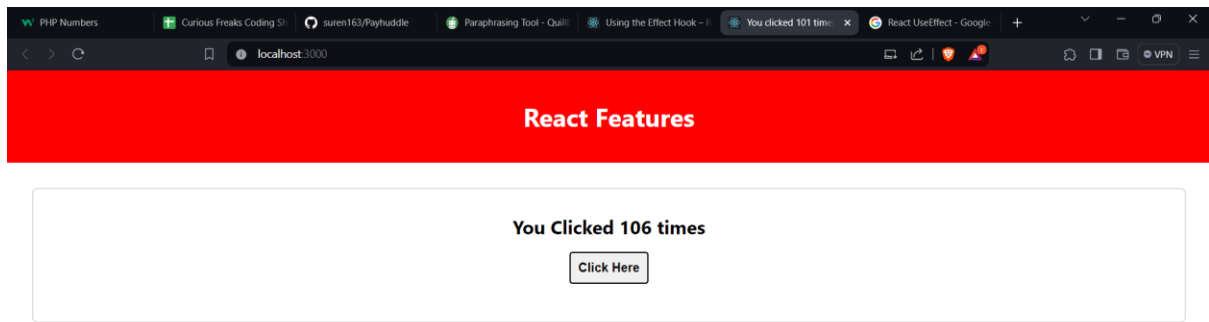
```
import React from 'react'
const Header = () => {
  return (
    <div className='header'>
```

```
        <h1>React Features</h1>
      </div>
    )
  }
export default Header

Content.js

import React, { useEffect, useState } from 'react'
const Content = () => {
  const[count, setCount] = useState(99);
  useEffect(()=>{
    console.log(`You Clicked ${count} times`);
  })
  const handleClick = () =>{
    setCount(count+1);
  }
  return (
    <div className='content'>
      <div className='sub-content'>
        You Clicked {count} times
      </div>
      <button className='btn'
onClick={()=>{handleClick()}}>Click Here</button>
    </div>
  )
}
export default Content
```

Output:



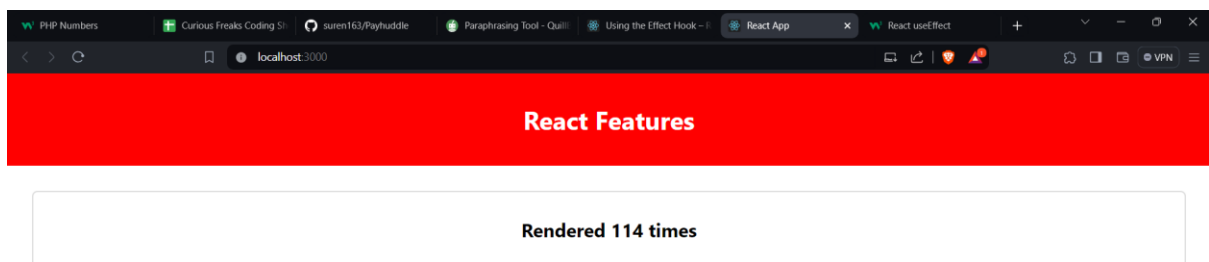
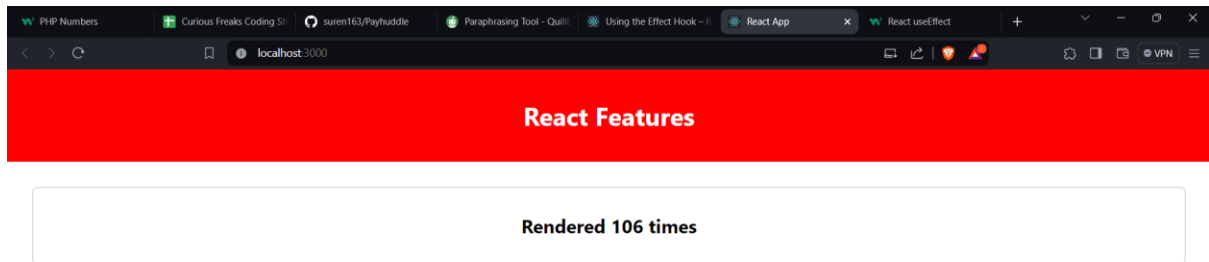
Automatic Time render: time = 1sec

Content.js

```
import React, { useEffect, useState } from 'react'
const Content = () => {
  const[count, setCount] = useState(99);
  useEffect(()=>{
    setTimeout(()=>{
      setCount(count+1);
    },1000)
  })
  return (
    <div className='content'>
      <div className='sub-content'>
        You Clicked {count} times
      </div>
    </div>
  )
}
```

```
</div>  
)  
}  
export default Content
```

Output:



To create a component and publish into the npmjs

Steps to publish...

- To create a component or function to upload into the npm.
- In cmd, `npm init` (for setting the config).
 - Package name
 - Version
 - Entry point
 - Git repository
 - Keywords
 - Author
 - License
- In cmd, **`npm login`** to login your npm account by using username and password.
- To create a Readme.md (Optional)
- In cmd, **`npm publish`**
- After publishing, To install the package on your project by using a necessary command.

- Package name
- Version
- Entry point
- Git repository
- Keywords
- Author
- License

- Then, In the index.js

Code:

```
const random_number_gen = (min=0, max=1000)=>{  
  ▪ return Math.round(Math.random()* (max-min) +  
    min);  
}  
  
export default random_number_gen;
```

- To create a link by using the command, **npm link** (Creating symbolic link).
- Inside the project, **npm link <package_name>** (Then, the package is automatically downloaded on the node module folder)
- Finally, import the package and use it.
 - import random_number_gen from 'random-number'
- To call the function by using the function name.
 - {random_number_gen()}

Example Program:

```
import React from 'react'  
import random_number_gen from 'random-number'  
const App = () => {  
  return (  
    <>
```

```
<div style={{backgroundColor:'red',
fontSize:'2rem',
textAlign:'center',padding:'1rem',fontWeight:'bold'}}>

  Payhuddle

</div>

<div style={{fontSize:'2rem',
textAlign:'center',padding:'1rem',fontWeight:'bold'}}>

  {random_number_gen()}

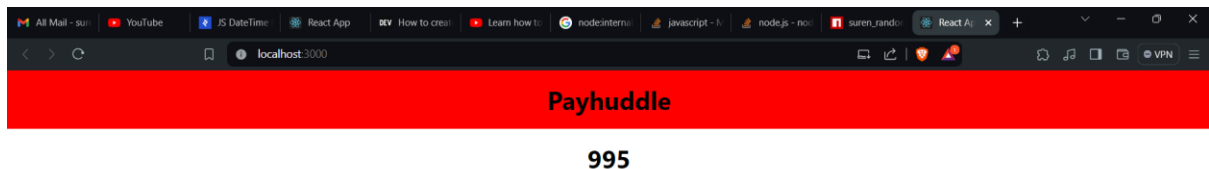
</div>

</>

)
}

export default App
```

Output:



To create a Component for Packaging:

Steps:

Files:

.babelrc

Webpack.config.js

Package.json

To create a folder name (myComponent).In terminal npm init (or) npm init -y

To install the webpack and necessary (Babel) loaders

Cmd => npm install --save-dev webpack webpack-cli babel-loader @babel/core @babel/preset-env @babel/preset-react

Inside the **package.json**:

```
{
  "name": "mycomponent",
  "version": "1.0.0",
  "description": "Reusable Component",
  "main": "dist/index.js",
  "scripts": {
    "build": "webpack --mode production"
  },
  "author": "Surendhar",
  "license": "ISC",
  "dependencies": {
    "react": "^17.0.2",
    "react-dom": "^17.0.2"
  },
}
```

```
"devDependencies": {  
  "@babel/core": "^7.16.7",  
  "@babel/preset-env": "^7.16.7",  
  "@babel/preset-react": "^7.24.1",  
  "babel-loader": "^8.2.2",  
  "webpack": "^5.68.0",  
  "webpack-cli": "^4.9.1"  
}  
}
```

Create a src/index.js:

Index.js:

```
import React from 'react';  
import 'bootstrap/dist/js/bootstrap.bundle.min.js';  
const MyComponent = ({text}) => {  
  return (  
    <div>  
      <h1>Welcome to MyComponent!</h1>  
      <p>Use the Component on the Other  
Projects</p>  
      <button className='btn btn-  
primary'>{text}</button>  
    </div>  
  );  
};  
export default MyComponent;
```

Setup the **webpack.Config.js** (**_Write it manually_**)

```
const path = require('path');
module.exports = {
  entry: './src/index.js',
  output: {
    path: path.resolve(__dirname, 'dist'),
    filename: 'index.js',
    libraryTarget: 'commonjs2',
  },
  module: {
    rules: [
      {
        test: /\.?(js|jsx)$/,
        exclude: /node_modules/,
        use: {
          loader: 'babel-loader',
          options: {
            presets: ['@babel/preset-env',
 '@babel/preset-react'],
          },
        },
      },
    ],
  },
}
```

```
    ],  
  },  
};
```

.babelrc

```
{  
  "presets": [  
    "@babel/preset-env",  
    "@babel/preset-react"  
  ]  
}
```

To build the component, **npm run build** the dist folder will be created automatically.

Create a link by using the command,

Inside the component => npm link

To install the component on your project

Inside the project => npm link <package name> mycomponent

On the Other Project Code:

App.js

```
import './App.css';  
import MyComponent from 'mycomponent';  
function App() {  
  const text = "Click Here"  
  return (  
    <div className="App">
```

```
<MyComponent
  text={text}
/>
</div>
);
}
export default App;
```

Output:

