

Lesson 03 Demo 03

Creating a React Program to Implement a Timer

Objective: To create a React program to implement a timer

Tools Required: Node terminal, React app, and Visual Studio Code

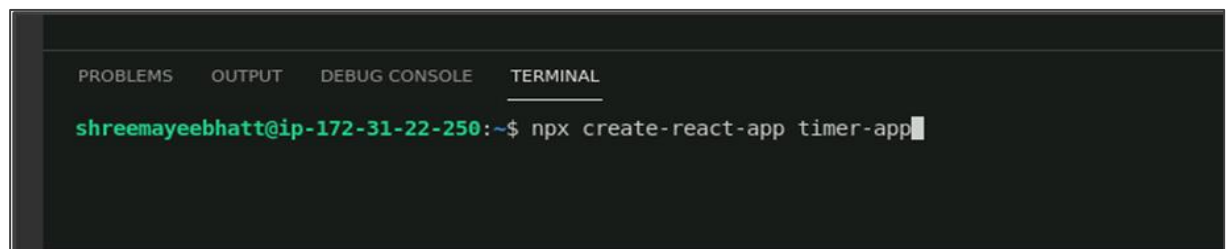
Prerequisites: Knowledge of creating a React app and an understanding of the folder structure

Steps to be followed:

1. Create a new React app
2. Configure the src/App.js file
3. Create a new file called Timer.js
4. Run the app and verify the functionality

Step 1: Create a new React app

- 1.1. Open your terminal and run the following command to create a new **React** app:
npx create-react-app timer-app



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
shreemayeebhatt@ip-172-31-22-250:~$ npx create-react-app timer-app
```

Note: This will create a new **React** app in a directory named **timer-app**.

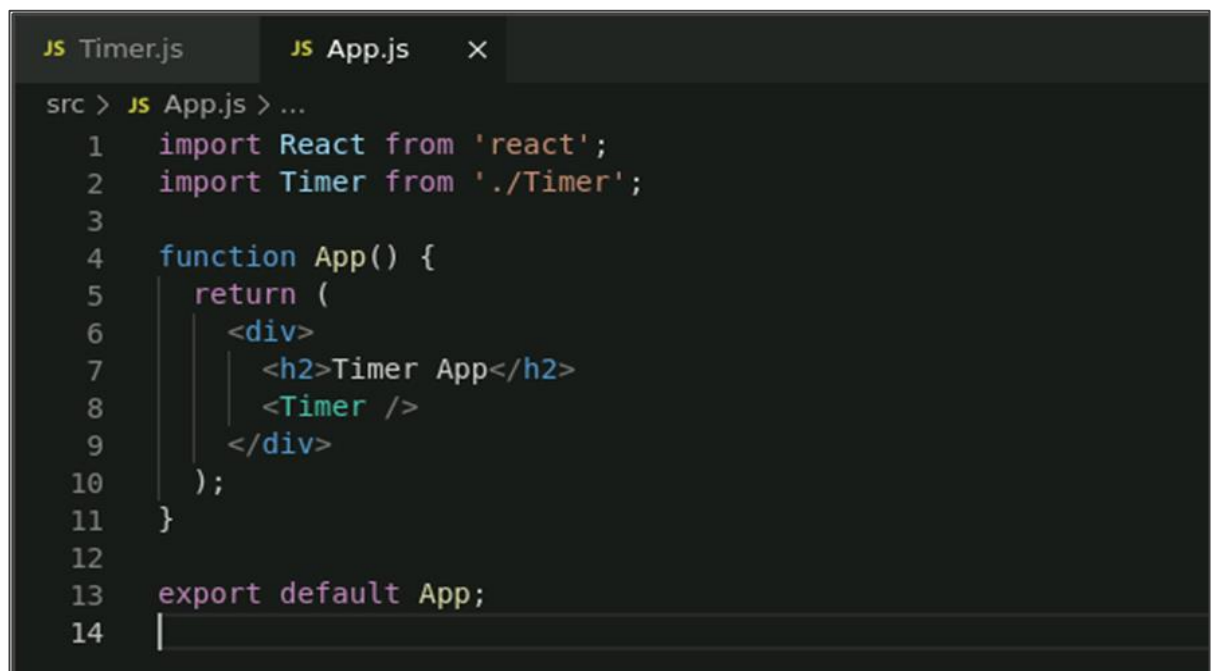
- 1.2. Navigate to the newly created directory by running the **cd timer-app** command

Step 2: Configure the src/App.js file

- 2.1 Open the React project in the Visual Studio code and navigate to the **src/App.js** file. Replace the existing code in **App.js** with the following code:

```
import React from 'react';  
import Timer from './Timer';
```

```
function App() {  
  return (  
    <div>  
      <h2>Timer App</h2>  
      <Timer />  
  
    </div>  
  );  
}  
export default App;
```



```
JS Timer.js JS App.js ×  
src > JS App.js > ...  
1  import React from 'react';  
2  import Timer from './Timer';  
3  
4  function App() {  
5    return (  
6      <div>  
7        <h2>Timer App</h2>  
8        <Timer />  
9      </div>  
10   );  
11 }  
12  
13 export default App;  
14 |
```

Note: This code defines the **App** component that renders the **Timer** component.

Step 3: Create a new file called Timer.js

3.1 In your code editor, create a new file called **Timer.js** inside the **src** directory

3.2 Copy and paste the provided code into the **Timer.js** file

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

class Timer extends Component {
  constructor(props) {
    super(props);
    this.state = {
      seconds: 0
    };
  }
  componentDidMount() {
    this.intervalId = setInterval(() => {
      this.setState(prevState => ({
        seconds: prevState.seconds + 1
      }));
    }, 1000);
  }
  componentWillUnmount() {
    clearInterval(this.intervalId);
  }
  render() {
    return (
      <div>
        <h1>Timer: {this.state.seconds} seconds</h1>
        <button onClick={() => this.setState({ seconds: 0 })}>Reset</button>
      </div>
    );
  }
}

export default Timer;
```

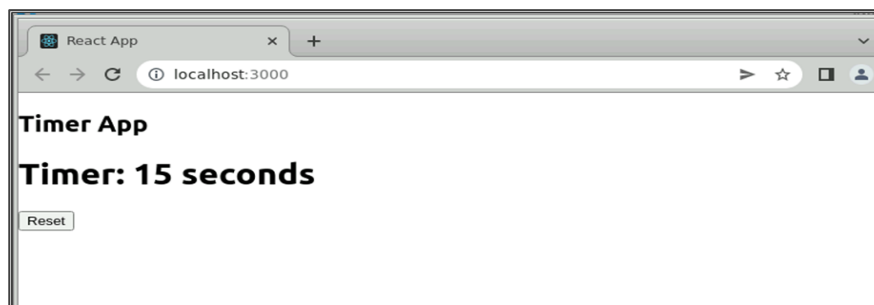
```

src > JS Timer.js
1 import React, { Component } from 'react';
2
3 class Timer extends Component {
4   constructor(props) {
5     super(props);
6     this.state = {
7       seconds: 0
8     };
9   }
10
11   componentDidMount() {
12     this.intervalId = setInterval(() => {
13       this.setState(prevState => ({
14         seconds: prevState.seconds + 1
15       }));
16     }, 1000);
17   }
18
19   componentWillUnmount() {
20     clearInterval(this.intervalId);
21   }
22
23   render() {
24     return (
25       <div>
26         <h1>Timer: {this.state.seconds} seconds</h1>
27         <button onClick={() => this.setState({ seconds: 0 })}>Reset</button>
28       </div>
29     );
30   }
31 }
32
33 export default Timer;
34

```

Step 4: Run the App and verify the functionality

- 4.1 Go to the terminal and execute the command **npm start** within the project directory **timer-app** to run the app
- 4.2 Once the server starts successfully, open **http://localhost:3000** in your browser to view and check the functionality of the app



In the browser, you will see the heading **Timer App** and a display of the seconds elapsed in the timer, starting from 0. The timer will increment by 1 every second. You can click on the **Reset** button to set the timer back to 0.

With this, you have implemented a **Timer app** using React.