Lesson-End Project

Creating a React Application Using Event Handler and Components

Project agenda: To create a notes application so that when the user types in the text and clicks on add notes, the text gets added

Description: The application allows users to input text, triggering an event handler that dynamically updates the state and adds new notes to the interface.

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

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

Expected deliverables: A fully functional React notes application with dynamic note addition, implemented components, and event handlers.

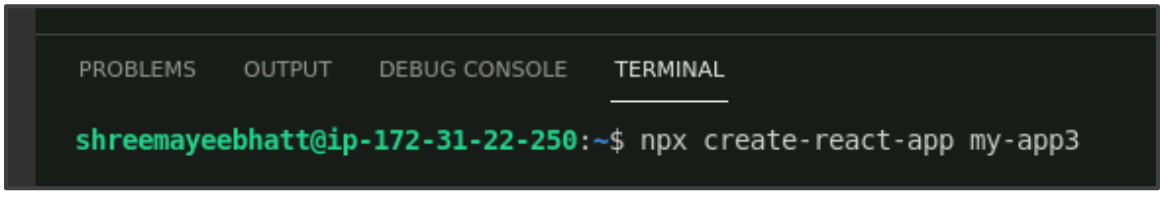
Steps to be followed:

1. Create a new React project
2. Implement the Notes Component
3. Implement the Default Component
4. Add State and Functionality to the Component
5. Run the app

Step 1: Create a new React project

1.1 Create a new React project by using the below command:

npx create-react-app my-app3



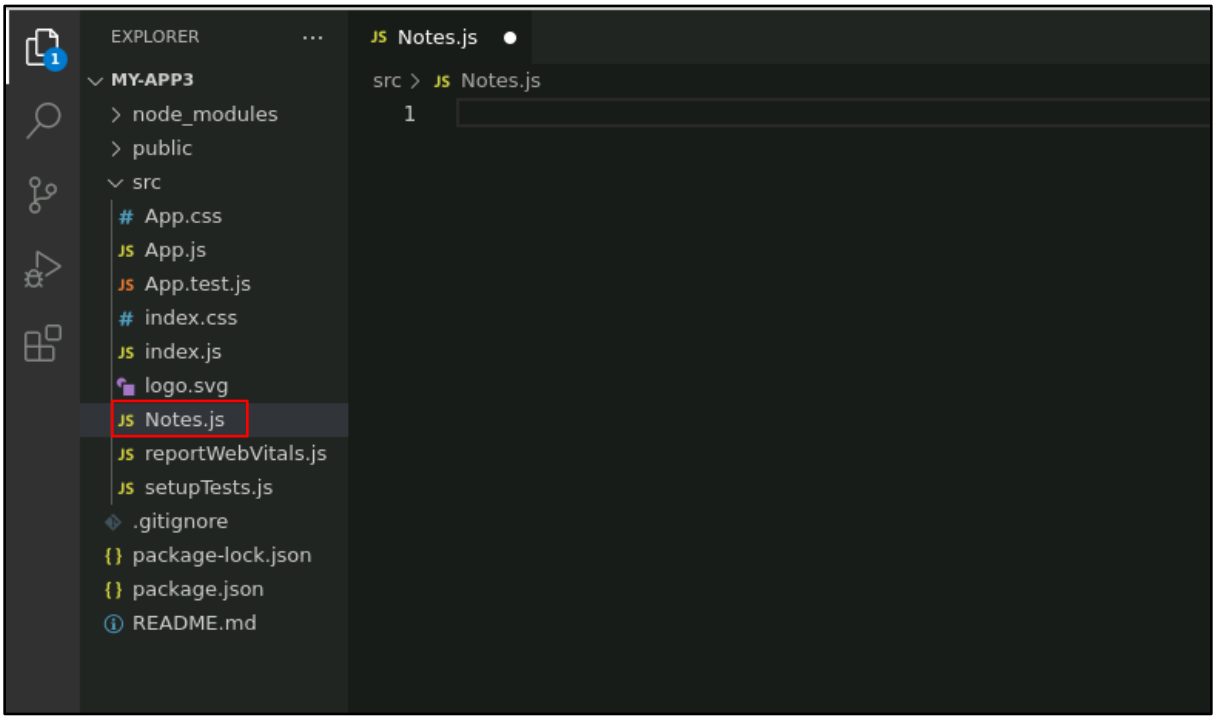
1.2 Move to the newly created directory by running the **cd my-app3** command

```
Happy hacking!
shreemayeebhatt@ip-172-31-22-250:~$ cd my-app3
```

1.3 Open **Visual Studio Code** and navigate to the project directory

Step 2: Implement the Notes Component

2.1 In the **src** directory, create a new file called **Notes.js**



2.2 In the **Notes.js** file, import React and define a functional component called **Notes** that accepts **props**

2.3 Implement the **Notes** component to map the **props.data** array and return a **div** for each note's text:

const Notes = props => props.data.map(note => {note.text});

2.4 Export the **Notes** component using the below command:

export default Notes;

```
> JS Notes.js > ...
1  import React from 'react';
2
3  const Notes = props => props.data.map(note => <div>{note.text}</div>);
4
5  export default Notes;
6
```

//Notes.js:
import React from react;
const Notes = props => props.data.map(note => {note.text});
export default Notes;

Step 3: Implement the Default Component

3.1 In the src directory, open the **App.js** file and modify the code as below

3.2 Import the **Notes** component from **./Notes**

```
import Notes from './Notes';
```

3.3 Create an array called **data** that contains objects representing notes:

const data = [{ text: 'Hey' }, { text: 'There' }];

```
const initialData = [{ text: 'Hey' }, { text: 'There' }];
```

3.4 Implement a functional component using an arrow function and export it as the default component

return (
<>
<input id="noteinput" style={{ width: '80%' }} type="text"
placeholder="Enter a new note" />

```
<button onClick={() => handleClick()}>Add note</button>

<Notes data={data} />
</>
);
```

```
return (
  <>
    <input id="noteinput" style={{ width: '80%' }} type="text" placeholder="Enter a new note" />
    <button onClick={handleClick}>Add note</button>
    <Notes data={data} />
  </>
);
```

Step 4: Add State and Functionality to the Component

4.1 Import the **useState** Hook from **React**

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

4.2 Inside the functional component, declare the **data** state variable and the **setData** function using **useState**

const [data, setData] = useState(initialData);

```
const App = () => {
  const [data, setData] = useState(initialData);
```

4.3 Add the code given below to handle the **button click** event:

```
const handleClick = () => {
const text = document.querySelector('#noteinput').value.trim();
if (text) {
const nextState = produce(data, draftState => {
draftState.push({ text });
});
document.querySelector('#noteinput').value = '';
setData(nextState);
 }
};
```

```
const handleClick = () => {
  const text = document.querySelector('#noteinput').value.trim();
  if (text) {
    const nextState = produce(data, draftState => {
      draftState.push({ text });
    });
    document.querySelector('#noteinput').value = '';
    setData(nextState);
  }
};
```

4.4 Modify the **JSX** to include an input field, a button, and the **Notes** component:

```
return (
 <>
<input id="noteinput" style={{ width: '80%' }} type="text" placeholder="Enter a
    new note" />
 <button onClick= {() => handleClick()}>Add note</button>
<Notes data={data} />
</>
 );
```

Notes.js | index.js | App.js

src > App.js > App > handleClick

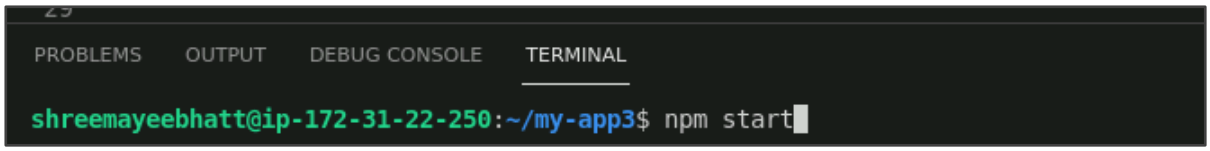
```
1  import React, { useState } from 'react';
2  import Notes from './Notes';
3  import produce from 'immer';
4
5  const initialData = [{ text: 'Hey' }, { text: 'There' }];
6
7  const App = () => {
8    const [data, setData] = useState(initialData);
9
10   const handleClick = () => {
11     const text = document.querySelector('#noteinput').value.trim();
12     if (text) {
13       const nextState = produce(data, draftState => {
14         draftState.push({ text });
15       });
16       document.querySelector('#noteinput').value = '';
17       setData(nextState);
18     }
19   };
20
21   return (
22     <>
23       <input id="noteinput" style={{ width: '80%' }} type="text" placeholder="Enter a new note" />
24       <button onClick={handleClick}>Add note</button>
25       <Notes data={data} />
26     </>
27   );
28 };
29
30 export default App;
31
```

```
//App.js
import React, { useState } from 'react';
import Notes from './Notes';
import {produce} from 'immer';

const initialData = [{ text: 'Hey' }, { text: 'There' }];
const App = () => {
 const [data, setData] = useState(initialData);

 const handleClick = () => {
  const text = document.querySelector('#noteinput').value.trim();
  if (text) {
   const nextState = produce(data, draftState => {
    draftState.push({ text });
   });
   document.querySelector('#noteinput').value = '';
   setData(nextState);
}
 };
 return (
  <>
   <input id="noteinput" style={{ width: '80%' }} type="text" placeholder="Enter a
        new note" />
   <button onClick={handleClick}>Add note</button>
   <Notes data={data} />
  </>
 );
};

export default App;
```

Step 5: Run the app

5.1 In your terminal, navigate to the project's root directory

5.2 Run the **npm start** command to start the development server



5.3 Open your browser and navigate to **http://localhost:3000**

You should see the app with the initial notes displayed and an input field and button to add more notes.

5.4 Enter text into the input field

5.5 Click the **Add note** button to add new notes

By following these steps, you have successfully created a React application with event handlers and components to manage and dynamically display notes.