Lesson-End Project

Creating a React Application Using All the Hooks

Project agenda: To develop an application that displays a random number and employs the useReducer, useEffect, and custom Hooks

Description: This React application showcases the practical implementation of React Hooks, including useReducer, useEffect, and custom Hooks. Users can input tasks, and event handlers dynamically update the state, leading to the seamless addition of new tasks to the interface for efficient task management.

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 application demonstrating task management through the practical implementation of React Hooks, including useReducer, useEffect, and custom Hooks

Steps to be followed:

1. Create a new React app
2. Create a new file called useRandomNumber.js in the src directory
3. Import useRandomNumber from the useRandomNumber.js file in App.js
4. Run the app and view it in the browser

Step 1: Create a new React app

1.1 Open the terminal and run the following command:

npx create-react-app state-management-demo

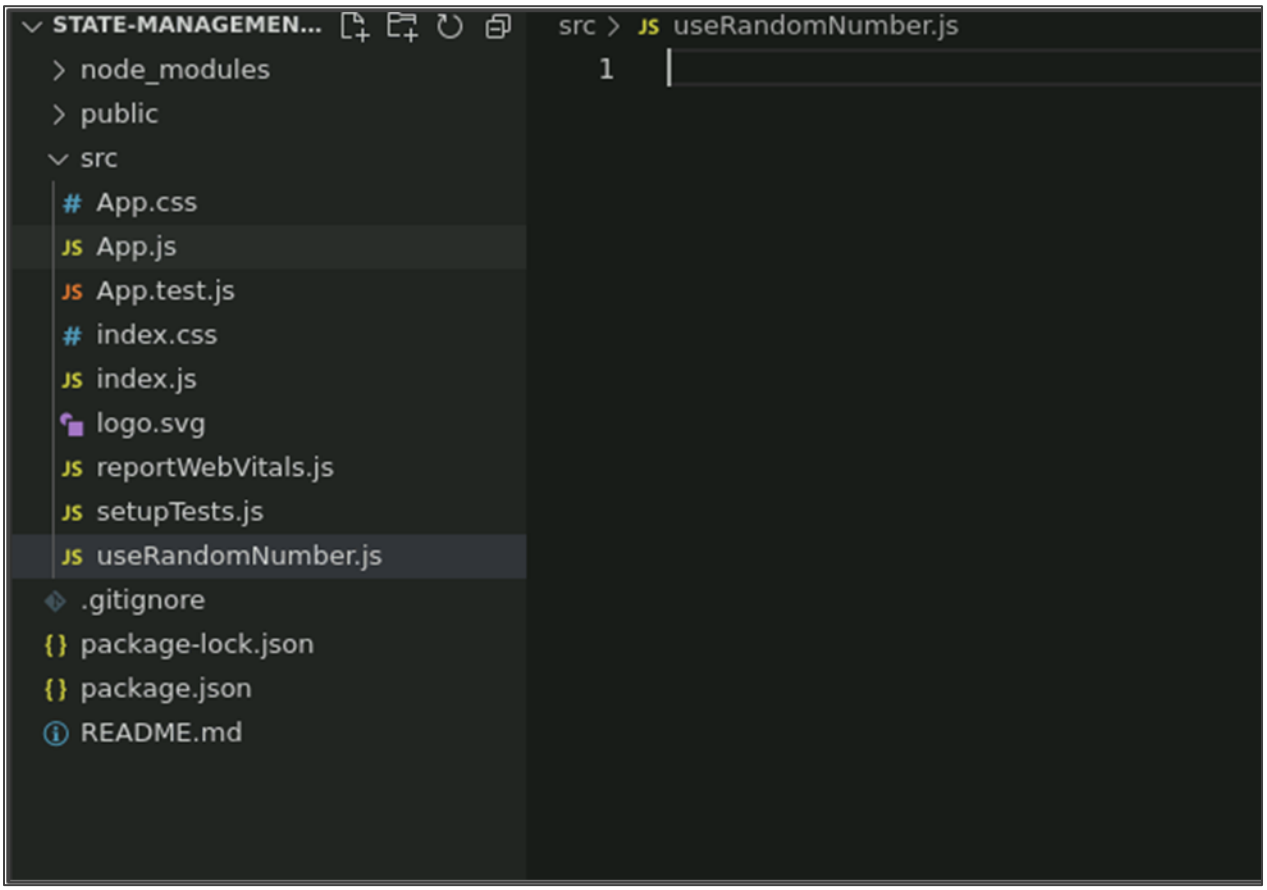
This command will create a new React app with the name **state-management-demo.**

1.2 Run the **cd state-management-demo** command in the terminal

This will change the current directory to the newly created **React** app directory.

Step 2: Create a new file called useRandomNumber.js in the src directory

2.1 Open your React project **state-management-demo** in the **Visual Studio Code** and create a new file called **useRandomNumber.js** within the **src** directory



2.2 Inside the **useRandomNumber.js** file, import the **useReducer** and **useEffect** Hooks from the **React** library

```
import { useReducer, useEffect } from 'react';
```

2.3 Define the **reducer** function that takes a **state** and **action** as parameters and returns a new state based on the action type

```
const reducer = (state, action) => {
switch (action.type) {
case 'SET_NUMBER':
return { number: action.payload };
default:
return state;
}
};
```

2.4 Implement the **useRandomNumber** function that utilizes the **useReducer** Hook to manage the state and dispatch actions

```
function useRandomNumber() {
const [state, dispatch] = useReducer(reducer, { number: null });
```

2.5 Inside the **useEffect** Hook, use **setInterval** to generate a random number every second and dispatch an action to update the state

```
useEffect(() => {
const intervalId = setInterval(() => {
const number = Math.floor(Math.random() * 100) + 1;
dispatch({ type: 'SET_NUMBER', payload: number });
}, 1000);
```

2.6 Return the current value of the **number** state

```
return () => clearInterval(intervalId);
}, []);

return state.number;
}
```

Note: Refer to the following code to configure the **useRandomNumber.js** file:

```
import { useReducer, useEffect } from 'react';

const reducer = (state, action) => {
switch (action.type) {
case 'SET_NUMBER':
return { number: action.payload };
default:
return state;
}
};

function useRandomNumber() {
const [state, dispatch] = useReducer(reducer, { number: null });

useEffect(() => {
```

```
const intervalId = setInterval(() => {
const number = Math.floor(Math.random() * 100) + 1;
dispatch({ type: 'SET_NUMBER', payload: number });
}, 1000);

return () => clearInterval(intervalId);
}, []);

return state.number;
}

export default useRandomNumber;
```

```
import { useReducer, useEffect } from 'react';

const reducer = (state, action) => {
switch (action.type) {
case 'SET_NUMBER':
return { number: action.payload };
default:
return state;
}
};

function useRandomNumber() {
const [state, dispatch] = useReducer(reducer, { number: null });

useEffect(() => {
const intervalId = setInterval(() => {
const number = Math.floor(Math.random() * 100) + 1;
dispatch({ type: 'SET_NUMBER', payload: number });
}, 1000);

return () => clearInterval(intervalId);
}, []);

return state.number;
}

export default useRandomNumber;
```

Step 3: Import useRandomNumber from the useRandomNumber.js file in App.js

3.1 Open the **App.js** file of the **state-management-demo** project and import the **useRandomNumber** Hook from the **useRandomNumber.js** file

```
import useRandomNumber from './useRandomNumber';
```

3.2 Inside the **App** function component, call the **useRandomNumber** Hook and assign the returned value to the **number** variable

```
6   function App() {
7   const number = useRandomNumber();
```

3.3 In the return statement, use the **number** variable in the JSX to display a random number

```
return (
<div className="App">
<h1>State Management Demo</h1>
<p>Random number: {number}</p>
</div>
);
}

export default App;
```

Note: Refer to the following code to configure the **App.js** file:

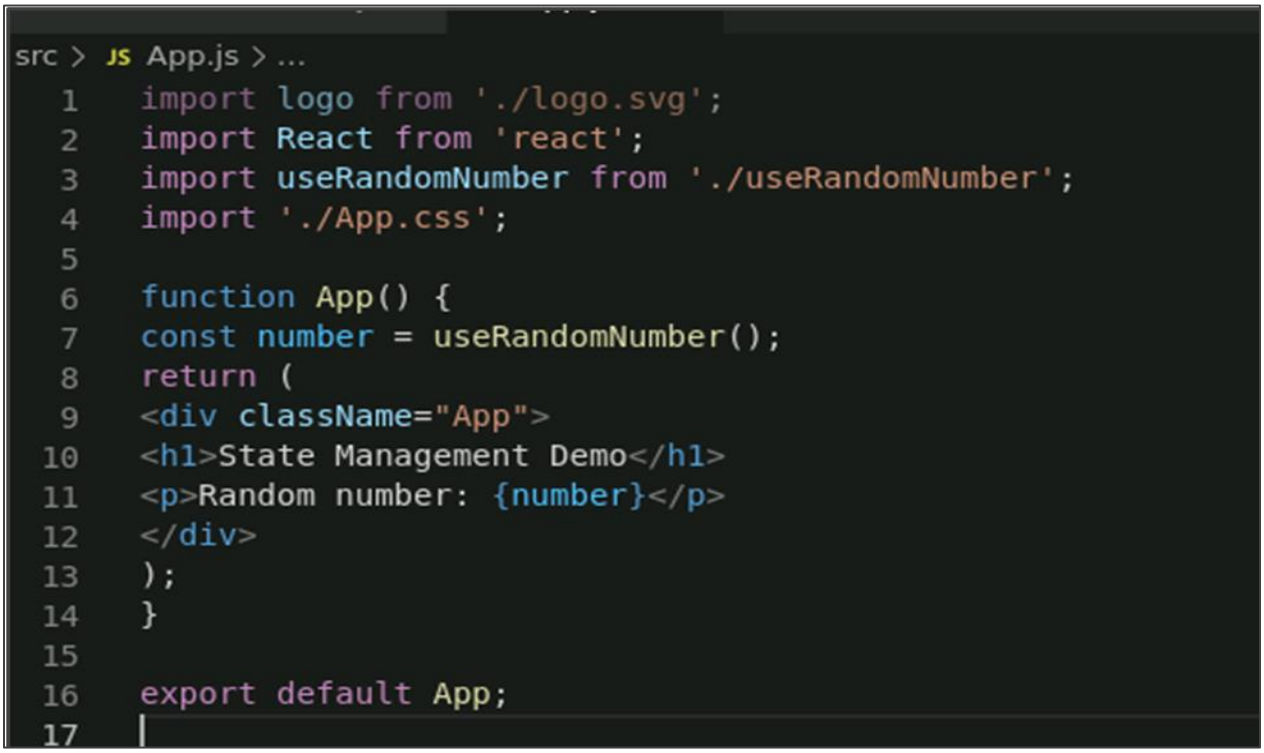
```
import React from 'react';
import useRandomNumber from './useRandomNumber';
import './App.css';

function App() {
const number = useRandomNumber();

return (
<div className="App">
    <h1>State Management Demo</h1>
    <p>Random number: {number}</p>
</div>

    );
    }

export default App;
```



Step 4: Run the app and view it in the browser

4.1 In the terminal, navigate to the project directory and run the **npm start** command to start the app

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

The app should be running, and you should see a simple app that displays a random number that changes every second.

With this, you have successfully created a React application to demonstrate the working of **useReducer**, **useEffect**, and custom Hooks.