Lesson 04 Demo 02

Fetch Data in a React Application Using useReducer Hook

Objective: To develop a React application that demonstrates fetching data using useReducer Hook

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

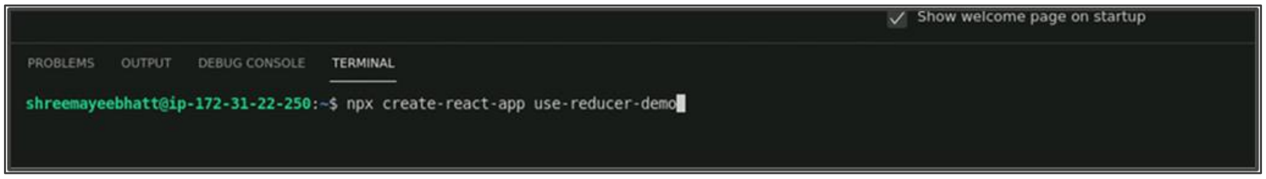
Prerequisites: Knowledge of creating a React app and understanding of the folder structure

Steps to be followed:

1. Create a new **React** app
2. Create a new reducer function in **App.js**
3. Initialize state using the **useReducer** Hook 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 command **npx create-react-app use-reducer-demo** to create a new **React** app with the name **use-reducer-demo**



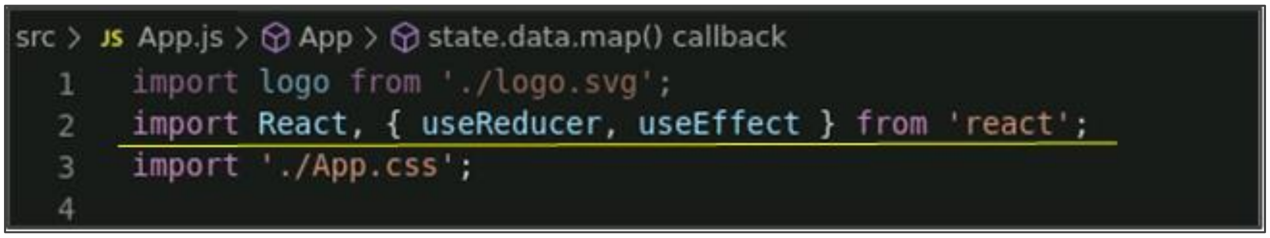
```
shreemayeebhatt@ip-172-31-22-250:~$ npx create-react-app use-reducer-demo
```

1.2 Run the command **cd use-reducer-demo** in the terminal to change the current directory to the newly created **React** app directory

Step 2: Create a new reducer function in App.js

2.1 Open your React project in **Visual Studio Code**, and navigate through the project structure to open the **App.js** file within the **src** directory

2.2 Modify the **App.js** file by importing the **useReducer** and **useEffect** Hooks from the **React** library using the following command:
import React, { useReducer, useEffect } from 'react';

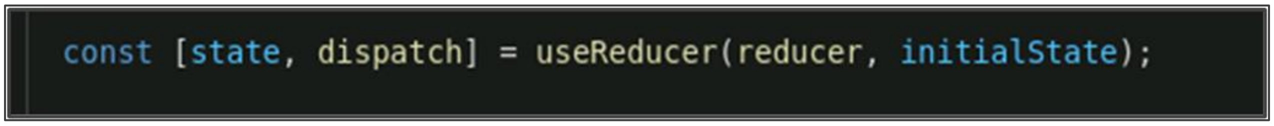


```
import logo from './logo.svg';
import React, { useReducer, useEffect } from 'react';
import './App.css';

```

Note: The **useReducer** is used to manage the app's state and **useEffect** to fetch data from an API when the component mounts

2.3 Declare an **initialState** object that represents the initial state of the component

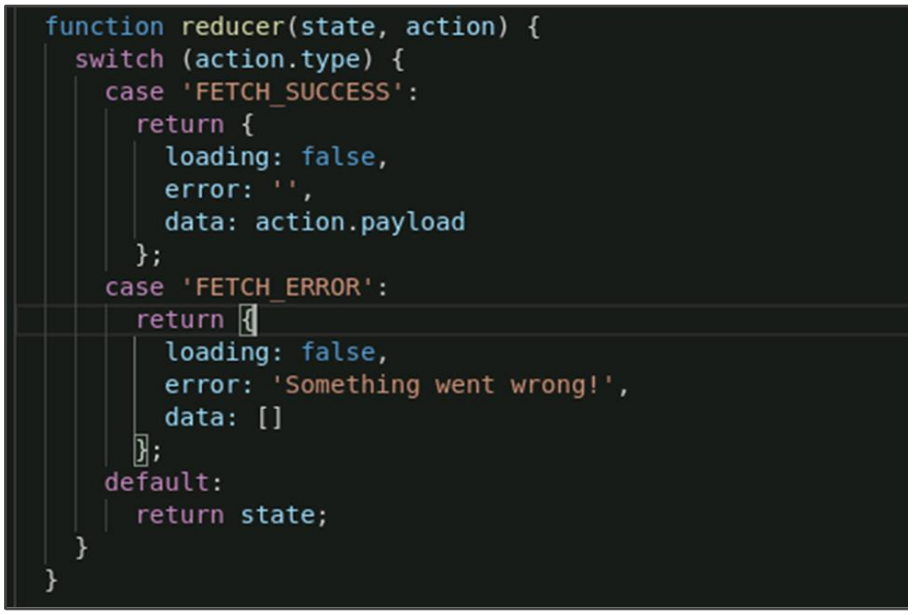
const [state, dispatch] = useReducer(reducer, initialState);



```
const [state, dispatch] = useReducer(reducer, initialState);
```

2.4 Create a **reducer** function that takes in the **state** and **action** parameters and returns a new state based on the action type

```
function reducer(state, action) {

 switch (action.type) {
   case 'FETCH_SUCCESS':
    return {
     loading: false,
     error: '',
     data: action.payload
    };
   case 'FETCH_ERROR':
    return {
     loading: false,
     error: 'Something went wrong!',
     data: []
    };
   default:
    return state;
  }
 }
```

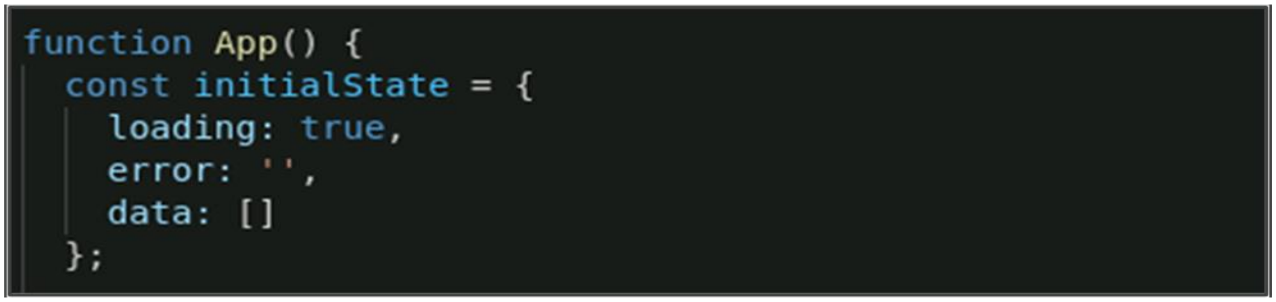


```
function reducer(state, action) {
  switch (action.type) {
    case 'FETCH_SUCCESS':
      return {
        loading: false,
        error: '',
        data: action.payload
      };
    case 'FETCH_ERROR':
      return {
        loading: false,
        error: 'Something went wrong!',
        data: []
      };
    default:
      return state;
  }
}
```

Step 3: Initialize state using the useReducer Hook in App.js

3.1 Inside the **App** function component, use the **useReducer** Hook to initialize the state using the **reducer** function and **initialState** object

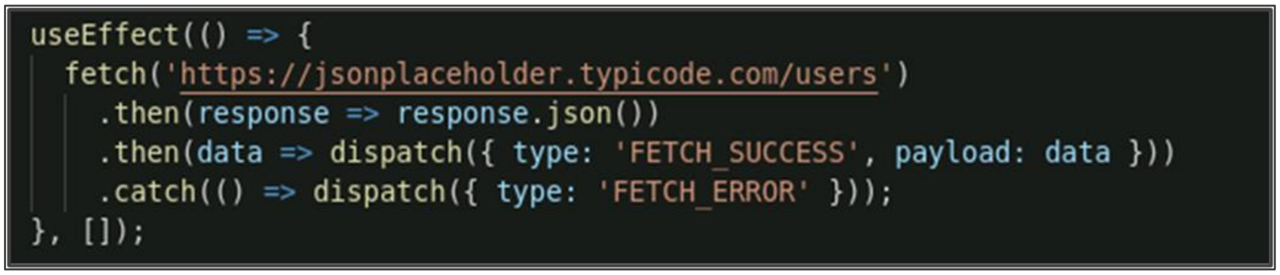
```
function App() {
const initialState = {
  loading: true,
      error: '',
   data: []
 };
```



```
function App() {
  const initialState = {
    loading: true,
    error: '',
    data: []
  };
```

3.2 Use the **useEffect** Hook to fetch data from an **API** when the component mounts

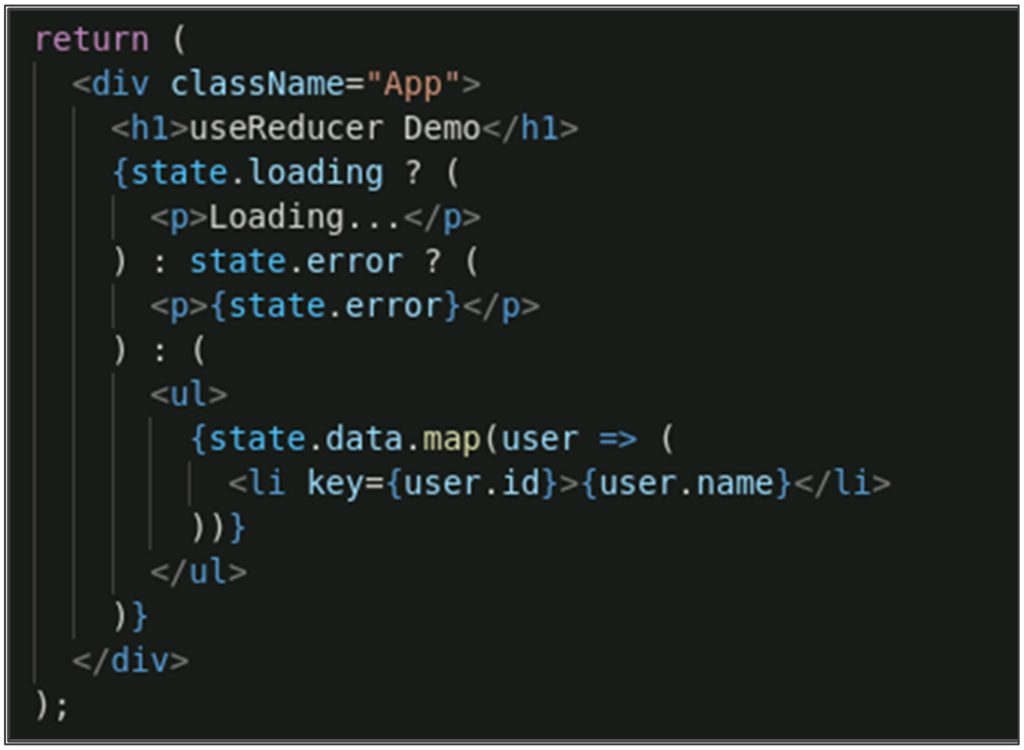
```
useEffect(() => {
  fetch('https://jsonplaceholder.typicode.com/users')
    .then(response => response.json())
    .then(data => dispatch({ type: 'FETCH_SUCCESS', payload: data }))
    .catch(() => dispatch({ type: 'FETCH_ERROR' }));
 }, []);
```



```
useEffect(() => {
  fetch('https://jsonplaceholder.typicode.com/users')
    .then(response => response.json())
    .then(data => dispatch({ type: 'FETCH_SUCCESS', payload: data }))
    .catch(() => dispatch({ type: 'FETCH_ERROR' }));
}, []);
```

3.3 In the **return** statement, display the data fetched from the **API** by handling the **loading** and **error** states

```
return (
  <div className="App">
    <h1>useReducer Demo</h1>
    {state.loading ? (
      <p>Loading...</p>
    ) : state.error ? (
      <p>{state.error}</p>
    ) : (
      <ul>
        {state.data.map(user => (
          <li key={user.id}>{user.name}</li>
        ))}
      </ul>
    )}
  </div>
 );
}
export default App;
```



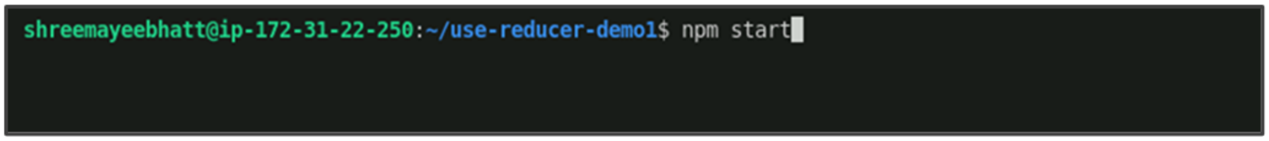
```
return (
  <div className="App">
    <h1>useReducer Demo</h1>
    {state.loading ? (
      <p>Loading...</p>
    ) : state.error ? (
      <p>{state.error}</p>
    ) : (
      <ul>
        {state.data.map(user => (
          <li key={user.id}>{user.name}</li>
        ))}
      </ul>
    )}
  </div>
);
```

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

```
import React, { useReducer, useEffect } from 'react';
import './App.css';

function App() {
  const initialState = {
    loading: true,
        error: '',
    data: []
  };

function reducer(state, action) {
   switch (action.type) {
      case 'FETCH_SUCCESS':
        return {
          loading: false,
          error: '',
          data: action.payload
        };
      case 'FETCH_ERROR':
        return {
          loading: false,
          error: 'Something went wrong!',
          data: []
        };
      default:
        return state;
    }
  }
  const [state, dispatch] = useReducer(reducer, initialState);
  useEffect(() => {
   fetch('https://jsonplaceholder.typicode.com/users')
    .then(response => response.json())
    .then(data => dispatch({ type: 'FETCH_SUCCESS', payload: data }))
    .catch(() => dispatch({ type: 'FETCH_ERROR' }));
  }, []);

 return (
    <div className="App">
      <h1>useReducer Demo</h1>
      {state.loading ? (
        <p>Loading...</p>
      ) : state.error ? (
        <p>{state.error}</p>
      ) : (
        <ul>
          {state.data.map(user => (
            <li key={user.id}>{user.name}</li>
          ))}
        </ul>
      )}
    </div>
  );
}
export default App;
```

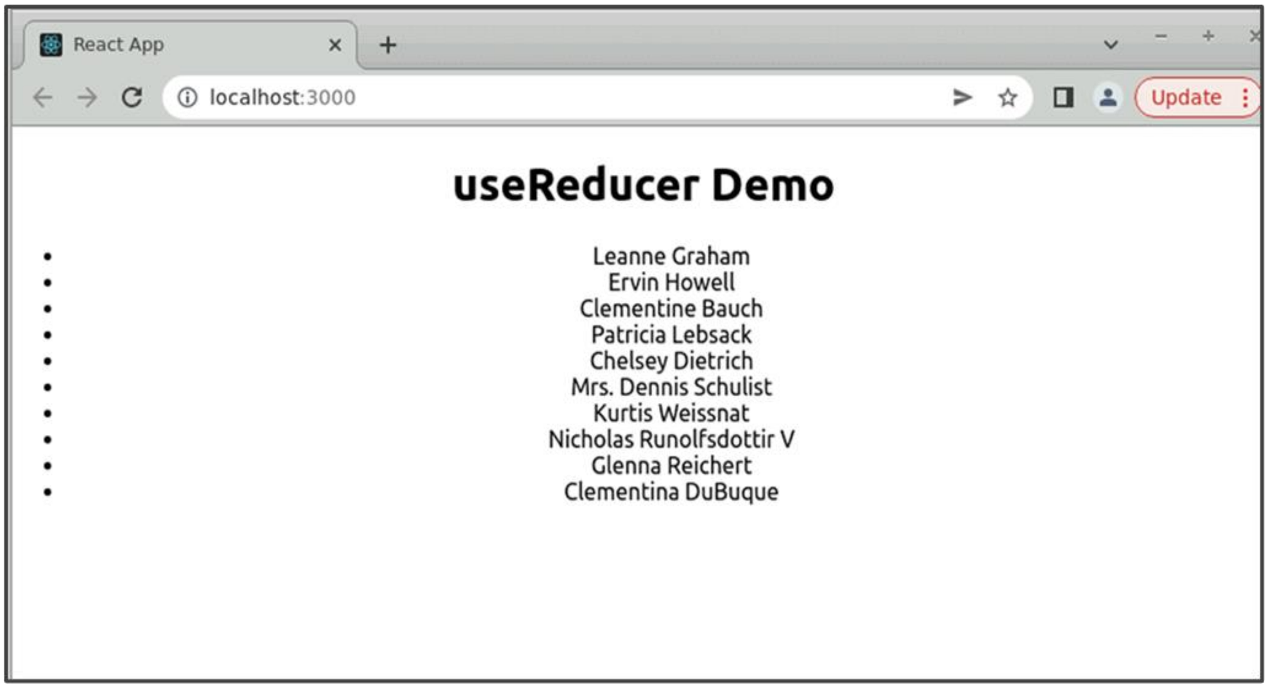
Step 4: Run the app and view it in the browser

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



```
shreemayeebhatt@ip-172-31-22-250:~/use-reducer-demo1$ npm start
```

4.2 Open your browser and navigate to http://localhost:3000 to see the final output



```
useReducer Demo

• Leanne Graham
• Ervin Howell
• Clementine Bauch
• Patricia Lebsack
• Chelsey Dietrich
• Mrs. Dennis Schulist
• Kurtis Weissnat
• Nicholas Runolfsdottir V
• Glenna Reichert
• Clementina DuBuque
```

You should see a simple app that displays a list of usernames fetched from the **JSONPlaceholder API.**

With this, you've successfully created a React application to fetch data using the **useReducer** Hook.