**Lesson 10 Demo 03**

**Creating a React Redux Testing Application**

**Objective:** To create a React Redux testing application for adding and displaying names using Jest and the testing library

**Tools required:** Node.js and React.js

**Prerequisites:** None

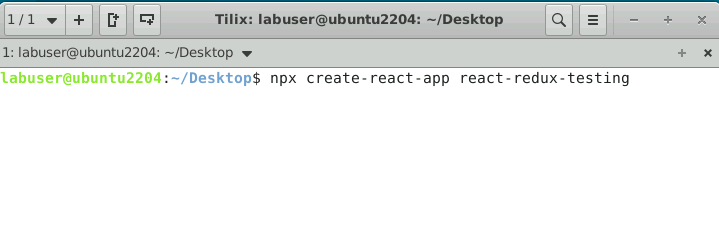
Steps to be followed:

1. Create and set up the React application
2. Create the characterSlice.js and buildStore.js file
3. Create an App.js and App.test.js file
4. Test the application

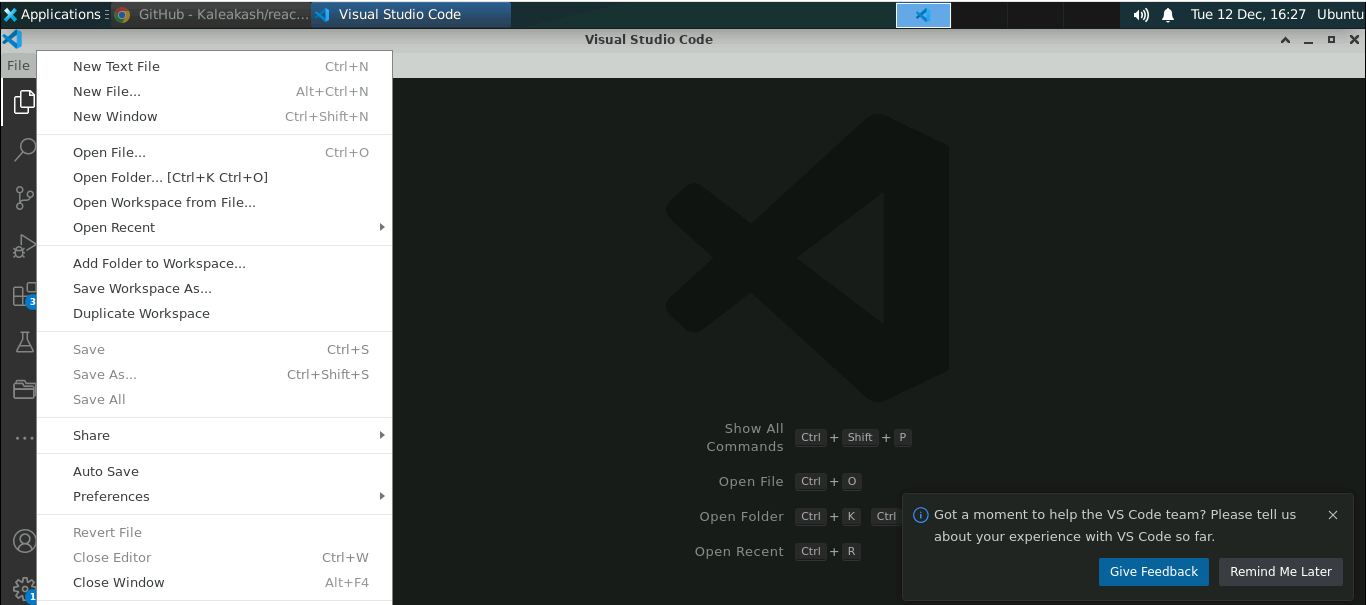
**Step 1: Create and set up the React application**

1. Open a terminal window to run the following command to create a React application:

**npx create-react-app react-redux-testing**



1. Open the created React application folder (**react-redux-testing**) in VS Code by clicking on **File** in the top left corner and selecting **Open Folder**

****

1. Click on **Open** button

A screenshot of a computer

Description automatically generated

The following project structure appears as follows:

A screenshot of a computer program

Description automatically generated

1. Inside the project, open the **TERMINAL** and run the following command to install the required dependencies:

**npm install**

A screenshot of a computer screen

Description automatically generated

1. Open the **package-lock.json** file and view the testing dependencies

A screen shot of a computer

Description automatically generated

**Step 2: Create the characterSlice.js and buildStore.js file**

1. Inside the **src** folder, create the file named **characterSlice.js** and enter the following code:

**import { createSlice } from "@reduxjs/toolkit";**

**const charactersSlice = createSlice({**

**name: "characters",**

**initialState: [],**

**reducers: {**

**add(state, action) {**

**state.unshift(action.payload);**

**},**

**remove(state, action) {**

**state.splice(state.indexOf(action.payload), 1);**

**}**

**}**

**});**

**export default charactersSlice;**

A screen shot of a computer

Description automatically generated

1. Create the file named **buildStore.js** and enter the following code:

**import { configureStore } from "@reduxjs/toolkit";**

**import charactersSlice from "./charactersSlice";**

**export default function buildStore() {**

**const store = configureStore({**

**reducer: charactersSlice.reducer,**

**preloadedState: ["Michael Scott", "Dwight Schrute", "Jim Halpert"]**

**});**

**return store;**

**}**

**A screen shot of a computer

Description automatically generated**

**Step 3: Create an App.js and App.test.js file**

1. Inside the **src** folder, create a file named **App.js** and enter the following code:

**import { useState } from "react";**

**import { Provider, useDispatch, useSelector } from "react-redux";**

**import charactersSlice from './charactersSlice';**

**import buildStore from "./buildStore";**

**import "./index.css";**

**const store = buildStore();**

**const actions = charactersSlice.actions;**

**export default function App() {**

**return (**

**<Provider store={store}>**

**<OfficeCharacters />**

**</Provider>**

**);**

**}**

**export function OfficeCharacters() {**

**const dispatch = useDispatch();**

**const characters = useSelector((state) => state);**

**const [newCharacter, setNewCharacter] = useState("");**

**function add(e) {**

**e.preventDefault();**

**dispatch(actions.add(newCharacter));**

**setNewCharacter("");**

**}**

**function deleteCharacter(character) {**

**dispatch(actions.remove(character));**

**}**

**return (**

**<div className="container">**

**<form onSubmit={add} className="form-group">**

**<div className="row">**

**<div className="col-2">**

**<label htmlFor="newCharacter">New Names</label>**

**</div>**

**<div className="col-3">**

**<input**

**type="text"**

**id="newCharacter"**

**value={newCharacter}**

**onChange={(e) => setNewCharacter(e.target.value)}**

**className="form-control"**

**/>**

**</div>**

**<div className="col-3">**

**<button className="btn btn-success">Add Names</button>**

**</div>**

**</div>**

**</form>**

**<ol>**

**{characters.map((character, i) => (**

**<li key={i} data-testid="character">**

**<div className="row offset-3">**

**<div className="col-3"> <span data-testid="name">{character}</span>{" "}</div>**

**<div className="col-3"> <button type="button" onClick={() => deleteCharacter(character)} className="btn btn-danger">**

**Delete**

**</button></div>**

**</div>**

**</li>**

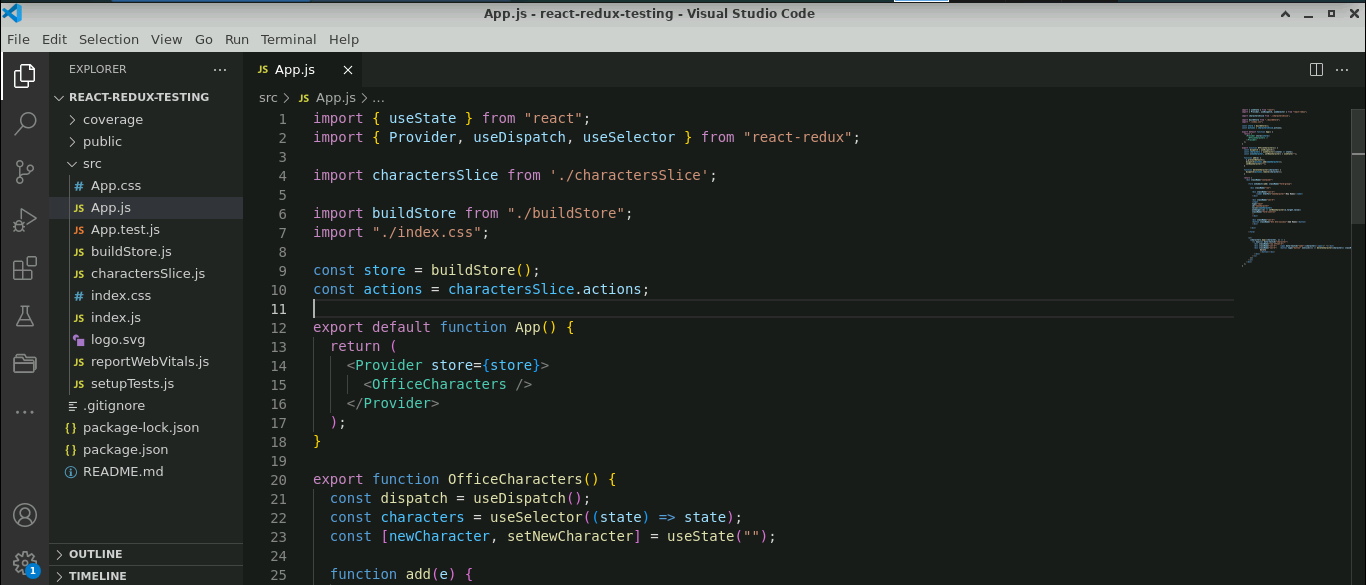
**))}**

**</ol>**

**</div>**

**);**

**}**

****

1. Create a file named **App.test.js** and enter the following code:

**import { render, screen, within, fireEvent } from "@testing-library/react";**

**import { Provider } from "react-redux";**

**import buildStore from "./buildStore";**

**import { OfficeCharacters } from "./App";**

**import charactersSlice from "./charactersSlice";**

**describe("OfficeCharacters", () => {**

**function renderOfficeCharacters() {**

**const store = buildStore();**

**render(**

**<Provider store={store}>**

**<OfficeCharacters />**

**</Provider>**

**);**

**return {**

**newCharacter: screen.getByLabelText("New Names"),**

**addButton: screen.getByText("Add Names"),**

**getCharacters() {**

**return screen.getAllByTestId("character").map((item) => ({**

**name: within(item).getByTestId("name").textContent,**

**deleteButton: within(item).getByText("Delete")**

**}));**

**}**

**};**

**}**

**it("should add a character", () => {**

**const { newCharacter, addButton, getCharacters } = renderOfficeCharacters();**

**const pam = "Pam Beesly";**

**// verify pam is NOT in the initial list**

**expect(**

**getCharacters().find((character) => character.name === pam)**

**).toBeFalsy();**

**// add pam**

**fireEvent.change(newCharacter, { target: { value: pam } });**

**fireEvent.click(addButton);**

**// verify pam is first in the list**

**expect(**

**getCharacters().findIndex((character) => character.name === pam)**

**).toBe(0);**

**});**

**it("should delete a character", () => {**

**const { getCharacters } = renderOfficeCharacters();**

**const jim = "Jim Halpert";**

**const deleteJim = getCharacters().find(**

**(character) => character.name === jim**

**).deleteButton;**

**// delete character**

**fireEvent.click(deleteJim);**

**// verify character is NOT in list**

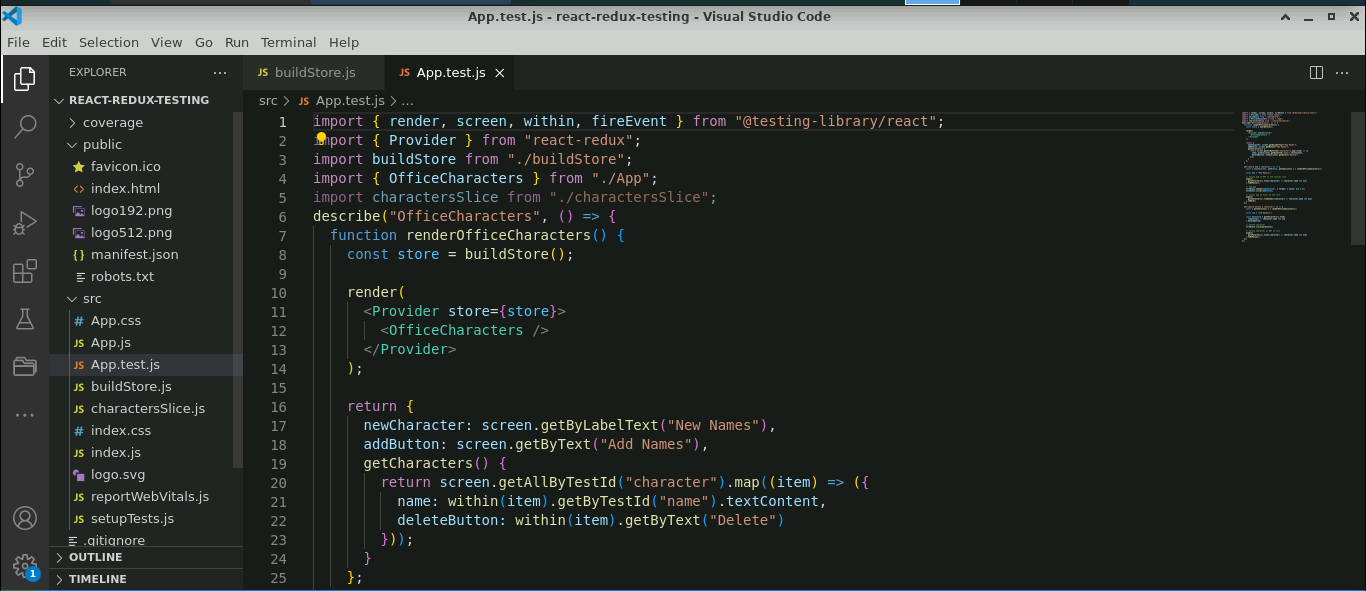
**expect(**

**getCharacters().find((character) => character.name === jim)**

**).toBeFalsy();**

**});**

**});**



**Step 4: Test the application**

1. Run the following command to test the application as shown below:

**npm test**

**A screen shot of a computer

Description automatically generated**

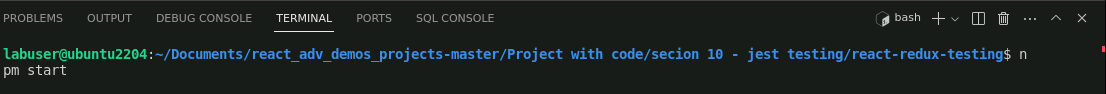
The output appears as shown below:

**A screen shot of a computer

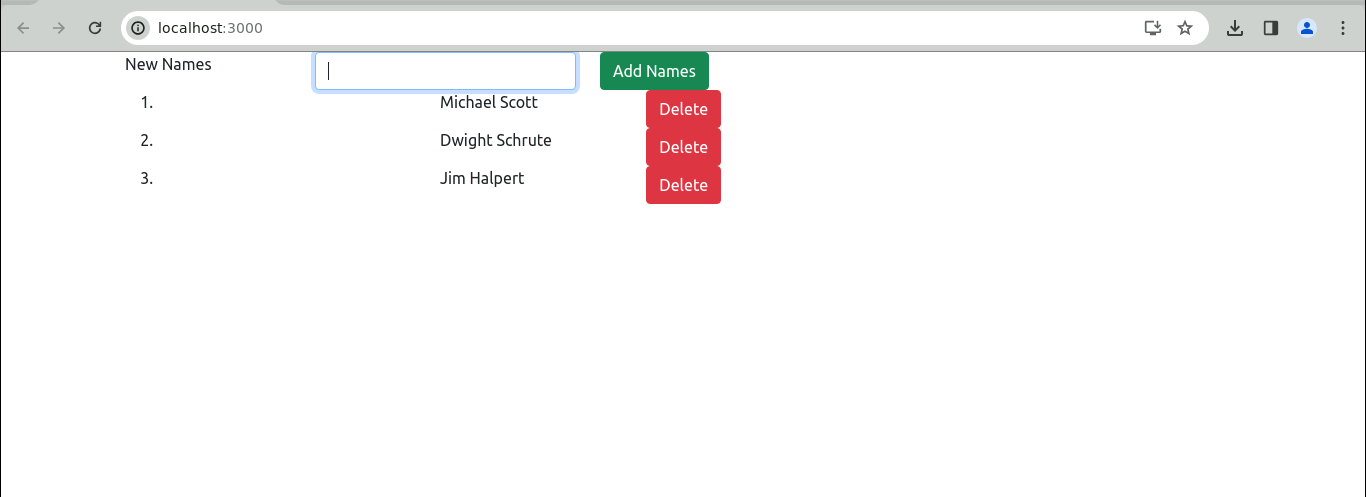
Description automatically generated**

1. Run the following command to execute the application as shown below:

**npm start**



After successful execution, the output can be viewed as shown below:

****

1. Enter the name under the **New Names** and click on the **Add Names** button to add the name

**A screenshot of a computer

Description automatically generated**

The output appears as shown below:

**A screenshot of a computer

Description automatically generated**

1. Select any name and click on the **Delete** button (refer to the above screenshot) to delete any name

**A screenshot of a computer

Description automatically generated**

You have now successfully created the React Redux testing application for adding and displaying names using Jest and the testing library.