**Lesson 08 Demo 02**

**Implementing Error Handling and Loading States**

**Objective:** To implement the error handling and loading states in the React application to enhance the user experience

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

**Prerequisites:** None

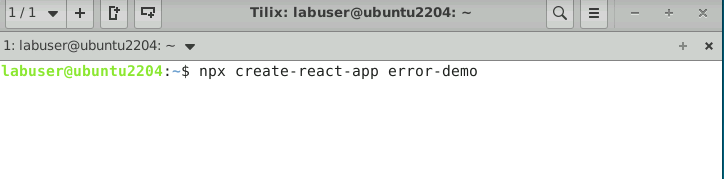
Steps to be followed:

1. Create and set up the React project
2. Modify the App.js file for error handling and run the application
3. Modify the App.js file for loading states and run the application

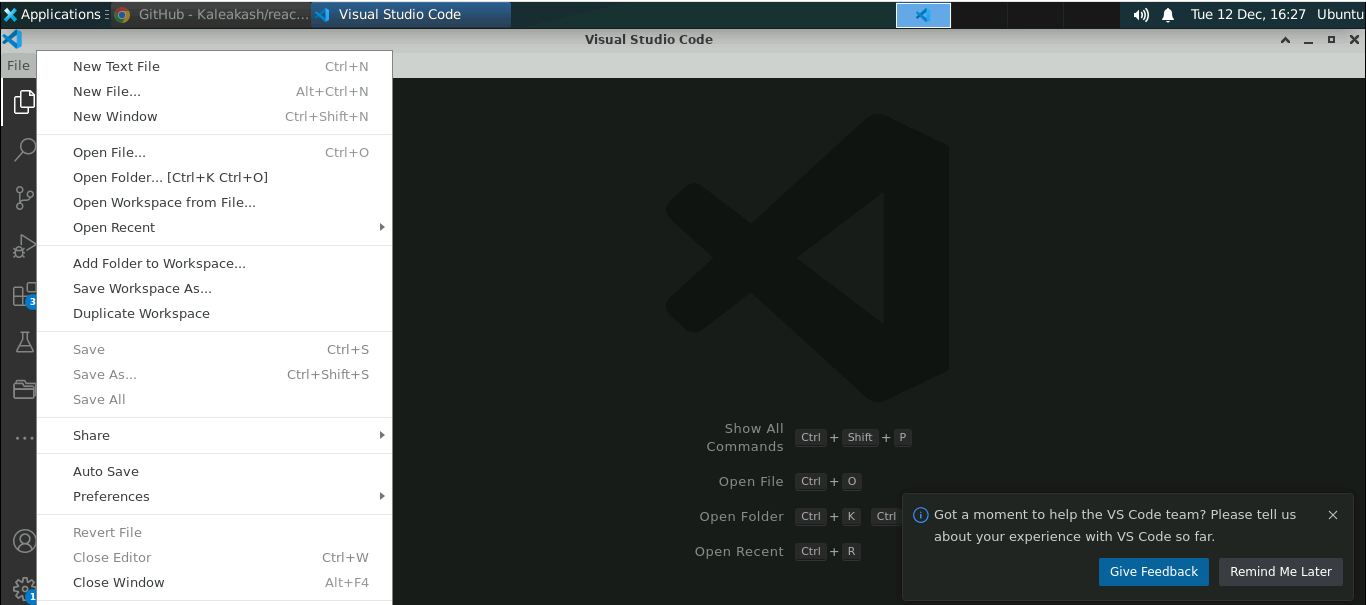
**Step 1: Create and set up the React project**

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

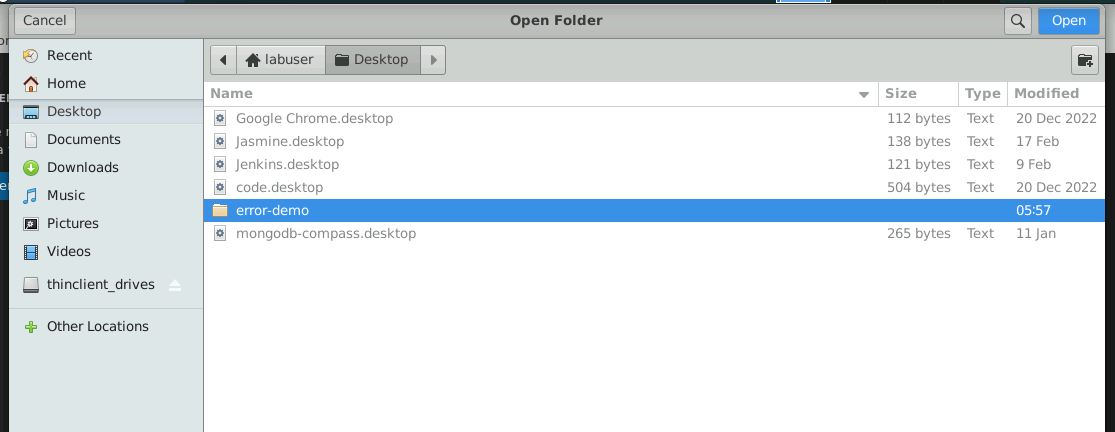
**npx create-react-app error-demo**

****

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

****

1. Click on **Open** button

****

The following project structure appears as follows:

A screenshot of a computer

Description automatically generated

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

**npm install**

**A black screen with white text

Description automatically generated**

**Note**: This command helps you install all required dependencies mentioned in the **package.json** file in the local machine as a **node\_module** folder.

**Step 2: Modify the App.js file for error handling and run the application**

* 1. Inside the **src** folder modify the **App.js** file by importing **useState** and **useEffect** hooks as shown below:

**import React, { useState, useEffect } from 'react';**

**import './App.css';**

**function App() {**

**const [error, setError] = useState(null);**

**async function fetchData() {**

**try {**

**const response = await fetch('https://jsonplaceholder.typicode.com/todos/1');**

**const jsonData = await response.json();**

**console.log(jsonData);**

**} catch (err) {**

**setError(err.message);**

**}**

**}**

**useEffect(() => {**

**fetchData();**

**}, []);**

**return (**

**<div className="App">**

**<h1>Error Demo</h1>**

**{error ? (**

**<p>Error: {error}</p>**

**) : (**

**<p>Loading...</p>**

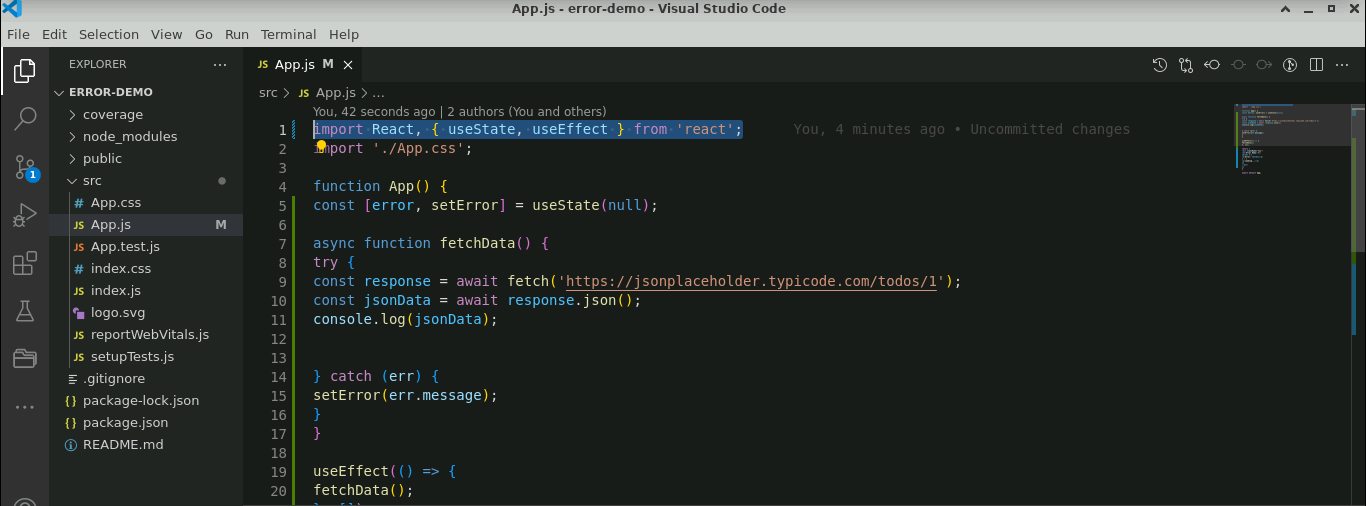
**)}**

**</div>**

**);**

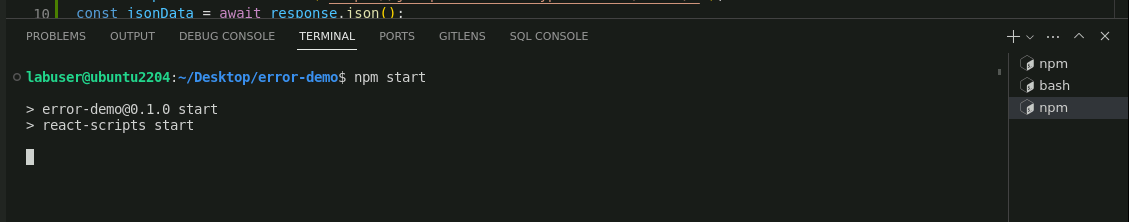
**}**

**export default App;**

****

* 1. Open the terminal and run the below command to execute the application:

**npm start**



The below screenshot shows the error handling process that displays a loading message while the APIcall is being made.

A screenshot of a computer error

Description automatically generated

**Step 3: Modify the App.js file for loading states and run the application**

* 1. Inside the **src** folder, modify the **App.js** file by importing **useState** and **useEffect** hooks as shown below:

**import React, { useState, useEffect } from 'react';**

**import './App.css';**

**function App() {**

**const [error, setError] = useState(null);**

**const [loading, setLoading] = useState(true);**

**const [data, setData] = useState(null);**

**async function fetchData() {**

**try {**

**const response = await fetch('https://jsonplaceholder.typicode.com/todos/1');**

**const jsonData = await response.json();**

**setData(jsonData);**

**} catch (err) {**

**setError(err.message);**

**} finally {**

**setLoading(false);**

**}**

**}**

**useEffect(() => {**

**fetchData();**

**}, []);**

**return (**

**<div className="App">**

**<h1>Error Demo</h1>**

**{loading ? (**

**<p>Loading...</p>**

**) : error ? (**

**<p>Error: {error}</p>**

**) : (**

**<div>**

**<p>Data:</p>**

**<pre>{JSON.stringify(data, null, 2)}</pre>**

**</div>**

**)}**

**</div>**

**);**

**}**

**export default App;**

**A screen shot of a computer

Description automatically generated**

* 1. Open the terminal and run the below command to execute the application:

**npm start**

A black screen with white text

Description automatically generated

The below screenshot shows the loading state process that displays either data or an error message while the APIcall is made.

A computer error message

Description automatically generated

With this, you have successfully implemented the error handling and loading states in API calls in the React application to enhance the user experience.