**Lesson 05 Demo 02**

**Building a Grocery App with GitHub Copilot and React with JSON Server.**

**Objective:** To build starter code for an e-commerce application using GitHub Copilot, React JS, and JSON Server to construct REST APIs, and Postman for comprehensive testing and interaction

**Tools required:** Visual Studio Code, node js, GitHub Copilot, JSON-Server, and Postman

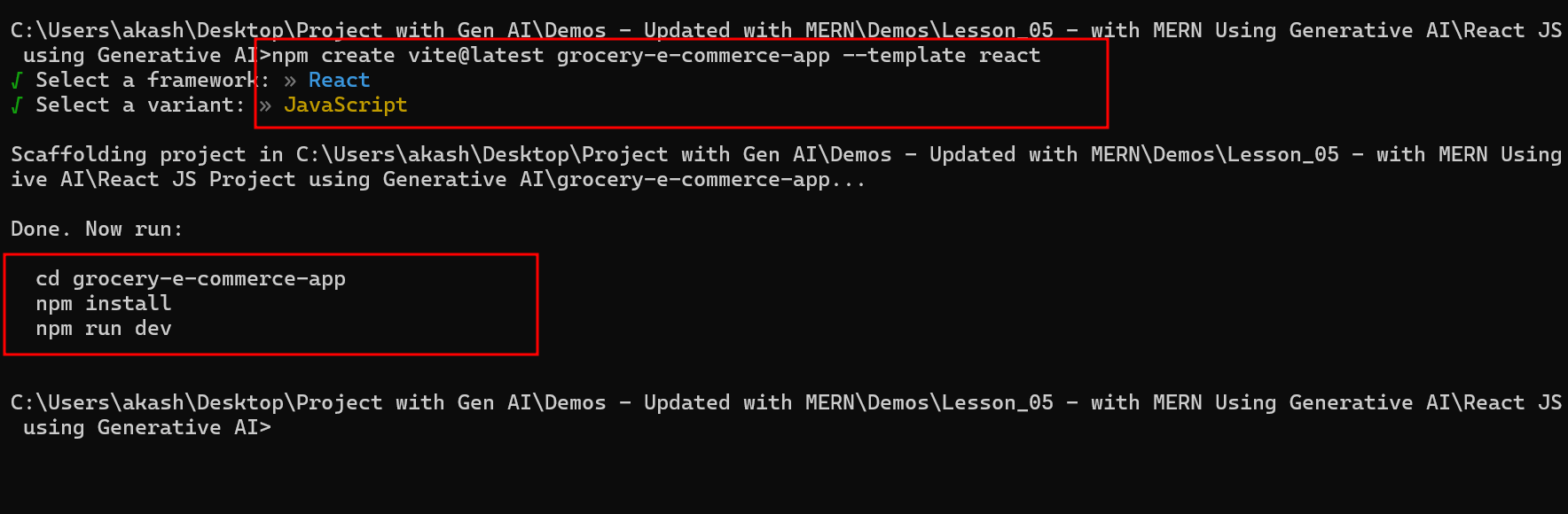
**Prerequisites:** Lesson 05 demo 01

Steps to be followed:

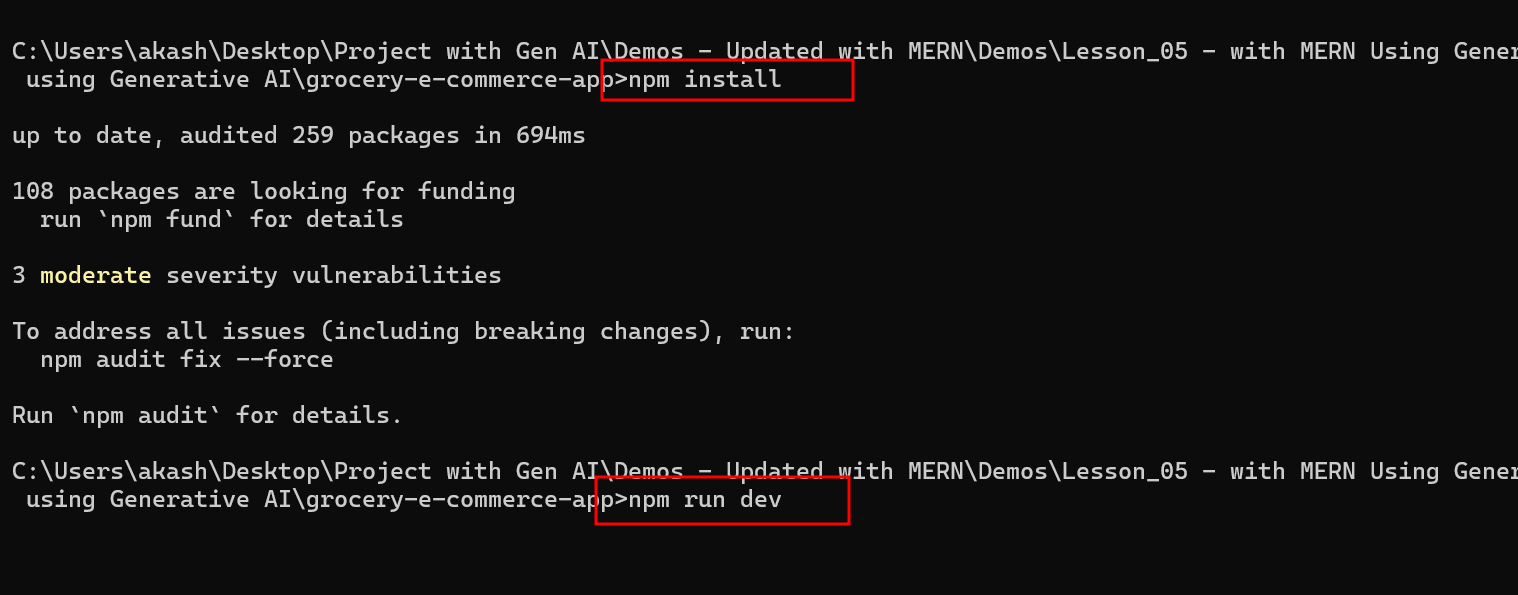
1. Follow the Demo\_01 step to create the project.
2. Create the **grocery.json** file and run this file using json-server module and insert the record using REST APIs through Postman
3. Utilize GitHub Copilot to generate the REST APIs for the grocery   
   e-commerce application. Fetch and display json data in the browser with help of react component.
4. Test the application.

**Note:** Generative AI tool used in this exercise can produce varied outputs even when presented with similar prompts. Thus, you may get different output for the same prompt

**Step 1:** **Create the React JS Project please follow the Demo1 step to create the project.**

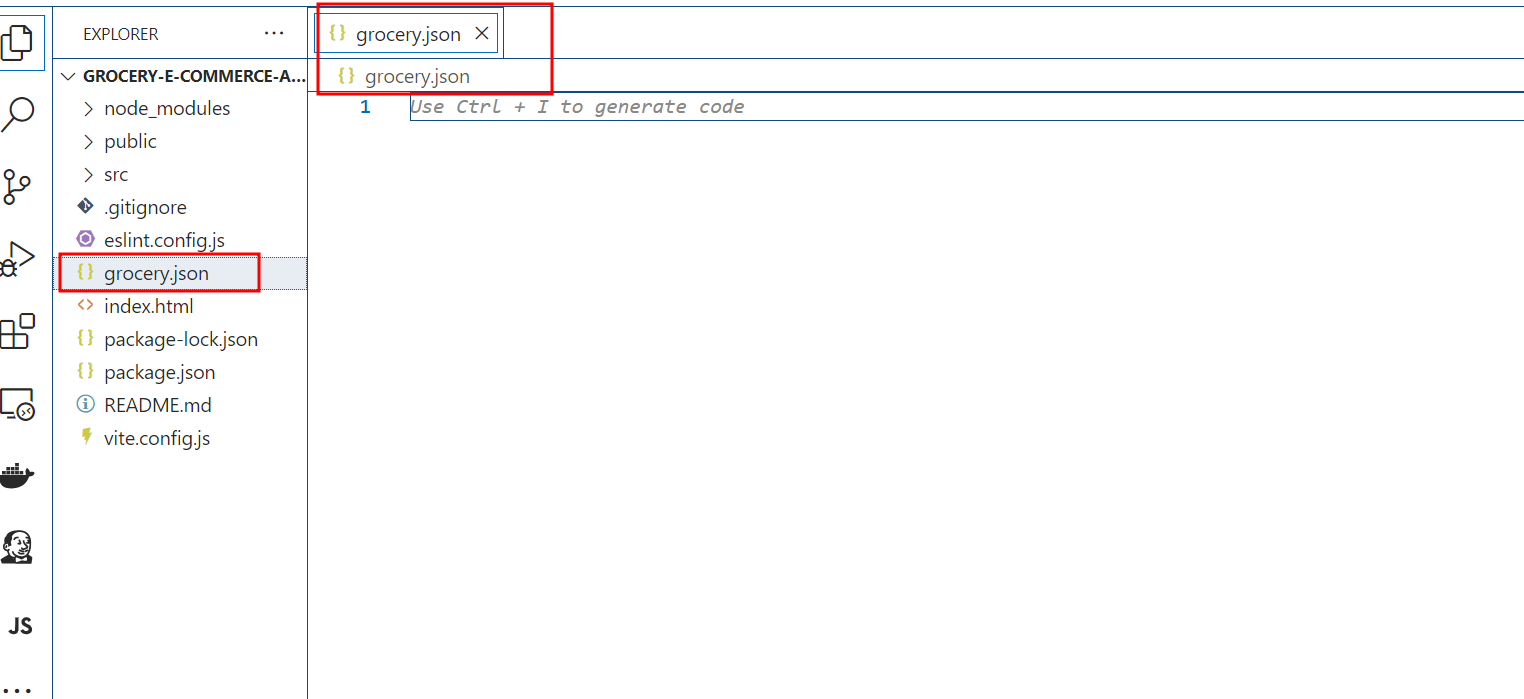


Installed all required dependencies and run the project.



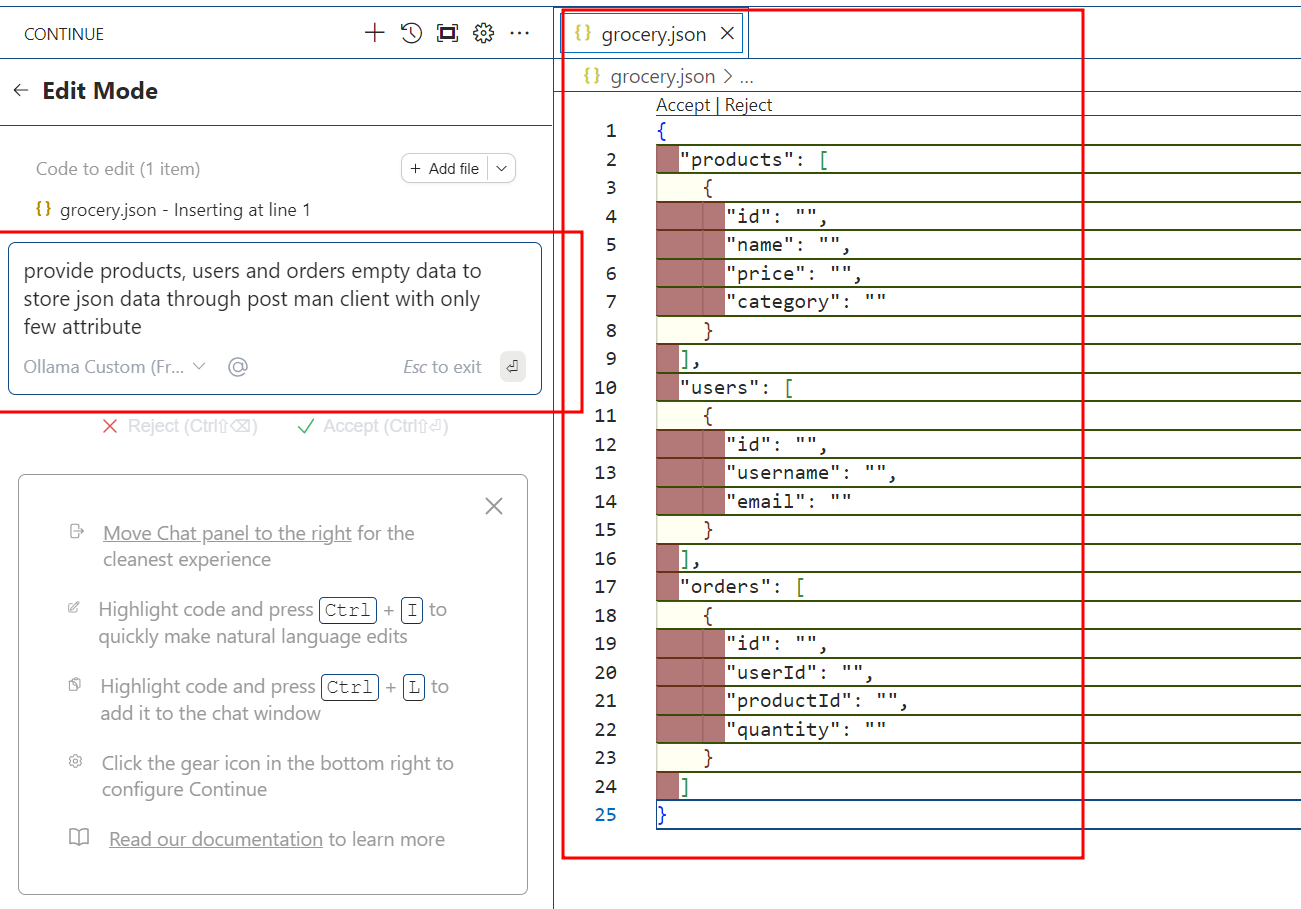
**Step 2:** **Create the grocery.json file and run this file using json-server module and insert the record using REST APIs through Postman**

2.1 create the grocery.json file

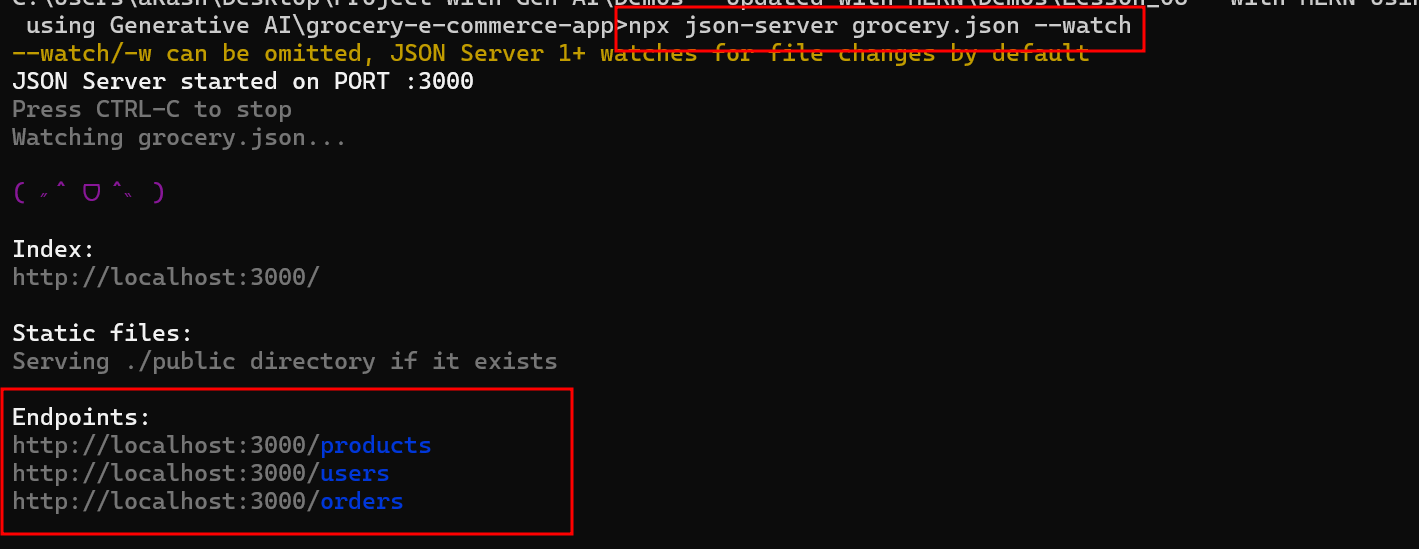


2.2 Here use Ctrl + I which open prompt text field and write below prompt message.

“provide products, users and orders empty data to store json data through post man client”

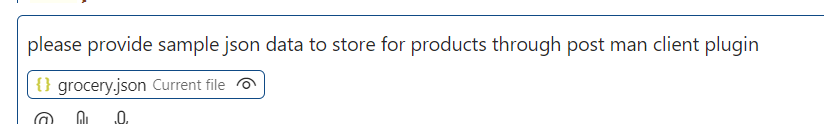


2.3 : Now open the terminal the location where grocery.json file present and run the below command to make this json file and dummy server which provide 3 rest api end point with empty data. So using Post man client we can insert few records.

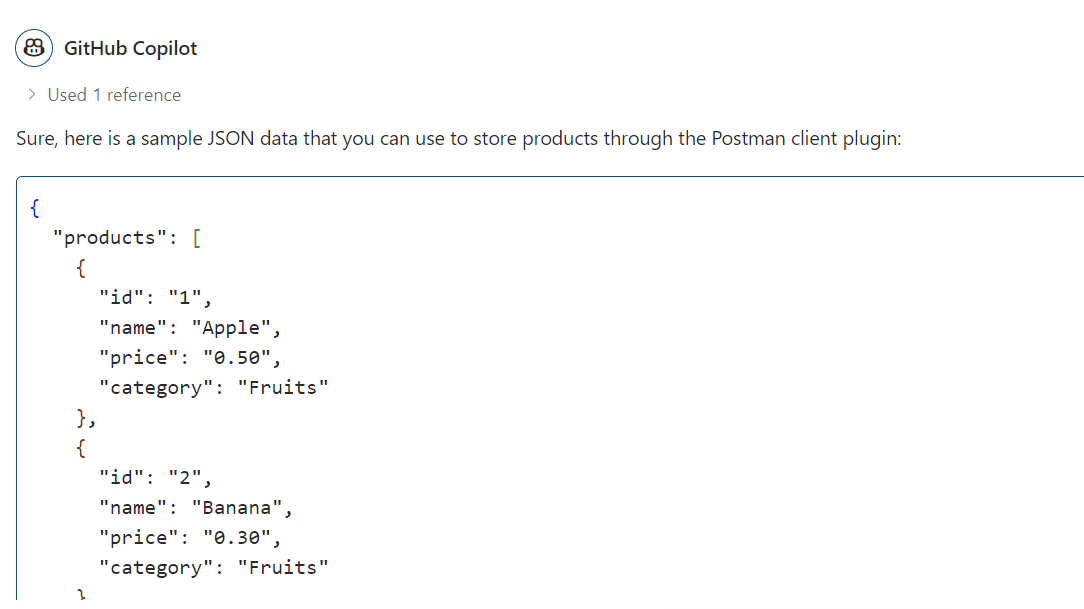


2.4 : Now open the GitHub copilot terminal and ask sample json data from products, users and orders sample data and these data you can insert through post man plugin.

“please provide sample json data to store for products through post man client plugin”

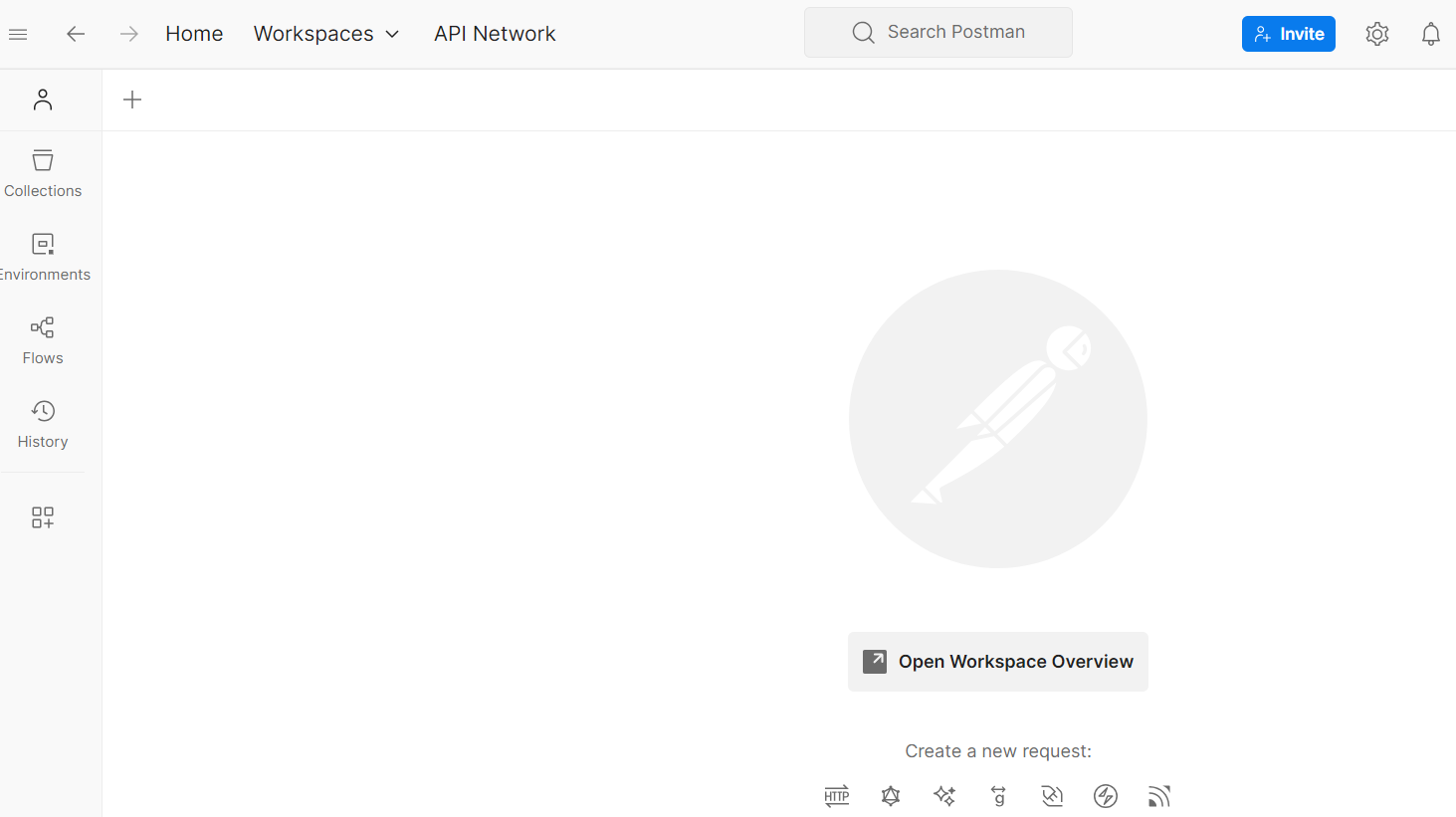


The output generated by GitHub Copilot

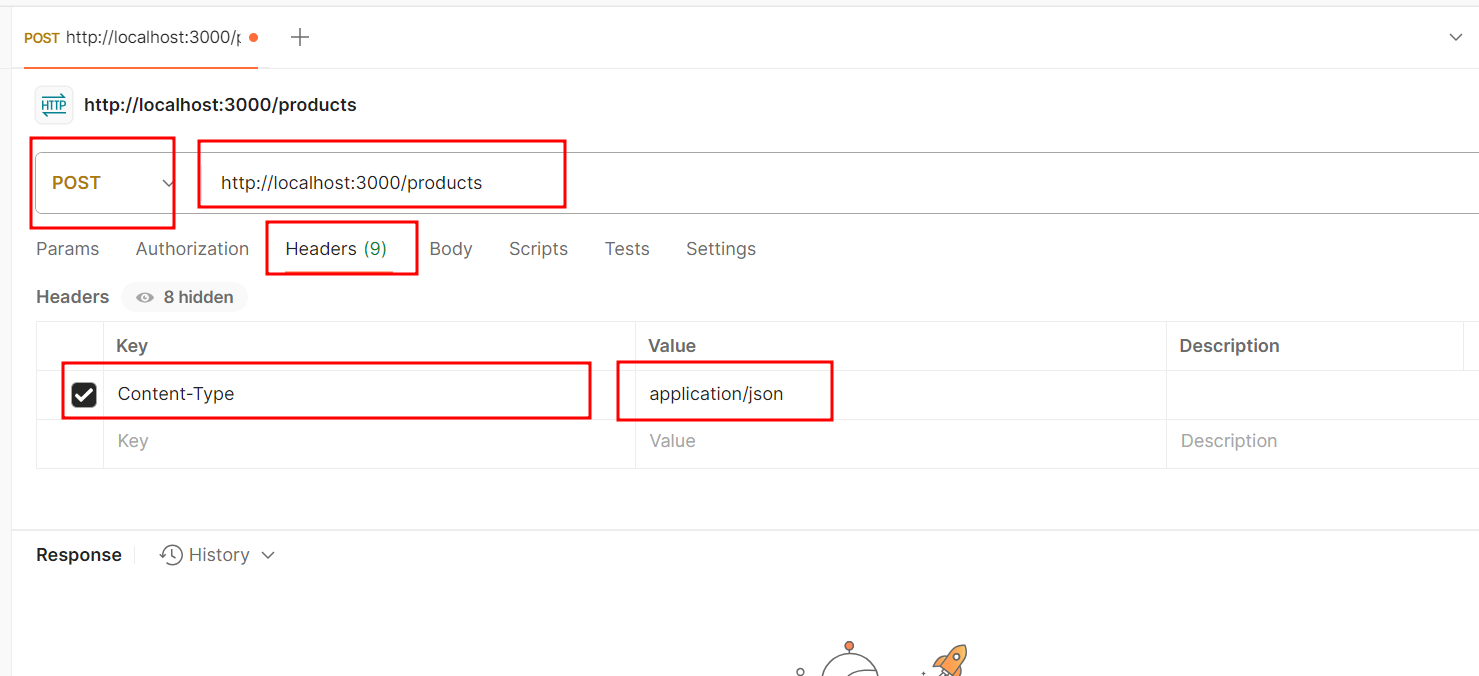


Now open the post man client. Please make sure post man client installed in your machine.

After open the post man client plugin



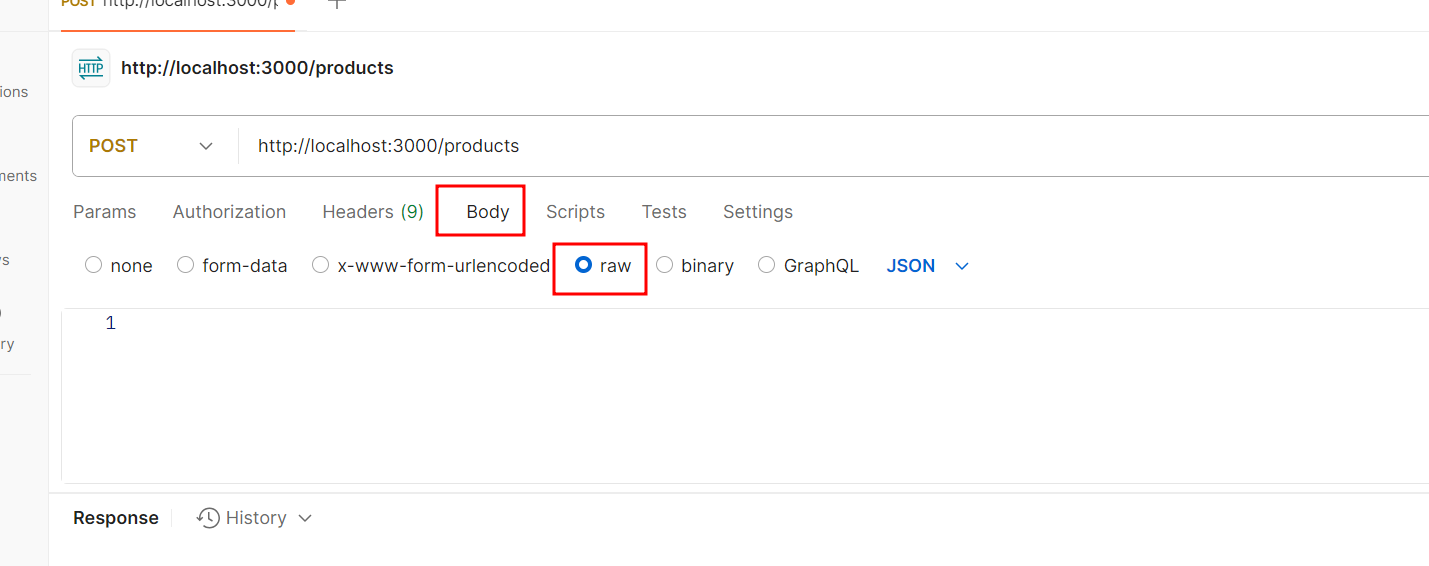
Click on + option to call end point ie rest api running through json-server module.



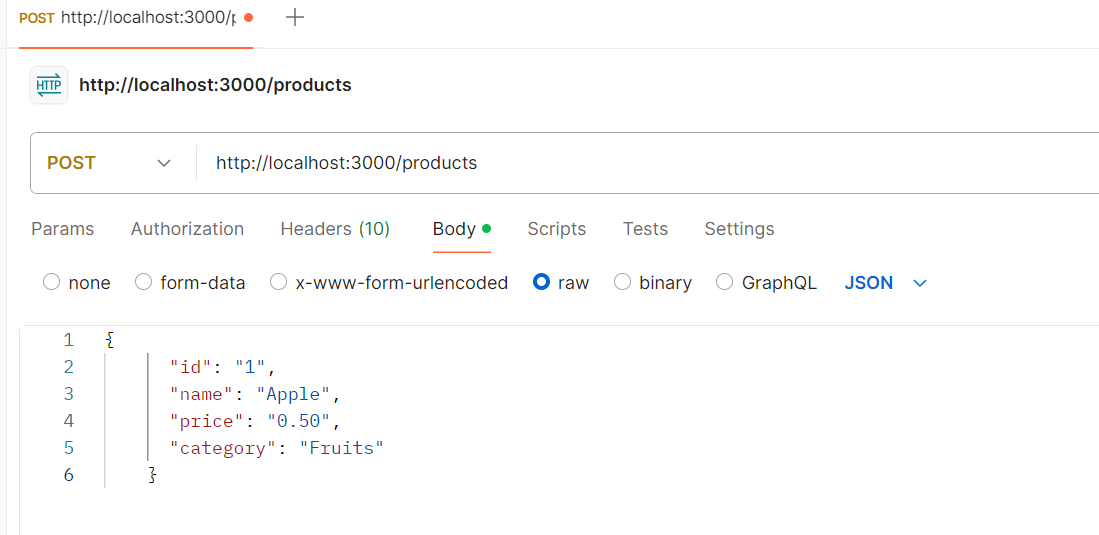
Here method is Post, URL is : <http://localhsot:3000/products>

set the header property as content-type and value is application/json

then click on body part



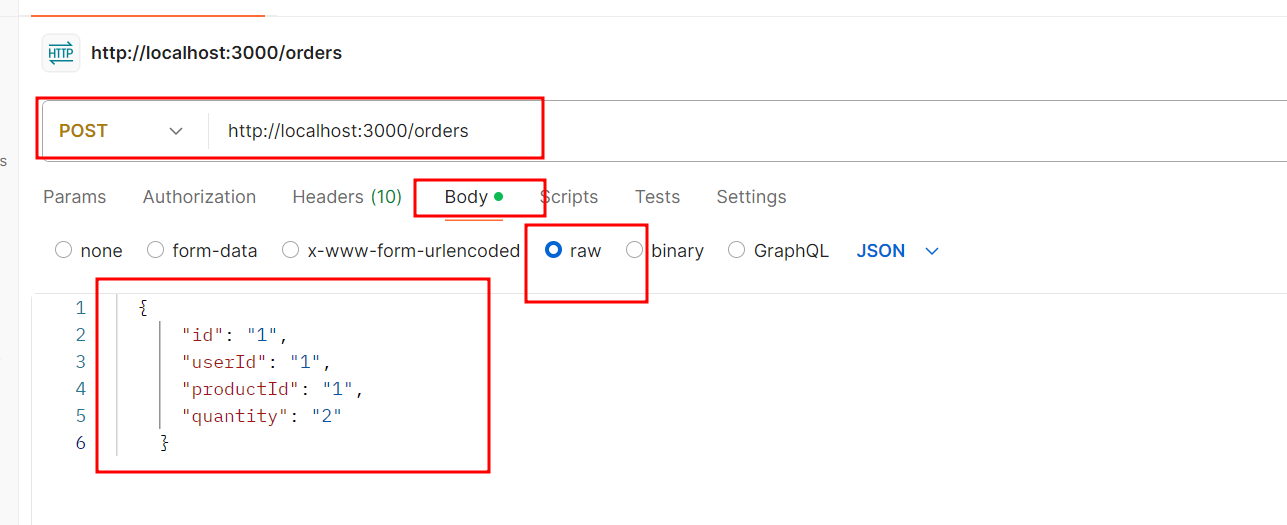
Then take sample product json data from Github copilot terminal and insert as many product details based upon requirements.



Same way stores the user details also.



Same way stores the order details.



After store products, users and orders details please check the grocery.json file store data or not.

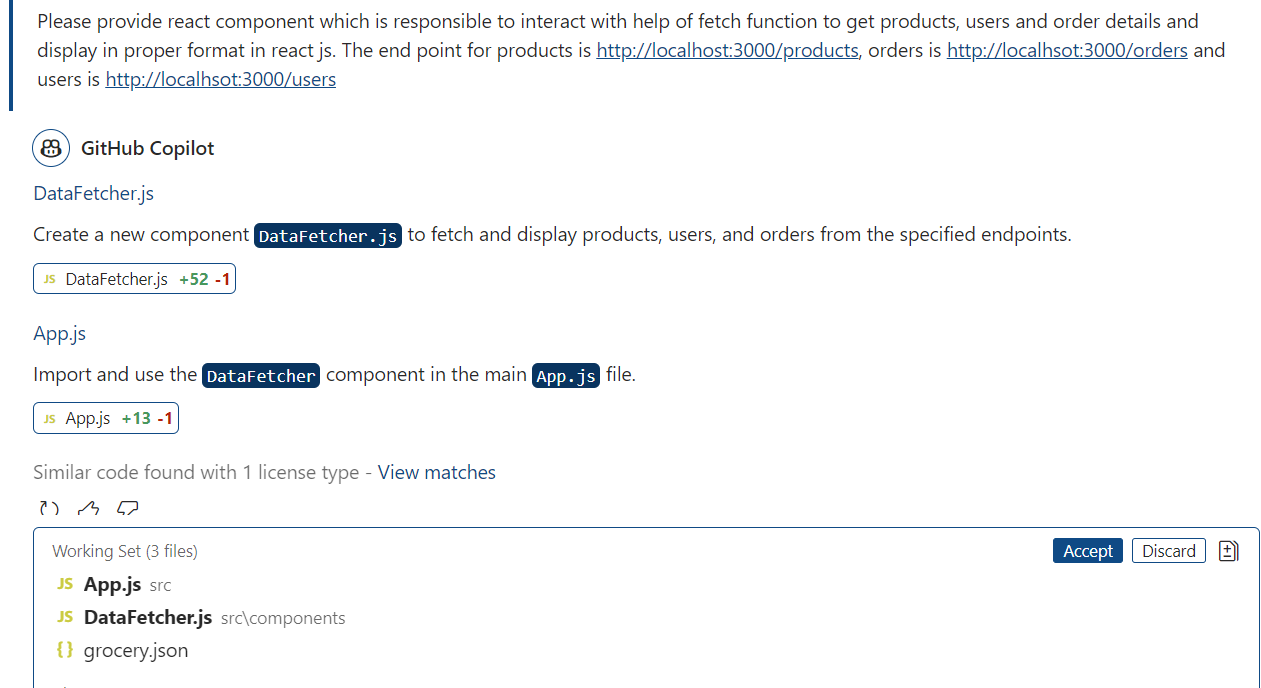
These are sample data present in file



**Step 3:** **Utilize GitHub Copilot to generate the Component which can interact with grocery api producing through json-server.**

* 1. Open the Github copilot prompt terminal and write the prompt message as

“Please provide react component which is responsible to interact with help of fetch function to get products, users and order details and display in proper format in react js. The end point for products is http://localhost:3000/products, orders is http://localhsot:3000/orders and users is http://localhsot:3000/users”



* 1. check, and copy and paste the DataFecher.js and app.js file code

**DataFetcher.jsx** code

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

const DataFetcher = () => {

  const [products, setProducts] = useState([]);

  const [users, setUsers] = useState([]);

  const [orders, setOrders] = useState([]);

  useEffect(() => {

    fetch('http://localhost:3000/products')

      .then(response => response.json())

      .then(data => setProducts(data))

      .catch(error => console.error('Error fetching products:', error));

    fetch('http://localhost:3000/users')

      .then(response => response.json())

      .then(data => setUsers(data))

      .catch(error => console.error('Error fetching users:', error));

    fetch('http://localhost:3000/orders')

      .then(response => response.json())

      .then(data => setOrders(data))

      .catch(error => console.error('Error fetching orders:', error));

  }, []);

  return (

    <div>

      <h2>Products</h2>

      <ul>

        {products.map(product => (

          <li key={product.id}>{product.name} - ${product.price} ({product.category})</li>

        ))}

      </ul>

      <h2>Users</h2>

      <ul>

        {users.map(user => (

          <li key={user.id}>{user.username} - {user.email}</li>

        ))}

      </ul>

      <h2>Orders</h2>

      <ul>

        {orders.map(order => (

          <li key={order.id}>User ID: {order.userId}, Product ID: {order.productId}, Quantity: {order.quantity}</li>

        ))}

      </ul>

    </div>

  );

};

export default DataFetcher;

**App.jsx** code

import React from 'react';

import DataFetcher from './components/DataFetcher';

function App() {

  return (

    <div className="App">

      <DataFetcher />

    </div>

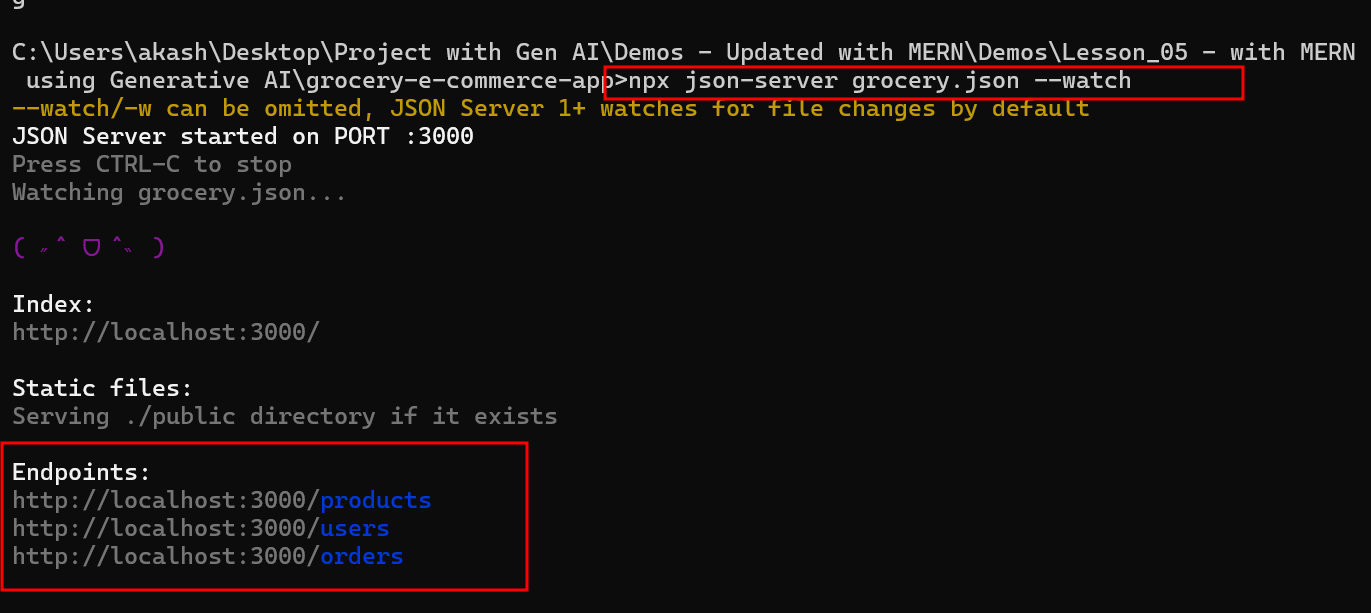
  );

}

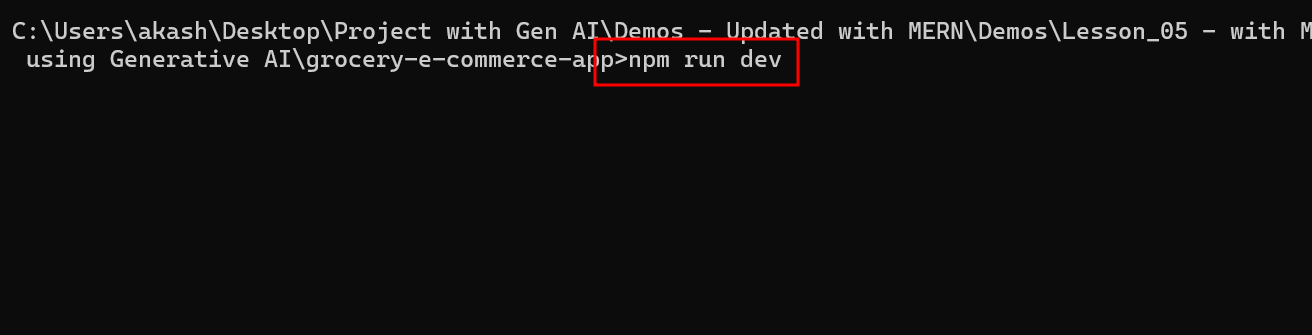
export default App;

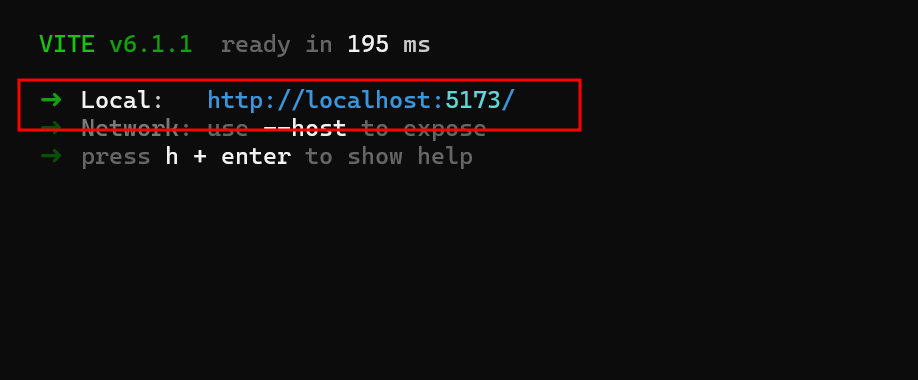
**Step 4:** Test the application using react js

4.1 grocery.json file running using json-server command>. It is like a server running and it provide rest api service using below url.



4.2 now run the react js project.





Now open these URL to test the application.

