**Lesson 04: Enhance E-Commerce Grocery Frontend Development with Generative AI**

**Overview:**

In this project, we develop into the practical application of generative AI prompts to expedite the frontend development process for a grocery e-commerce website.

By leveraging ChatGPT, we aim to streamline the creation of essential web pages such as view all products details and search the particular product based upon product id with live rest api develop using any backend technology. As a frontend developer you need to interact with backend technology using Rest API and display that information in proper format.

**Instructions:**

1. Navigate to ChatGPT and initiate the development process by providing specific prompts to create web pages
2. Create required generated web page like html, css, js code.
3. Connect all those pages to perform the task.
4. Apply styling for all web pages.
5. CSS you can use internal or external depending upon your requirements.

**Tasks:**

1. Create required project folder structure and quired pages.
   1. Index ie landing page which display appropriate header and link two pages as display all product and search the product base upon product id.
   2. View Product page responsible to display all product in proper format.
   3. Search product which contains search text file and search button. If product present it will display specific product details else product not present.
   4. productOperationAPI.js file which contains two function 1st function to view all product details and 2nd function to search the particular product using product id.
   5. Create styles.css file which contains all css rules.
2. Search required code for html, css and JS with help of ChatGPT AI tool.
   1. Code for index.html page which contains link between two pages ie searchProduct as well as viewAllProduct details.
   2. searchProduct page contains search text field with search button call function to get all product information from backend rest api.
   3. viewProduct page contains viewAllProduct button which interact with backedn rest api to get all product details.
   4. productOperationAPI.js file contains function to interact with backend rest api.
   5. <https://fakestoreapi.com/products> This URL provide all product information in json format.
   6. <https://fakestoreapi.com/products/1> This URL provide specific product details if present else return null.
3. Test the program. If need working or not. If need please debug the code.

**Tools required:**

1. ChatGPT

**Guided Practice Solution**

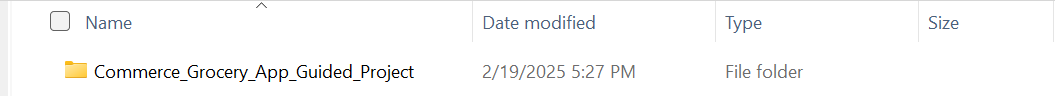
**Disclaimer:** Please note that all the GenAI tools used in this exercise can produce varied outputs even when presented with similar prompts. Thus, you may get different output for the same prompt.

**Task 1: Create a Landing page for the Grocery website using**

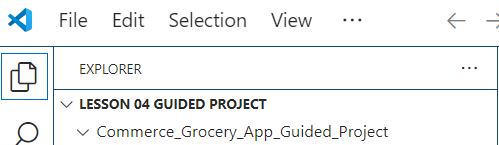
**ChatGPT**

**Step 1: create all required pages to develop the application**

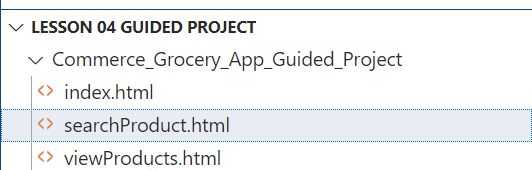
1.1Create the folder with name as E-Commerce\_Grocery\_App\_Guided\_Project



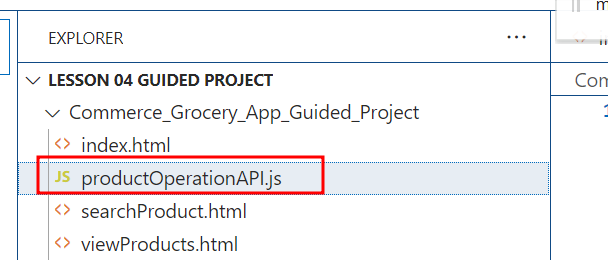
1.2 Open this folder inside a VSCode using File options with sub option as open folder.



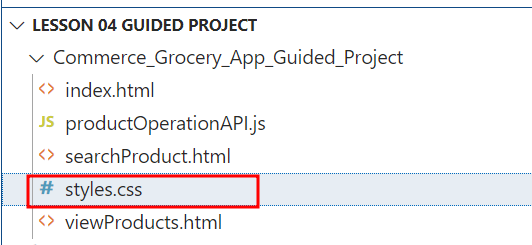
1.3 Now create index.html, viewProducts.html, searchProduct.html page.



1.4 create productOperationAPI.js file which contains function which is responsible to interact with end point to consume Rest API.



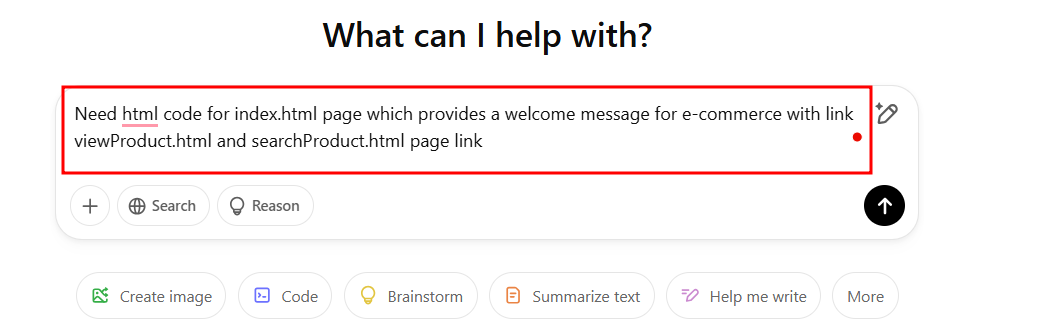
1.5 create styles.css file which contains all styling for the web page.



**Task 2: creating code for all html, css and JavaScript file.**

1. Search required code for html, css and JS with help of ChatGPT AI tool
   1. Open the Chat GPT and write the prompt message for index.html page code.

“Need html code for index.html page which provides a welcome message for e-commerce with link viewProduct.html and searchProduct.html page link ”



**index.html code**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Welcome to Our E-Commerce</title>

<style>

body {

font-family: Arial, sans-serif;

text-align: center;

margin: 50px;

}

.container {

max-width: 600px;

margin: auto;

padding: 20px;

border: 1px solid #ddd;

border-radius: 10px;

box-shadow: 2px 2px 12px rgba(0, 0, 0, 0.1);

}

a {

display: block;

margin: 10px 0;

padding: 10px;

background-color: #007bff;

color: white;

text-decoration: none;

border-radius: 5px;

}

a:hover {

background-color: #0056b3;

}

</style>

</head>

<body>

<div class="container">

<h1>Welcome to Our E-Commerce Website</h1>

<p>Explore a variety of products and find what you love!</p>

<a href="viewProducts.html">View Products</a>

<a href="searchProduct.html">Search Products</a>

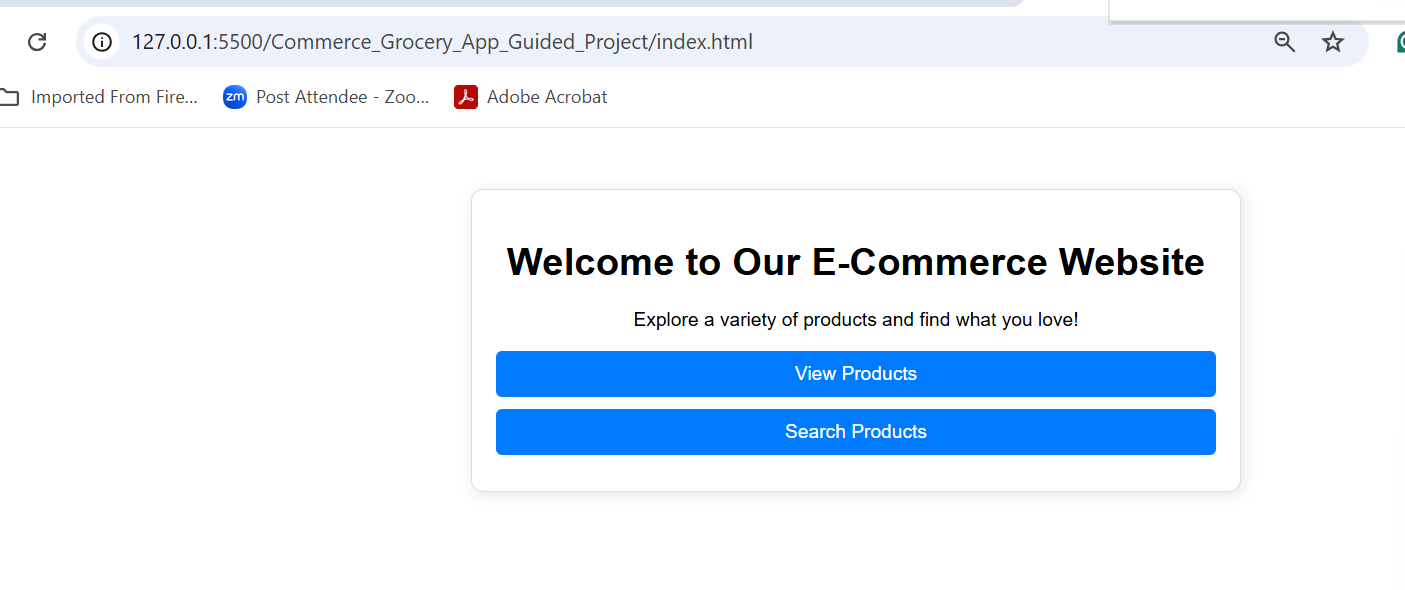
</div>

</body>

</html>

Run this page with help of VS Code Live server features.

Output for index.html page

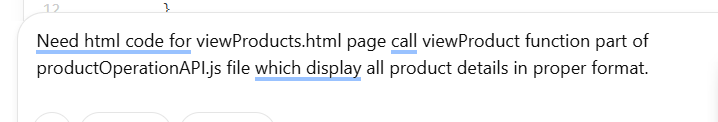


Note : Look and feel can be different depending upon the prompt message.

* 1. Open the Chat GPT and write the prompt message for viewProducts.html page code.

“Need html code for viewProducts.html page call viewProduct function part of productOperationAPI.js file which display all product details in proper format

”



**viewProducts.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>View Products</title>

<script src="productOperationAPI.js" defer></script>

<style>

body {

font-family: Arial, sans-serif;

text-align: center;

margin: 20px;

}

.product-container {

display: flex;

flex-wrap: wrap;

justify-content: center;

}

.product {

border: 1px solid #ddd;

border-radius: 10px;

padding: 15px;

margin: 10px;

width: 250px;

box-shadow: 2px 2px 12px rgba(0, 0, 0, 0.1);

}

.product img {

max-width: 100%;

border-radius: 5px;

}

</style>

</head>

<body>

<h1>Product Listings</h1>

<div class="product-container" id="productContainer"></div>

<script>

document.addEventListener("DOMContentLoaded", function() {

viewProduct();

});

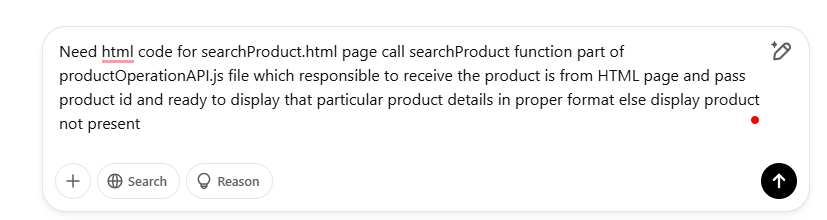
</script>

</body>

</html>

* 1. Open the Chat GPT and write the prompt message for searchProduct.html page code.

“Need html code for searchProduct.html page call searchProduct function part of productOperationAPI.js file which responsible to receive the product is from HTML page and pass product id and ready to display that particular product details in proper format else display product not present ”



<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Search Product</title>

<script src="productOperationAPI.js" defer></script>

<style>

body {

font-family: Arial, sans-serif;

text-align: center;

margin: 20px;

}

.container {

max-width: 400px;

margin: auto;

padding: 20px;

border: 1px solid #ddd;

border-radius: 10px;

box-shadow: 2px 2px 12px rgba(0, 0, 0, 0.1);

}

.product {

margin-top: 20px;

padding: 15px;

border: 1px solid #ddd;

border-radius: 10px;

box-shadow: 2px 2px 12px rgba(0, 0, 0, 0.1);

}

</style>

</head>

<body>

<div class="container">

<h1>Search Product</h1>

<input type="text" id="productId" placeholder="Enter Product ID">

<button onclick="searchProduct()">Search</button>

<div id="productDetails" class="product"></div>

</div>

<script>

function searchProduct() {

const productId = document.getElementById("productId").value;

if (productId) {

searchProductAPI(productId);

} else {

document.getElementById("productDetails").innerHTML = "<p>Please enter a product ID.</p>";

}

}

</script>

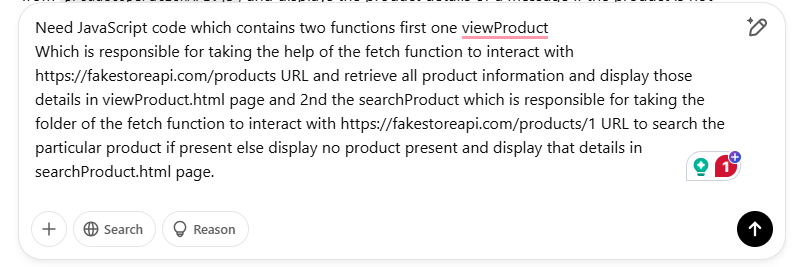
</body>

</html>

* 1. Open the Chat GPT and write the prompt message for productOperationAPI.html page code.

“Need JavaScript code which contains two functions first one viewProduct

Which is responsible to take the help of fetch function to interact with <https://fakestoreapi.com/products> URL and retrieve all product information and display those detail in viewProduct.html page and 2nd one searchProduct which is reponsbile to take the folder of fetch function to interact with <https://fakestoreapi.com/products/1> URL to search the particular product if present else display no product present and display that details in searchProduct.html page. ”



async function viewProduct() {

try {

const response = await fetch('https://fakestoreapi.com/products');

if (!response.ok) {

throw new Error('Failed to fetch product data');

}

const products = await response.json();

const productContainer = document.getElementById('productContainer');

productContainer.innerHTML = '';

products.forEach(product => {

const productElement = document.createElement('div');

productElement.classList.add('product');

productElement.innerHTML = `

<img src="${product.image}" alt="${product.title}" width="100">

<h3>${product.title}</h3>

<p>Price: $${product.price}</p>

<p>${product.description}</p>

`;

productContainer.appendChild(productElement);

});

} catch (error) {

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

}

}

async function searchProductAPI(productId) {

try {

const response = await fetch(`https://fakestoreapi.com/products/${productId}`);

if (!response.ok) {

throw new Error('Product not found');

}

const product = await response.json();

const productDetails = document.getElementById('productDetails');

productDetails.innerHTML = `

<img src="${product.image}" alt="${product.title}" width="100">

<h3>${product.title}</h3>

<p>Price: $${product.price}</p>

<p>${product.description}</p>

`;

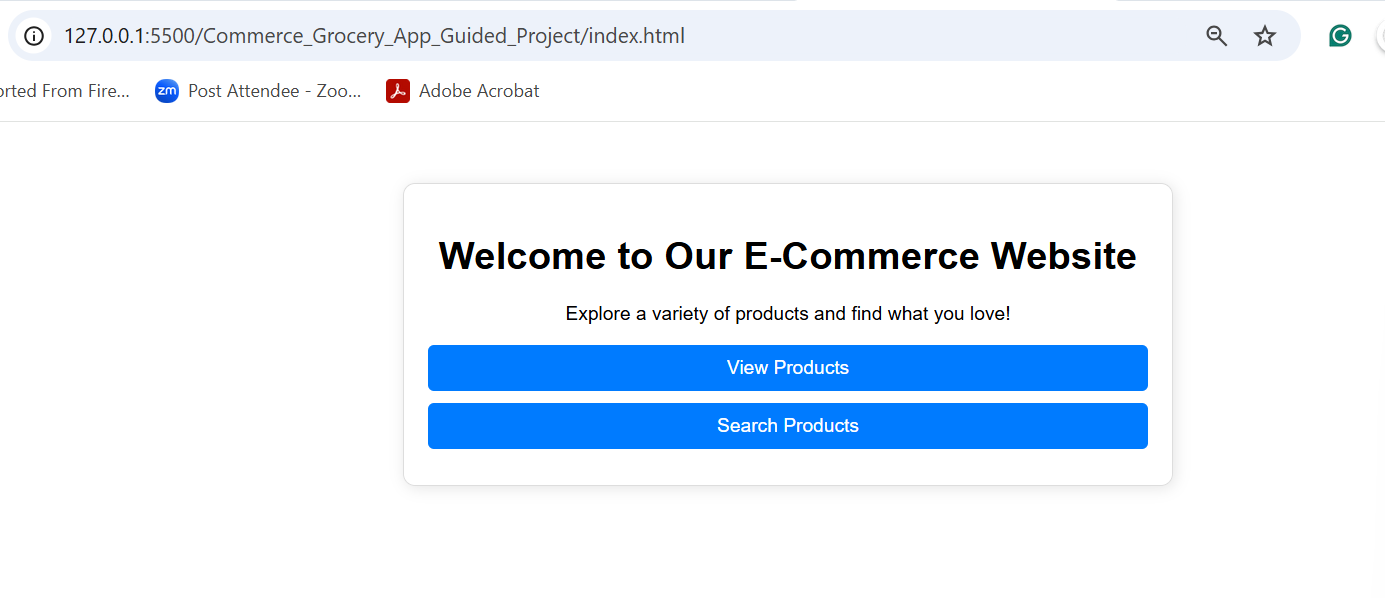
} catch (error) {

document.getElementById('productDetails').innerHTML = '<p>No product present</p>';

}

}

3. Need to test working or not. If need please debug the code.



When you click on View Product button

