

Lesson 04 Demo 06

Building and enhancing a PHP E-Commerce Website Using Generative AI

Objective: To build and deploy a PHP-based e-commerce website for groceries using generative AI for initial code generation and user interface enhancement

Tools required: ChatGPT and XAMPP

Prerequisites: NA

Steps to be followed:

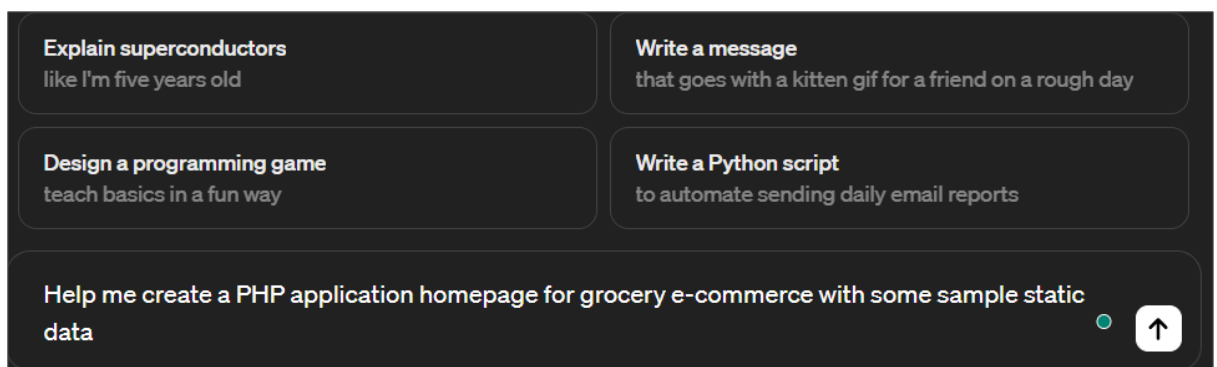
1. Create and enhance a PHP application for grocery e-commerce using Generative AI

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

Step 1: Create and enhance a PHP application for grocery e-commerce using Generative AI

- 1.1 Use the following prompt to generate a PHP application homepage:

Help me create a PHP application homepage for grocery e-commerce with some sample static data



Below is the generated PHP script from ChatGPT:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Grocery Ecommerce</title>
  <style>
    /* Add your CSS styles here */
  </style>
</head>
<body>
  <header>
    <h1>Welcome to our Grocery Ecommerce Store</h1>
    <nav>
      <ul>
        <li><a href="#">Home</a></li>
        <li><a href="#">Shop</a></li>
        <li><a href="#">Contact</a></li>
      </ul>
    </nav>
  </header>

  <main>
    <section class="featured-products">
      <h2>Featured Products</h2>
      <div class="product-list">
        <?php
          // Sample static data for featured products
          $featured_products = [
            ["name" => "Bananas", "price" => "$0.50", "image" => "banana.jpg"],
            ["name" => "Apples", "price" => "$1.00", "image" => "apple.jpg"],
            ["name" => "Oranges", "price" => "$0.75", "image" => "orange.jpg"]
          ];

          // Loop through each product and display it
          foreach ($featured_products as $product) {
            echo "<div class='product'>";
            echo "<img src='images/" . $product['image'] . "' alt='" .
$product['name'] . "'>";
            echo "<h3>" . $product['name'] . "</h3>";
            echo "<p>Price: " . $product['price'] . "</p>";
            echo "<button>Add to Cart</button>";
            echo "</div>";
          }
        </?php>
      </div>
    </section>
  </main>
</body>
</html>
```

```


        ?>
    </div>
</section>
</main>

<footer>
    <p>&copy; <?php echo date("Y"); ?> Grocery Ecommerce</p>
</footer>
</body>
</html>

```

Certainly! Below is a PHP script for the homepage of a grocery e-commerce application with some sample static data:

php

 Copy code

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Grocery Ecommerce</title>
    <style>
        /* Add your CSS styles here */
    </style>
</head>
<body>
    <header>
        <h1>Welcome to our Grocery E-commerce Store</h1>
        <nav>


```

```
<h1>Welcome to our Grocery Ecommerce Store</h1>
<nav>
  <ul>
    <li><a href="#">Home</a></li>
    <li><a href="#">Shop</a></li>
    <li><a href="#">Contact</a></li>
  </ul>
</nav>
</header>

<main>
  <section class="featured-products">
    <h2>Featured Products</h2>
    <div class="product-list">
      <?php
        // Sample static data for featured products
        $featured_products = [
          ["name" => "Bananas", "price" => "$0.50", "image" => "banana.jpg"],
          ["name" => "Apples", "price" => "$1.00", "image" => "apple.jpg"],
          ["name" => "Oranges", "price" => "$0.75", "image" => "orange.jpg"]
        ];

        // Loop through each product and display it
        foreach ($featured_products as $product) {
          echo "<div class='product'>";
```


Below is the generated folder structure:

 **ChatGPT**

Sure, here's the folder structure for the codebase:

CSS

Copy code

```
myGrocery/  
├── css/  
│   └── style.css  
├── images/  
│   ├── banana.jpg  
│   ├── apple.jpg  
│   └── orange.jpg  
├── includes/  
│   └── header.php  
├── index.php  
└── footer.php
```

This structure organizes the code into separate directories for CSS, images, includes, and the main PHP files. It helps keep the codebase tidy and makes it easier to locate specific files when working on the project.

Below is the explanation for the folder structure:

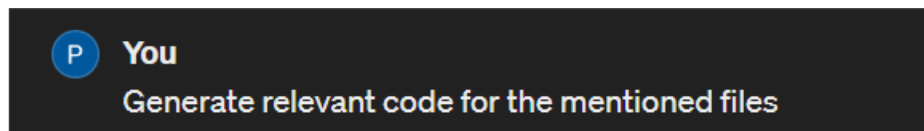
- **css/**: This directory contains the CSS files for styling the application. You can have a `style.css` file here.
- **images/**: Store your product images in this directory.
- **includes/**: This directory contains PHP files that are included in multiple pages. In this example, we have a `header.php` file which contains the header section of the HTML.
- **index.php**: This is the main PHP file for the homepage of your application. It contains the HTML structure along with PHP code for displaying dynamic content.
- **footer.php**: This file contains the footer section of the HTML, which can be included in multiple pages for consistency.

1.3 Create a folder on the local drive based on the generated structure and name the project as **myGrocery**:

Name	Date modified	Type	Size
css	06-05-2024, Mon 12:03 PM	File folder	
images	06-05-2024, Mon 11:58 AM	File folder	
includes	06-05-2024, Mon 11:54 AM	File folder	
footer.php	06-05-2024, Mon 11:40 AM	PHP Source File	1 KB
index.php	06-05-2024, Mon 11:40 AM	PHP Source File	2 KB

1.4 Use the following prompt to generate script for **index.php**, **style.css**, **header.php**, and **footer.php** files:

Generate relevant code for the mentioned files



Below is the script; copy and save it in the relevant files per the directory structure:

```
../myGrocery/css/style.css
/* CSS styles for the grocery ecommerce application */
/* Add your CSS styles here */
header {
  background-color: #f8f9fa;
  padding: 20px;
  text-align: center;
}

nav ul {
  list-style-type: none;
  padding: 0;
}

nav ul li {
  display: inline;
  margin-right: 20px;
}

.product {
  display: inline-block;
  margin: 20px;
}
```

```
.product img {
    width: 200px;
    height: 200px;
}
```

```
footer {
    background-color: #343a40;
    color: #ffffff;
    padding: 20px;
    text-align: center;
}
```

```
../myGrocery/includes/header.php
```

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
    <meta charset="UTF-8">
```

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

```
    <title>Grocery Ecommerce</title>
```

```
    <link rel="stylesheet" href="css/style.css">
```

```
</head>
```

```
<body>
```

```
    <header>
```

```
        <h1>Welcome to our Grocery Ecommerce Store</h1>
```

```
        <nav>
```

```
            <ul>
```

```
                <li><a href="#">Home</a></li>
```

```
                <li><a href="#">Shop</a></li>
```

```
                <li><a href="#">Contact</a></li>
```

```
            </ul>
```

```
        </nav>
```

```
    </header>
```

```
../myGrocery/index.php
```

```
<?php include 'includes/header.php'; ?>
```

```
<main>
```

```
    <section class="featured-products">
```

```
        <h2>Featured Products</h2>
```

```
        <div class="product-list">
```

```
            <?php
```

```
            // Sample static data for featured products
```

```
            $featured_products = [
```

```
                ["name" => "Bananas", "price" => "$0.50", "image" => "banana.jpg"],
```

```
                ["name" => "Apples", "price" => "$1.00", "image" => "apple.jpg"],
```

```
                ["name" => "Oranges", "price" => "$0.75", "image" => "orange.jpg"]
```

```
            ];
```



```

        // Loop through each product and display it
        foreach ($featured_products as $product) {
            echo "<div class='product'>";
            echo "<img src='images/" . $product['image'] . "' alt='" . $product['name']
. "'>";

            echo "<h3>" . $product['name'] . "</h3>";
            echo "<p>Price: " . $product['price'] . "</p>";
            echo "<button>Add to Cart</button>";
            echo "</div>";
        }
    ?>
</div>
</section>
</main>
<?php include 'footer.php'; ?>
../myGrocery/footer.php
<footer>
    <p>&copy; <?php echo date("Y"); ?> Grocery Ecommerce</p>
</footer>
</body>
</html>

```

Here's the relevant code for the mentioned files:

CSS styles (style.css):

```

CSS
Copy code

/* CSS styles for the grocery ecommerce application */
/* Add your CSS styles here */
header {
    background-color: #f8f9fa;
    padding: 20px;
    text-align: center;
}

nav ul {
    list-style-type: none;
    padding: 0;
}

nav ul li {
    display: inline;
    margin-right: 20px;
}

```

```

.product {
    display: inline-block;
    margin: 20px;
}


.product img {
    width: 200px;
    height: 200px;
}

footer {
    background-color: #343a40;
    color: #ffffff;
    padding: 20px;
    text-align: center;
}

```

Header file (header.php):

php

 Copy code

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Grocery Ecommerce</title>
    <link rel="stylesheet" href="css/style.css">
</head>
<body>
    <header>
        <h1>Welcome to our Grocery Ecommerce Store</h1>
        <nav>
            <ul>
                <li><a href="#">Home</a></li>
                <li><a href="#">Shop</a></li>
                <li><a href="#">Contact</a></li>
            </ul>
        </nav>
    </header>

```



Index file (index.php):

php

Copy code

```
<?php include 'includes/header.php'; ?>

<main>
    <section class="featured-products">
        <h2>Featured Products</h2>
        <div class="product-list">
            <?php
                // Sample static data for featured products
                $featured_products = [
                    ["name" => "Bananas", "price" => "$0.50", "image" => "banana.jpg"],
                    ["name" => "Apples", "price" => "$1.00", "image" => "apple.jpg"],
                    ["name" => "Oranges", "price" => "$0.75", "image" => "orange.jpg"]
                ];

                // Loop through each product and display it
                foreach ($featured_products as $product) {
                    echo "<div class='product'>";
                    echo "<img src='images/" . $product['image'] . "' alt='" . $product['name'] . "'>";
                    echo "<h3>" . $product['name'] . "</h3>";
                    echo "<p>Price: " . $product['price'] . "</p>";
                    echo "<button>Add to Cart</button>";

                    echo "</div>";

                }
            ?>
        </div>
    </section>
</main>

<?php include 'footer.php'; ?>

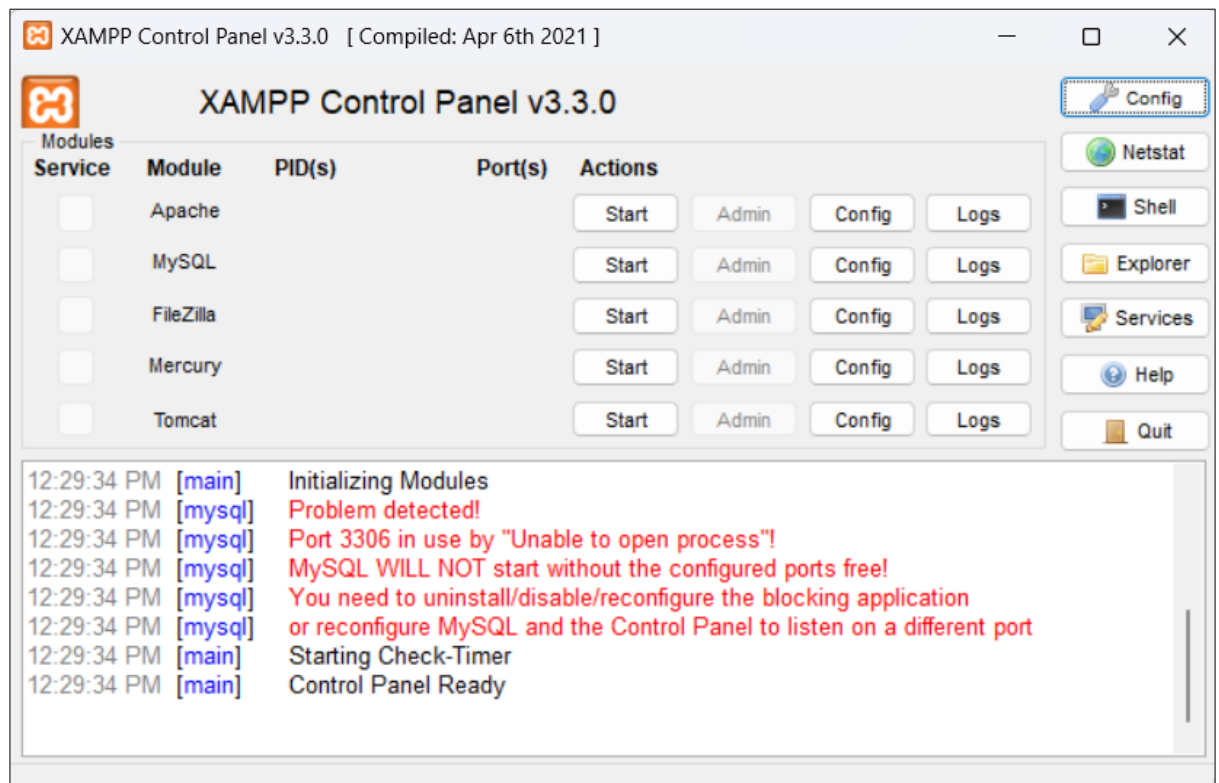
<footer>
    <p>&copy; <?php echo date("Y"); ?> Grocery Ecommerce</p>
</footer>

</body>
</html>
```

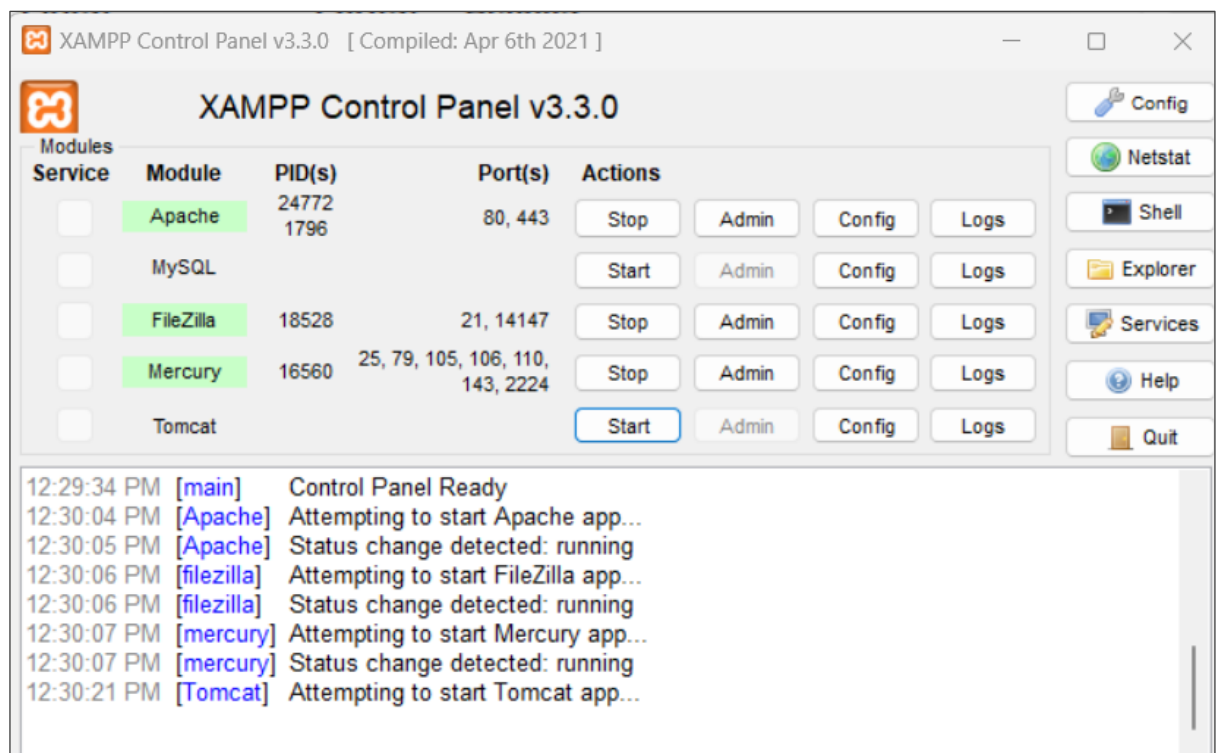
1.5 Download and install XAMPP from the link below:

<https://www.apachefriends.org/download.html>

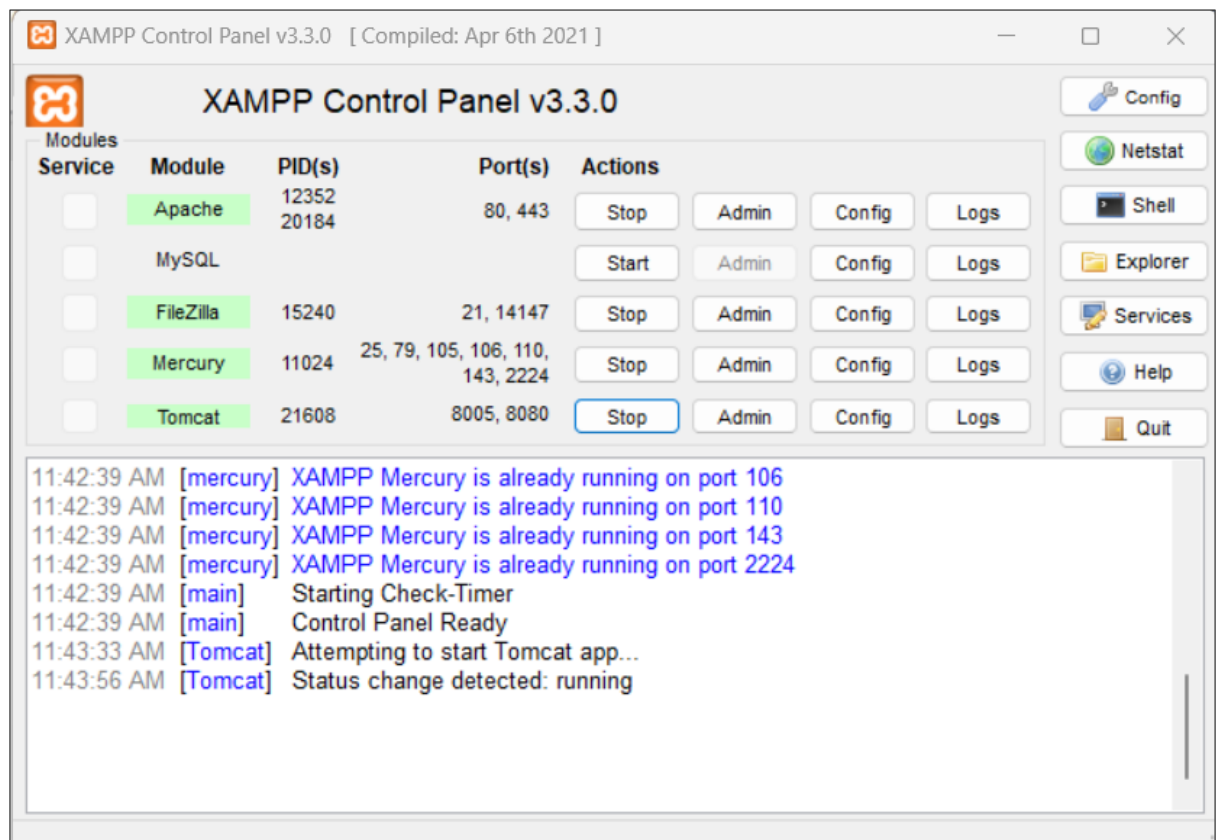
Once installed, run the **XAMPP control panel** as follows:



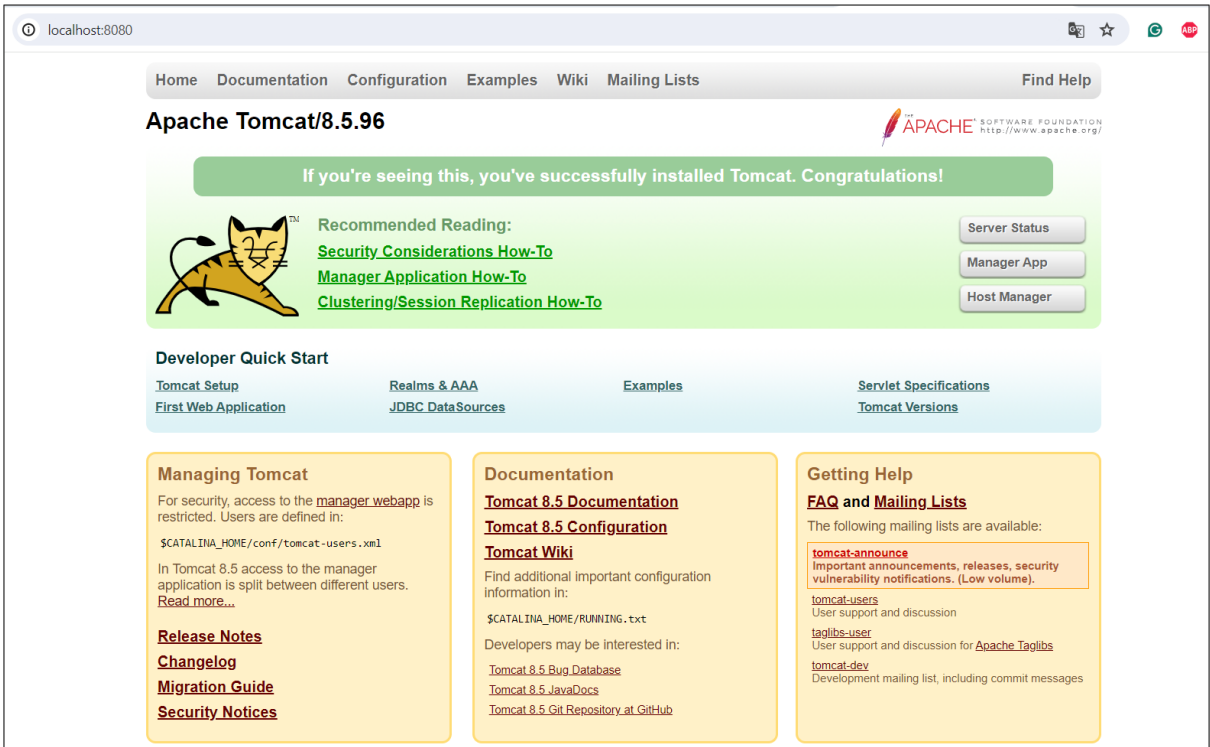
Start the required services, as shown in the screenshot below:



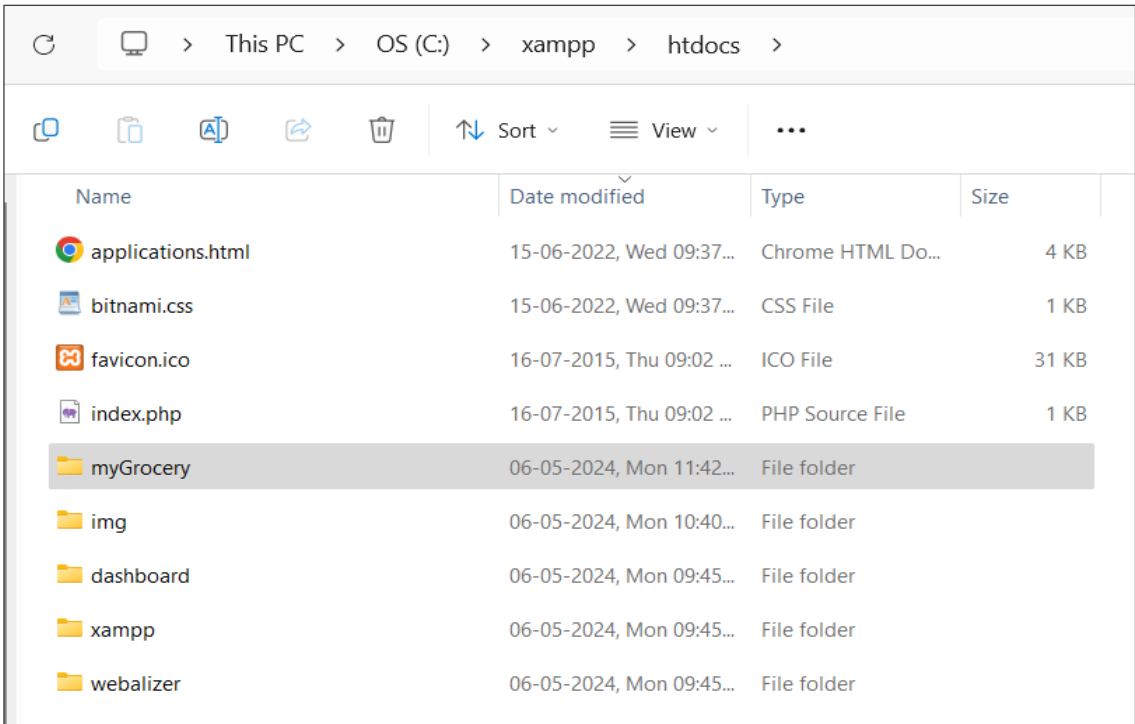
- 1.6 Start Tomcat in the **XAMPP control panel** and open the browser to test it using **http://localhost:8080/**; it checks if the local server is running on the computer.



Apache Tomcat appears if the service is running successfully, as shown in the screenshot below:

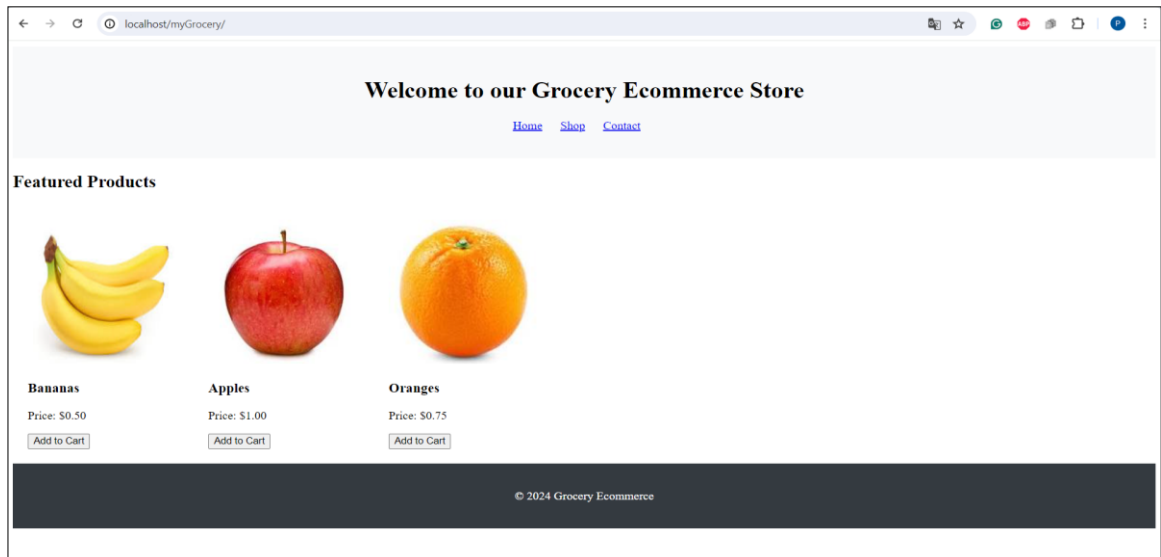


1.7 Copy the **myGrocery** folder and paste it into the **../xampp/htdocs/** directory. In this case, it is **C:/xampp/htdocs** as the folder is located on the C drive.



- 1.8 Open the browser and run the project **myGrocery** by entering the following URL:
http://localhost/myGrocery/

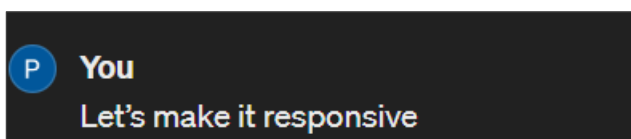
Below screenshot is the grocery e-commerce application:



Note: Keep the XAMPP application open with the Tomcat service running continuously. Avoid closing the localhost browser to facilitate quick checking of script changes by simply refreshing the page instead of reopening XAMPP and restarting the service.

Note: Download images of an apple, banana, and orange from a website and rename them as apple.jpg, banana.jpg, and orange.jpg. Then save these files in the **../xampp/htdocs/myGrocery/images/** folder.

- 1.9 Use the following prompt in ChatGPT to generate a responsive CSS script:
Let's make it responsive



- 1.10 Copy the script and replace the existing script in the file `../xampp/htdocs/myGrocery/css/style.css` and save it.

```
/* CSS styles for the grocery ecommerce application */
/* Add your CSS styles here */

/* Header */
header {
    background-color: #f8f9fa;
    padding: 20px;
    text-align: center;
}

nav ul {
    list-style-type: none;
    padding: 0;
}

nav ul li {
    display: inline;
    margin-right: 20px;
}

/* Main */
main {
    padding: 20px;
}

.featured-products {
    margin-bottom: 40px;
}

.product {
    display: inline-block;
    margin: 20px;
    text-align: center;
}

.product img {
    width: 100%;
    max-width: 200px;
    height: auto;
}

/* Footer */
footer {
```



```

background-color: #343a40;
color: #ffffff;
padding: 20px;
text-align: center;
}

/* Responsive Styles */
@media screen and (max-width: 768px) {
  nav ul li {
    display: block;
    margin-bottom: 10px;
  }

  .product {
    width: 45%;
    margin: 2.5%;
  }
}

```



ChatGPT

To make the grocery e-commerce application responsive, we need to adjust the CSS styles to ensure that the layout and elements adapt well to different screen sizes. Here's the updated CSS code with responsive styles:

CSS

Copy code

```

/* CSS styles for the grocery ecommerce application */
/* Add your CSS styles here */

/* Header */
header {
  background-color: #f8f9fa;
  padding: 20px;
  text-align: center;
}

nav ul {
  list-style-type: none;
  padding: 0;
}

```



```
}

nav ul li {
  display: inline;
  margin-right: 20px;
}

/* Main */
main {
  padding: 20px;
}

.featured-products {
  margin-bottom: 40px;
}

.product {
  display: inline-block;
  margin: 20px;
  text-align: center;
}

.product img {
  width: 100%;
  height: 100%;
```




```
    max-width: 200px;
    height: auto;
}

/* Footer */
footer {
    background-color: #343a40;
    color: #ffffff;
    padding: 20px;
    text-align: center;
}

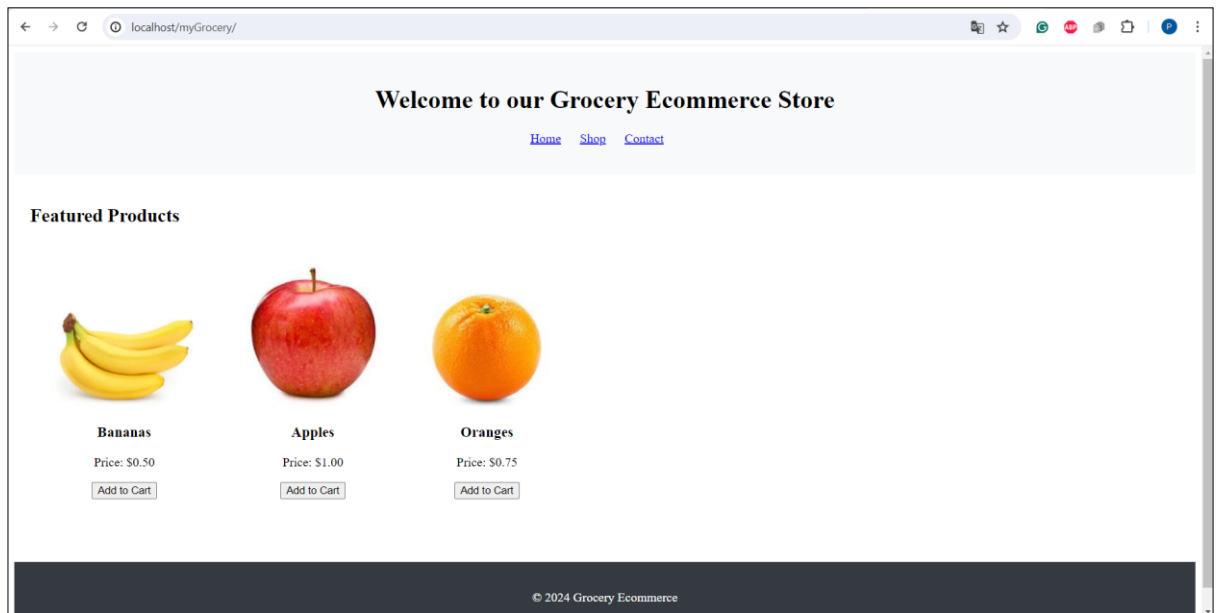
/* Responsive Styles */
@media screen and (max-width: 768px) {
    nav ul li {
        display: block;
        margin-bottom: 10px;
    }

    .product {
        width: 45%;
        margin: 2.5%;
    }
}
```



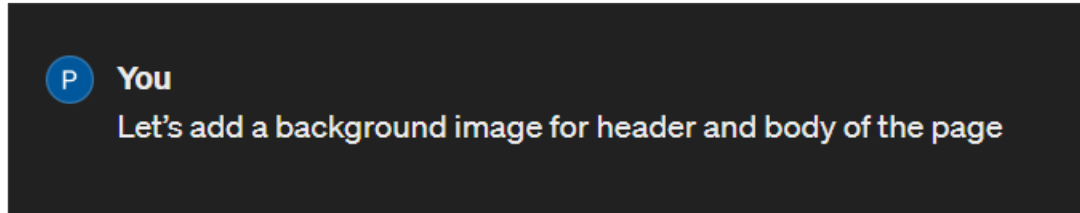
These responsive styles will adjust the layout and sizing of elements such as navigation links and product images to provide a better user experience on smaller screens, such as tablets and smartphones.

1.11 Refresh the page to see the dynamic changes. Below is the screenshot of the homepage:



1.12 Use the following prompt to enhance the application by adding background and body images to the page:

Let's add a background image for header and body of the page



Note: Download header and body background images from the website and rename them as **header-background.jpg** and **body-background.jpg**. Then, save these files in the **../xampp/htdocs/myGrocery/images/** folder.

- 1.13 Copy the below script and replace the existing one in
../xampp/htdocs/myGrocery/css/style.css

```
/* CSS styles for the grocery ecommerce application */
/* Add your CSS styles here */

/* Header */
header {
    background-image: url('../images/header-background.jpg');
    background-size: cover;
    background-position: center;
    padding: 20px;
    text-align: center;
    color: #ffffff; /* Adjust text color for better readability */
}

nav ul {
    list-style-type: none;
    padding: 0;
}

nav ul li {
    display: inline;
    margin-right: 20px;
}

/* Main */
main {
    background-image: url('../images/body-background.jpg');
    background-size: cover;
    background-position: center;
    padding: 20px;
    color: #333333; /* Adjust text color for better readability */
}

.featured-products {
    margin-bottom: 40px;
}

.product {
    display: inline-block;
    margin: 20px;
    text-align: center;
}

.product img {
```

```

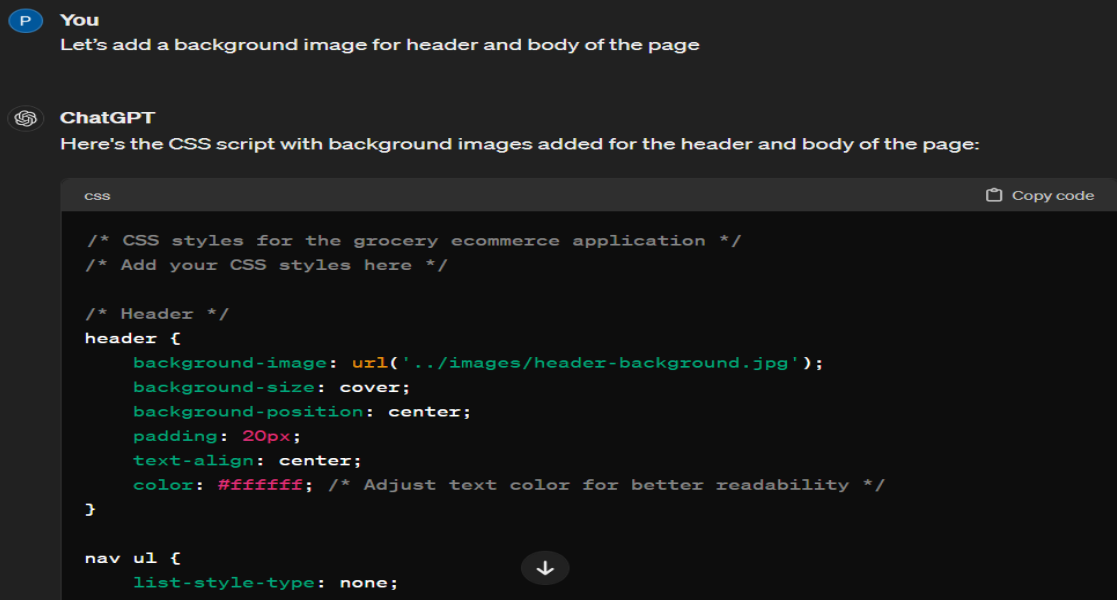
width: 100%;
max-width: 200px;
height: auto;
}

/* Footer */
footer {
  background-color: #343a40;
  color: #ffffff;
  padding: 20px;
  text-align: center;
}

/* Responsive Styles */
@media screen and (max-width: 768px) {
  nav ul li {
    display: block;
    margin-bottom: 10px;
  }

  .product {
    width: 45%;
    margin: 2.5%;
  }
}

```




```
padding: 0;
}

nav ul li {
display: inline;
margin-right: 20px;
}

/* Main */
main {
background-image: url('../images/body-background.jpg');
background-size: cover;
background-position: center;
padding: 20px;
color: #333333; /* Adjust text color for better readability */
}

.featured-products {
margin-bottom: 40px;
}

.product {
display: inline-block;
margin: 20px;
```



```

        text-align: center;
    }

    .product img {
        width: 100%;
        max-width: 200px;
        height: auto;
    }

    /* Footer */
    footer {
        background-color: #343a40;
        color: #ffffff;
        padding: 20px;
        text-align: center;
    }

    /* Responsive Styles */
    @media screen and (max-width: 768px) {
        nav ul li {
            display: block;
            margin-bottom: 10px;
        }
    }

```



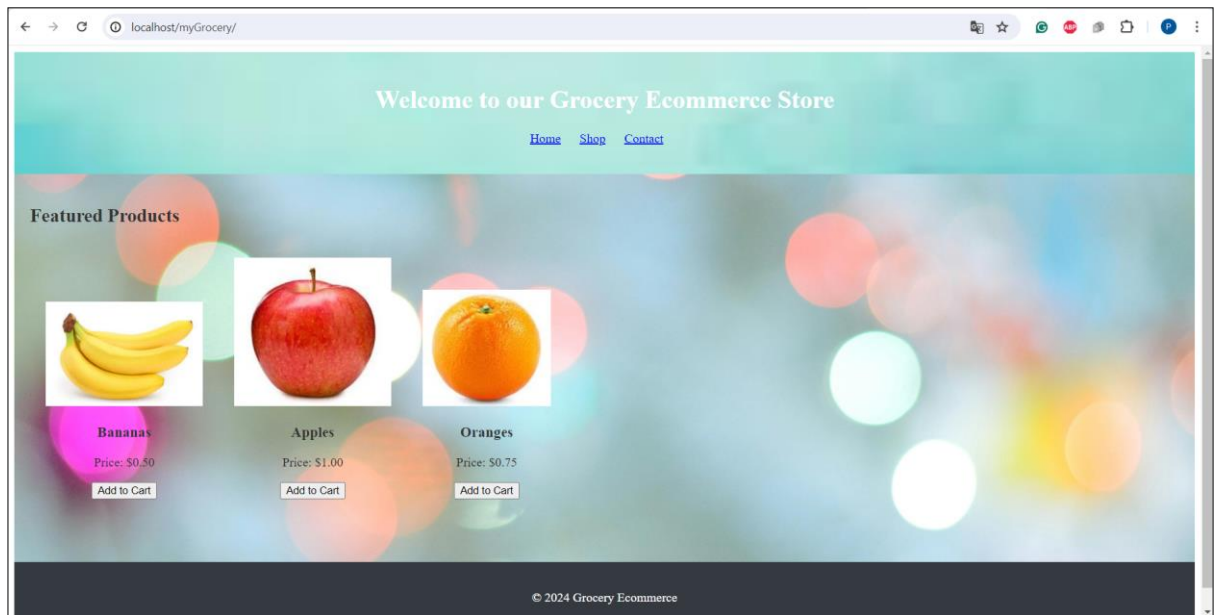
```

    .product {
        width: 45%;
        margin: 2.5%;
    }
}

```

This CSS script sets background images for the header and main sections of the page, enhancing the visual appeal and providing a better user experience. Adjustments to text color are also made for better readability against the background images.

1.14 Refresh the page and the background appearance changes as shown in the screenshot below:



By following the above steps, you have successfully built and deployed a PHP-based e-commerce website for groceries, utilizing generative AI for initial code generation and user interface enhancement.