



Wholly owned by UTAR Education Foundation
(Co. No. 578227-M)
DU012(A)

UECS2094 / UECS2194 WEB APPLICATION DEVELOPMENT

Cup Corner Café Website

Group Name: G10			
Student Name	Student ID	Practical Group	Programme (SE/ET)
Lai Jien Weng	2104438	P4	AM
Janice Ng Zhi Yan	2104332	P4	AM
Chow Ping Ching	1903301	P1	SE
WONG SIEW KUAN	2006530	P5	SE

**Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman**

Table of Contents

Introduction.....	4
Flowchart.....	6
Detailed Explanation and Implementation of Code.....	8
Frontend.....	8
navbar.php.....	8
footer.php.....	11
index.php.....	13
style.css.....	18
about.php.....	22
aboutUs.css.....	25
menu.php.....	29
menu.css.....	32
contact.php.....	36
contactus.css.....	39
cart.php.....	43
Cart.css.....	47
ourteam.php.....	52
our.css.....	55
login.php.....	59
login.css.....	63
register.php.....	67
registerme.css.....	70
Admin.....	74
admin/index.php.....	74
admin/insert_product.php.....	76
admin/view_products.php.....	77
admin/edit_product.php.....	79
admin/delete_product.php.....	80
admin/insert_category.php.....	81
admin/view_categories.php.....	83
admin/edit_category.php.....	84
admin/delete_category.php.....	86
admin/view_orders.php.....	87
admin/order_details.php.....	89
admin/refund.php.....	91
admin/all_payment.php.....	92
admin/user_list.php.....	95
admin/edit_user.php.....	97
admin/delete_user.php.....	99
admin/inbox.php.....	100
admin/message_details.php.....	101

Include.....	103
Include/connect.php.....	103
General.....	104
add_to_cart.php.....	104
check_login_status.php.....	105
checkout.php.....	106
config.php.....	107
submit_contact.php.....	108
remove_from_cart.php.....	109
order_details.php.....	109
function/common_function.php.....	111
JavaScript.....	112
Menu.js.....	112
MySQL.....	113
cup_corner.sql.....	113
Code Snippets.....	119
Admin.....	119
admin/product_images.....	119
admin/all_payments.php.....	120
admin/edit_product.php.....	134
admin/edit_user.php.....	140
admin/inbox.php.....	144
admin/index.php.....	146
admin/insert_category.php.....	149
admin/insert_product.php.....	151
admin/message_details.php.....	154
admin/order_details.php.....	157
admin/refund.php.....	162
admin/user_list.php.....	164
admin/view_categories.php.....	167
admin/view_orders.php.....	170
admin/view_products.php.....	172
Functions.....	175
functions/common_function.php.....	175
Image.home.....	181
Images.....	181
Include.....	182
include/connect.php.....	182
include/footer.php.....	182
include/navbar.php.....	186
General.....	192
about.php.....	192
aboutus.css.....	194
account.php.....	198

add_to_cart.php.....	203
cart.css.....	204
cart.php.....	208
check_login_status.php.....	215
checkout.php.....	215
config.php.....	219
contact.php.....	219
contactus.css.....	222
index.php.....	228
login.css.....	236
login.php.....	241
menu.css.....	244
menu.php.....	252
order_details.css.....	254
order_details.php.....	256
our.css.....	260
ourteam.php.....	264
reAccount.css.....	266
register.php.....	270
registerme.css.....	274
remove_from_cart.php.....	279
style.css.....	280
submit_contact.php.....	289
JavaScript.....	290
menu.js.....	290
MySQL.....	292
cup_corner.sql.....	293
Conclusion.....	308
Workload Summary.....	310
References.....	311

Introduction

Cup_Corner is a meticulously engineered online platform designed for coffee aficionados, developed using PHP, HTML, JavaScript and MySQL to deliver a robust and comprehensive web experience. The backbone of the site is formed by a series of PHP scripts, such as index.php for the homepage, menu.php for listing coffee products, and about.php which details the ethos and mission of Cup_Corner. These scripts are complemented by login.php and register.php, which handle user authentication, enhancing the security and personalization of the user experience.

CSS files like style.css and menu.css ensure the website is not only functional but also aesthetically pleasing, adhering to modern design principles for a responsive and engaging layout. JavaScript is utilized to enrich interactivity, evident in files like menu.js, which aids in dynamic content loading and enhances user interactions without page reloads.

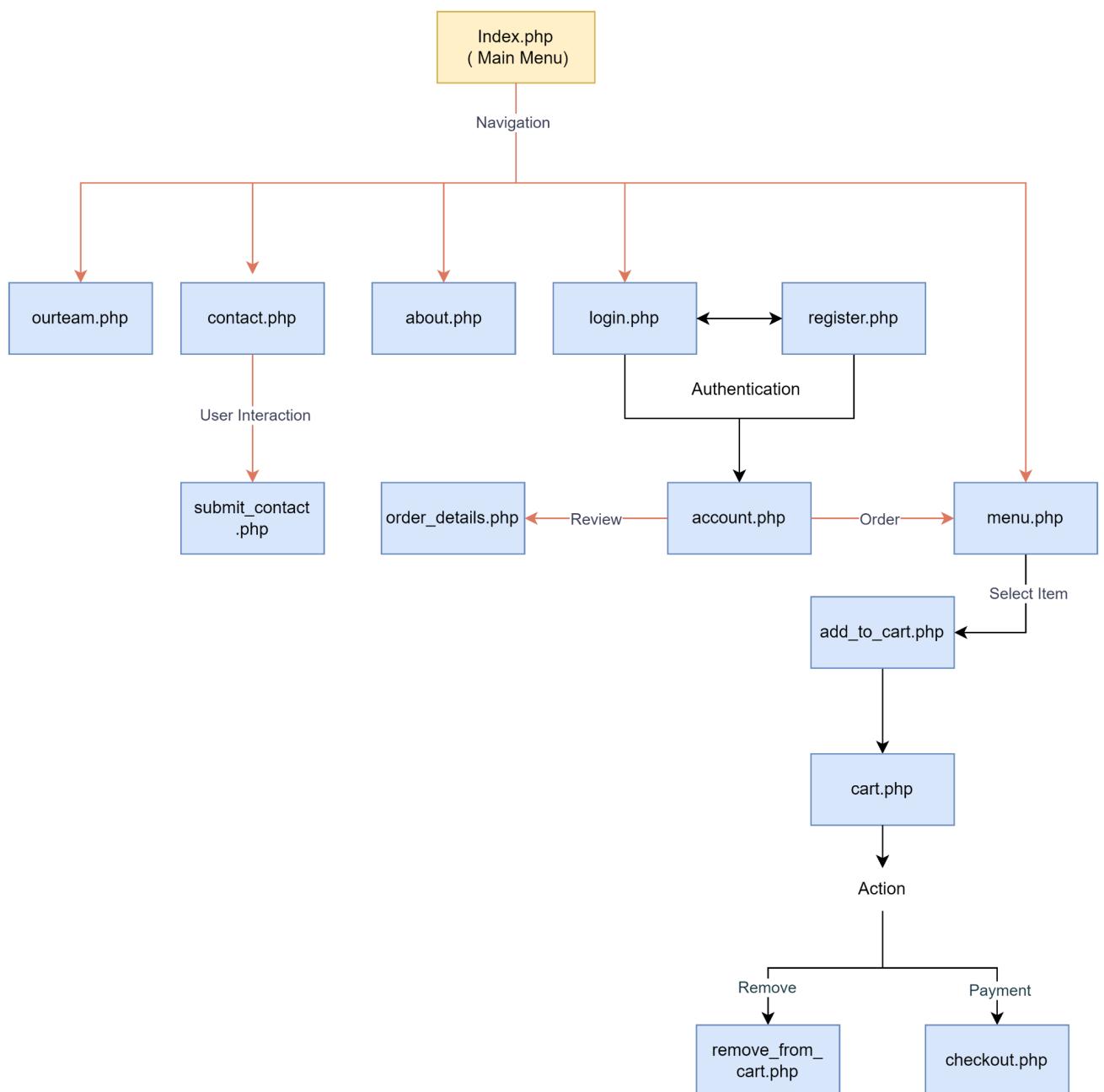
The platform's infrastructure is fortified with a robust backend database connection handled by connect.php included within the include directory, which securely manages data transactions between the website and its SQL database. This setup supports a complex yet efficient cart system implemented through add_to_cart.php and remove_from_cart.php, providing users with a seamless shopping experience.

Further, the website's content-rich approach is supported by an array of assets in the images and image.home directories, which host visual content that ranges from product showcases to background elements that enhance the site's thematic consistency. The admin folder contains scripts for backend management, allowing site administrators to update product listings, manage blog posts, and interact with customer data effectively.

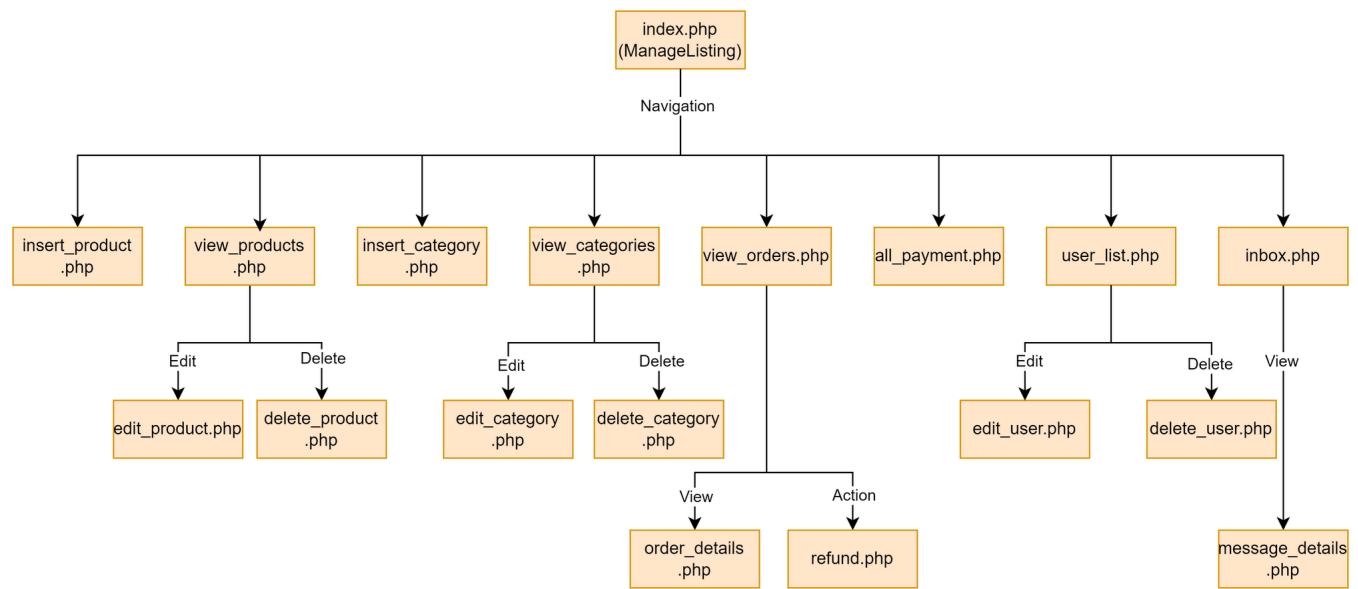
Through this detailed and well-structured use of web technologies, Cup_Corner offers a premier online experience that educates, engages, and excites coffee lovers globally, ensuring that every aspect of coffee enjoyment is just a click away.

Flowchart

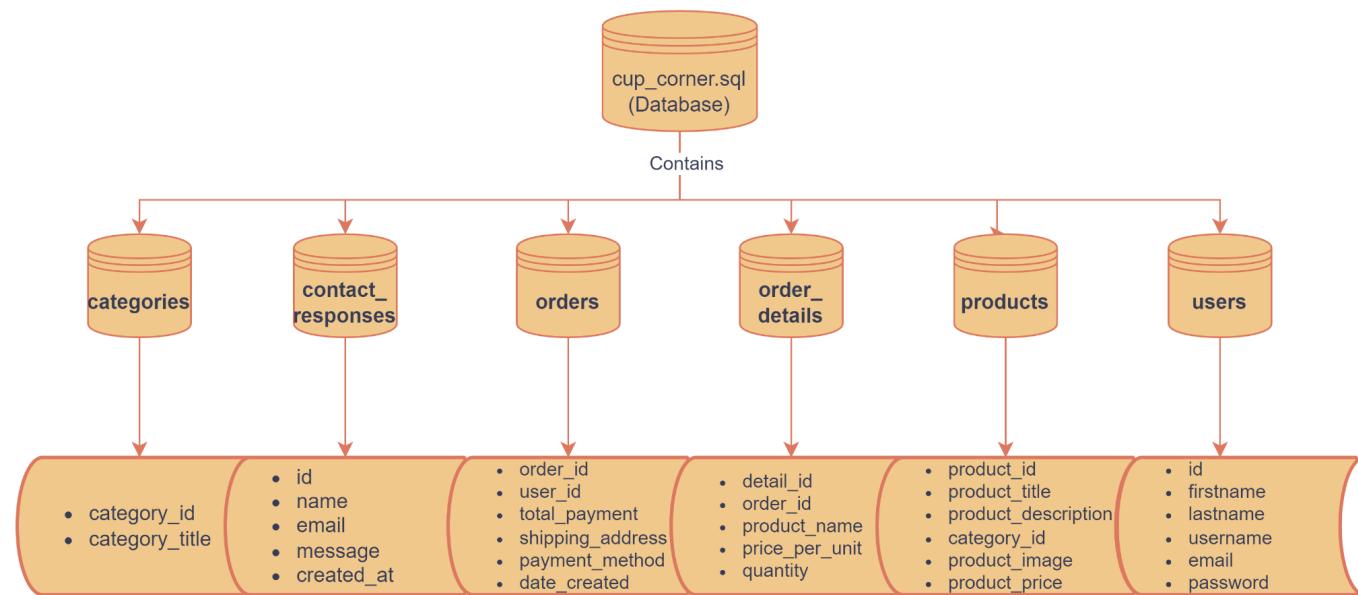
Main Menu (Index.php)



Admin Menu (Index.php)



Database (cup_corner.sql)



Detailed Explanation and Implementation of Code

Cup Corner is a fictional coffee shop, and the provided php, css and js files represent the structure and content of its website. In this essay, we will delve into the implementation details of the code, exploring its various sections and functionalities.

Frontend

navbar.php

The "navbar.php" file contains the code for the navigation bar of the Cup Corner website. It provides users with a consistent and accessible way to navigate between different pages of the website. The navigation bar includes the Cup Corner logo and links to various sections such as Home, About, Menu, Contact, Cart, and Account.

Detailed Explanation and Implementation of Code:

HTML Structure:

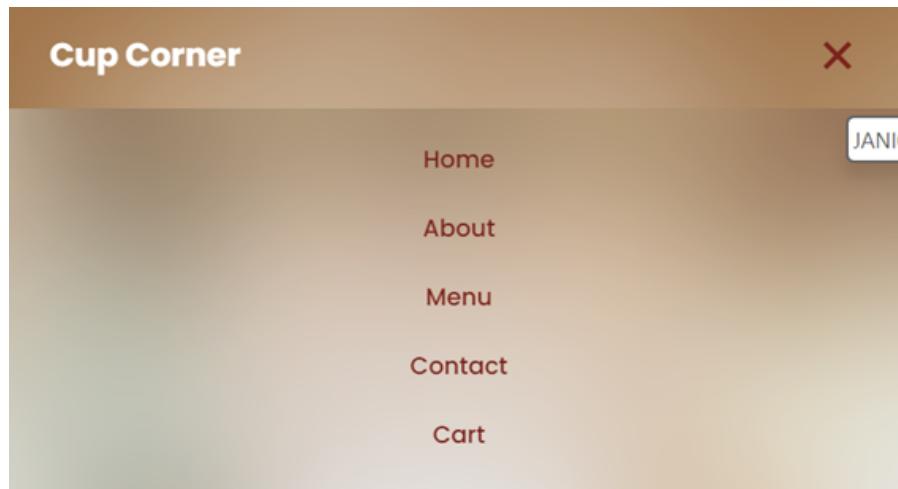
The HTML structure defines a header element (<header>) containing the Cup Corner logo (<a>), a checkbox input (<input type="checkbox">) for toggling the menu on small screens, and a navigation bar (<nav>). Inside the navigation bar, there are anchor tags (<a>) representing different sections of the website.

CSS Styling:

The CSS styling applies visual enhancements to the navigation bar and its components. It sets global styles for elements such as margin, padding, box-sizing, and font-family using the universal selector (*). The header (header) is styled to have a sticky position at the top of the viewport, with padding, flexbox alignment, and a background image.



Pseudo-elements (::before and ::after) are used to create visual effects such as a backdrop filter and gradient overlay.



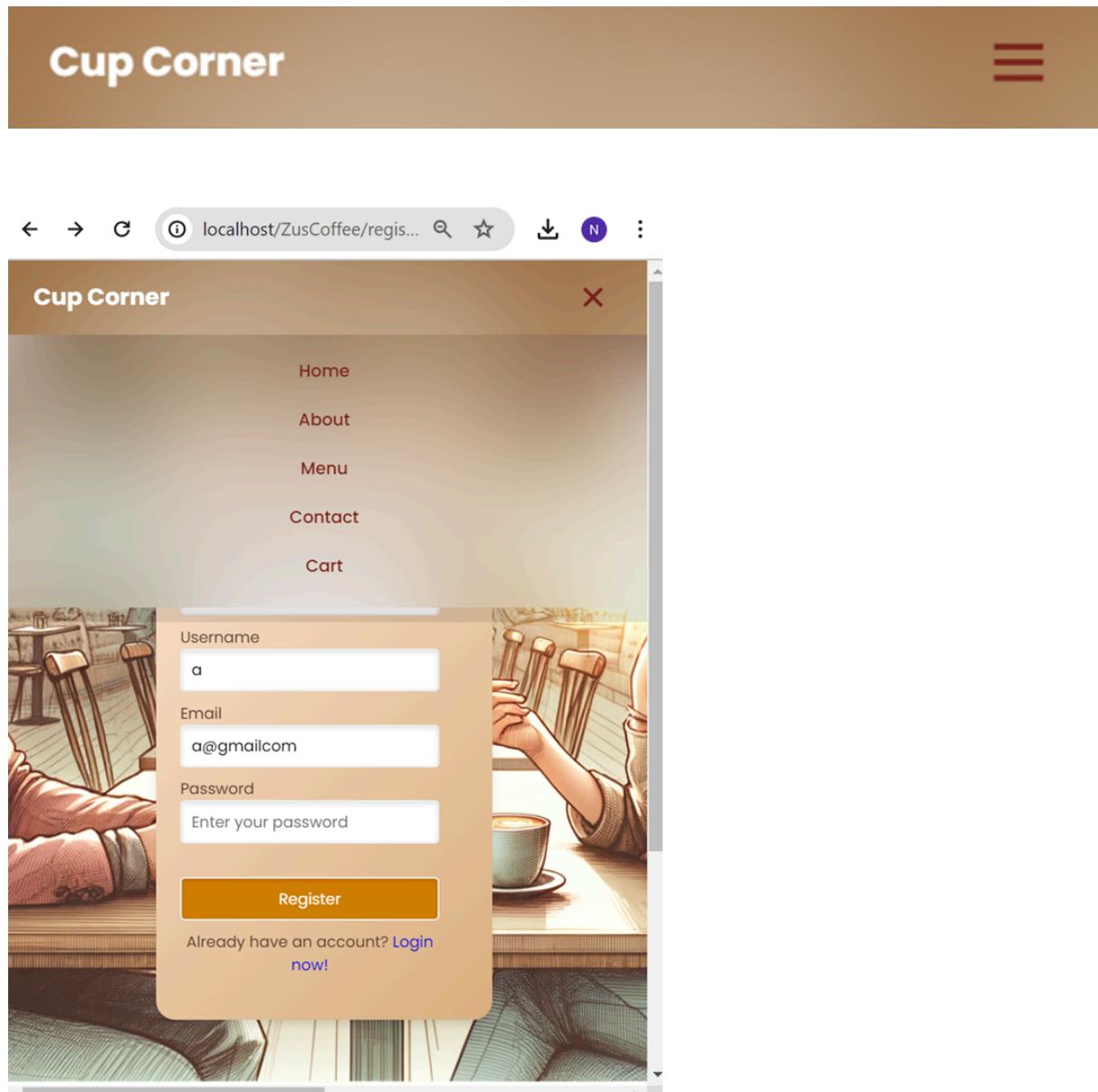
The Cup Corner logo (logo) is styled with a larger font size, white color, and bold font weight.



Navigation links (navbar a) are styled with a specific font size, color, and margin. The checkbox input (#check) is hidden by default, and its state is used to toggle the visibility of the menu on small screens.

Media Queries:

Media queries are used to make the navigation bar responsive and adapt its layout to different screen sizes. At smaller screen widths, the navigation bar is converted into a collapsible menu. The display of icons and navigation links is adjusted based on the checkbox state (#check:checked) to show or hide the menu.



footer.php

The "footer.php" file contains the code for the footer section of the Cup Corner website. It is an essential part of the website's layout, providing links to social media profiles, navigation links to other sections of the website, and copyright information.

Detailed Explanation and Implementation of Code:

HTML Structure:

The HTML structure defines a footer element (<div class="footer">) containing three main sections: social media links (<div class="footer-col-1">), navigation links (<div class="footer-col-2">), and copyright information (<div class="footerBottom">). The Social media links are represented by anchor tags (<a>) with Font Awesome icons inside them. The Navigation links are represented by an unordered list () containing list items () with anchor tags for each link. Besides, the copyright information is displayed as a paragraph (<p>).



- [Our Team](#)
- [About](#)
- [Contact Us](#)

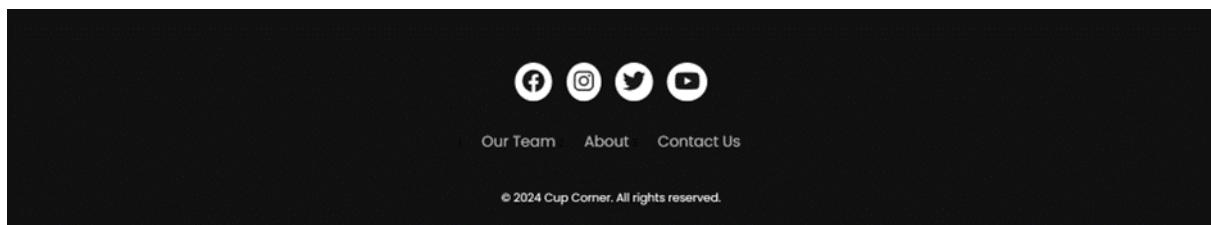
© 2024 Cup Corner. All rights reserved.

CSS Styling:

The CSS styling enhances the visual appearance of the footer section. The footer background color is set to dark gray (#111) to create contrast with the rest of the page. The padding and margin properties are used to adjust the spacing within the footer elements. Flexbox is utilized to align the social media icons horizontally (footer-col-1) and center the navigation links vertically (footer-col-2).

Social media icons are styled with a white background, a black color, and a circular shape. Hover effects change the background to black and the icon color to white. The navigation links are styled with a white color, margin, and opacity. Besides, the hover effects increase the opacity of the links.

Copyright text is centered and styled in white color.



index.php

The index.php file is the main file responsible for managing the design, content, and features of Cup Corner's homepage. The website plays a vital role in providing key information about the coffee shop, attracting visitors to delve deeper, and potentially directing them to specific actions like viewing the menu, discovering special promotions, or reading customer feedback.

HTML Structure:

The HTML document begins with standard declarations (`<!DOCTYPE html>`), followed by the `<html>` element with the `lang` attribute set to "en". Inside the `<head>` section, metadata such as character encoding (`<meta charset="UTF-8">`) and page title (`<title>Cup Corner</title>`) are specified.

External Resources:

The `<head>` section also includes references to external resources such as CSS files for styling and designing the home page (`<link rel="stylesheet" href="style.css">`) and font libraries from Google Fonts and Font Awesome. Additionally, a favicon is linked to the browser tab icon.

Navigation Bar:

The navigation bar is included via PHP using the `include()` function, pointing to a separate file (navbar.php). This promotes code modularity and reusability by separating the navigation logic from the main HTML document.



Content Section:

The main content section (`<div class="content">`) contains a header (`<h1>`) with a title and a paragraph describing the coffee experience offered by Cup Corner. An anchor tag (`<a>`) is used to link to the menu page (`menu.php`). The content is divided into two columns using Bootstrap grid system classes (`col-6`).

Embrace the Aroma: Elevate Your Coffee Experience Today!

Savoring a perfect cup of coffee isn't just about that initial burst of flavor, it's about consistently delivering an unparalleled experience with every sip.

[Explore Now ➔](#)

Featured Categories:

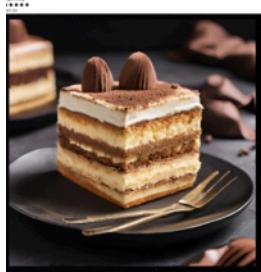
A section showcasing featured categories of coffee products is displayed using a grid layout (`<div class="categories">`). Each category is represented by an image wrapped in a `<div>` with the class `col-3`.



Featured Menu:

The featured menu section lists seasonal flavors and all-time favorites for the coffee shop.

Each menu item is presented within a grid layout (<div class="row">), with details such as image, name, rating, and price displayed using appropriate HTML elements.



Special Offer:

An offer section (<div class="offer">) promotes a seasonal promotion with descriptive text and an accompanying image. The content is structured using grid layout classes.



Spooktacular saving

Pumpkin Patch Pals

Add some festive flair to your Halloween celebrations with our adorable Pumpkin Patch Pals plush sponsored by Disney! This cuddly companion is the perfect addition to your spooky decor, featuring soft, plush fabric and charming pumpkin design. Receive this delightful plush as a free gift with the purchase of any of our seasonal drinks, and bring a touch of whimsy to your festivities. Hurry, while supplies last!

Customer Reviews:

Customer reviews are displayed in a section (<div class="cusreview">), featuring testimonials, ratings, and images of the reviewers. The reviews are arranged in a grid layout for visual appeal.



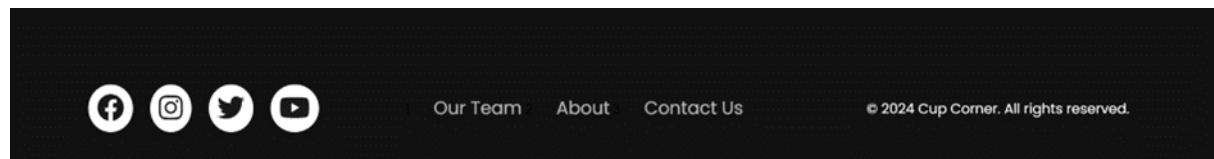
Brand Logos:

The brand section (<div class="brands">) showcases logos of partner brands, arranged in a grid layout for symmetry.



Footer:

Finally, the footer section is included via PHP (include('include/footer.php');). This section typically contains links, contact information, and other miscellaneous details.



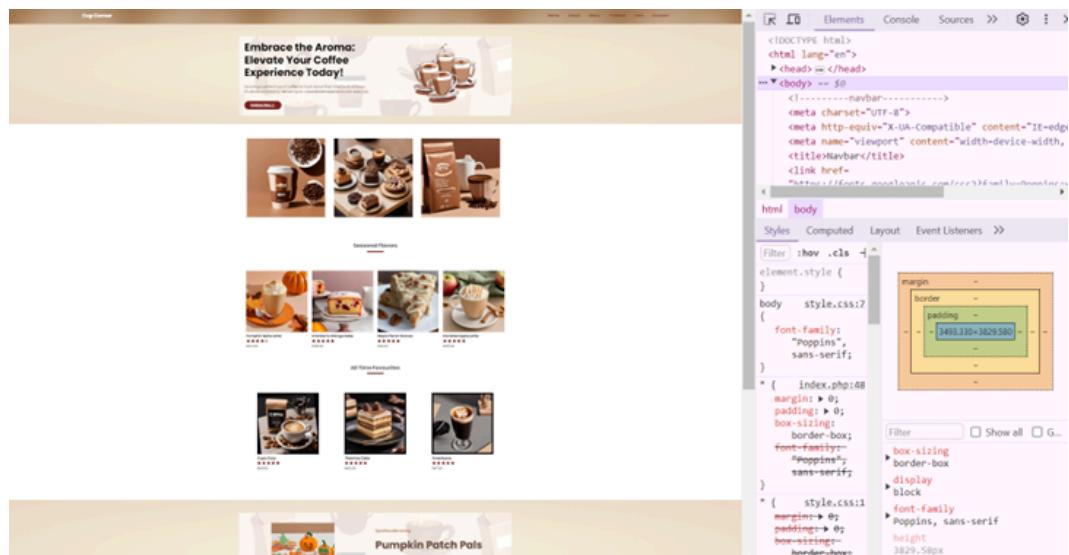
style.css

The CSS stylesheet provided is responsible for styling Cup Corner's homepage ([index.php](#)).

It defines various visual aspects such as layout, typography, colors, and transitions to create an aesthetically pleasing and user-friendly interface for visitors.

Global Styles:

- The `*` selector resets margins, paddings, and sets the `box-sizing` property to `border-box` for consistent layout handling across elements.
- The `body` rule sets the default font family to "Poppins", a sans-serif typeface, ensuring readability and consistency throughout the page.



Layout Styles:

- Container classes (**container** and **small-container**) define the maximum width, centering, and padding of content to maintain readability and visual appeal.
- The **.row** class sets up a flex container for organizing content into rows, allowing for alignment, wrapping, and spacing between elements.
- Column classes (e.g., **.col-6**, **.col-2**, **.col-3**, **.col-4**, **.col-5**) define the layout and sizing of content columns using flexbox, enabling responsive design for different screen sizes.



Typography and Colors:

- Headings (**h1**) are styled with custom font sizes, line heights, and colors to create hierarchy and visual impact.
- Paragraphs (**p**) have defined colors and font weights for readability and emphasis.
- Anchor tags (**a**) styled as buttons (**btn**) feature background colors, text colors, padding, and border-radius for interactive elements. Hover effects enhance user interaction by changing background colors on hover.

Content Sections:

- Background images and gradients are used to style content sections (**content**, **categories**, **offer**, **cusreview**, **brands**), providing visual interest and separation between different parts of the page.
- Section titles (**title**) are centered, styled with custom colors, and feature an underline effect created using pseudo-elements (**::after**), adding visual distinction and hierarchy.



Seasonal Flavors



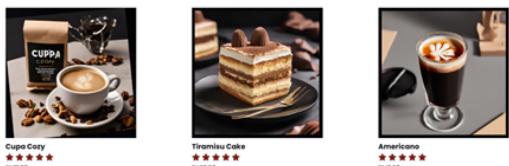
Pumpkin Spice Latte
RM14.00
★★★★★

Cranberry Orange Cake
RM18.00
★★★★★

Maple Pecan Scones
RM15.00
★★★★★

Caramel Apple Latte
RM15.00
★★★★★

All Time Favourites



Cupa Cozy
RM13.00
★★★★★

Tiramisu Cake
RM12.00
★★★★★

Americano
RM11.00
★★★★★



Pumpkin Patch Pals

Add some festive flair to your Halloween celebrations with our adorable Pumpkin Patch Pals plush softies! Made by Disney! These super-soft pals are the perfect addition to your spooky decor, featuring soft, pliable arms and charming pumpkin design. Receive this delightful plush as a free gift with the purchase of any of our seasonal drinks, and bring a touch of whimsy to your festivities! (Only while supplies last!)

Special Effects and Transitions:

- Shadow effects and transitions (**box-shadow, transition**) are applied to certain elements (e.g., customer review cards, brand logos) to create depth and smooth visual effects on hover interactions.

When the cursor reaches the middle box, the box will bounce upwards.



When the cursor reaches the eco logo, the logo will show its colour.

about.php

The "About Us" page plays a crucial role in presenting Cup Corner's backstory, values, and achievements to visitors. It serves as a platform to establish a connection with customers by sharing the brand's journey, mission, and key statistics. This page aims to evoke trust, credibility, and interest in Cup Corner's offerings, ultimately fostering a sense of community and loyalty among its audience.

HTML Structure:

The HTML document begins with standard declarations and specifies the language as English. The external resources such as CSS files for navbar and aboutUs stylesheets are linked, along with a Google Fonts stylesheet for typography.

Navbar Inclusion:

The navbar is included using PHP's **include()** function, pointing to a separate file (**navbar.php**). This promotes code reusability and maintains consistency across multiple pages.



Content Section:

The content section is encapsulated within a `<section>` element with the class **about**. This provides a semantic structure to the page and allows for targeted styling. Inside the section, a container (**section_container**) houses the main content and image elements of the about page. The content is divided into two parts: text content (**about_content**) and an accompanying image (**about_image**). The text content consists of a heading (`<h2>`), a subheading (`<p>`), and a flex container (**about_flex**) containing statistical cards (**about_card**).

About us

Cup Corner is founded by a group of coffee enthusiasts and has grown from humble beginnings to become a beloved destination for coffee lovers all over the world. Our journey began with a simple desire: to create a space where the rich aroma of freshly brewed coffee and the warmth of community could coexist seamlessly. Cup Corner stands today as a testament to our passion for coffee, serving a diverse selection of meticulously sourced and expertly crafted beverages. At the heart of our brand is a dedication to quality and a strong belief in building relationships over coffee. Join us at Cup Corner, where each sip reveals a story of passion and excellence.

No.1

Coffee Brand

2094

Stores worldwide

3

Michelin Stars



Text Content:

The heading (`<h2>`) introduces the section as "About us", clearly indicating the purpose of the content that follows. The subheading (`<p>`) provides a detailed narrative of Cup Corner's history, values, and offerings. It emphasizes the brand's passion for coffee and commitment to quality and community.

About us

Cup Corner is founded by a group of coffee enthusiasts and has grown from humble beginnings to become a beloved destination for coffee lovers all over the world. Our journey began with a simple desire: to create a space where the rich aroma of freshly brewed coffee and the warmth of community could coexist seamlessly. Cup Corner stands today as a testament to our passion for coffee, serving a diverse selection of meticulously sourced and expertly crafted beverages. At the heart of our brand is a dedication to quality and a strong belief in building relationships over coffee. Join us at Cup Corner, where each sip reveals a story of passion and excellence.

Statistical Cards:

Three statistical cards (**about_card**) are displayed side by side using flexbox layout within the flex container (**about_flex**). Each card contains a numerical value (**<h4>**) and a corresponding label (**<p>**), highlighting key metrics such as coffee brand ranking, store count, and Michelin stars.

No.1

Coffee Brand

2094

Stores worldwide

3

Michelin Stars

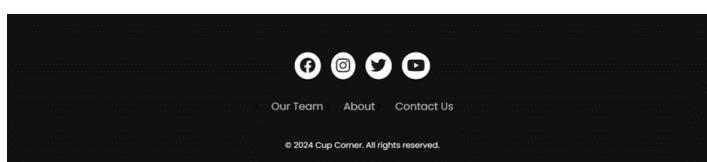
Image:

An image (****) is included to visually complement the text content and provide additional context about Cup Corner's brand identity. The image is sourced from a file named "backgroundabout.webp" and is set as the background of the about section.



Footer Inclusion:

The footer is included using PHP's **include()** function, pointing to a separate file (**footer.php**). This ensures consistency in footer content across all pages of the website.



aboutUs.css

The "aboutUs.css" file is responsible for styling the "About Us" section of Cup Corner's website. It defines the visual appearance, layout, and interactive effects to effectively convey the brand's story, values, and achievements to visitors. By employing a combination of colors, typography, layout techniques, and responsive design principles, this stylesheet enhances the user experience and reinforces Cup Corner's brand identity.

Global Styles:

The `*` selector resets margins, paddings, and sets the font family to 'Poppins' for all elements, ensuring consistency and a uniform appearance throughout the page. The `:root` pseudo-class defines custom variables for color values, facilitating easy color management and theming across the stylesheet.

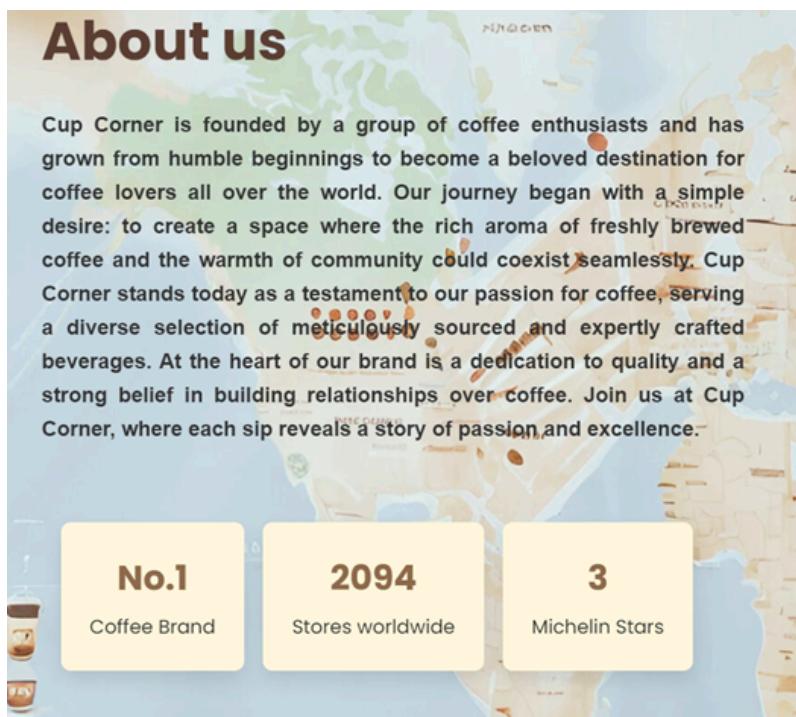
About Section Styles:

The `.about` class sets the background image, padding, and flexbox properties to center-align content both vertically and horizontally within the section. The background image creates a visually engaging backdrop for the content. Minimum height (`min-height: 100vh`) ensures that the section occupies at least the height of the viewport, maintaining a consistent layout regardless of screen size.



Typography and Colors:

.section_header and .section_subheader classes style the heading and subheading text respectively, defining font sizes, weights, colors, and line heights to create hierarchy and readability. The custom color variables (**--primary-color**, **--dark-color**, **--accent-color**, **--text-color**, **--background-color**) are used throughout the stylesheet to maintain consistency and allow for easy color adjustments.



Grid Layout for Content:

The **.about_container** class employs CSS Grid layout to arrange content into two columns on larger screens (**repeat(2, 1fr)**), providing a balanced and visually appealing layout. The **gap** property adds spacing between grid items, enhancing readability and visual separation.

Flexbox for Statistical Cards:

.about_flex class utilizes flexbox to horizontally align and space out statistical cards (**about_card**), creating a visually appealing and responsive layout.

Each card (**about_card**) features background color, padding, border radius, and box shadow for a polished appearance.



Image Styling and Effects:

The .about_image img class styles the accompanying image, setting maximum width, margins, border radius, and box shadow for visual appeal. On hover (:hover), the image enlarges and displays an enhanced box shadow effect, creating an interactive and engaging user experience.

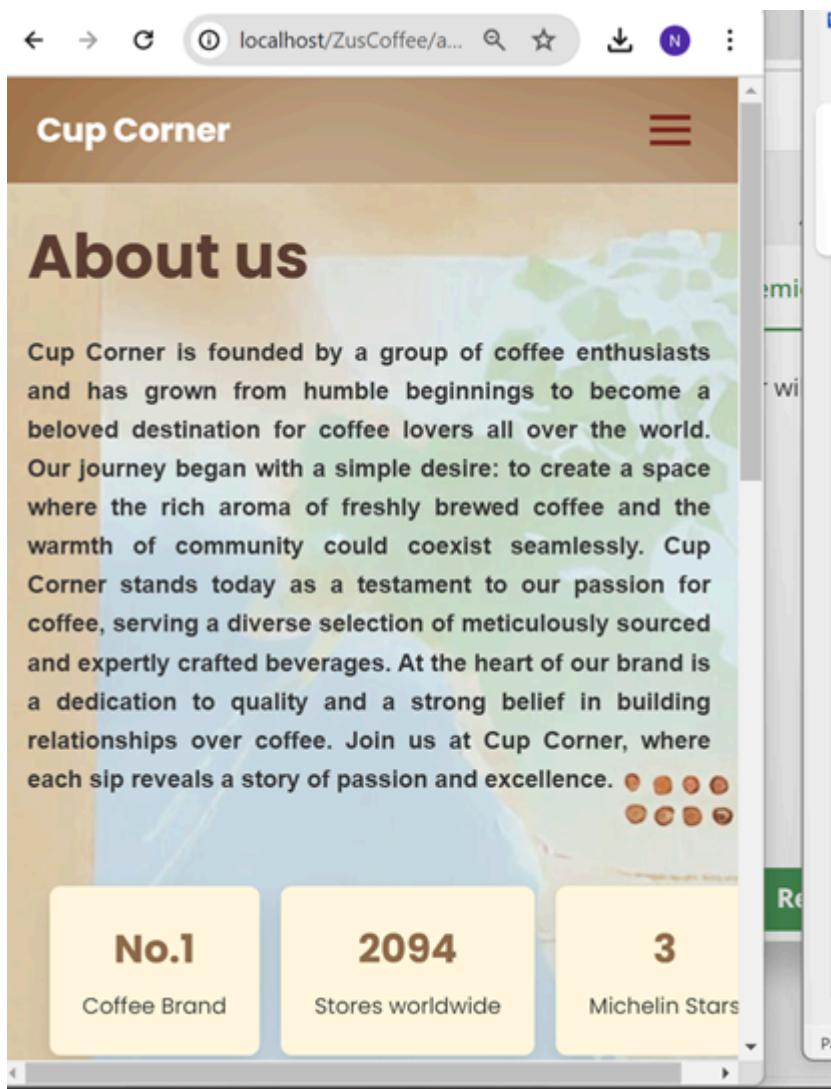


When the cursor hovers over the image, the border will illuminate with a brown shadow.

Media Queries for Responsiveness:

Media queries (`@media`) are used to adjust styles based on screen size, ensuring optimal layout and readability across different devices.

In smaller viewports (max-width: 768px), the grid layout switches to a single column, and the image spans the full width for improved responsiveness.



menu.php

The "menu.php" file is a dynamic webpage that presents the coffee menu of Cup Corner's website. It utilizes PHP to fetch and display products and categories stored in a database, allowing users to browse through different coffee options. The webpage is designed to provide an interactive and visually appealing experience for customers, featuring product previews and category navigation.

HTML Structure:

The HTML document starts with standard declarations and includes meta tags for character set and viewport settings. The external resources such as Bootstrap CSS and JavaScript, Font Awesome, custom CSS (**menu.css**), and custom JavaScript (**menu.js**) are linked for styling and functionality.

Navbar Inclusion:

The navbar is included at the top of the page using PHP's **include()** function, pointing to a separate file (**navbar.php**). This promotes code reusability and maintains consistency across multiple pages.

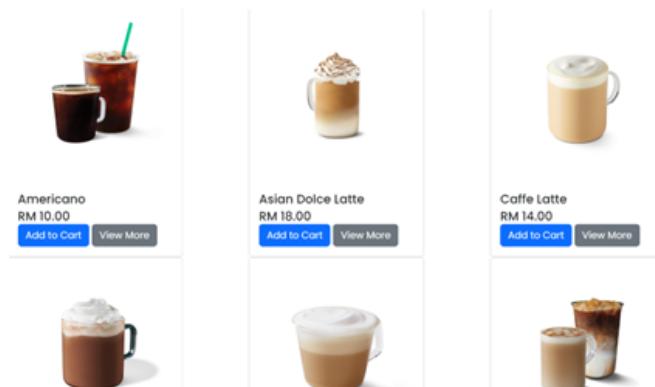


Database Connection and Function Includes:

PHP includes statements are used to include files for database connection (**connect.php**) and common functions (**common_function.php**). This allows for retrieving product data and performing necessary operations.

Product Display:

The main content area (`<div class="col-md-10">`) is dedicated to displaying products in a grid layout. Inside this area, a PHP function (`getProducts()`) is called to fetch and display coffee products dynamically. This function retrieves product data from the database and generates HTML markup for each product.



Category Navigation:

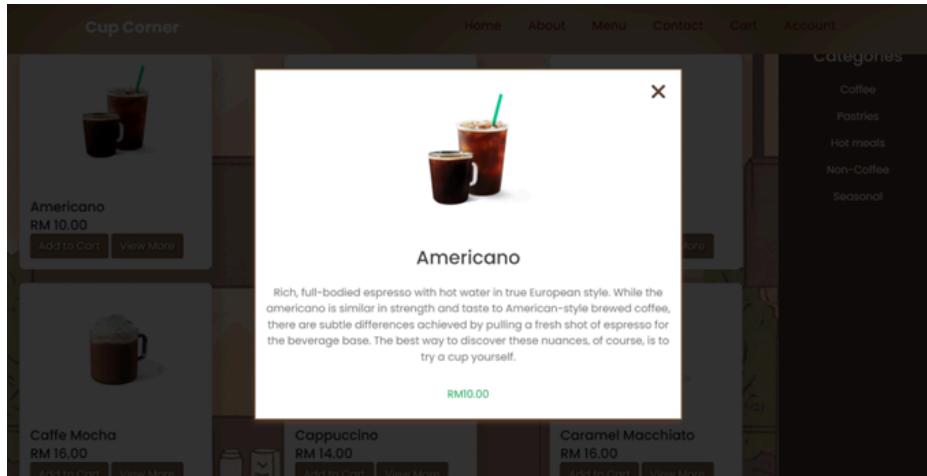
A side navigation section (`<div class="side-nav col-md-2">`) is provided to allow users to navigate through different coffee categories. The PHP code (`getCategories()`) fetches category data from the database and generates HTML markup for category links.

Categories

- Coffee
- Pastries
- Hot meals
- Non-Coffee
- Seasonal

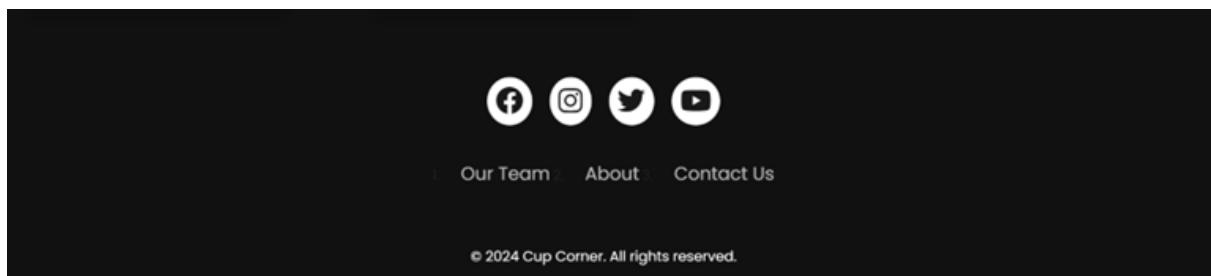
Product Preview:

Below the main product grid, a section (<div class='products-preview'>) displays previews of selected products or featured items. PHP code (**getProductPreview()**) is used to retrieve and display these previews dynamically.



Footer Inclusion:

The footer is included at the bottom of the page using PHP's **include()** function, pointing to a separate file (**footer.php**). This ensures consistency in footer content across all pages of the website.



menu.css

The "menu.css" file is responsible for styling the menu page of Cup Corner's website, providing an attractive and user-friendly interface for customers to browse through coffee products. It defines various styles for elements such as cards displaying products, navigation bars, buttons, and pop-up previews. The CSS file utilizes custom variables for color schemes, ensuring consistency and easy customization across the website.

Global Reset:

The CSS file starts with a global reset, resetting margins, padding, and box-sizing for all elements to ensure consistent layout rendering across different browsers. The font-family is set to 'Open Sans' for the entire document, ensuring a consistent typography style.

Custom Color Variables:

Custom color variables are defined using CSS custom properties (--primary-brown, --light-brown, --dark-brown, --text-color, --button-hover-color, --button-text-color) to maintain consistency and allow easy color theme adjustments throughout the stylesheet.

Body Styling:

The body background is set to an image (backgroundmenu.png) using a relative URL.

Card Styling:

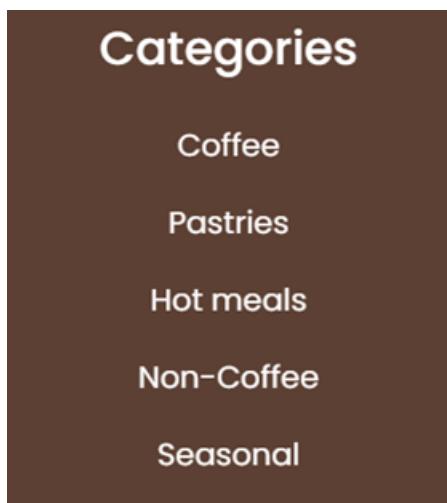
Cards (<div class="card">) displaying individual coffee products are styled with a white background, border radius, and box-shadow for a modern and clean look. Hover effects (transform and box-shadow) are applied to cards to provide a subtle interactive effect when users interact with them.



The border would bounce upward if the cursor reached the box.

Side Navigation Styling:

Side navigation (<div class="side-nav">) elements are styled with a dark brown background color, white text color, and centered alignment for improved readability and visual appeal.



Navbar Styling:

Navbar elements (<ul class="navbar-nav">) are styled with a dark brown background color for consistency with the side navigation.

Button Styling:

Buttons (<button> and <a> elements with .btn class) are styled with custom background colors, text colors, border properties, and hover effects to enhance visual appeal and usability. The .btn-outline-primary class is defined for buttons with primary color borders and transparent backgrounds, suitable for use in specific contexts.



Americano

RM 10.00

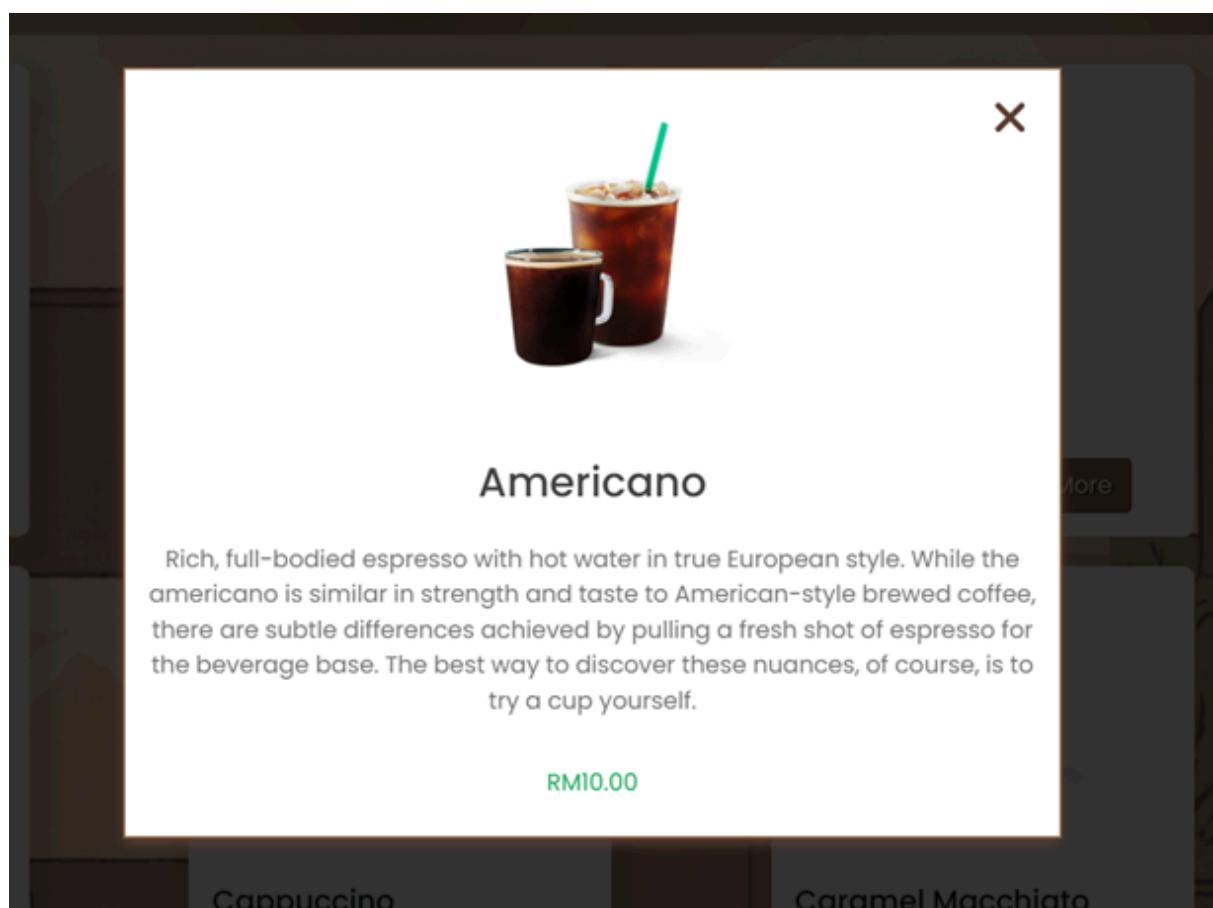
[Add to Cart](#)

[View More](#)

The colour will change to light brown when the cursor reaches the button.

Product Preview Styling:

The `.products-preview` class defines styles for a pop-up preview of individual coffee products. It uses absolute positioning, background transparency, and box-shadow to create a modal-like appearance. The product preview content (image, title, description, price, buttons) is styled for readability and visual appeal, with appropriate padding, font sizes, and colors.

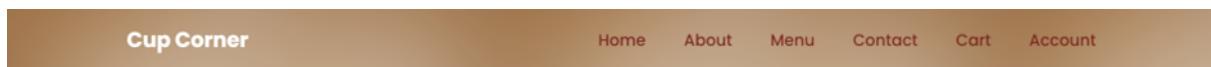


contact.php

The "contact.php" file serves as the page for customers to contact Cup Corner regarding any inquiries, feedback, or support-related issues. It provides a form for users to input their contact details and messages, which are then sent to Cup Corner's support team for review and response. Additionally, the page displays essential information such as the company's address, contact information, and business hours.

Session Handling and Includes:

PHP session is started to maintain user sessions and handle feedback messages. The file includes the navigation bar (navbar.php) for consistent navigation across the website.



Feedback Message Display:

If a feedback message is stored in the session variable `$_SESSION['feedback']`, it is displayed at the top of the page. After displaying the feedback message, it is unset to ensure it is shown only once.

Contact Information Display:

The contact information section displays Cup Corner's address, phone number, and email address in separate boxes. Each box contains an icon (from Font Awesome) corresponding to the type of information (address, phone, or email) and the respective contact details.



Address

1, Jalan Sungai Long,
43200 Kajang, Selangor



Phone

012-1232094



Email

cupcorner@gmail.com

Feedback Form:

Below the contact information section, a feedback form is provided for users to submit their messages. The form action is set to "submit_contact.php", indicating that form submissions will be handled by that PHP script. The input fields include "Full Name", "Email", and "Message", all marked as required. Upon form submission, the data is sent to "submit_contact.php" via the POST method.

Feedback

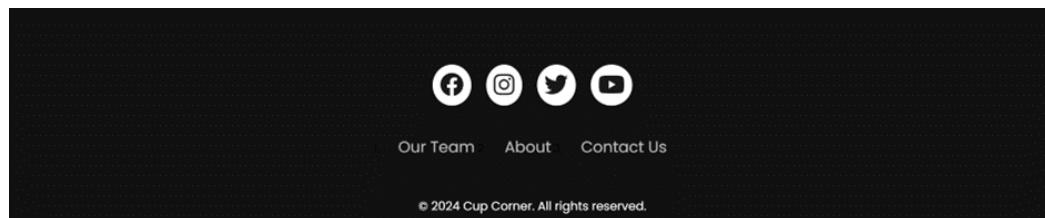
<input type="text"/>	Full Name
<input type="text"/>	Email
<input type="text"/>	Type your Message...
<input type="button" value="Send"/>	

Styling and Layout:

The page is styled using external CSS files (navbar.css and contactus.css). The fonts from Google Fonts and icons from Font Awesome are imported to enhance the visual appearance. The layout is structured into sections, with the contact information and feedback form placed side by side within a container.

Footer Inclusion:

At the end of the page, the footer is included (footer.php) to provide additional navigation or informational links.

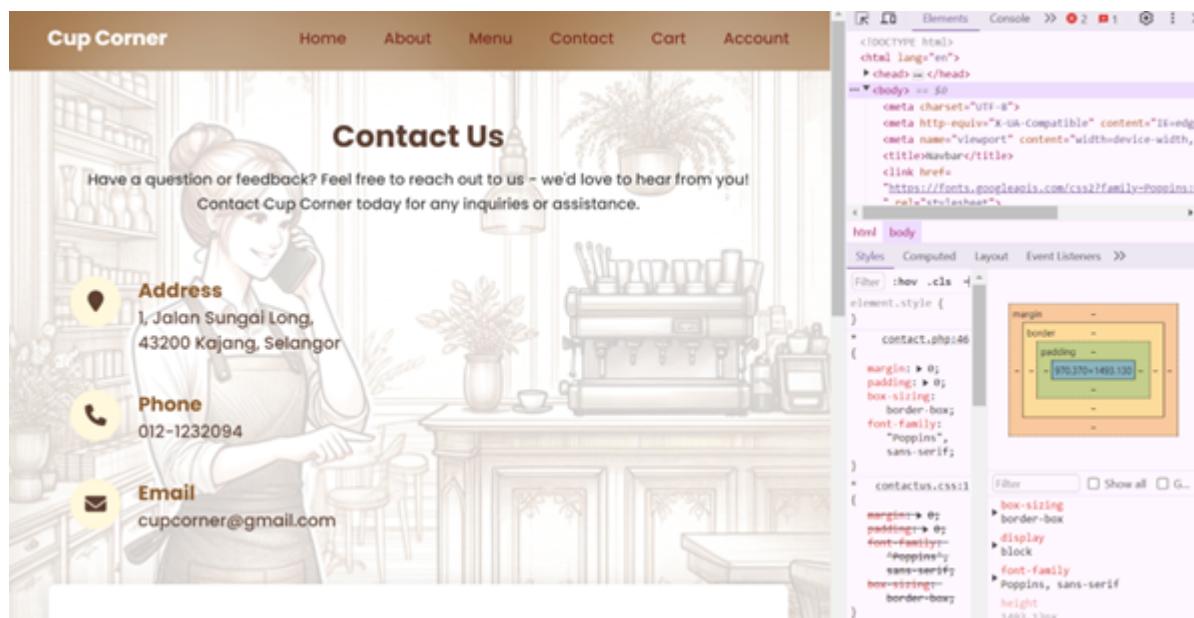


contactus.css

The "contactus.css" file contains the CSS styles for the contact page of Cup Corner's website. It defines the layout, positioning, and styling of various elements on the contact page, including the contact information section and the feedback form.

Global Styles:

The CSS file starts with global styles that reset margins, paddings, and set the font family to 'Poppins', a sans-serif font. The box-sizing property is set to border-box for consistent box model sizing.

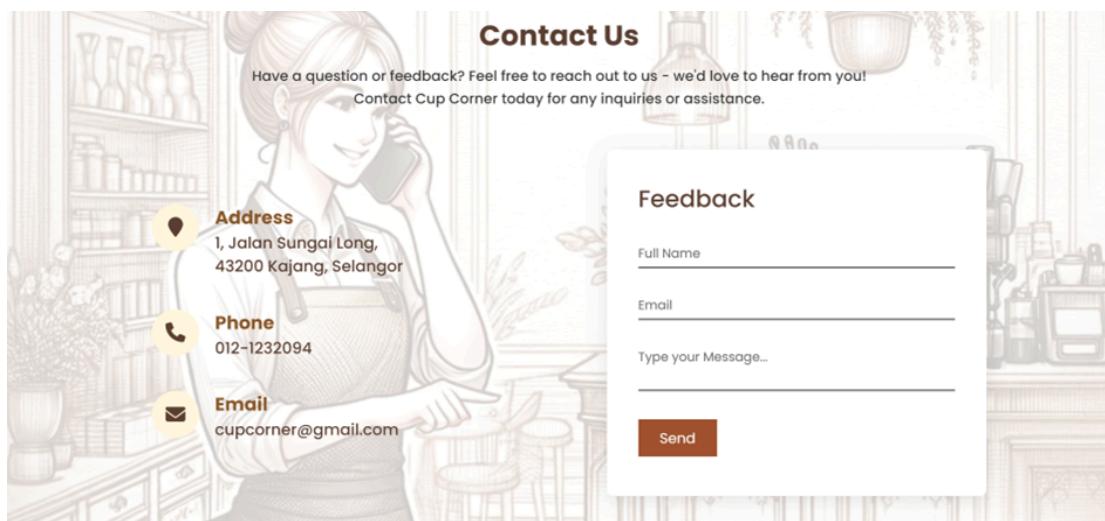


Contact Section:

The .contact class styles the main section of the contact page. It sets the background image using the background property with a fixed position, cover size, and centered alignment.

Content Styling:

The .contact .content class styles the content inside the contact section. It sets the maximum width and aligns the text center. Headings (h2) and paragraphs (p) inside the content are styled for font size, weight, and color.



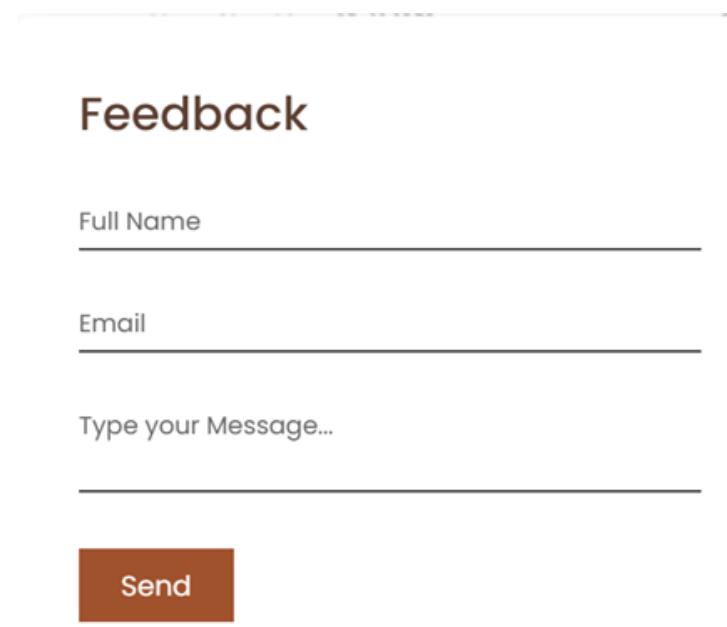
Contact Information Section:

The .container .contactInfo class styles the container for contact information. It sets the width to 50% of the parent container and arranges the content in a column layout. Each contact information box (address, phone, email) is styled using the .container .contactInfo .box class. It positions the icon and text side by side with appropriate margins and colors.



Feedback Form Styling:

The .contactform class styles the feedback form section. It sets the width, padding, background color, box shadow, and border radius to create a visually appealing form container. Headings (h2) inside the form are styled for font size, weight, and color. The input fields and text areas are styled with consistent width, padding, font size, and border properties. The bottom border is used for input focus indication. The .contactform .inputBox span class styles the placeholder text for input fields. It positions the placeholder text at the top of the input field when it is focused or contains valid content. The submit button is styled with background color, text color, border, cursor, padding, and font size.



A screenshot of a feedback form titled 'Feedback'. The form includes fields for 'Full Name', 'Email', and a message area with placeholder text 'Type your Message...'. A brown 'Send' button is at the bottom.

Feedback

Full Name

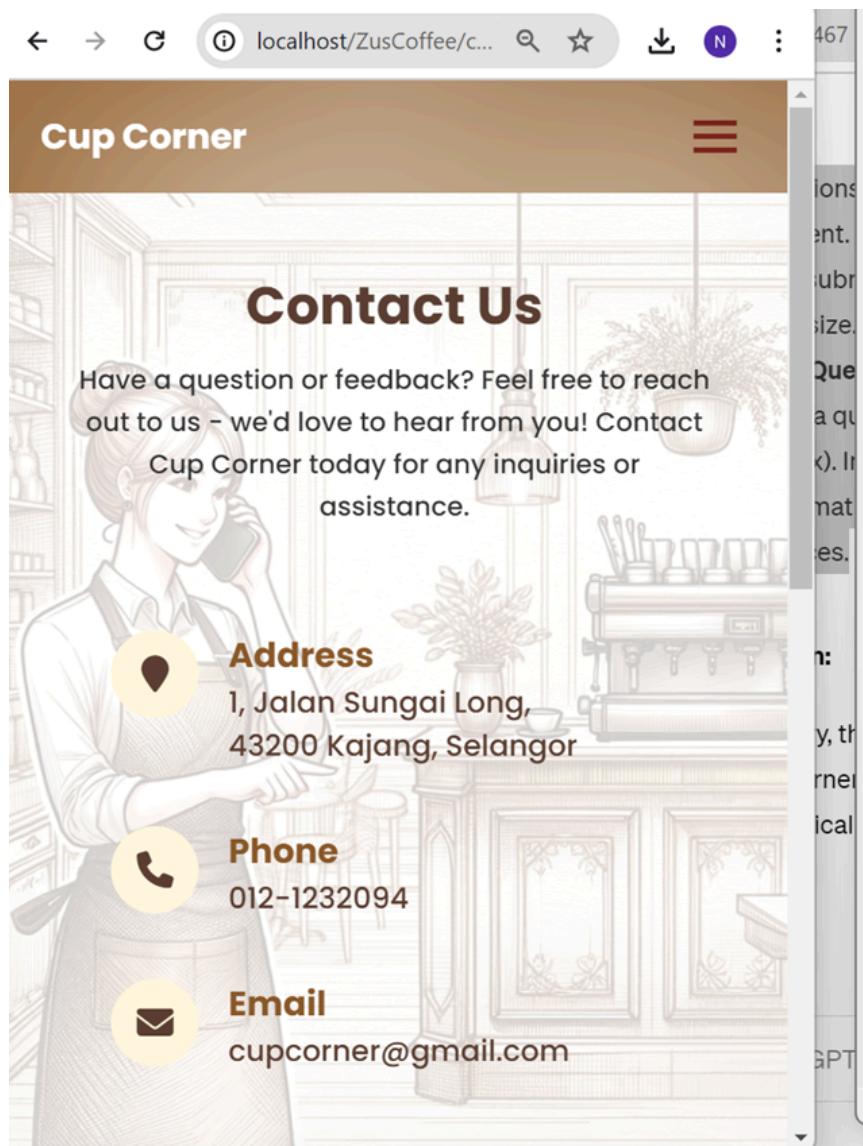
Email

Type your Message...

Send

Media Queries:

Media queries are used to adjust the layout and styling for smaller screens (max-width: 991px). In this case, the padding of the contact section is reduced, and the contact information and feedback form are stacked vertically for better responsiveness on smaller devices.



cart.php

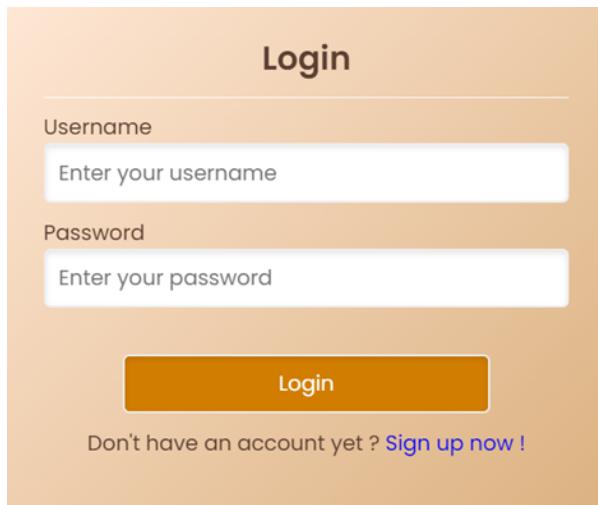
The "cart.php" file is responsible for displaying the shopping cart functionality on Cup Corner's website. It allows users to view the items they have added to their cart, adjust quantities, remove items, and proceed to checkout. The file retrieves cart data from the session and fetches corresponding product details from the database to display in the cart.

Session Initialization and Database Inclusion:

The `session_start()` function is called to initialize the session. Database connection and common functions are included using `include` statements.

User Authentication Check:

The code checks if the user is authenticated by verifying the existence of the valid key in the session. If not, the user is redirected to the login page before entering the cart webpage.

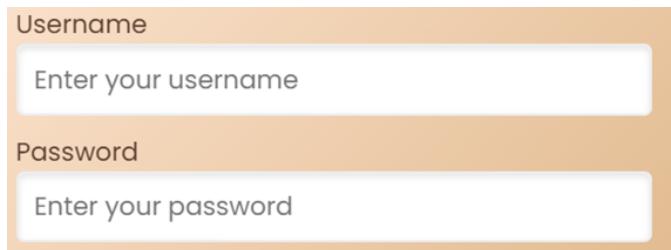


The image shows a login form with the following structure:

- Header:** The word "Login" is centered at the top in a dark brown font.
- Fields:** There are two input fields: "Username" and "Password". Each field has a placeholder text: "Enter your username" for the username field and "Enter your password" for the password field.
- Buttons:** A large orange button with the word "Login" in white is centered below the input fields.
- Text:** At the bottom, there is a line of text that reads "Don't have an account yet? [Sign up now!](#)"

Function to Retrieve User Information:

The `getUserInfo()` function is defined to retrieve user information from the session, specifically the user ID and username.



Username
Enter your username

Password
Enter your password

Cart Initialization:

If the cart session variable is not set, it is initialized as an empty array.

Fetching Cart Products:

The code iterates through each product in the cart session variable and retrieves corresponding product details (such as title, price, and image) from the database using SQL queries. The retrieved product details are stored in an array along with the quantity and total price for each product.

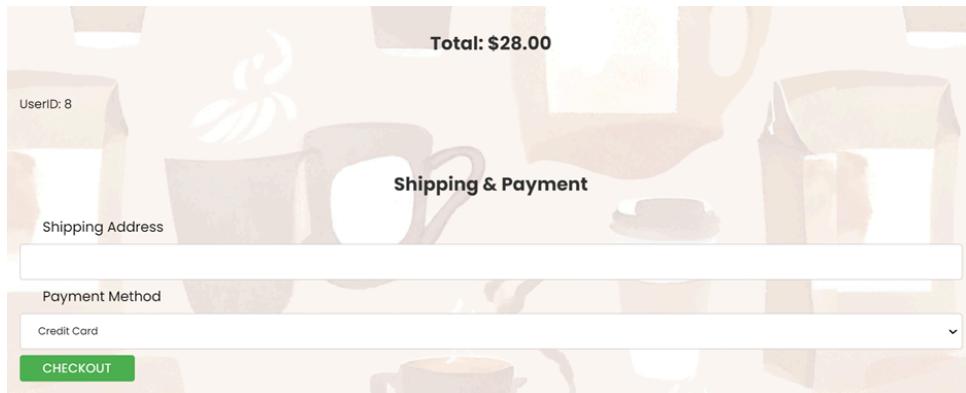
HTML Content Generation:

The HTML structure is generated to display the cart contents using PHP. If the cart is not empty, a table is created to list the items, quantities, prices, and an option to remove items. If the cart is empty, a message is displayed informing the user that the cart is empty.



Checkout Form:

A form is provided for users to enter shipping details and select a payment method. The input fields are provided for shipping address and a dropdown menu for selecting the payment method. A submit button allows users to proceed to checkout.



Total: \$28.00

UserID: 8

Shipping Address

Payment Method

Credit Card

CHECKOUT

JavaScript for Item Removal:

JavaScript code is included to handle the removal of items from the cart. Event listeners are added to the remove buttons, and when clicked, a POST request is sent to the server to remove the selected item from the cart. Upon successful removal, the page is refreshed to reflect the updated cart contents.



Item	Quantity	Price	
Americano	1	\$10.00	REMOVE
Asian Dolce Latte	1	\$18.00	REMOVE

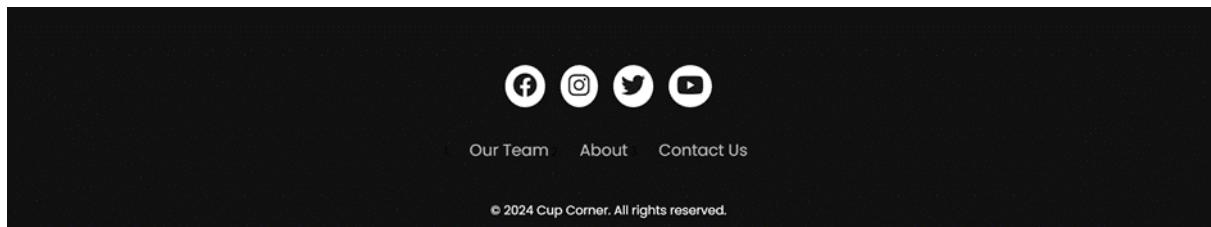


Cart			
Item	Quantity	Price	
Asian Dolce Latte	1	\$18.00	REMOVE

External Script and Library Inclusion:

External scripts for jQuery and Popper.js are included for additional functionality.

Footer content is included using include statements.

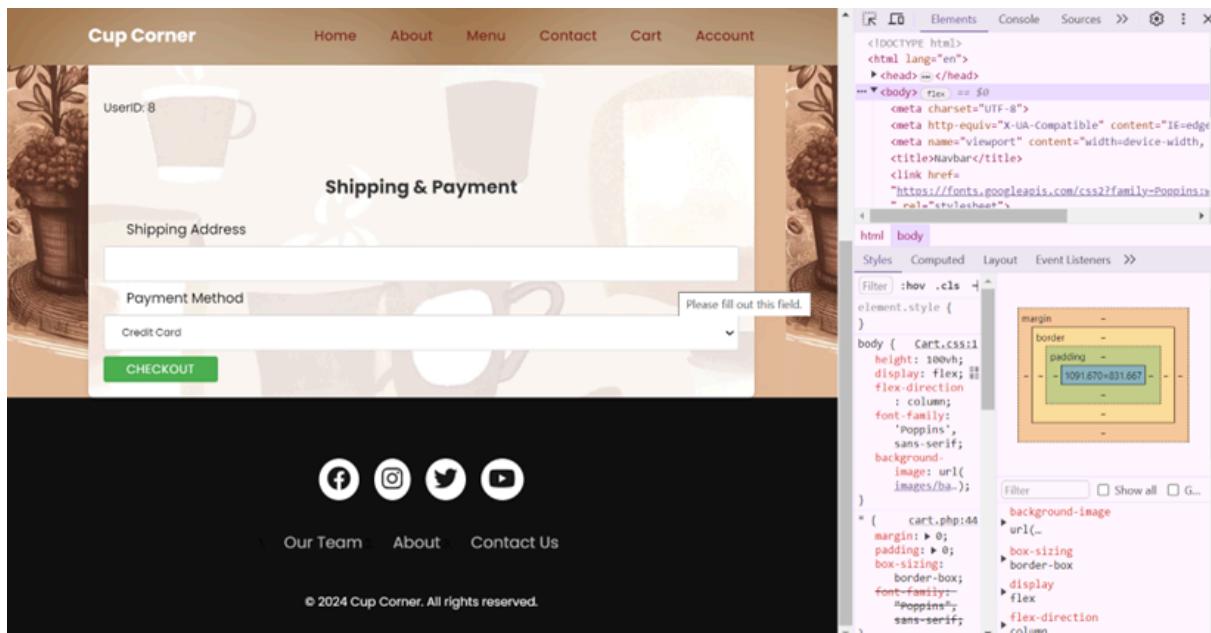


Cart.css

The "cart.php" file is responsible for displaying the shopping cart functionality on Cup Corner's website. It allows users to view the items they have added to their cart, adjust quantities, remove items, and proceed to checkout. This file primarily handles the presentation layer, utilizing HTML and CSS to create the cart interface, and incorporates PHP to dynamically fetch and display cart contents from the session and database.

Body Styles:

The body element is styled to occupy the full viewport height (100vh) and use flexbox (display: flex) with a column direction (flex-direction: column) to align its children vertically. The font-family is set to 'Poppins' and sans-serif as a fallback for text rendering consistency across different platforms. A background image is applied to the body using background-image, enhancing the visual appeal of the page.



Container Styles:

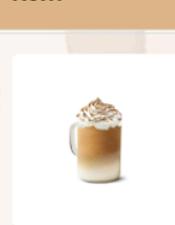
The .container class defines styles for the main container that holds the cart content. It sets the width to 80% of the parent element, centers it horizontally (margin: auto), and provides padding and border-radius for visual aesthetics. A box-shadow effect is applied to create depth and separation from the background.

Heading Styles:

Styles for h1 and h2 elements are specified to center-align text, set color, and add margin for spacing.

Table Styles:

The .cart-table class styles the cart table, ensuring it spans the full width and adds margin at the bottom for separation. The table cells (th and td) are styled with borders, padding, and text alignment. The header row (th) has a background color and distinct text color for visibility.

Item	Quantity	Price
 Asian Dolce Latte	1	\$18.00

Total: \$18.00

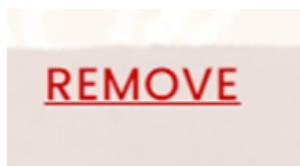
Image and Item Name Styles:

Styles for .cart-item-image and .cart-item-name define dimensions, border-radius, and color for consistency in displaying product images and names.



Remove Button Styles:

The .btn-remove class styles the remove buttons with color, cursor, font size, and transition effect for hover state.



When the cursor reached the word “remove”, it will shows underline.

Button Enhancements:

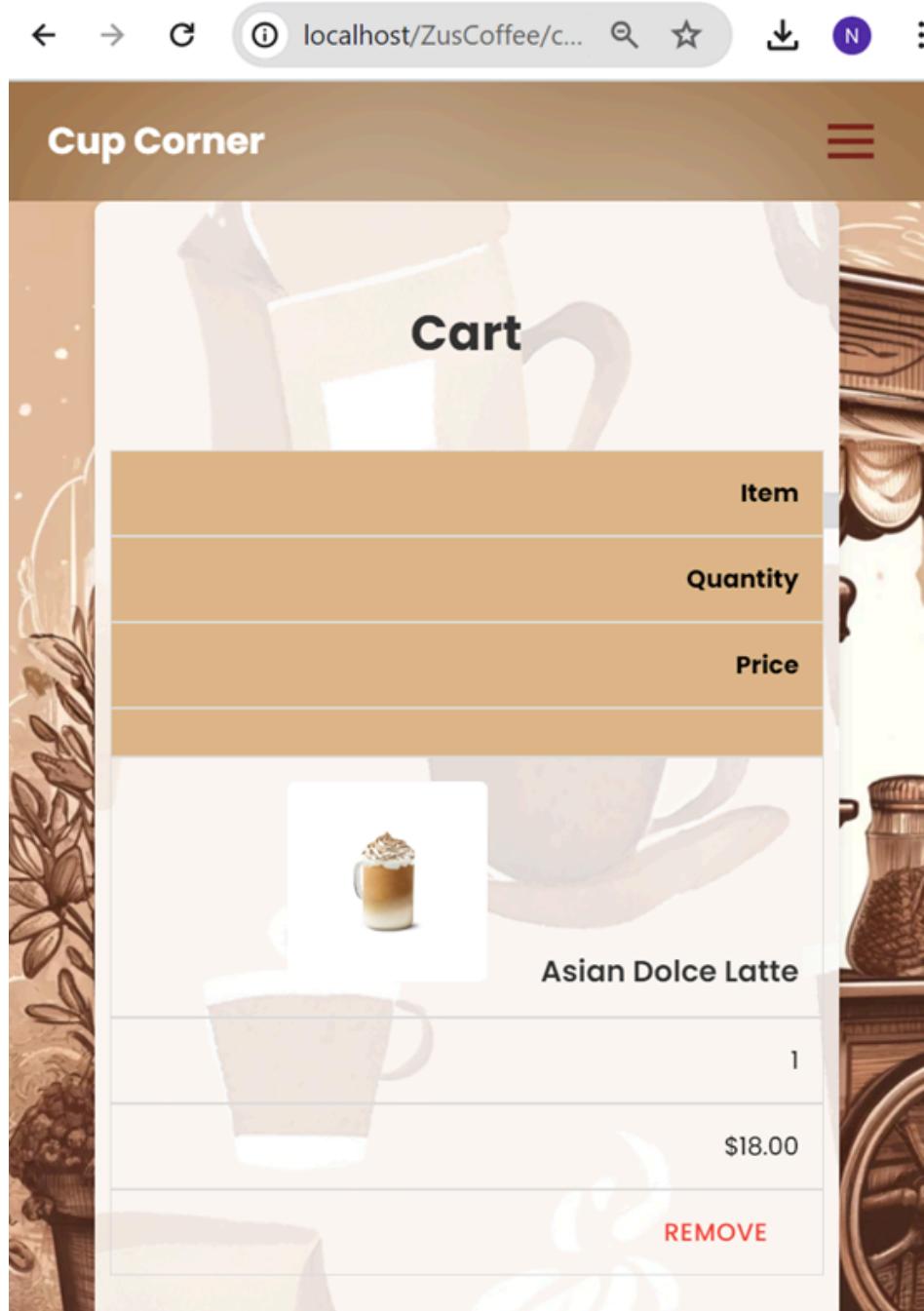
Both success and remove buttons (btn-success and btn-remove) share common styling for padding, border, border-radius, text transformation, and letter spacing.

Success button has a green background color while remove button inherits color from the theme.



Responsive Table Styles:

Media queries are used to make the cart table responsive for smaller screens (max-width: 768px). The table elements are displayed as blocks with full width, and header labels are shown before cell contents for clarity on narrow screens. Container padding is adjusted for better spacing on mobile devices.



The screenshot shows a mobile browser interface with a navigation bar at the top. The main content area is for a coffee shop named "Cup Corner". A large image of a coffee cup is the background. The word "Cart" is centered in the upper portion of the cart area. Below it is a table with three columns: "Item", "Quantity", and "Price". The table lists one item: "Asian Dolce Latte" with a quantity of "1" and a price of "\$18.00". At the bottom of the table is a red "REMOVE" button. The browser's address bar shows the URL "localhost/ZusCoffee/c...".

Item	Quantity	Price
Asian Dolce Latte	1	\$18.00

REMOVE

Additional Button and Input Styles:

Styles for text inputs and select elements (form-select) are provided for consistency and usability. Hover effect is added to buttons and submit inputs for visual feedback.

ourteam.php

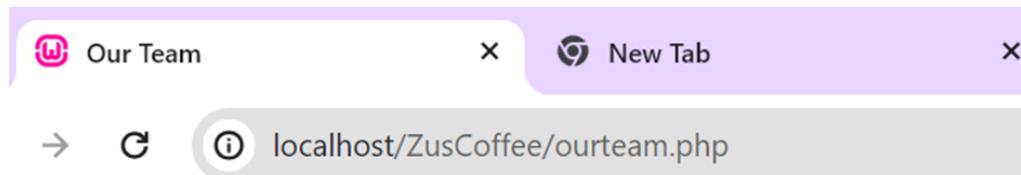
The "ourteam.php" file showcases the team members of Cup Corner, providing a glimpse into the individuals who contribute to the success and operation of the business. It utilizes HTML, CSS, and PHP to create a visually appealing and informative section on the website where visitors can learn about key team members and their roles within the company.

HTML Structure:

The file begins with the standard HTML5 doctype declaration and specifies the language as English. The meta tags are included to define the viewport for responsive design.

Title and External Resources:

The page title is set to "Our Team" for clarity. The external stylesheets for the navbar, custom styles, and font-awesome icons are linked to enhance the visual presentation.



Navbar Inclusion:

PHP is used to include the navbar component from a separate file (include/navbar.php), ensuring consistency across the website.

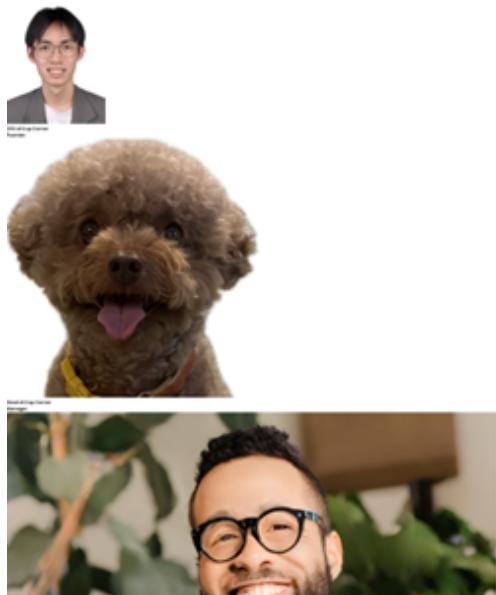


Content Section:

The main content section is wrapped in a <section> element for semantic structuring. Within this section, a container div (<div class="container">) holds individual cards representing team members.

Team Member Cards:

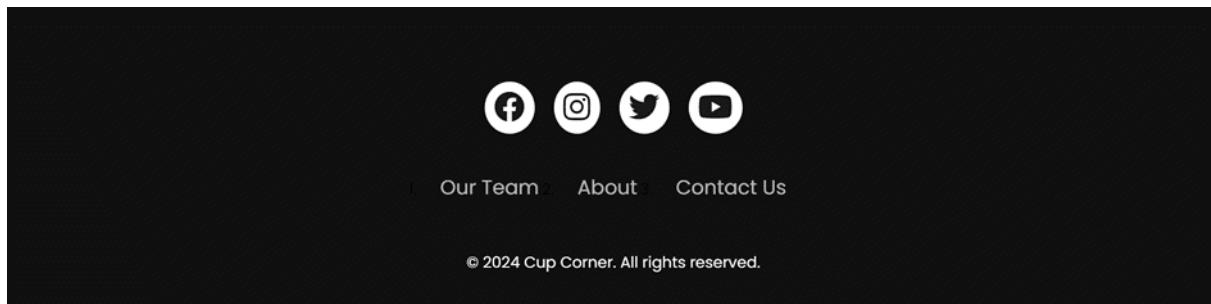
Each team member is represented by a card (<div class="card">) containing an image and text content. The image () of the team member is placed inside a div with the class imgBx. Text content, including the team member's role and designation, is contained within a div with the class contentBx.



There are three team member cards displayed, each with an image representing the team member and their respective role and designation within Cup Corner. The images are sourced from the "images" directory within the project folder, providing visual representation of team members.

PHP Footer Inclusion:

PHP is used again to include the footer component from a separate file (include/footer.php), ensuring consistent footer across the website.



our.css

The "our.css" file contains styles for the team section of the Cup Corner website. It defines the layout, appearance, and animations applied to the team member cards displayed on the page. The styles aim to create an attractive and engaging presentation of the team members, enhancing the overall user experience.

Reset Styles:

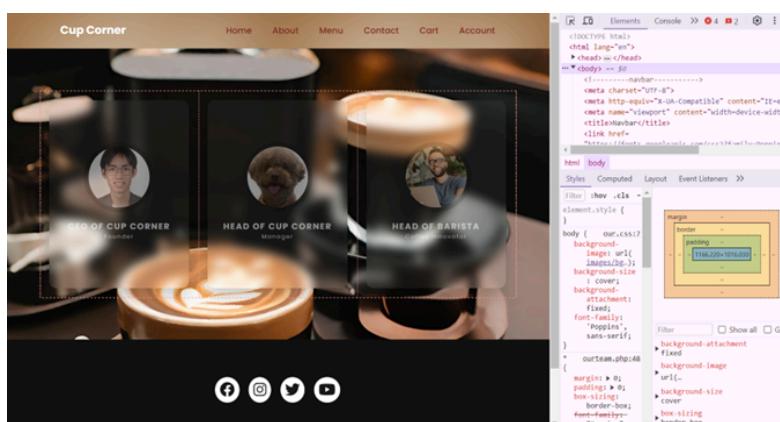
The universal selector (*) is used to reset margins, paddings, and set the font family to 'Poppins' for consistency across elements.

Body Styles:

Background image properties are applied to the body to set a visually appealing background for the team section. The background image is fixed and covers the entire viewport.

Section Styles:

Padding, flexbox properties (display, flex-direction, align-items, justify-content), and background properties are applied to the <section> element to create spacing and center the content vertically and horizontally.

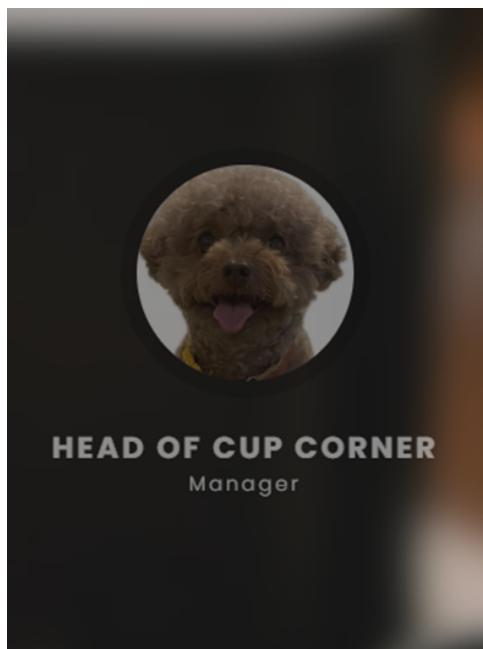


Container Styles:

The container class styles the wrapper for the team member cards. Flexbox properties are used to center the container both horizontally and vertically. The z-index is applied to ensure the cards appear above the background image. Besides, Flex-wrap is set to wrap, allowing cards to wrap onto a new line if the container width is exceeded.

Card Styles:

Each team member card is represented by the .card class. Specific dimensions, background color with transparency, margin, box-shadow, and border-radius properties are applied to create a card-like appearance. The Flexbox properties are used to center the content within each card. Then, the backdrop-filter is applied to create a blur effect on the background of the card.



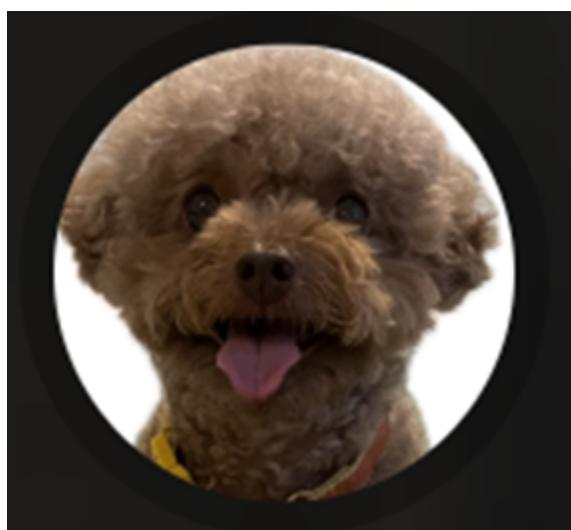
Content Styles:

The .content class styles the content within each card. The opacity and transition properties are applied to create a hover effect where the content becomes more visible and moves slightly upward. The Flexbox properties are used to center the content vertically and horizontally within each card.



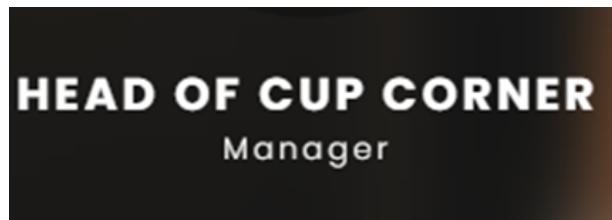
Image Box Styles:

The .imgBx class styles the container for the team member's image. The specific dimensions, border-radius, overflow, and border properties are applied to create a circular frame for the image.



Text Styles:

Heading styles are defined for the team member's name and designation within the `.contentBx h3` element. Text color, text-transform, letter-spacing, font-weight, font-size, margin, and line-height properties are applied to enhance readability and visual appeal. Apart from that, additional styles are applied to the `` element within the heading for secondary text.



login.php

The "login.php" file is responsible for providing a login interface for users to access the Cup Corner website. It handles user input, validates login credentials, and grants access to authorized users. Additionally, it displays error messages if the login attempt fails.

Session Initialization:

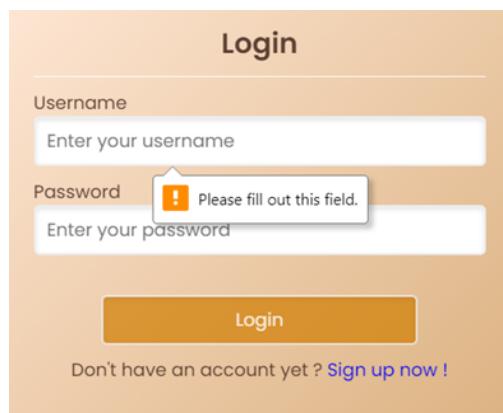
`session_start()` is used to initialize a session or resume the current session if one exists.

Including Configuration File:

The config.php file is included, which likely contains database connection information and other configuration settings.

Variable Initialization:

`$error` is initialized to an empty string. This variable will hold error messages if the login attempt fails.



The image shows a screenshot of a login form. The form has a light orange header with the word "Login". Below the header, there are two input fields: "Username" and "Password". The "Username" field contains the placeholder text "Enter your username". The "Password" field contains the placeholder text "Enter your password" and has a red exclamation mark icon with the text "Please fill out this field." above it. Below the input fields is a large orange "Login" button. At the bottom of the form, there is a link that says "Don't have an account yet? [Sign up now!](#)".

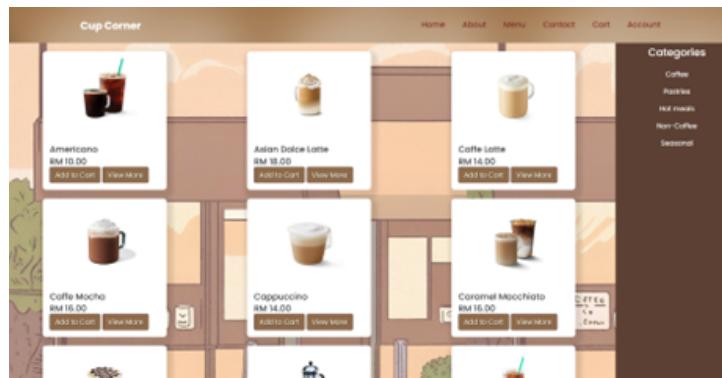
Login Form Processing:

Upon form submission (`$_POST['submit']`), the username and password entered by the user are retrieved. A SQL query is constructed to select a user from the database where the provided username and password match. The query is executed using `mysqli_query()`.

Handling Query Results:

If the query returns one or more rows (`mysqli_num_rows($result) > 0`), indicating a successful login attempt:

- The user's ID is stored in the session (`$_SESSION['userId']`).
- The valid session variable is set to true.
- The user is redirected to the "menu.php" page.
- `exit()` is called to stop further script execution.



If the query returns no rows, indicating an unsuccessful login attempt:

An error message is assigned to the `$error` variable.

HTML Structure:

The HTML structure includes a login form with input fields for username and password. The form action is left empty (action=""), indicating that the form submits data to the same page. The error messages are displayed within a <div> element with the class message if \$error is not empty.

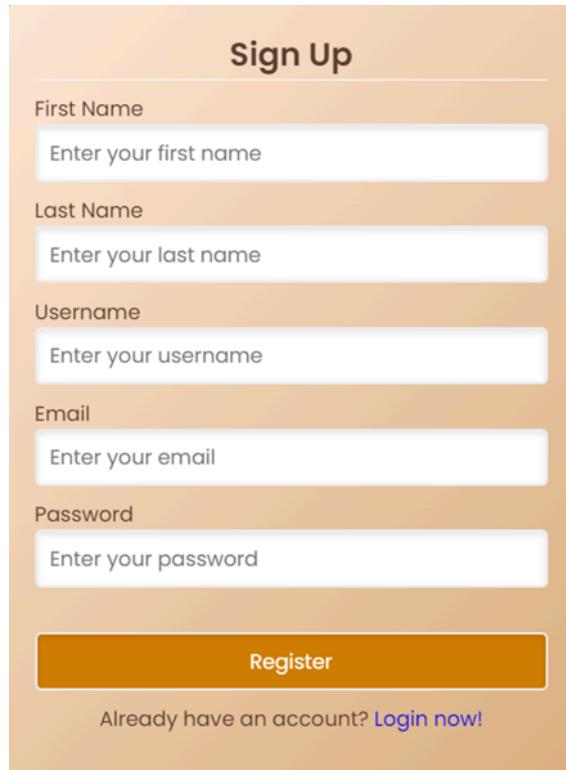
Input Fields:

Input fields for username and password have associated <label> elements for accessibility.

The required attribute is added to ensure that both fields are filled before submitting the form.

Link to Registration Page:

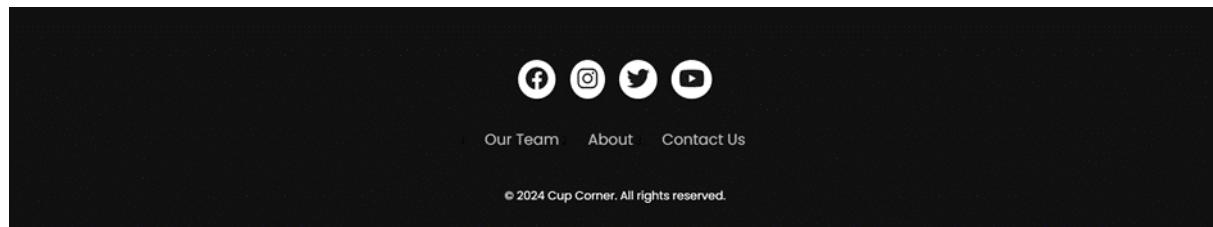
A link is provided for users who don't have an account yet. Clicking the link redirects users to the "register.php" page.



The image shows a 'Sign Up' form interface. The background is a light orange color. At the top, the word 'Sign Up' is centered in a dark green font. Below it is a horizontal line. The form consists of six input fields, each with a label and a text input box. The labels are: 'First Name', 'Last Name', 'Username', 'Email', and 'Password'. Each label is followed by a text input box with a placeholder text: 'Enter your first name', 'Enter your last name', 'Enter your username', 'Enter your email', and 'Enter your password'. Below these input fields is a large orange button with the word 'Register' in white. At the bottom of the form, there is a link in blue text that says 'Already have an account? [Login now!](#)'.

Including Footer:

The footer.php file is included at the bottom of the page, likely containing common footer elements and scripts.

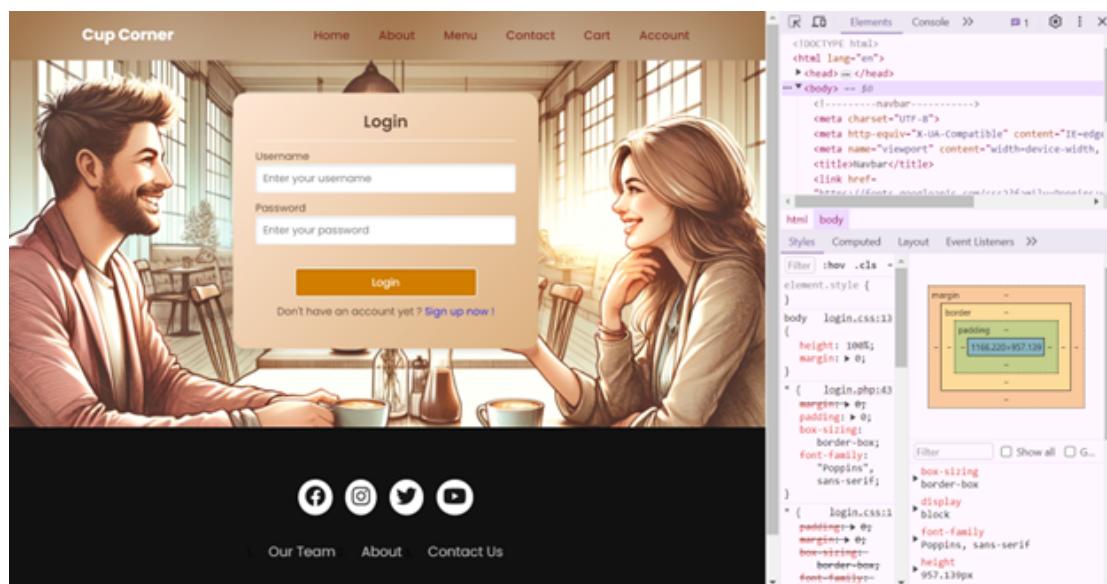


login.css

The "login.css" file contains the styles for the login page of the Cup Corner website. It defines the layout, colors, fonts, and other visual aspects to create an appealing and user-friendly login interface.

Global Styles:

The * selector resets the default margin, padding, and box-sizing properties to ensure consistent styling across different elements. The font-family property sets the default font to "Poppins" or a fallback sans-serif font.



Root Variable:

:root defines custom CSS variables, such as --primary-color, which is set to a shade of orange (#d17d00). These variables can be used throughout the stylesheet for easy color management.

Body Styles:

Sets the height of the body to 100% and removes any default margin.

Container Styles:

Defines styles for the container element that holds the login form. The uses flexbox to center the form vertically and horizontally. We apply a background image to the container with properties to ensure it covers the entire viewport.



Box Styles and Form Box Styles:

Styles the box containing the login form with a gradient background, padding, border-radius, box-shadow, and text color. Then, we create a visually appealing container for the login form. We also set the width and margin for the form box. Styles of the heading of the form with font size, weight, and border-bottom.

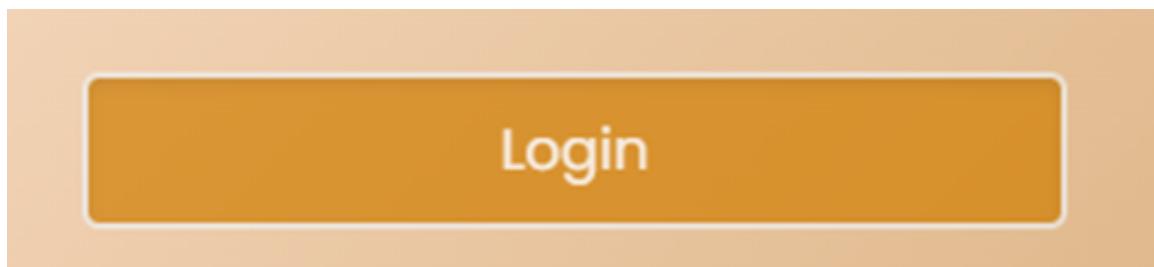


Input Styles:

For the input styles, we define styles for input fields within the form, including height, width, font size, padding, border-radius, and border. Then adding transitions for smooth visual effects on input focus.

Button Field and Button Styles:

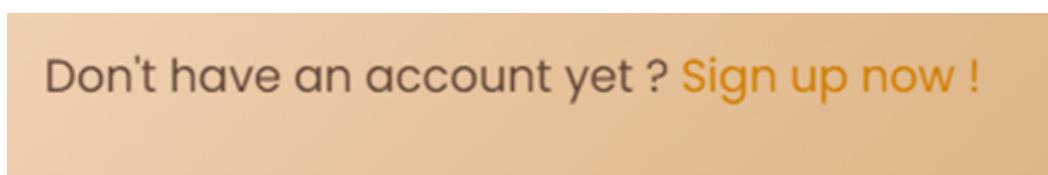
We set the width and margin for the button field. Besides, we also style the login button with background color, border-radius, text color, font size, cursor, and transitions for hover effects.



When the cursor reaches the button, it will turn into light orange.

Link Styles:

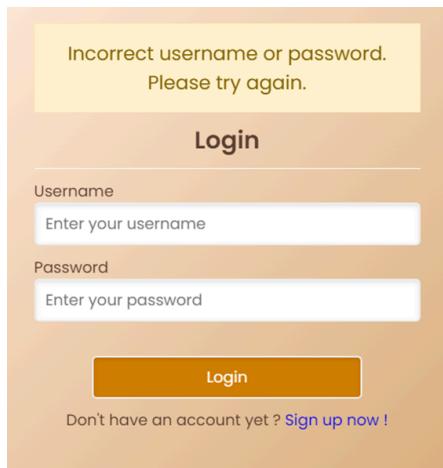
The main part of this section is to styling the link for registering a new account with text decoration, color, and hover effects.



The “Sign up now” will turn into a light orange colour when the cursor reaches there.

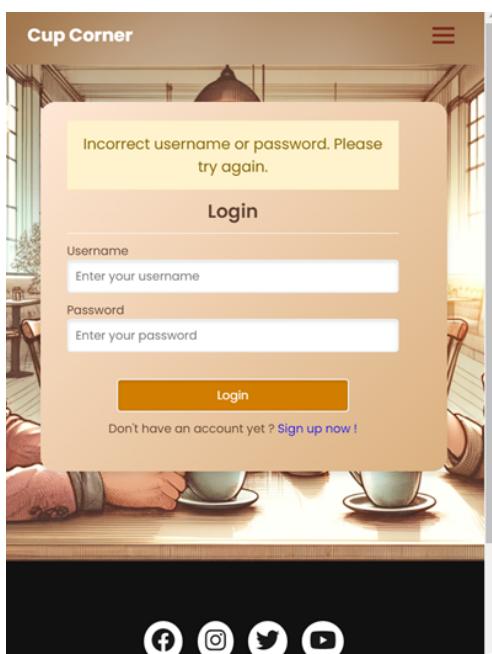
Error Message Styles:

Then, the code helps to define styles for error messages displayed if the login attempt fails and sets font size, text alignment, background color, padding, border, and text color.



Media Queries:

The functions help to adjust the layout for smaller screen sizes using media queries. Besides, we reduce the width of the form box for screens up to 768px wide. Then, it also Sets the form box width to 100% for screens up to 576px wide, ensuring responsiveness on smaller devices.



register.php

The "register.php" file is responsible for handling user registration on the Cup Corner website. It provides a form where users can enter their personal information such as first name, last name, username, email, and password to create a new account.

PHP Session and Configuration:

The PHP session is started to maintain user session data. The "config.php" file is included, which contains the database configuration details.

HTML Structure:

The file follows the standard HTML5 structure with a <head> section containing metadata and a <body> section for the page content.

Header Section:

Includes the website's navigation bar using the include PHP function.



Container and Box:

Defines a container and a box to visually encapsulate the registration form.

Sign Up

First Name	<input type="text" value="Enter your first name"/>
Last Name	<input type="text" value="Enter your last name"/>
Username	<input type="text" value="Enter your username"/>
Email	<input type="text" value="Enter your email"/>
Password	<input type="text" value="Enter your password"/>
<input type="button" value="Register"/>	

Already have an account? [Login now!](#)

Form Handling:

At this part, we check if the form has been submitted using `isset($_POST["submit"])`.

- If the form is submitted, it retrieves the user input values for first name, last name, username, email, and password.
- Validates whether the email and username are already registered in the database.
- If the email or username is not already in use, insert the user data into the database.
- Displays appropriate success or error messages based on the registration outcome.

Sign Up

First Name

Last Name

Username

Email

 Please include an '@' in the email address. 'a' is missing an '@'.

Already have an account? [Login now!](#)

Username is already used! Please try another one.

[Go Back](#)

Registration Form:

It displays the registration form with input fields for first name, last name, username, email, and password. Then the use of HTML `<input>` elements with appropriate attributes such as type, placeholder, and name. This also includes a submit button to submit the form data.

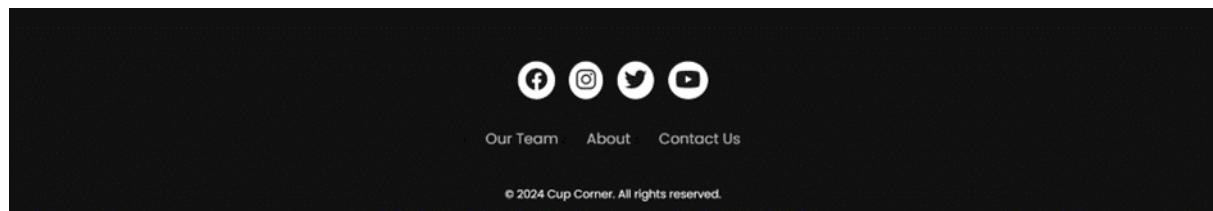
Link to Login Page:

This provides a link to the login page for users who already have an account. It enables users to easily navigate between the registration and login pages.

Already have an account? Login now!

Footer Section:

Includes the website's footer using the include PHP function.

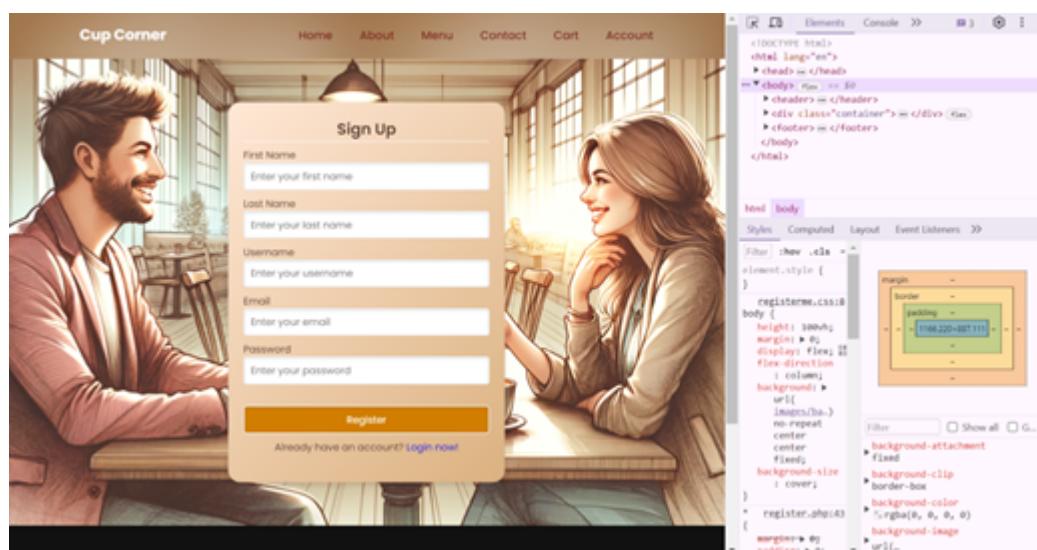


registerme.css

The "registerme.css" file is responsible for styling the registration form on the Cup Corner website. It defines the visual appearance and layout of various elements such as form inputs, buttons, messages, and links to create a cohesive and user-friendly registration experience.

Global Styles:

- Applies global styles using the universal selector * to reset padding, margin, and box-sizing.
- Sets the font family to 'Poppins' and sans-serif as fallback.



Body Styling:

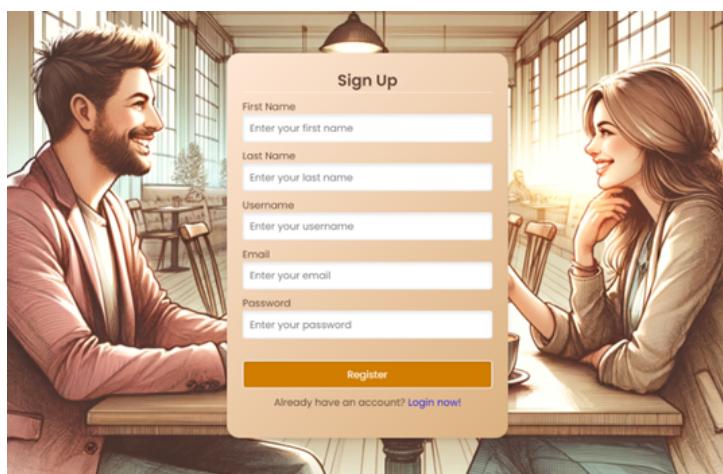
- Sets the body's height to 100vh (viewport height) to ensure full-screen display.
- Uses flexbox to vertically center the content by setting the display property to flex and flex-direction to column.
- Applies a background image with background property to the body using a URL.

Header, Main, Footer:

Sets the width of the header, main content, and footer to 100% to occupy the entire viewport width.

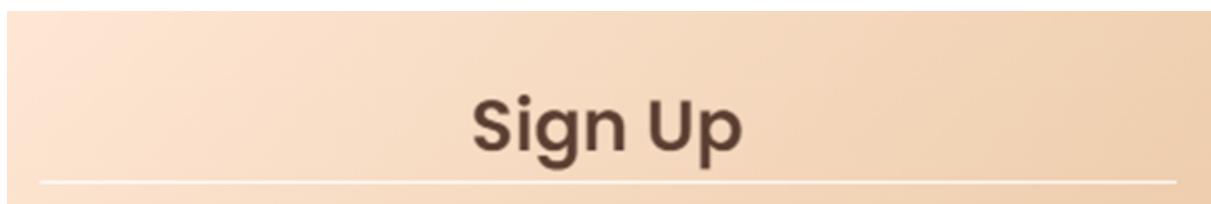
Container and Box:

- Defines a container to hold the registration form, using flexbox to centre its contents vertically and horizontally.
- Styles the box containing the form with a gradient background, padding, border-radius, and box-shadow to create a visually appealing container.



Form Box:

- Specifies the width of the form box to maintain consistency and readability.
- Sets the form box's heading (h1) style with a larger font size, bold font weight, and center alignment. Adds a bottom border for visual separation.



Input Fields:

- Styles input fields using flexbox to create a column layout with proper spacing.
- Sets the height, width, font size, padding, border, border radius, and box-shadow for input fields to ensure a consistent and user-friendly appearance.
- Uses outline: none; to remove the default focus outline.

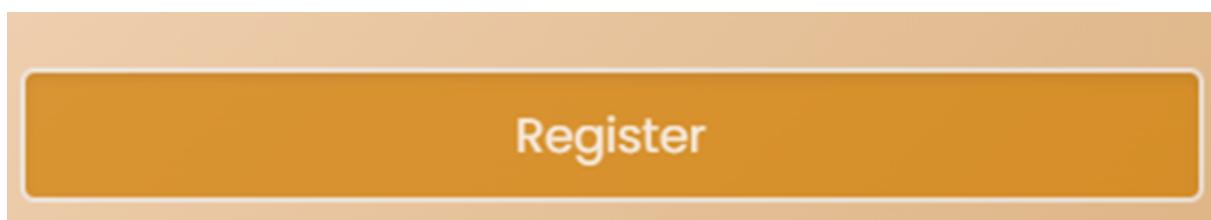
The image shows a 'Sign Up' form with the following fields:

- First Name: Input field with placeholder 'Enter your first name'.
- Last Name: Input field with placeholder 'Enter your last name'.
- Username: Input field with placeholder 'Enter your username'.
- Email: Input field with placeholder 'Enter your email'.
- Password: Input field with placeholder 'Enter your password'.

Below the input fields is a large orange 'Register' button. At the bottom of the form, there is a link 'Already have an account? [Login now!](#)'.

Buttons:

- Styles the registration button (btn) with specific height, background color, border, border radius, text color, font size, and cursor pointer.
- Adds transition effects for smoother hover interactions.



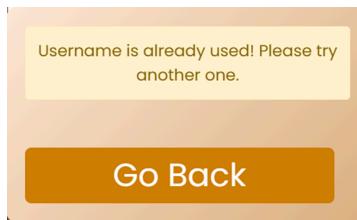
The buttons will turn into light orange when the cursor reaches there.

Links and Messages:

- Defines styles for links (a) and messages (message and message2) to enhance readability and visual appeal.
- Sets colors, font size, padding, border-radius, and border properties for messages to differentiate between success and error messages.

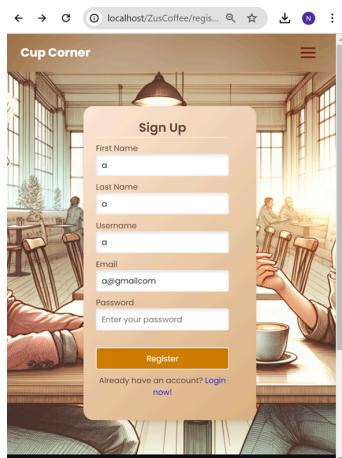
Already have an account? [Login now!](#)

The “Login now!” will change into light orange when the cursor reaches there.



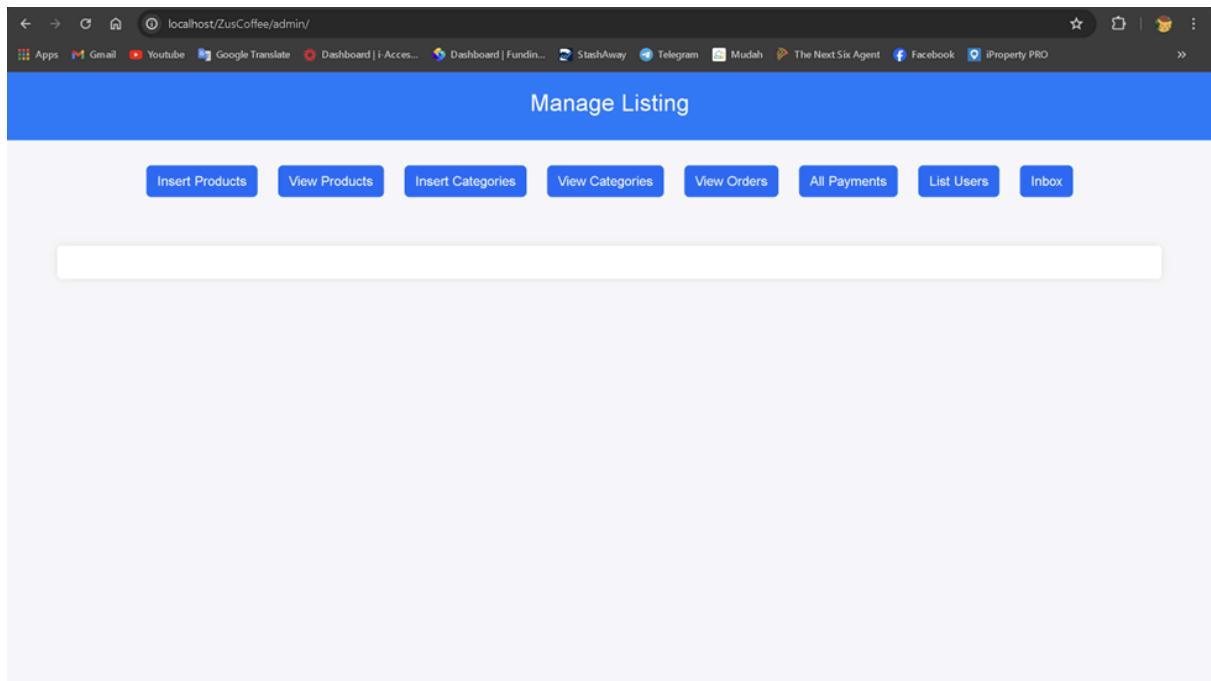
Media Queries:

- Utilizes media queries to ensure responsiveness across different screen sizes.
- Adjusts the width of the form box to accommodate smaller screens, maintaining readability and usability.



Admin

admin/index.php



1. HTML Structure:

- `<!DOCTYPE html>`: Specifies the document type and version of HTML being used.
- `<head>`: Contains metadata about the document, such as character encoding, viewport settings, and the title of the page.
- `<body>`: Encloses the visible content of the webpage.

2. Bootstrap CSS:

- `<link>`: Includes the Bootstrap CSS framework from a CDN (Content Delivery Network).
- Bootstrap provides a robust set of predefined styles and components for creating responsive web layouts.

3. Header Section:

- `<div class="header">`: Represents a container with the class "header", defining the header section of the webpage.
- The header has a blue background color (#007bff) and white text color (#fff) for contrast.

- It contains a centered **<h3>** heading displaying the title "Manage Listing".

4. Button Container:

- **<div class="button-container">**: Defines a container for organizing and styling buttons.
- Inside this container, there are several **<a>** elements representing clickable buttons.
- Each button is styled using Bootstrap classes (**btn btn-primary**) to create a uniform appearance.
- The **href** attribute of each button points to **index.php** with different query parameters to indicate different actions.

5. Content Container:

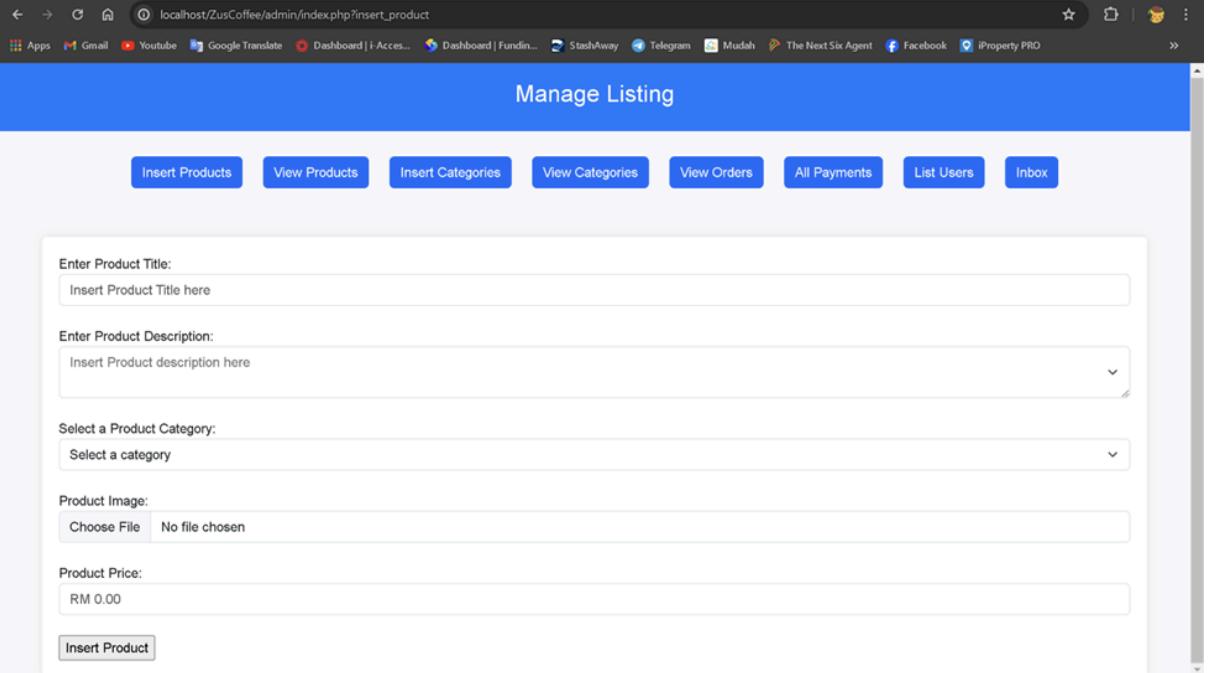
- **<div class="container">**: Sets up a container for the main content area of the webpage.
- This container has a white background color and rounded corners (**border-radius: 5px**), giving it a visually distinct appearance.

6. PHP Logic:

- Inside the content container, PHP code is used to dynamically include specific files based on the query parameters received via **\$_GET**.
- **isset(\$_GET['parameter'])** checks if a particular query parameter is present in the URL.
- If a specific parameter is detected (e.g., **insert_product**, **view_orders**), the corresponding PHP file is included using **include()**.
- This dynamic inclusion mechanism allows the webpage to display different content based on the user's interactions with the buttons.

Overall, this HTML and PHP code combination creates a structured and visually appealing admin panel interface. The use of Bootstrap ensures consistent styling and responsiveness across different devices, while the PHP logic enables dynamic content generation based on user actions.

admin/insert_product.php



The screenshot shows a web browser window with the URL `localhost/ZusCoffee/admin/index.php?insert_product`. The page title is "Manage Listing". Below the title, there is a horizontal navigation bar with buttons for "Insert Products", "View Products", "Insert Categories", "View Categories", "View Orders", "All Payments", "List Users", and "Inbox". The main content area is titled "Enter Product Title:" with a text input field containing "Insert Product Title here". Below it is "Enter Product Description:" with a text input field containing "Insert Product description here". There is a dropdown menu titled "Select a Product Category:" with the placeholder "Select a category". Under "Product Image:", there is a file input field showing "Choose File No file chosen". Under "Product Price:", there is a text input field showing "RM 0.00". At the bottom of the form is a "Insert Product" button.

1. PHP Include:

- `include('../include/connect.php');`: This line includes the PHP file **connect.php** located in the **include** directory one level above the current directory. This file likely contains code to establish a database connection.

2. HTML Form:

- The form allows users to input details for a new product.
- It uses the **POST** method to send form data to the same page ("") and allows file uploads (`enctype="multipart/form-data"`).
- Input fields include:
 - Product title (**product_title**)
 - Product description (**product_description**)
 - Product category (**product_category**) with options dynamically populated from the **categories** table in the database.
 - Product image (**product_image**) uploaded via file input.
 - Product price (**product_price**).

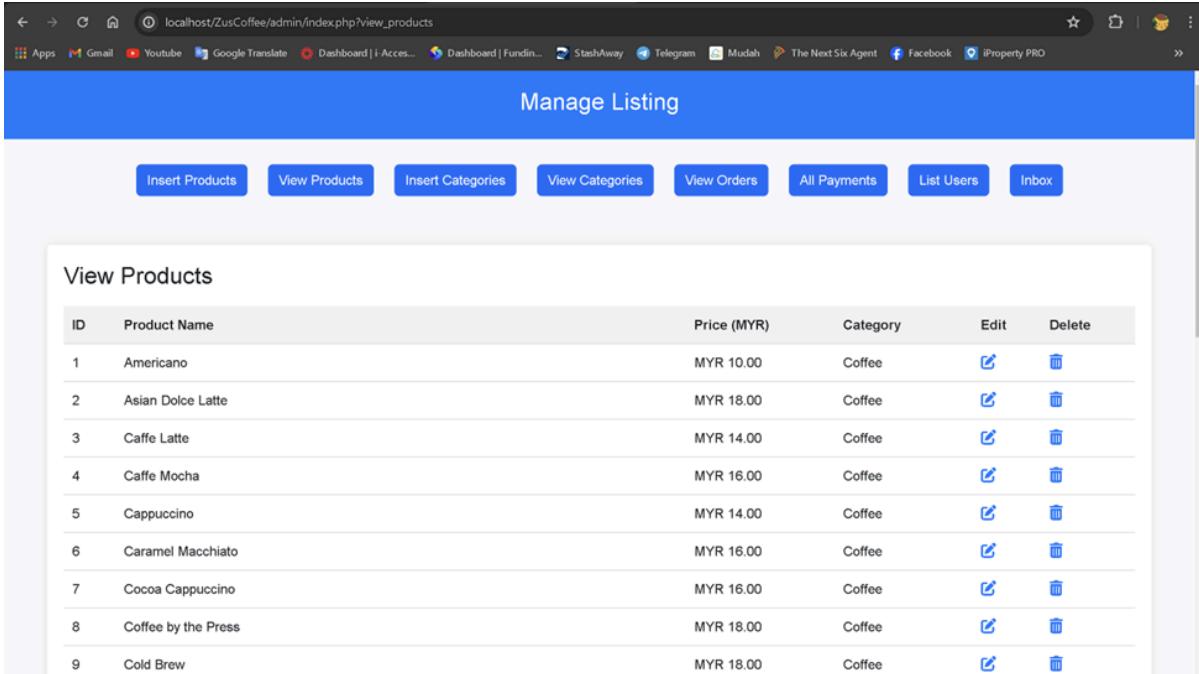
3. PHP Logic:

- `if(isset($_POST['insert_product']))`: Checks if the form has been submitted by detecting the presence of the **insert_product** submit button.
- Inside this conditional block, form data is retrieved using `$_POST`.

- The script checks if any of the form fields are empty. If so, it displays an alert message.
- If all fields are filled, the product image is moved from its temporary location to the **product_images** directory.
- An SQL **INSERT** query is constructed to add the new product details into the **products** table in the database.
- The query is executed using **mysqli_query()**.
- If the insertion is successful, a success message is displayed via JavaScript alert.

Overall, this code segment allows administrators to insert new products into the database through a form interface. It validates form inputs and handles the file upload for product images, ensuring that all necessary data is captured and stored correctly.

admin/view_products.php



The screenshot shows a web browser window with the URL `localhost/ZusCoffee/admin/index.php?view_products`. The page has a blue header bar with the title "Manage Listing". Below the header is a navigation bar with buttons: "Insert Products", "View Products", "Insert Categories", "View Categories", "View Orders", "All Payments", "List Users", and "Inbox". The main content area is titled "View Products" and contains a table with 9 rows of product data. The table has columns: ID, Product Name, Price (MYR), Category, Edit, and Delete. Each row includes a "Edit" button (pencil icon) and a "Delete" button (trash icon). The products listed are: 1. Americano, 2. Asian Dolce Latte, 3. Caffe Latte, 4. Caffe Mocha, 5. Cappuccino, 6. Caramel Macchiato, 7. Cocoa Cappuccino, 8. Coffee by the Press, and 9. Cold Brew.

ID	Product Name	Price (MYR)	Category	Edit	Delete
1	Americano	MYR 10.00	Coffee		
2	Asian Dolce Latte	MYR 18.00	Coffee		
3	Caffe Latte	MYR 14.00	Coffee		
4	Caffe Mocha	MYR 16.00	Coffee		
5	Cappuccino	MYR 14.00	Coffee		
6	Caramel Macchiato	MYR 16.00	Coffee		
7	Cocoa Cappuccino	MYR 16.00	Coffee		
8	Coffee by the Press	MYR 18.00	Coffee		
9	Cold Brew	MYR 18.00	Coffee		

1. Database Connection:

- `include('..../include/connect.php');`: This line includes the PHP file **connect.php**, which likely contains code to establish a database connection.

2. Query to Retrieve Products:

- `$select_query`: SQL query to select all products from the database along with their corresponding categories.
- The query joins the **products** table (**p**) with the **categories** table (**c**) based on the **category_id** foreign key.

3. HTML Structure:

- The HTML structure includes a table to display the products retrieved from the database.
- It utilizes Bootstrap for styling.

4. Table Headers:

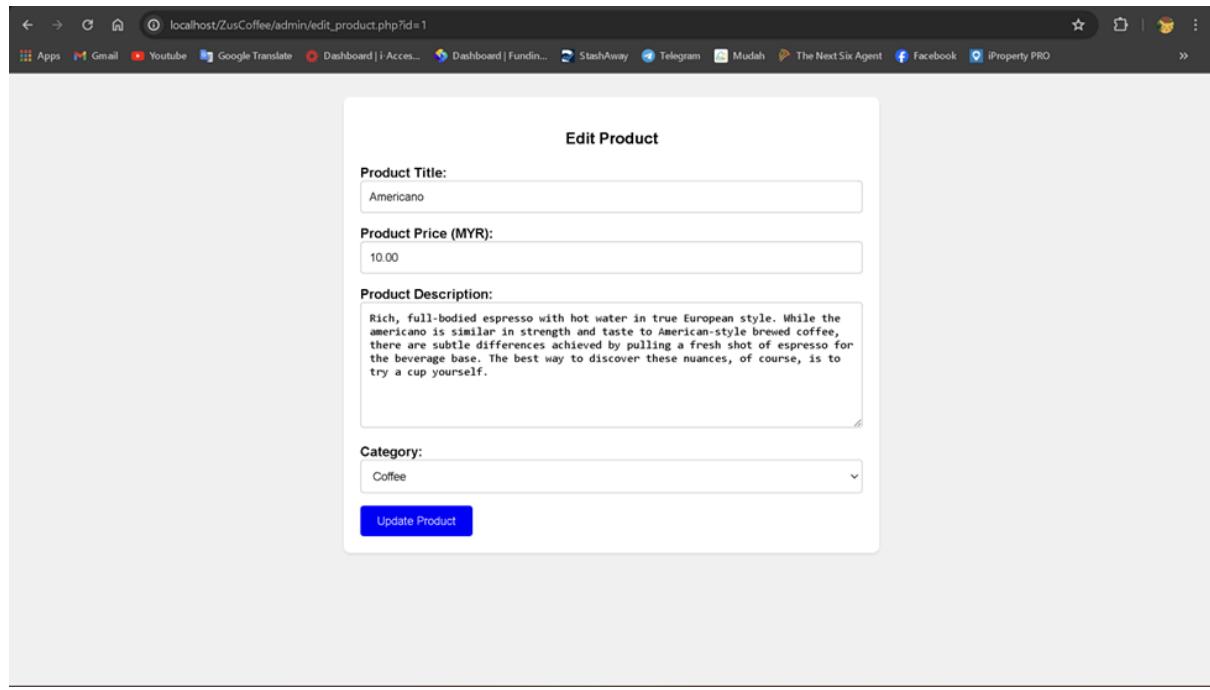
- The table headers include columns for product ID, product name, price, category, edit action, and delete action.

5. PHP Logic:

- Inside the table body, PHP code fetches and displays product data retrieved from the database.
- `mysqli_fetch_assoc($result)` fetches each row from the query result as an associative array.
- The fetched data is used to populate table rows (`<tr>`) and cells (`<td>`).
- Product price is formatted as MYR (Malaysian Ringgit) using `number_format()` for better readability.
- Edit and delete actions are represented by icons from the Font Awesome library (`<i class='fas fa-edit icon'></i>` for edit and `<i class='fas fa-trash-alt icon'></i>` for delete).
- Edit action links to `edit_product.php` with the product ID as a query parameter (`id`).
- Delete action links to `delete_product.php` with the product ID as a query parameter (`id`) and includes a confirmation prompt using JavaScript `confirm()`.

Overall, this code segment retrieves product data from the database and presents it in a tabular format, allowing administrators to view and manage products. It includes functionality for editing and deleting products, enhancing the admin interface's usability.

admin/edit_product.php



The screenshot shows a web browser window with the URL `localhost/ZusCoffee/admin/edit_product.php?id=1`. The page title is "Edit Product". The form contains the following fields:

- Product Title:** Americano
- Product Price (MYR):** 10.00
- Product Description:** Rich, full-bodied espresso with hot water in true European style. While the americano is similar in strength and taste to American-style brewed coffee, there are subtle differences achieved by pulling a fresh shot of espresso for the beverage base. The best way to discover these nuances, of course, is to try a cup yourself.
- Category:** Coffee

At the bottom is a blue "Update Product" button.

1. Database Connection:

- `include('../include/connect.php');`: Includes the PHP file `connect.php`, establishing a connection to the database.

2. Product Retrieval:

- The code checks if a product ID is provided in the URL (`$_GET['id']`).
- If provided, it fetches the details of the product from the database using the provided ID.
- If the product does not exist, the page redirects to the view products page.

3. Category Retrieval:

- It retrieves all categories from the database to populate a dropdown list in the form.

4. Form Submission:

- If the form is submitted (`isset($_POST['update_product'])`), it retrieves the form data.
- It updates the product details in the database with the provided information.
- After updating, it redirects to the view products page.

5. HTML Structure:

- The HTML structure includes a form to edit the product details.
- It uses Bootstrap for styling.

6. Form Fields:

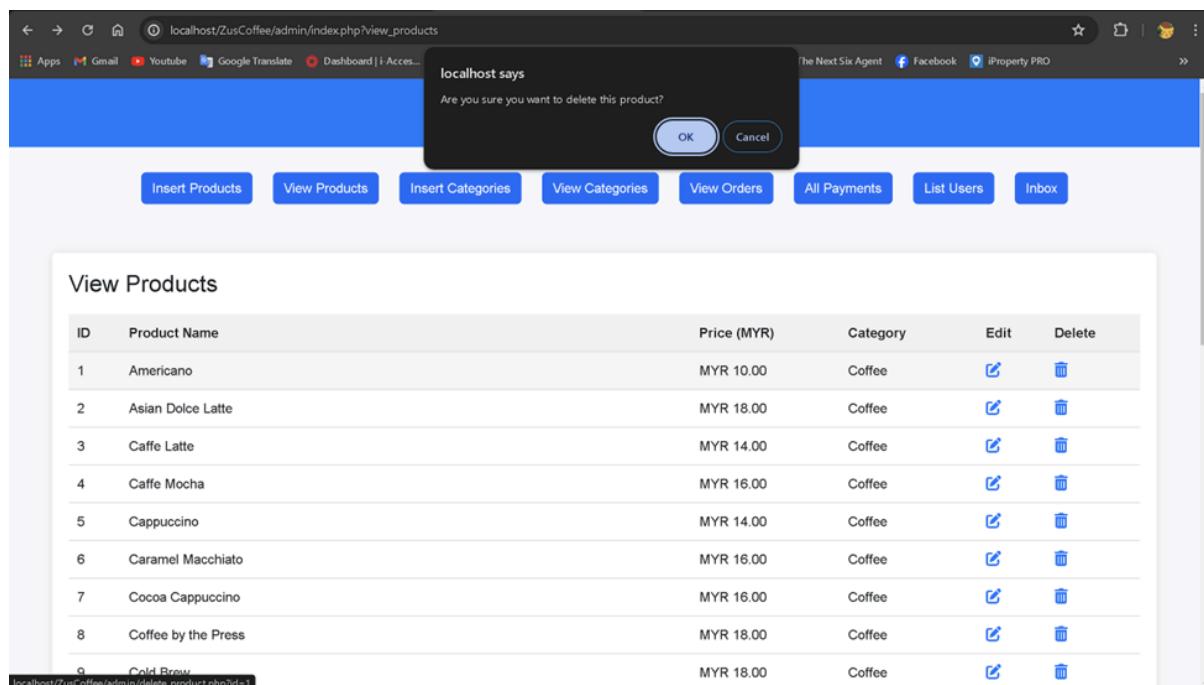
- Text fields and textareas are populated with the current product details fetched from the database.
- The category dropdown is populated with options retrieved from the database.
- The selected category matches the current product's category.

7. Styling:

- Styling is applied to the form elements for a better visual presentation.

Overall, this code segment allows administrators to edit product details. It retrieves the current details of the product from the database, displays them in the form fields, and allows users to update and save the changes. Additionally, it provides a dropdown list for selecting the product's category, enhancing the editing functionality.

admin/delete_product.php



The screenshot shows a web browser window with the URL `localhost/ZusCoffee/admin/index.php?view_products`. A modal dialog box is centered on the screen, asking "Are you sure you want to delete this product?". The dialog has "OK" and "Cancel" buttons. Below the dialog, the main content area displays a table titled "View Products" with 9 rows of data. The table columns are: ID, Product Name, Price (MYR), Category, Edit, and Delete. Each row contains a product name, its price, category (all listed as "Coffee"), and edit/delete buttons. The edit button is represented by a pencil icon and the delete button by a trash bin icon. The browser's navigation bar and a toolbar with various icons are visible at the top.

ID	Product Name	Price (MYR)	Category	Edit	Delete
1	Americano	MYR 10.00	Coffee		
2	Asian Dolce Latte	MYR 18.00	Coffee		
3	Caffe Latte	MYR 14.00	Coffee		
4	Caffe Mocha	MYR 16.00	Coffee		
5	Cappuccino	MYR 14.00	Coffee		
6	Caramel Macchiato	MYR 16.00	Coffee		
7	Cocoa Cappuccino	MYR 16.00	Coffee		
8	Coffee by the Press	MYR 18.00	Coffee		
9	Cold Brew	MYR 18.00	Coffee		

1. Database Connection:

- `include('..../include/connect.php');`: Includes the PHP file `connect.php`, establishing a connection to the database.

2. Product Deletion:

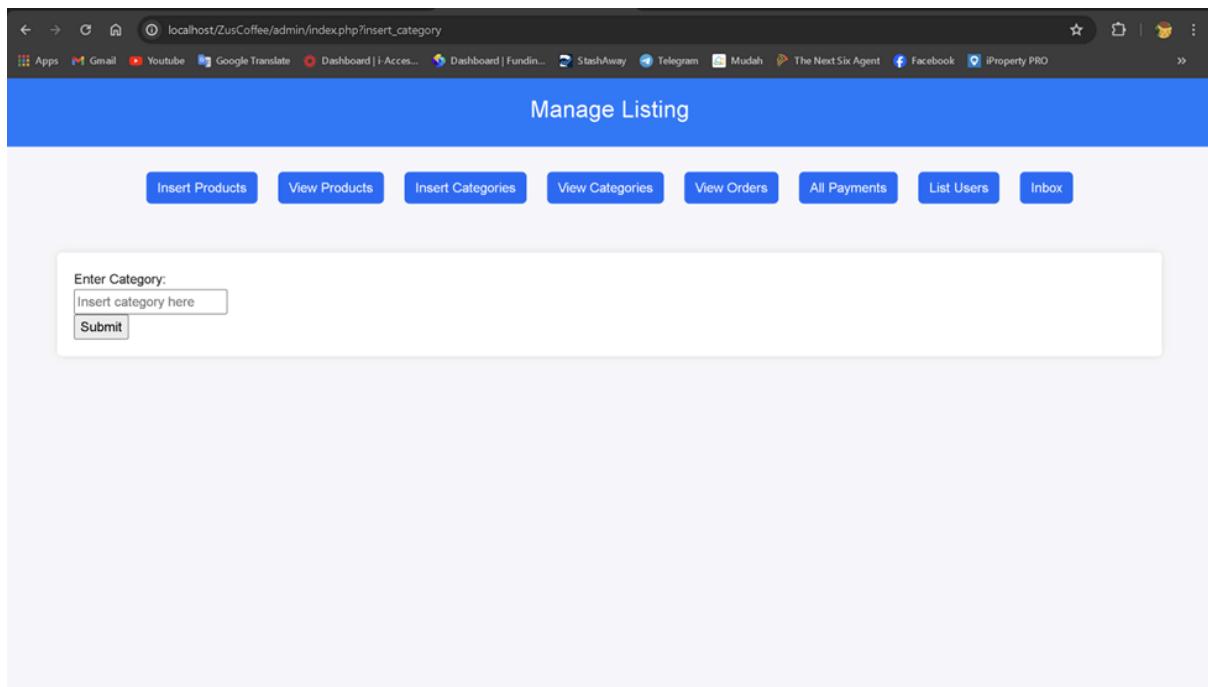
- The code checks if a product ID is provided in the URL (`$_GET['id']`).
- If provided, it retrieves the product ID from the URL parameter.
- It constructs an SQL query to delete the product with the specified ID from the database.
- The **DELETE** query is executed using `mysqli_query()`.
- If the deletion is successful, the page redirects to the view products page.
- If an error occurs during deletion, an error message is displayed.

3. Redirection:

- If the product ID is not provided in the URL, or if there is an issue with the deletion process, the page redirects to the view products page.

Overall, this code segment allows administrators to delete a product from the database. It checks for the presence of a product ID in the URL, deletes the corresponding product from the database, and then redirects the user to the view products page. If any errors occur during the deletion process, an error message is displayed.

admin/insert_category.php



1. Database Connection:

- `include('..../include/connect.php');`: Includes the PHP file **connect.php**, establishing a connection to the database.

2. Form Submission Handling:

- The code checks if the form for inserting a category is submitted (`isset($_POST['insert_category'])`).
- If the form is submitted, it retrieves the category title from the form input.

3. Category Validation:

- It checks if the entered category title already exists in the database to prevent duplicates.
- An SQL query is executed to select data from the **categories** table where the category title matches the entered title.
- If a matching category is found (`$number > 0`), an alert is displayed informing the user that the category already exists.

4. Category Insertion:

- If the entered category title is unique, an SQL **INSERT** query is executed to insert the new category into the **categories** table.
- Upon successful insertion, an alert is displayed indicating that the category has been inserted successfully.

5. HTML Form:

- The HTML form allows users to input a new category title.

- It uses the **POST** method to send form data to the same page ("").
- Input fields include:
 - Category title (**category**).

6. Submit Button:

- The submit button triggers the form submission process when clicked.

Overall, this code segment enables administrators to insert new categories into the database. It ensures that duplicate categories are not inserted by validating the entered category title against existing entries in the database. If the insertion is successful, the user is notified via an alert message.

admin/view_categories.php

ID	Category Name	Edit	Delete
1	Coffee		
3	Pastries		
4	Hot meals		
5	Non-Coffee		
7	Seasonal		

1. Database Connection:

- `include('../include/connect.php');`: Includes the PHP file **connect.php**, which establishes a connection to the database.

2. Category Retrieval:

- The code selects all categories from the **categories** table in the database using an SQL query.

- The result is stored in the `$result` variable.

3. HTML Structure:

- The HTML structure includes a table to display the categories retrieved from the database.
- It uses Bootstrap for styling.

4. Table Headers:

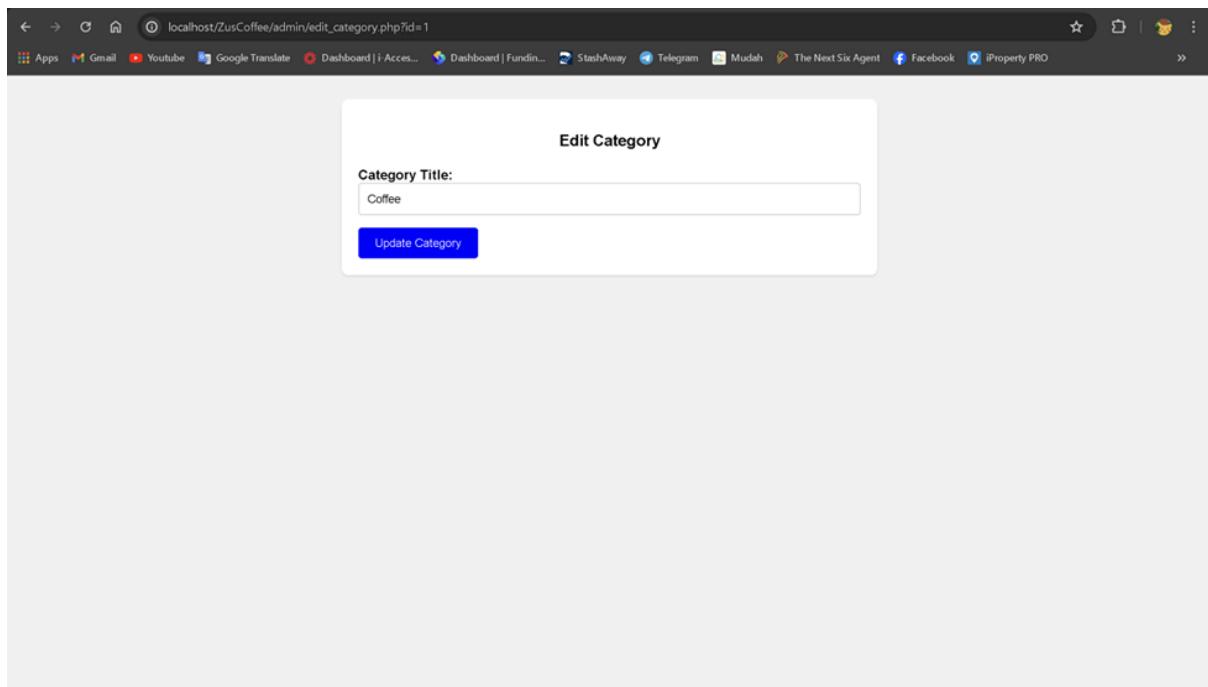
- The table headers include columns for category ID, category name, edit action, and delete action.

5. PHP Logic:

- Inside the table body, PHP code fetches and displays category data retrieved from the database.
- `mysqli_fetch_assoc($result)` fetches each row from the query result as an associative array.
- The fetched data is used to populate table rows (`<tr>`) and cells (`<td>`).
- Edit and delete actions are represented by icons from the Font Awesome library (`<i class='fas fa-edit icon'></i>` for edit and `<i class='fas fa-trash-alt icon'></i>` for delete).
- Edit action links to `edit_category.php` with the category ID as a query parameter (`id`).
- Delete action links to `delete_category.php` with the category ID as a query parameter (`id`) and includes a confirmation prompt using JavaScript `confirm()`.

Overall, this code segment allows administrators to view the categories stored in the database. It retrieves category data from the database and presents it in a tabular format, allowing users to edit or delete categories as needed. The use of icons enhances the user interface, and the confirmation prompt adds an extra layer of user interaction to the deletion process.

admin/edit_category.php



1. Database Connection:

- It includes the **connect.php** file to establish a connection with the database.

2. Category Retrieval:

- It checks if a category ID is provided in the URL (**\$_GET['id']**).
- If a category ID is provided, it fetches the details of the category from the database using an SQL query.
- If the category exists, its details are stored in the **\$category** variable.

3. Form Submission Handling:

- It checks if the form for updating a category is submitted (**isset(\$_POST['update_category'])**).
- If the form is submitted, it retrieves the updated category title from the form input.

4. Category Update:

- It constructs an SQL query to update the category title in the database based on the provided category ID.
- The **UPDATE** query is executed using **mysqli_query()**.

5. Redirection:

- After updating the category, the page redirects to the view categories page (**index.php?view_categories**).

6. HTML Structure:

- The HTML structure includes a form to edit the category details.

- It uses inline CSS for basic styling.

7. Form Fields:

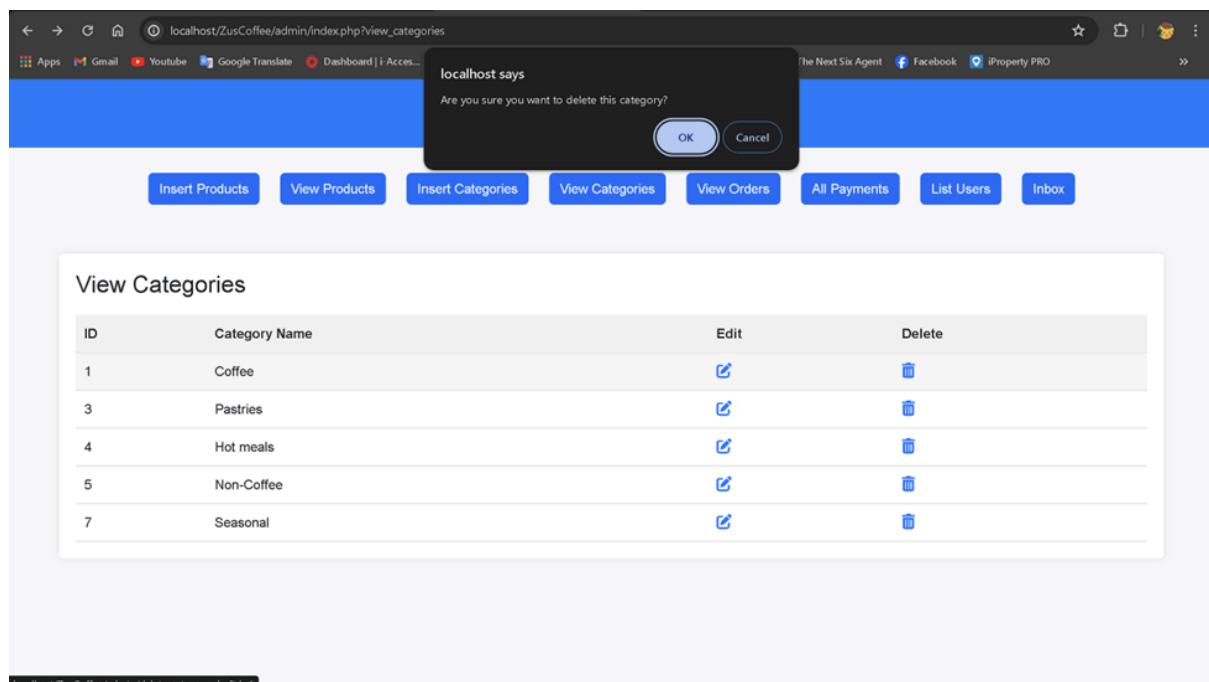
- The form contains an input field for the category title.
- The value of the input field is populated with the current category title fetched from the database.

8. Submit Button:

- The submit button triggers the form submission process when clicked.

Overall, this code segment allows administrators to edit the title of a category. It retrieves the current category details from the database, displays them in the form field, and allows users to update and save the changes. Upon successful update, the user is redirected to the view categories page.

admin/delete_category.php



1. Database Connection:

- The code includes the `connect.php` file to establish a connection with the database.

2. Category Deletion:

- It checks if a category ID is provided in the URL (`$_GET['id']`).
- If a category ID is provided, it retrieves the category ID from the URL parameter.
- It constructs an SQL query to delete the category with the specified ID from the database.
- The **DELETE** query is executed using `mysqli_query()`.

3. Redirection:

- If the deletion is successful, the page redirects to the view categories page (`index.php?view_categories`).
- If there is an error during the deletion process, an error message is displayed.

4. Conditional Redirection:

- If the category ID is not provided in the URL, or if there is an issue with the deletion process, the page redirects to the view categories page.

Overall, this code segment allows administrators to delete a category from the database. It checks for the presence of a category ID in the URL, deletes the corresponding category from the database, and then redirects the user to the view categories page. If any errors occur during the deletion process, an error message is displayed.

admin/view_orders.php

Order ID	User ID	Total Payment (MYR)	Shipping Address	Payment Method	Date Created	Action
4_6620d4eec6cf3	4	MYR 232.00	bubu home	Bank Transfer	2024-04-18 10:08:14	Refund
4_6620d50a58bb8	4	MYR 92.00	utar	Bank Transfer	2024-04-18 10:08:42	Refund
6_6620e148738d8	6	MYR 236.00	1, Jalan Besar, KL	Credit Card	2024-04-18 11:00:56	Refund
6_6620e18251b29	6	MYR 278.00	Shell Mahkota Cheras, Selangor	PayPal	2024-04-18 11:01:54	Refund
7_6620e1dd847a	7	MYR 138.00	Halloween Ground, Putrajaya	Credit Card	2024-04-18 11:03:25	Refund
8_66212a2353c44	8	MYR 50.00	Oscar home	PayPal	2024-04-18 16:11:47	Refund

1. HTML Structure:

- The HTML structure contains a table to display order details fetched from the database.
- Basic styling is applied using inline CSS.

2. Table Headers:

- The table headers include columns for order ID, user ID, total payment, shipping address, payment method, date created, and action.

3. Database Connection:

- It includes the **connect.php** file to establish a connection with the database.

4. Order Retrieval:

- It selects all orders from the **orders** table in the database using an SQL query.
- The result is stored in the **\$result** variable.

5. Order Display:

- Inside the table body, PHP code fetches and displays order data retrieved from the database.
- **mysqli_fetch_assoc(\$result)** fetches each row from the query result as an associative array.
- The fetched data is used to populate table rows (**<tr>**) and cells (**<td>**).
- The order ID is displayed as a hyperlink (**<a>**) which leads to the **order_details.php** page with the order ID as a query parameter (**order_id**).

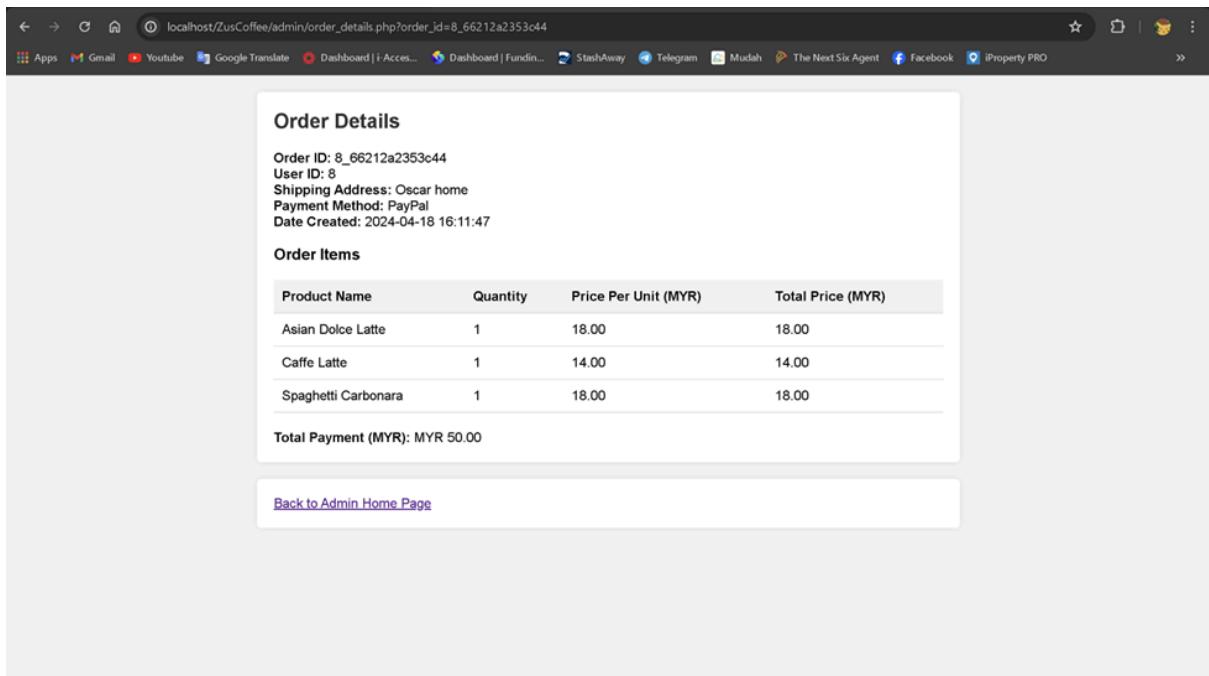
- Order details such as user ID, total payment, shipping address, payment method, and date created are displayed.
- An action column contains a "Refund" link for each order. Clicking the link triggers a refund action and prompts a confirmation dialog using JavaScript **confirm()**.

6. Refund Action:

- Each "Refund" link points to **refund.php** with the order ID as a query parameter (**order_id**).
- When clicked, it prompts the user with a confirmation dialog asking if they are sure they want to refund the order.
- If confirmed, it redirects to the **refund.php** page to process the refund.

Overall, this code segment allows administrators to view orders placed by users. It retrieves order details from the database and presents them in a tabular format. The use of hyperlinks for order IDs and a refund action provides interactivity for administrators to manage orders efficiently.

admin/order_details.php



The screenshot shows a web browser window with the URL `localhost/ZusCoffee/admin/order_details.php?order_id=8_66212a2353c44`. The page is titled "Order Details" and displays the following information:

Order Details

- Order ID: 8_66212a2353c44
- User ID: 8
- Shipping Address: Oscar home
- Payment Method: PayPal
- Date Created: 2024-04-18 16:11:47

Order Items

Product Name	Quantity	Price Per Unit (MYR)	Total Price (MYR)
Asian Dolce Latte	1	18.00	18.00
Caffe Latte	1	14.00	14.00
Spaghetti Carbonara	1	18.00	18.00

Total Payment (MYR): MYR 50.00

[Back to Admin Home Page](#)

1. HTML Structure:

- The page structure includes elements for displaying order details retrieved from the database, such as order ID, user ID, shipping address, payment method, date created, order items, and total payment.
- Basic inline CSS styling is applied for layout and readability.

2. Database Connection:

- The code includes the **connect.php** file to establish a connection with the database, ensuring seamless data retrieval.

3. Order Retrieval:

- It checks if the order ID is provided in the URL (`$_GET['order_id']`).
- If provided, it retrieves the corresponding order details from the **orders** table in the database using an SQL query.
- The fetched order details are stored in the **\$order** variable.

4. Order Details Retrieval:

- Another SQL query fetches order item details from the **order_details** table based on the order ID.
- The fetched order item details are stored in the **\$orderDetails** variable.

5. Total Price Calculation:

- The code calculates the total price of the order by summing up the product of quantity and price per unit for each order item.

6. Order Display:

- The retrieved order details, including order ID, user ID, shipping address, payment method, and date created, are displayed using PHP echo statements.
- A table is used to display order items, with columns for product name, quantity, price per unit, and total price for each item.
- PHP loops through the **\$orderDetails** array to populate table rows with order item details.

7. Total Payment Display:

- The total payment amount for the order is displayed at the end of the order items table.

8. Back to Admin Home Page Button:

- A button/link is provided to navigate back to the admin home page (**index.php?view_orders**).

Overall, this code segment provides administrators with a detailed view of an individual order, including order items and total payment. It enhances administrative efficiency by presenting essential order information in a structured and user-friendly manner.

admin/refund.php

Order ID	User ID	Total Payment (MYR)	Shipping Address	Payment Method	Date Created	Action
4_6620d44ee6cf3	4	MYR 232.00	bubu home	Bank Transfer	2024-04-18 10:08:14	Refund
4_6620d50a58bb8	4	MYR 92.00	utar	Bank Transfer	2024-04-18 10:08:42	Refund
6_6620e148738d8	6	MYR 236.00	1, Jalan Besar, KL	Credit Card	2024-04-18 11:00:56	Refund
6_6620e18251b29	6	MYR 278.00	Shell Mahkota Cheras, Selangor	PayPal	2024-04-18 11:01:54	Refund
7_6620e1dd847a	7	MYR 138.00	Halloween Ground, Putrajaya	Credit Card	2024-04-18 11:03:25	Refund
8_66212a2353c44	8	MYR 50.00	Oscar home	PayPal	2024-04-18 16:11:47	Refund

1. Database Connection:

- The code includes the **connect.php** file to establish a connection with the database, ensuring seamless data operations.

2. Function to Delete an Order and Its Details:

- There's a PHP function **deleteOrder(\$orderId, \$connection)** defined to delete an order and its corresponding details.
 - Inside this function:
 - The order details are deleted first using a DELETE query targeting the **order_details** table based on the provided order ID.
 - Then, the order itself is deleted using a DELETE query targeting the **orders** table based on the provided order ID.

- The function returns true if both queries execute successfully; otherwise, it returns false.

3. Delete Request Handling:

- The code checks if a delete request is submitted via the GET method (`$_GET['order_id']`).
- If a valid order ID is provided, the code calls the **deleteOrder** function passing the order ID and the database connection as arguments.
- Upon successful deletion, an alert message is displayed indicating the successful refund of the order and its details.
- If there's an error during the deletion process, an alert message indicating the error is displayed.
- After processing the deletion request, the code redirects the user to the view orders page (**index.php?view_orders**).

4. Redirect Handling:

- If the order ID is not provided in the URL, or if there's an attempt to access the page directly without a valid order ID, the code redirects the user to the view orders page to ensure proper navigation.

Overall, this code segment provides functionality to refund an order and its details by deleting them from the database. It ensures data integrity and provides feedback to the user regarding the success or failure of the refund process.

admin/all_payment.php

localhost/ZusCoffee/admin/index.php?all_payments

Insert Products View Products Insert Categories View Categories View Orders All Payments List Users Inbox

Sales Report

Product Sales

Product	Unit Sold	Price per Unit	Total Price
Americano	1	MYR 10.00	MYR 10.00
Apple Turnover	3	MYR 16.00	MYR 48.00
Asian Dolce Latte	5	MYR 18.00	MYR 90.00
Brewed Tea	2	MYR 10.00	MYR 20.00
Caffe Latte	3	MYR 14.00	MYR 42.00
Cappuccino	2	MYR 14.00	MYR 28.00
Cinnamon Roll	4	MYR 16.00	MYR 64.00

localhost/ZusCoffee/admin/index.php?all_payments

Insert Products View Products Insert Categories View Categories View Orders All Payments List Users Inbox

Category Sales

Category	Frequency of Category Sold	Total Category Sales
Coffee	16	MYR 276.00
Hot meals	8	MYR 316.00
Non-Coffee	5	MYR 126.00
Pastries	11	MYR 168.00
Seasonal	4	MYR 140.00

User Sales

User	User Purchase Frequency	Total User Spending
Jr	2	MYR 324.00
enyi	2	MYR 514.00
max	1	MYR 138.00
Oscarr	1	MYR 50.00

Payment Method Sales

Payment Method	Frequency of Payment Method Used	Total Sales by Payment Method
Bank Transfer	2	MYR 324.00
Credit Card	2	MYR 374.00

Payment Method Sales		
Payment Method	Frequency of Payment Method Used	Total Sales by Payment Method
Bank Transfer	2	MYR 324.00
Credit Card	2	MYR 374.00
PayPal	2	MYR 328.00

Order Sales	
Order	Order Price
4_6620d4eec6cf3	MYR 232.00
4_6620d50a58bb8	MYR 92.00
6_6620e148738d8	MYR 236.00
6_6620e18251b29	MYR 278.00
7_6620e1dda847a	MYR 138.00
8_66212a2353c44	MYR 50.00

Total Sales	
Total:	MYR 1,026.00

Here's an overview of the code for **admin/all_payment.php**:

1. Database Connection:

- The code includes the **connect.php** file to establish a connection with the database, ensuring seamless data operations.

2. Query Execution and Data Retrieval:

- **Product Sales:** The script executes a SQL query (**\$query_product_sales**) to retrieve product sales data from the database. The query selects the product name, quantity sold, price per unit, and total price for each product from the **order_details** table. The result is stored in **\$result_product_sales**.
- **Category Sales:** Similar to product sales, this query (**\$query_category_sales**) retrieves data on category sales. It joins the **order_details**, **products**, and **categories** tables to get the category title, frequency of category sold, and total category sales. The result is stored in **\$result_category_sales**.
- **User Sales:** This query (**\$query_user_sales**) retrieves data on user sales. It joins the **orders** and **users** tables to get the username, user purchase frequency, and total user spending. The result is stored in **\$result_user_sales**.
- **Payment Method Sales:** Similar to the above, this query (**\$query_payment_method_sales**) retrieves data on sales by payment method. It selects the payment method, frequency of payment method used, and total

sales by payment method from the **orders** table. The result is stored in **\$result_payment_method_sales**.

- **Order Sales:** This query (**\$query_order_sales**) simply retrieves data on individual orders, including the order ID and order price, from the **orders** table. The result is stored in **\$result_order_sales**.
- **Total Sales:** Lastly, this query (**\$query_total_sales**) calculates the total sales by summing up the total payment from all orders. The result is stored in **\$total_sales_row**, and the total sales value is extracted and stored in **\$total_sales**.

3. HTML Structure:

- The HTML structure contains various tables to display sales data, including product sales, category sales, user sales, payment method sales, and individual order sales.
- Basic styling is applied using Bootstrap classes.

4. JavaScript for Charts:

- The code utilizes Chart.js library to create pie charts for visualizing sales data.
- Separate pie charts are generated for product sales, category sales, user sales, and payment method sales.
- JavaScript code dynamically populates chart data based on the fetched database results.

Overall, this script generates a comprehensive sales report by querying the database for various sales metrics and visualizing them using pie charts for better analysis and understanding.

admin/user_list.php

Manage Listing

ID	Username	Email	Name	Edit	Delete
4	Jr	reallyhat@gmail.com	Jien Weng Lai		
5	Oscar	oscar@gmail.com	Oscar Chong		
6	enyi	liewenyi@yahoo.com	En Yi Liew		
7	max	johnmax@hotmail.com	John Max		
8	Oscarr	oscar@hotmail.com	Oscar Lai		

1. Database Connection:

- The code includes the **connect.php** file to establish a connection with the database, ensuring seamless data retrieval.

2. User Retrieval:

- It selects all users from the **users** table in the database using an SQL query.
- The query result is stored in the **\$result** variable.

3. HTML Structure:

- The HTML structure contains a table to display user details fetched from the database.
- Basic styling is applied using inline CSS for table formatting.

4. Table Headers:

- The table headers include columns for user ID, username, email, name (concatenation of first name and last name), edit action, and delete action.

5. User Display:

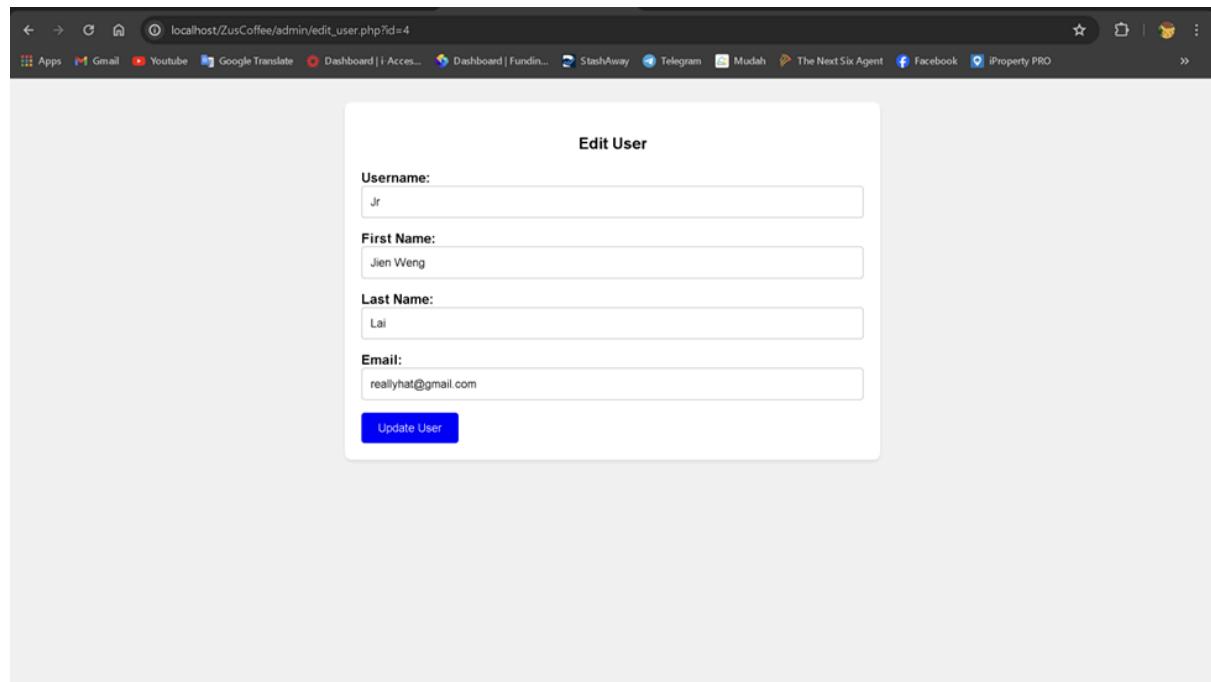
- Inside the table body, PHP code fetches and displays user data retrieved from the database.
- **mysqli_fetch_assoc(\$result)** fetches each row from the query result as an associative array.
- User details such as ID, username, email, and name are displayed in respective table cells.
- Edit and delete actions are provided for each user in the form of icons.

6. Edit and Delete Icons:

- An edit icon (pencil icon) is linked to the **edit_user.php** page, passing the user's ID as a query parameter.
- A delete icon (trash icon) is linked to the **delete_user.php** page, passing the user's ID as a query parameter.
- Both edit and delete actions prompt a confirmation dialog using JavaScript **confirm()** before proceeding with the action.

Overall, this script generates a user list interface for administrators to view user details and perform edit or delete actions on individual users as needed.

admin/edit_user.php



The screenshot shows a web browser window with the URL `localhost/ZusCoffee/admin/edit_user.php?id=4`. The page title is "Edit User". The form contains the following fields:

- Username:** Jr
- First Name:** Jien Weng
- Last Name:** Lai
- Email:** reallyhat@gmail.com

At the bottom of the form is a blue "Update User" button.

1. Database Connection:

- The code includes the **connect.php** file to establish a connection with the database, ensuring seamless data retrieval and updates.

2. User Retrieval:

- It checks if a user ID is provided in the URL.

- If provided, it selects the user details from the **users** table in the database based on the user ID.
- If the user exists, their details are fetched and stored in the **\$user** variable. Otherwise, the user is redirected to the user list page.

3. Form Submission:

- The code checks if the form is submitted (**\$_POST['update_user']**).
- If submitted, it retrieves the updated user information from the form fields.
- It updates the user's first name, last name, and email in the database using an **SQL UPDATE** query.
- Upon successful update, the user is redirected to the user list page to view the changes. If an error occurs during the update process, an error message is displayed.

4. HTML Structure:

- The HTML structure includes a form to allow administrators to edit user details.
- Basic styling is applied for form elements and layout using inline CSS.

5. Form Fields:

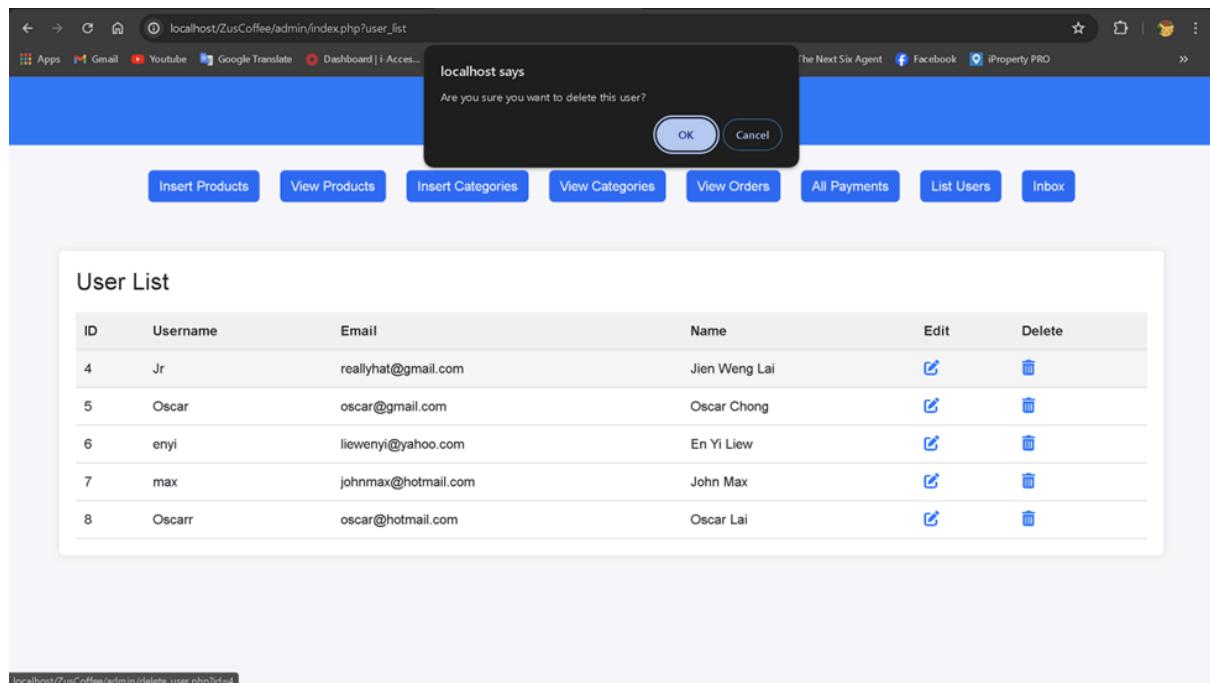
- Input fields are provided for the username, first name, last name, and email.
- The username field is set to read-only, preventing modifications as it is assumed to be a unique identifier for the user.
- The current user details are populated in the respective input fields for editing.

6. Submit Button:

- A submit button allows administrators to update the user details after making changes.

Overall, this script provides administrators with a user-friendly interface to edit user information, ensuring efficient management of user data within the system.

admin/delete_user.php



1. Database Connection:

- The code includes the `connect.php` file to establish a connection with the database, ensuring seamless data deletion.

2. User Deletion:

- It checks if the user ID parameter (`$_GET['id']`) is set in the URL.
- If the ID parameter is set, it retrieves the user ID from the URL.
- It constructs an SQL **DELETE** query to remove the user from the **users** table in the database based on the provided user ID.
- The query is executed using `mysqli_query`.

3. Deletion Result:

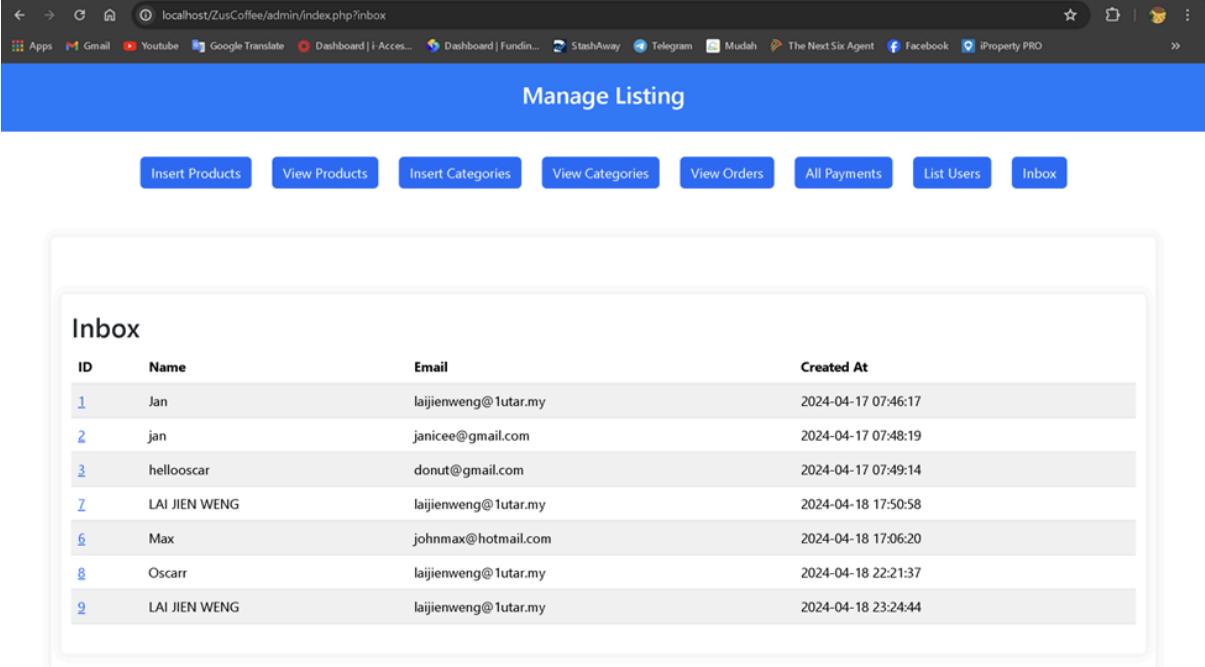
- After executing the deletion query, the code checks if the deletion was successful.
- If successful, it displays a success message indicating that the user was deleted successfully.
- If an error occurs during the deletion process, it displays an error message indicating that there was an issue deleting the user.

4. Invalid Request Handling:

- If the ID parameter is not set in the URL, the code displays an "Invalid request" message. This ensures that the deletion process is only initiated when a valid user ID is provided.

Overall, this script provides functionality to delete a user from the database based on their ID, allowing administrators to manage user data effectively within the system.

admin/inbox.php



The screenshot shows a web browser window with the URL `localhost/ZusCoffee/admin/index.php?inbox` in the address bar. The page title is "Manage Listing". Below the title, there is a navigation bar with buttons: "Insert Products", "View Products", "Insert Categories", "View Categories", "View Orders", "All Payments", "List Users", and "Inbox". The main content area is titled "Inbox" and contains a table with the following data:

ID	Name	Email	Created At
1	Jan	laijenweng@1utar.my	2024-04-17 07:46:17
2	jan	janicee@gmail.com	2024-04-17 07:48:19
3	hellooscar	donut@gmail.com	2024-04-17 07:49:14
7	LAI JIEN WENG	laijenweng@1utar.my	2024-04-18 17:50:58
6	Max	johnmax@hotmail.com	2024-04-18 17:06:20
8	Oscarr	laijenweng@1utar.my	2024-04-18 22:21:37
9	LAI JIEN WENG	laijenweng@1utar.my	2024-04-18 23:24:44

1. Database Connection:

- The code includes the `connect.php` file to establish a connection with the database, ensuring seamless retrieval of inbox messages.

2. Fetching Messages:

- It constructs an SQL query to select all messages from the `contact_responses` table in the database.
- The query is executed using `mysqli_query`.
- The result set containing the messages is stored in the `$result` variable.

3. Displaying Messages:

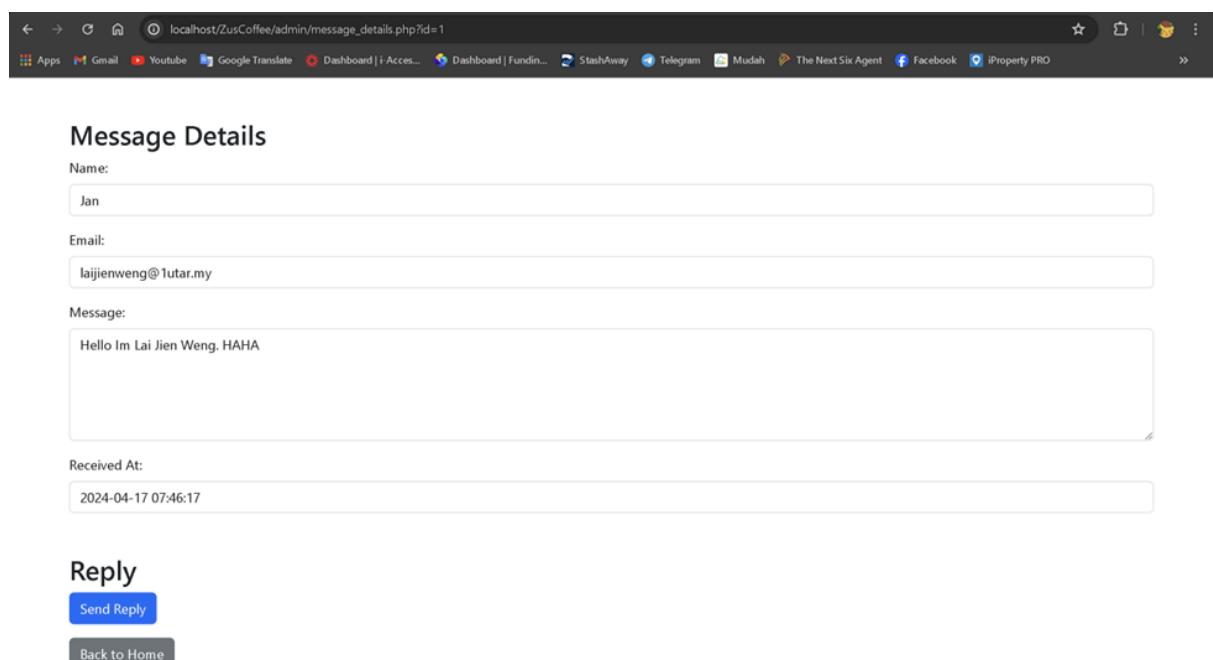
- Inside the HTML table body, PHP code iterates through each row of the result set using a **while** loop.
- For each message, it displays the following details in a table row:
 - Message ID: It's displayed as a hyperlink (<a>) leading to the **message_details.php** page with the message ID as a query parameter.
 - Name: The name of the sender.
 - Email: The email address of the sender.
 - Created At: The timestamp indicating when the message was created.

4. Table Styling:

- The table is styled using Bootstrap classes for a clean and organized appearance.

Overall, this script provides functionality to display a list of inbox messages, allowing administrators to view basic details of each message and navigate to the details of a specific message for further actions or responses.

admin/message_details.php



Message Details

Name:

Email:

Message:

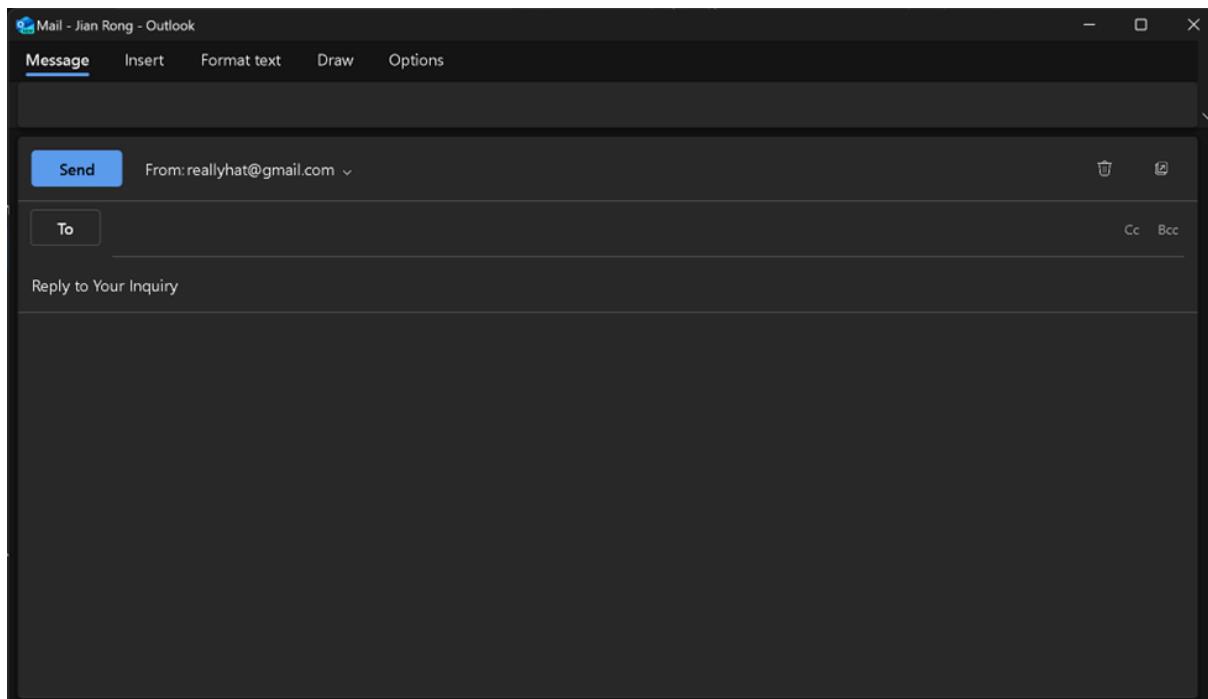
```
Hello Im Lai Jen Weng. HAHA
```

Received At:

Reply

Send Reply

Back to Home



1. Database Connection:

- The code includes the **connect.php** file to establish a connection with the database, ensuring seamless retrieval of message details.

2. Fetching Message Details:

- It checks if the **id** parameter is set in the URL.
- If set, it retrieves the message details from the **contact_responses** table based on the provided ID.
- The fetched details include the name of the sender, email address, message content, and the timestamp indicating when the message was created.
- If the message exists (checked by the number of rows returned), it assigns the retrieved details to variables for later use.

3. Displaying Message Details:

- Inside the HTML body, it displays the message details in a structured format.
- It includes form elements for each message detail such as name, email, message content, and the timestamp.
- These form elements are set to read-only to prevent modification of the message details.

- Additionally, it provides a "Reply" button that allows administrators to send a reply to the sender via email. Clicking the button opens the default email client with a pre-filled email addressed to the sender.

4. Navigation:

- It includes a "Back to Home" button that redirects users to the inbox page ([index.php?inbox](#)).

Overall, this script facilitates the viewing of detailed information about a specific message from the inbox, allowing administrators to review the message content and send a reply if necessary.

Include

Include/connect.php

1. **Connection Establishment:** The `mysqli_connect()` function is used to create a connection to the MySQL database server.
2. **Connection Parameters:**
 - `'localhost'`: This specifies the hostname where the MySQL database server is running. In this case, it's assumed to be running on the same machine where the PHP script is hosted.
 - `'root'`: This is the username used to connect to the MySQL database server. It's common to use "root" as the default username for local development environments. In production environments, a more restricted user with appropriate privileges should be used.
 - `''`: This empty string represents the password for the MySQL user. If no password is set for the user, an empty string is used. In production environments, passwords should always be set and securely managed.
 - `'cup_corner'`: This is the name of the MySQL database to which the script connects.
3. **Error Handling:** If the connection attempt fails (`!$con`), the `mysqli_error()` function is used to retrieve the error message from MySQL, and the script terminates execution with `die()`. This ensures that any connection errors are displayed for debugging purposes.

Overall, this file serves the purpose of establishing a connection to the "cup_corner" database, allowing other PHP scripts in the project to interact with the database.

General

add_to_cart.php

1. **Session Start:** The script begins by starting a PHP session using `session_start()`. This allows the script to store and access session variables across multiple pages.
2. **Initialization of Cart Session Variable:** If the `$_SESSION['cart']` variable is not set, meaning the shopping cart hasn't been initialized yet, it initializes it as an empty array.
3. **Receiving Data:** The script expects JSON data to be sent via a POST request. It decodes the JSON data using `json_decode()` and sets it to the `$data` variable.
4. **Handling Product Addition:**
 - If the JSON data contains a key named "productId", it extracts the product ID and assigns it to the `$productId` variable.
 - It also checks if an optional key named "increment" is set to true. If set and true, it indicates that the quantity of the product in the cart should be incremented.
 - If the product ID is not already in the shopping cart, it adds it with a quantity of 1.
 - If the product ID is already in the shopping cart and the "increment" flag is true, it increments the quantity of the product by 1.
5. **Response:** After processing the request, the script sends a JSON response back to the client with a message indicating whether the product was successfully added to the cart or if the product ID was missing.

Here's a breakdown of the JSON responses:

- If the product is successfully added to the cart, the response contains a message: **'Product added to cart successfully.'**.

- If the product ID is missing from the request, the response contains a message: **'Product ID is missing.'**.

Overall, this script provides a simple way to add products to a shopping cart using session variables and JSON data.

check_login_status.php

1. **Session Start:** The script begins by starting a PHP session using `session_start()`. This allows the script to access session variables set in other parts of the application.
2. **Initialization of Response Array:** An associative array named `$response` is initialized with a key-value pair where `'loggedIn'` is set to `false` by default. This means that the user is assumed to be not logged in initially.
3. **Checking Login Status:**
 - The script checks if the session variable `$_SESSION['valid']` is set. This variable likely indicates whether the user is considered logged in by the application.
 - If `$_SESSION['valid']` is set, it means the user is logged in. In this case, the value of `'loggedIn'` in the response array is set to `true`.
4. **Sending Response:** After checking the login status, the script encodes the `$response` array into JSON format using `json_encode()` and echoes it back to the client.

Here's a breakdown of the JSON response:

- If the user is logged in (`$_SESSION['valid']` is set), the response contains:
`{"loggedIn": true}`.
- If the user is not logged in (`$_SESSION['valid']` is not set), the response contains:
`{"loggedIn": false}`.

This script can be used by client-side JavaScript to determine whether a user is logged in and adjust the user interface or behavior accordingly.

checkout.php

1. **Session Start and Configuration Inclusion:** The script starts a PHP session using `session_start()` and includes a configuration file named "config.php" which likely contains database connection settings.
2. **User Authentication Check:** The script checks whether the user is authenticated by verifying the existence of the session variable `$_SESSION['userId']`. If the user is not authenticated, the script redirects them to the login page using `header('Location: login.php')`. The `exit` statement ensures that no further code is executed after redirection.
3. **POST Request Handling:** The script checks if the request method is POST, indicating that the checkout form has been submitted.
4. **Order Processing:**
 - **Unique Order ID Generation:** An order ID is generated using the `uniqid()` function, which combines the user's ID and a unique identifier.
 - **Form Data Retrieval:** Form data such as user ID, shipping address, payment method, and current date and time are retrieved from the POST request.
 - **Database Transaction:** A database transaction is initiated using `mysqli_begin_transaction()` to ensure that all database operations are atomic.
 - **Total Payment Calculation:** The total payment for the order is calculated by iterating over each product in the shopping cart, retrieving its price from the database, and multiplying it by the quantity.
 - **Order Insertion:** The order details are inserted into the "orders" table using a prepared statement.
 - **Order Details Insertion:** For each product in the shopping cart, its details are retrieved from the database, and an entry is inserted into the "order_details" table.
 - **Transaction Commit:** If all database operations are successful, the transaction is committed using `mysqli_commit()`. If an error occurs, a rollback is performed to ensure data consistency.
 - **Session Cleanup:** The shopping cart session variable `$_SESSION['cart']` is unset to clear the cart after placing the order.
 - **Redirection:** The user is redirected back to the cart page after a successful order placement, with a delay of 3 seconds.

Overall, this script handles the checkout process by processing the order details, inserting them into the database, and redirecting the user to the cart page after a successful order placement.

config.php

1. Database Connection Parameters:

- **\$servername:** This variable stores the hostname where the MySQL database server is running. In this case, it's set to "localhost", indicating that the database server is on the same machine where the PHP script is hosted.
- **\$username:** This variable stores the username used to connect to the MySQL database server. It's set to "root", which is a common default username for local development environments.
- **\$password:** This variable stores the password used to authenticate the connection to the MySQL database server. It's currently set to an empty string, indicating that no password is required. In production environments, passwords should always be set and securely managed.
- **\$dbname:** This variable stores the name of the MySQL database to which the script will connect. It's set to "cup_corner".

2. Database Connection: The script attempts to establish a connection to the MySQL database using **mysqli_connect()** with the provided connection parameters.

3. Connection Error Handling: If the connection attempt fails (**!\$con**), the script terminates execution with **die()** and outputs an error message containing the reason for the connection failure, obtained using **mysqli_connect_error()**.

Overall, this file serves the purpose of establishing a connection to the "cup_corner" database, allowing other PHP scripts in the project to interact with the database.

submit_contact.php

1. **Database Connection Inclusion:** The script includes the "connect.php" file using `include()`. This file likely contains the code to establish a connection to the database. Ensure that the path to the "connect.php" file is correct relative to the location of the "submit_contact.php" script.
2. **POST Request Handling:** The script checks if the HTTP request method is POST, indicating that form data has been submitted.
3. **Data Retrieval:** The script retrieves the submitted data from the POST request parameters (`$_POST`). The variables `$name`, `$email`, and `$message` are assigned the corresponding values from the form fields.
4. **Database Insertion:**
 - A prepared statement is created to insert the contact form data into the "contact_responses" table of the database. The statement includes placeholders `(?, ?, ?)` for the values to be inserted.
 - The `bind_param()` function binds the variables `$name`, `$email`, and `$message` to the placeholders in the prepared statement.
 - The `execute()` function is called to execute the prepared statement.
5. **Response Handling:**
 - If the insertion is successful (`execute()` returns true), a JavaScript alert is displayed indicating that the message has been sent, and the user is redirected to the "contact.php" page.
 - If there's an error during insertion, a JavaScript alert is displayed indicating the error, and the user is redirected back to the "contact.php" page.
6. **Statement and Connection Closure:** After executing the statement, both the statement (`$stmt`) and the database connection (`$con`) are closed using the `close()` method.

Overall, this script handles the submission of contact form data by inserting it into the database and providing appropriate feedback to the user. Make sure to properly handle user input validation and prevent SQL injection vulnerabilities by sanitizing user input or using prepared statements.

remove_from_cart.php

1. **Session Start:** The script starts a PHP session using `session_start()` to ensure that session variables can be accessed.
2. **Cart Initialization Check:** The script checks if the session variable `$_SESSION['cart']` is set, indicating that the shopping cart has been initialized. If the cart is not initialized, it returns a JSON response with an error message and exits the script.
3. **Input Retrieval:** The script retrieves JSON data sent via a POST request using `file_get_contents('php://input')`. It then decodes the JSON data into an associative array using `json_decode()`.
4. **Input Validation:**
 - The script checks if the input array contains a key named "productId" and if the value is a numeric string. If the input is valid, it casts the "productId" to an integer.
 - If the input is invalid (missing or non-numeric), it returns a JSON response with an error message indicating an invalid product ID.
5. **Item Removal:**
 - If the product ID exists in the shopping cart (`$_SESSION['cart']`), the script checks if the quantity of the item is greater than 1.
 - If the quantity is greater than 1, it decrements the quantity by 1.
 - If the quantity is 1, it removes the item from the cart by unsetting the corresponding key.
 - After updating the cart, the script returns a JSON response with a success message indicating that the item was updated successfully.
 - If the product ID does not exist in the cart, it returns a JSON response with an error message indicating that the item was not found in the cart.

Overall, this script provides functionality to remove items from a shopping cart stored in a session variable and communicates the status of the operation via JSON responses.

order_details.php

Order Details

Product Name	Quantity	Price Per Unit (MYR)	Total Price (MYR)
Asian Dolce Latte	1	18.00	18.00
Caffe Latte	1	14.00	14.00
Cappuccino	1	14.00	14.00
Cinnamon Roll	1	16.00	16.00
Croissant	1	10.00	10.00
Pumpkin Spice Latte	1	20.00	20.00
Total Price (MYR): 92.00			

Order Date
2024-04-18 10:08:42

Shipping Address
utar

Order Date
2024-04-18 10:08:42

Shipping Address
utar

Payment Method
Bank Transfer

[Back to Account](#)

[Our Team](#) [About](#) [Contact Us](#)

© 2024 Cup Corner. All rights reserved.

- Session Start and Configuration Inclusion:** The script starts a PHP session and includes the "config.php" file to establish a database connection.
- Order ID Validation:** It checks if the order ID is provided in the URL (`$_GET['order_id']`). If not, it redirects the user to the account page or displays an error message.
- Data Retrieval:** The script retrieves order details and order information from the database based on the provided order ID using SQL queries.

4. **Data Display:** It dynamically generates HTML content to display the order details, including product name, quantity, price per unit, and total price. It also displays the shipping address, payment method, and order date.
5. **Styling:** The script includes a separate CSS file ("order_details.css") to style the order details page, defining the layout, typography, and table styling.
6. **Error Handling:** It includes error handling for database queries to handle situations where fetching order details or order information fails.
7. **Navigation:** It provides a button to navigate back to the account page.
8. **Footer Inclusion:** Finally, it includes a footer section using the "include/footer.php" file.

Overall, the script provides a user-friendly interface to view detailed information about an order, enhancing the user experience for managing orders within the system.

function/common_function.php

1. **getProducts() Function:**
 - Retrieves products from the database, specifically from a default category if no category is specified.
 - Displays each product as a card with its image, title, price, and buttons for adding to cart or viewing more details.
2. **getUniqueCategory() Function:**
 - Retrieves products from a specified category if a category ID is provided in the URL.
 - Displays each product similarly to the **getProducts()** function.
3. **getCategories() Function:**
 - Retrieves categories from the database.
 - Displays each category as a navigation item with a link to filter products by that category.
4. **getProductPreview() Function:**
 - Retrieves products for preview purposes, typically used when clicking "View More" buttons.

- Displays a preview of each product with its image, title, description, and price. This function is likely used for displaying detailed information about a product when a user clicks on a "View More" button.

Overall, these functions handle the retrieval and display of products and categories on the website, enhancing the user experience by providing organized navigation and detailed product information.

Top of Form

JavaScript

Menu.js

1. View More Buttons Event Listener:

- It adds an event listener to all elements with the class **.btn-secondary**, presumably "View More" buttons.
- When clicked, it prevents the default action of the link.
- It retrieves the product ID from the closest ancestor element with the class **.product** using the **getAttribute()** method.
- It then calls the **showProductPreview()** function, passing the product ID as an argument.

2. showProductPreview() Function:

- It hides any currently active product previews by removing the **active** class from elements with the class **.preview**.
- It shows the preview corresponding to the product ID by adding the **active** class to the element with the attribute **data-target** equal to the product ID.
- It displays the products preview container by setting its **display** style property to **'flex'**.

3. Close Buttons Event Listener:

- It adds an event listener to all elements with the class **.fa-times**, presumably close buttons within product previews.
- When clicked, it hides the products preview container by setting its **display** style property to '**none**'.

4. Add to Cart Buttons Event Listener:

- It adds an event listener to all elements with the class **.btn-primary**, presumably "Add to Cart" buttons.
- When clicked, it prevents the default form submission behavior.
- It retrieves the product ID from the closest ancestor element with the class **.product** using the **getAttribute()** method.
- It makes an AJAX POST request to **add_to_cart.php** with the product ID and an increment flag set to **true** in the request body.
- It handles the response from the server, displaying an alert message with the response data or logging an error to the console if the request fails.

Overall, this JavaScript code enhances the user experience by providing functionality to view product previews, add products to the shopping cart asynchronously, and close product previews without reloading the page.

MySQL

cup_corner.sql

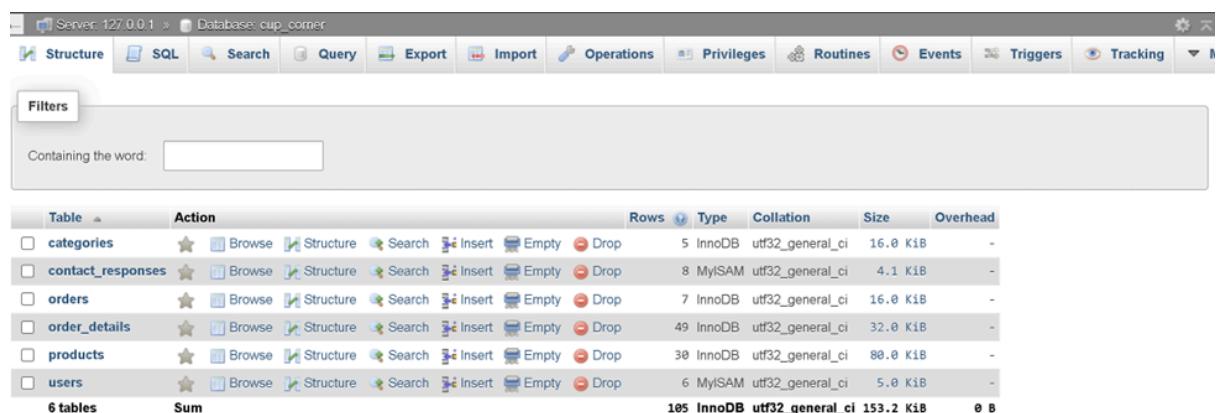


Table	Action	Rows	Type	Collation	Size	Overhead
categories		5	InnoDB	utf8_general_ci	16.0 Kib	-
contact_responses		8	MyISAM	utf8_general_ci	4.1 Kib	-
orders		7	InnoDB	utf8_general_ci	16.0 Kib	-
order_details		49	InnoDB	utf8_general_ci	32.0 Kib	-
products		30	InnoDB	utf8_general_ci	80.0 Kib	-
users		6	MyISAM	utf8_general_ci	5.0 Kib	-
6 tables	Sum	105	InnoDB	utf8_general_ci	153.2 Kib	0 B

Showing rows 0 - 4 (5 total, Query took 0.0004 seconds.)

SELECT * FROM `categories`

Profile [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

	category_id	category_title
<input type="checkbox"/>	1	Coffee
<input type="checkbox"/>	3	Pastries
<input type="checkbox"/>	4	Hot meals
<input type="checkbox"/>	5	Non-Coffee
<input type="checkbox"/>	7	Seasonal

1. Categories Table:

- This table stores different categories of products offered by Cup Corner.
- Columns:
 - **category_id**: An auto-incrementing integer serving as the primary key.
 - **category_title**: A varchar field to store the title/name of each category.
- Example Data:
 - Category ID 1: Coffee
 - Category ID 3: Pastries
 - Category ID 4: Hot meals
 - Category ID 5: Non-Coffee
 - Category ID 7: Seasonal

Showing rows 0 - 7 (8 total, Query took 0.0004 seconds.)

SELECT * FROM `contact_responses`

Profile [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

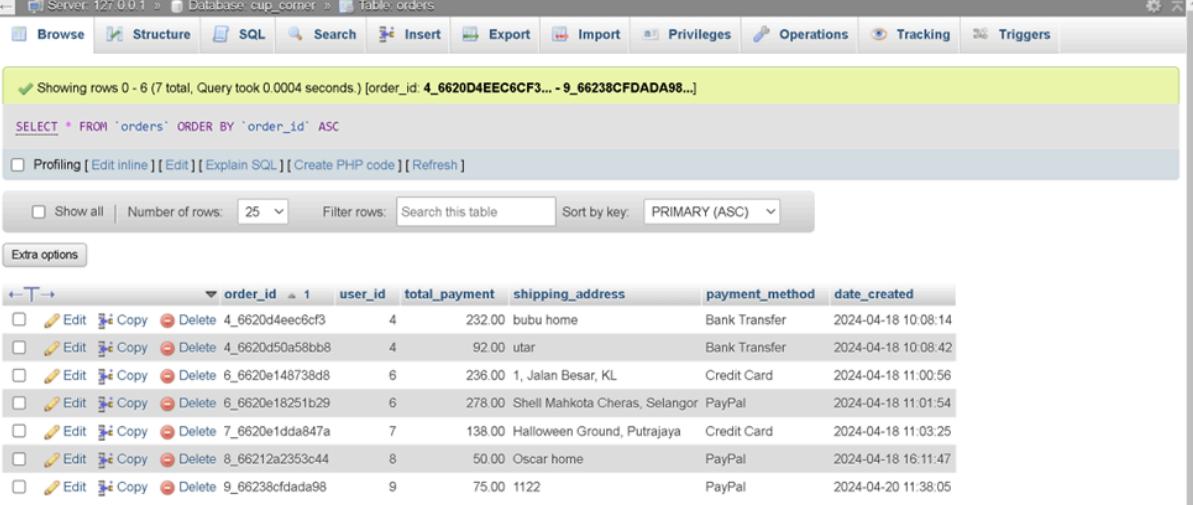
Extra options

	id	name	email	message	created_at
<input type="checkbox"/>	1	Jan	laijenweng@1utar.my	Hello Im Lai Jien Weng. HAHA	2024-04-17 07:46:17
<input type="checkbox"/>	2	jan	janiceee@gmail.com	deafrsf	2024-04-17 07:48:19
<input type="checkbox"/>	3	helloscar	donut@gmail.com	my donut is not served	2024-04-17 07:49:14
<input type="checkbox"/>	7	LAI JIEN WENG	laijenweng@1utar.my	The coffee is too bitter.	2024-04-18 17:50:58
<input type="checkbox"/>	6	Max	johnmax@hotmail.com	Very good service!! Please add more seasonal coffee	2024-04-18 17:06:20
<input type="checkbox"/>	8	Oscarr	laijenweng@1utar.my	Coffee is too bitter, please compensate me!!!	2024-04-18 22:21:37
<input type="checkbox"/>	9	LAI JIEN WENG	laijenweng@1utar.my	Your spana very lembik, cannot fix my pipe.	2024-04-18 23:24:44
<input type="checkbox"/>	10	Enyi	liewenyi0112@gmail.com	Your coffee is too bad. I need compensation!!	2024-04-20 17:42:27

2. Contact Responses Table:

- This table stores responses submitted through the contact form on the Cup Corner website.
- Columns:
 - **id**: An auto-incrementing integer serving as the primary key.
 - **name**: Name of the person submitting the response.
 - **email**: Email address of the person submitting the response.
 - **message**: The message content submitted through the contact form.

- **created_at**: Timestamp indicating when the response was submitted.
- Example Data:
 - ID 1: From "Jan" with email "laijienweng@1utar.my" and message "Hello Im Lai Jien Weng. HAHA"



Showing rows 0 - 6 (7 total, Query took 0.0004 seconds.) [order_id: 4_6620d4eec6cf3... - 9_66238cfada98...]

SELECT * FROM `orders` ORDER BY `order_id` ASC

Profile Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all Number of rows: 25 Filter rows: Search this table Sort by key: PRIMARY (ASC)

Extra options

	order_id	user_id	total_payment	shipping_address	payment_method	date_created
<input type="checkbox"/>	Edit Copy Delete 4_6620d4eec6cf3	4	232.00	bubu home	Bank Transfer	2024-04-18 10:08:14
<input type="checkbox"/>	Edit Copy Delete 4_6620d50a58bb8	4	92.00	utar	Bank Transfer	2024-04-18 10:08:42
<input type="checkbox"/>	Edit Copy Delete 6_6620e148738d8	6	236.00	1, Jalan Besar, KL	Credit Card	2024-04-18 11:00:56
<input type="checkbox"/>	Edit Copy Delete 6_6620e18251b29	6	278.00	Shell Mahkota Cheras, Selangor	PayPal	2024-04-18 11:01:54
<input type="checkbox"/>	Edit Copy Delete 7_6620e1dda847a	7	138.00	Halloween Ground, Putrajaya	Credit Card	2024-04-18 11:03:25
<input type="checkbox"/>	Edit Copy Delete 8_66212a2353c44	8	50.00	Oscar home	PayPal	2024-04-18 16:11:47
<input type="checkbox"/>	Edit Copy Delete 9_66238cfada98	9	75.00	1122	PayPal	2024-04-20 11:38:05

3. Orders Table:

- This table stores information about orders placed by customers.
- Columns:
 - **order_id**: A unique identifier for each order.
 - **user_id**: ID of the user who placed the order.
 - **total_payment**: Total payment amount for the order.
 - **shipping_address**: Address where the order will be shipped.
 - **payment_method**: Method used for payment (e.g., Bank Transfer, Credit Card).
 - **date_created**: Timestamp indicating when the order was created.
- Example Data:
 - Order ID "4_6620d4eec6cf3": Placed by user ID 4, total payment of 232.00, shipped to "bubu home" on April 18, 2024.

Showing rows 0 - 24 (49 total, Query took 0.0004 seconds.)

SELECT * FROM `order_details`

1 > >> Show all Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

	detail_id	order_id	product_name	price_per_unit	quantity
<input type="checkbox"/>	36	4_6620d4eec6cf3	Asian Dolce Latte	18.00	1
<input type="checkbox"/>	37	4_6620d4eec6cf3	Caffe Latte	14.00	1
<input type="checkbox"/>	38	4_6620d4eec6cf3	Cappuccino	14.00	1
<input type="checkbox"/>	39	4_6620d4eec6cf3	Cold Brew	18.00	1
<input type="checkbox"/>	40	4_6620d4eec6cf3	Coffee by the Press	18.00	1
<input type="checkbox"/>	41	4_6620d4eec6cf3	Cocoa Cappuccino	16.00	1
<input type="checkbox"/>	42	4_6620d4eec6cf3	Matcha Latte	18.00	2
<input type="checkbox"/>	43	4_6620d4eec6cf3	Spaghetti Carbonara	18.00	2
<input type="checkbox"/>	44	4_6620d4eec6cf3	Spaghetti Bolognese	18.00	2
<input type="checkbox"/>	45	4_6620d4eec6cf3	Cinnamon Roll	16.00	1
<input type="checkbox"/>	46	4_6620d4eec6cf3	Wholemeal Tuna Sandwich	10.00	1
<input type="checkbox"/>	47	4_6620d50a58bb8	Asian Dolce Latte	18.00	1
<input type="checkbox"/>	48	4_6620d50a58bb8	Caffe Latte	14.00	1
<input type="checkbox"/>	49	4_6620d50a58bb8	Cappuccino	14.00	1
<input type="checkbox"/>	50	4_6620d50a58bb8	Cinnamon Roll	16.00	1
<input type="checkbox"/>	51	4_6620d50a58bb8	Croissant	10.00	1

4. Order Details Table:

- This table stores details of each product within an order.
- Columns:
 - **detail_id**: An auto-incrementing integer serving as the primary key.
 - **order_id**: The ID of the order to which the product belongs.
 - **product_name**: Name of the product.
 - **price_per_unit**: Price per unit of the product.
 - **quantity**: Quantity of the product ordered.
- Example Data:
 - Detail ID 36: Product "Asian Dolce Latte" ordered as part of order ID "4_6620d4eec6cf3" with a price of 18.00 and quantity of 1.

Showing rows 0 - 24 (30 total, Query took 0.0007 seconds.)

SELECT * FROM `products`

Profile [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

1 > >> Show all Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

	product_id	product_title	product_description	category_id	product_image	product_price
<input type="checkbox"/>	1	Americano	Rich, full-bodied espresso with hot water in true ...	1	american.jpg	10.00
<input type="checkbox"/>	2	Asian Dolce Latte	Introducing Asian Dolce Latte, the new smooth and ...	1	asianDolceLatte.jpg	18.00
<input type="checkbox"/>	3	Caffe Latte	Rich, full-bodied espresso in steamed milk, light...	1	caffLatte.jpg	14.00
<input type="checkbox"/>	4	Caffe Mocha	Espresso with bittersweet mocha sauce and steamed ...	1	caffMocha.jpg	16.00
<input type="checkbox"/>	5	Cappuccino	Espresso with steamed milk, topped with a deep lay...	1	cappuccino.jpg	14.00
<input type="checkbox"/>	6	Caramel Macchiato	Freshly steamed milk with vanilla-flavored syrup i...	1	caramelMacchiato.jpg	16.00
<input type="checkbox"/>	7	Cocoa Cappuccino	Dark, rich espresso with bittersweet mocha sauce i...	1	cocoaCappuccino.jpg	16.00
<input type="checkbox"/>	8	Coffee by the Press	The coffee press is a classic, straightforward bre...	1	caffPress.jpg	18.00
<input type="checkbox"/>	9	Cold Brew	Slow-steeped, small-batch and super smooth. We use...	1	coldBrew.jpg	18.00
<input type="checkbox"/>	10	Matcha Cold Foam Iced Americano	A dream come true for coffee and tea lovers – Cup ...	1	matchaFoamAmericano.jpg	20.00

5. Products Table:

- This table stores information about the products available for purchase.
- Columns:
 - **product_id**: An auto-incrementing integer serving as the primary key.
 - **product_title**: Title/name of the product.
 - **product_description**: Description providing details about the product.
 - **category_id**: ID of the category to which the product belongs.
 - **product_image**: Filename of the product image.
 - **product_price**: Price of the product.
- Example Data:
 - Product ID 1: "Americano" with a description and price, belonging to the "Coffee" category.

Showing rows 0 - 5 (6 total, Query took 0.0004 seconds.)

SELECT * FROM `users`

Profile [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

	id	firstname	lastname	username	email	password
<input type="checkbox"/>	4	Jien	Weng	Jr	reallyhat@gmail.com	12345678
<input type="checkbox"/>	5	Oscar	Chong	Oscar	oscar@gmail.com	Oscar1234
<input type="checkbox"/>	6	En	Yi	enyi	liewenyi@yahoo.com	12345678
<input type="checkbox"/>	7	John	Max	max	johnmax@hotmail.com	12345678
<input type="checkbox"/>	8	Oscar	Lai	Oscarr	oscar@hotmail.com	12345678
<input type="checkbox"/>	9	Enyi	Liew	ey	liewenyi0112@gmail.com	12345678

6. Users Table:

- This table stores information about users who have accounts on the Cup Corner website.
- Columns:
 - **id**: An auto-incrementing integer serving as the primary key.
 - **firstname**: First name of the user.
 - **lastname**: Last name of the user.
 - **username**: Unique username chosen by the user.
 - **email**: Email address of the user.
 - **password**: Encrypted password of the user.
- Example Data:
 - User ID 4: "Jien Weng Lai" with username "Jr", email "reallyhat@gmail.com", and encrypted password.

Code Snippets

Admin

The dedicated admin folder, suggested by the context, likely contains PHP scripts and other resources essential for administrative tasks. This includes management of the website's content, overseeing user accounts, handling product listings, and possibly viewing reports and analytics that guide decision-making.

admin/product_images

Name	Last commit message	Last commit date
..		
american.jpg	upload new version	2 days ago
appleTurnover.jpg	upload new version	2 days ago
asianDolceLatte.jpg	upload new version	2 days ago
atf.cake.png	add season products	yesterday
bolognese.jpg	upload new version	2 days ago
brewedTea.jpg	upload new version	2 days ago
caffelatte.jpg	upload new version	2 days ago
caffemocha.jpg	upload new version	2 days ago
caffepress.jpg	upload new version	2 days ago
cappuccino.jpg	upload new version	2 days ago
caramelMacchiato.jpg	upload new version	2 days ago
carbonara.jpg	upload new version	2 days ago
cinnamonRool.jpg	upload new version	2 days ago
cocoaCappuccino.jpg	upload new version	2 days ago
coldBrew.jpg	upload new version	2 days ago
croissant.jpg	upload new version	2 days ago
danish.jpg	upload new version	2 days ago
doubleShot.jpg	upload new version	2 days ago
hotChocolate.jpg	upload new version	2 days ago
lasagna.jpg	upload new version	2 days ago
lemonTea.jpg	upload new version	2 days ago
macAndCheese.jpg	upload new version	2 days ago
matchaFoamAmericano.jpg	upload new version	2 days ago
matchaLatte.jpg	upload new version	2 days ago
mfood.jpg	add season products	yesterday
new.cake.png	add season products	yesterday
new.coffee2.jpg	add season products	yesterday
pumpkinSpiceLatte.png	upload new version	2 days ago
strawberryLemonade.jpg	upload new version	2 days ago
tunaSandwich.jpg	upload new version	2 days ago

admin/all_payments.php

```
<?php

include("../include/connect.php");

// Product Sales

$query_product_sales = "SELECT product_name AS 'Product',
                           SUM(quantity) AS 'Unit Sold',
                           price_per_unit AS 'Price per Unit',
                           SUM(quantity * price_per_unit) AS 'Total Price'
                      FROM order_details
                     GROUP BY product_name";

$result_product_sales = mysqli_query($con, $query_product_sales);

// Category Sales

$query_category_sales = "SELECT c.category_title AS 'Category',
                           COUNT(od.product_name) AS 'Frequency of Category Sold',
                           SUM(od.quantity * od.price_per_unit) AS 'Total Category Sales'
                      FROM order_details od
                     INNER JOIN products p ON od.product_name = p.product_title
                     INNER JOIN categories c ON p.category_id = c.category_id
                     GROUP BY c.category_title";

$result_category_sales = mysqli_query($con, $query_category_sales);

// User Sales
```

```

$query_user_sales = "SELECT CONCAT(u.username) AS 'User',
                           COUNT(o.order_id) AS 'User Purchase Frequency',
                           SUM(o.total_payment) AS 'Total User Spending'
                     FROM orders o
                     INNER JOIN users u ON o.user_id = u.id
                     GROUP BY u.id";

$result_user_sales = mysqli_query($con, $query_user_sales);

// Payment Method Sales

$query_payment_method_sales = "SELECT payment_method AS 'Payment Method',
                           COUNT(order_id) AS 'Frequency of Payment Method Used',
                           SUM(total_payment) AS 'Total Sales by Payment Method'
                     FROM orders
                     GROUP BY payment_method";

$result_payment_method_sales = mysqli_query($con, $query_payment_method_sales);

// Order Sales

$query_order_sales = "SELECT order_id AS 'Order',
                           total_payment AS 'Order Price'
                     FROM orders";

$result_order_sales = mysqli_query($con, $query_order_sales);

// Total Sales

$query_total_sales = "SELECT SUM(total_payment) AS 'Total Sales'

```

```

    FROM orders";
```



```

$result_total_sales = mysqli_query($con, $query_total_sales);

$total_sales_row = mysqli_fetch_assoc($result_total_sales);

$total_sales = $total_sales_row['Total Sales'];

?>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Sales Report</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+ALEwiH" crossorigin="anonymous">

        <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-YvpcrYf0tY3IHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcldsIK1eN7N6jleHz" crossorigin="anonymous"></script>

        <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>

</head>

<body>

    <div class="container mt-5">

        <h2>Sales Report</h2>

        <!-- Product Sales -->

        <h3>Product Sales</h3>
```

```

<table class="table table-striped">

<thead>

<tr>

<th>Product</th>

<th>Unit Sold</th>

<th>Price per Unit</th>

<th>Total Price</th>

</tr>

</thead>

<tbody>

<?php while ($row = mysqli_fetch_assoc($result_product_sales)) { ?>

<tr>

<td><?php echo $row['Product']; ?></td>

<td><?php echo $row['Unit Sold']; ?></td>

<td>MYR <?php echo number_format($row['Price per Unit'], 2); ?></td>

<td>MYR <?php echo number_format($row['Total Price'], 2); ?></td>

</tr>

<?php } ?>

</tbody>

</table>

<!-- Category Sales -->

<h3>Category Sales</h3>

<table class="table table-striped">

<thead>

<tr>

<th>Category</th>

```

```

<th>Frequency of Category Sold</th>
<th>Total Category Sales</th>
</tr>
</thead>
<tbody>
<?php while ($row = mysqli_fetch_assoc($result_category_sales)) { ?>
<tr>
<td><?php echo $row['Category']; ?></td>
<td><?php echo $row['Frequency of Category Sold']; ?></td>
<td>MYR <?php echo number_format($row['Total Category Sales'], 2); ?></td>
</tr>
<?php } ?>
</tbody>
</table>

<!-- User Sales -->
<h3>User Sales</h3>
<table class="table table-striped">
<thead>
<tr>
<th>User</th>
<th>User Purchase Frequency</th>
<th>Total User Spending</th>
</tr>
</thead>
<tbody>
<?php while ($row = mysqli_fetch_assoc($result_user_sales)) { ?>

```

```

<tr>

<td><?php echo $row['User']; ?></td>

<td><?php echo $row['User Purchase Frequency']; ?></td>

<td>MYR <?php echo number_format($row['Total User Spending'], 2); ?></td>

</tr>

<?php } ?>

</tbody>

</table>

<!-- Payment Method Sales --&gt;

&lt;h3&gt;Payment Method Sales&lt;/h3&gt;

&lt;table class="table table-striped"&gt;

&lt;thead&gt;

&lt;tr&gt;

&lt;th&gt;Payment Method&lt;/th&gt;

&lt;th&gt;Frequency of Payment Method Used&lt;/th&gt;

&lt;th&gt;Total Sales by Payment Method&lt;/th&gt;

&lt;/tr&gt;

&lt;/thead&gt;

&lt;tbody&gt;

&lt;?php while ($row = mysqli_fetch_assoc($result_payment_method_sales)) { ?&gt;

&lt;tr&gt;

&lt;td&gt;&lt;?php echo $row['Payment Method']; ?&gt;&lt;/td&gt;

&lt;td&gt;&lt;?php echo $row['Frequency of Payment Method Used']; ?&gt;&lt;/td&gt;

&lt;td&gt;MYR &lt;?php echo number_format($row['Total Sales by Payment Method'], 2); ?&gt;&lt;/td&gt;

&lt;/tr&gt;

&lt;/tbody&gt;
</pre>

```

```

<?php } ?>

</tbody>

</table>

<!-- Order Sales --&gt;

&lt;h3&gt;Order Sales&lt;/h3&gt;

&lt;table class="table table-striped"&gt;

&lt;thead&gt;

&lt;tr&gt;

&lt;th&gt;Order&lt;/th&gt;

&lt;th&gt;Order Price&lt;/th&gt;

&lt;/tr&gt;

&lt;/thead&gt;

&lt;tbody&gt;

&lt;?php while ($row = mysqli_fetch_assoc($result_order_sales)) { ?&gt;

&lt;tr&gt;

&lt;td&gt;&lt;?php echo $row['Order']; ?&gt;&lt;/td&gt;

&lt;td&gt;MYR &lt;?php echo number_format($row['Order Price'], 2); ?&gt;&lt;/td&gt;

&lt;/tr&gt;

&lt;?php } ?&gt;

&lt;/tbody&gt;

&lt;/table&gt;

<!-- Total Sales --&gt;

&lt;h3&gt;Total Sales&lt;/h3&gt;

&lt;p&gt;Total: MYR &lt;?php echo number_format($total_sales, 2); ?&gt;&lt;/p&gt;

&lt;/div&gt;
</pre>

```

```
</body>
```

```
</html>
```

admin/delete_category.php

```
<?php

// Include the database connection file

include('../include/connect.php');

// Check if category ID is provided in the URL

if(isset($_GET['id'])) {

    $category_id = $_GET['id'];

    // Delete the category from the database

    $delete_query = "DELETE FROM categories WHERE category_id =
$category_id";

    $result = mysqli_query($con, $delete_query);

    if($result) {

        // Redirect to view categories page after successful deletion

        header("Location: index.php?view_categories");

        exit();

    } else {

        echo "Error deleting category.';

    }

} else {

    // Redirect to view categories page if category ID is not provided

    header("Location: index.php?view_categories");

    exit();

}
```

```
}
```

```
?>
```

admin/delete_product.php

```
<?php

// Include the database connection file
include('../include/connect.php');

// Check if product ID is provided in the URL
if(isset($_GET['id'])) {
    $product_id = $_GET['id'];

    // Delete the product from the database
    $delete_query = "DELETE FROM products WHERE product_id = " .
    $product_id;
    $result = mysqli_query($con, $delete_query);

    if($result) {
        // Redirect to view products page after successful deletion
        header("Location: index.php?view_products");
        exit();
    } else {
        echo "Error deleting product.";
    }
} else {
    // Redirect to view products page if product ID is not provided
    header("Location: index.php?view_products");
    exit();
}
```

```
}
```

```
?>
```

```
admin\delete_user.php
```

```
<?php

// Include database connection file
include("../include/connect.php");

// Check if ID parameter is set
if(isset($_GET['id'])) {
    $id = $_GET['id'];

    // Delete user from the database
    $query = "DELETE FROM users WHERE id = $id";
    $result = mysqli_query($con, $query);

    // Check if deletion was successful
    if($result) {
        echo "User deleted successfully.";
    } else {
        echo "Error deleting user.";
    }
} else {
    echo "Invalid request.";
}

?>
```

admin\edit_category.php

```
<?php

// Include the database connection file
include('../include/connect.php');

// Check if category ID is provided in the URL
if(isset($_GET['id'])) {
    $category_id = $_GET['id'];

    // Select the category details from the database
    $select_query = "SELECT * FROM categories WHERE category_id = $category_id";
    $result = mysqli_query($con, $select_query);

    // Check if the category exists
    if(mysqli_num_rows($result) > 0) {
        $category = mysqli_fetch_assoc($result);
    } else {
        // Redirect to view categories page if category not found
        header("Location: index.php?view_categories");
        exit();
    }
} else {
    // Redirect to view categories page if category ID is not provided
    header("Location: index.php?view_categories");
    exit();
}

// Check if form is submitted
```

```

if(isset($_POST['update_category'])) {
    $category_title = $_POST['category_title'];

    // Update the category in the database
    $update_query = "UPDATE categories SET
category_title='$category_title' WHERE category_id=$category_id";
    $result = mysqli_query($con, $update_query);

    if($result) {
        // Redirect to view categories page after successful update
        header("Location: index.php?view_categories");
        exit();
    } else {
        echo "Error updating category.";
    }
}

?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Edit Category</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            padding: 20px;
        }
    </style>

```

```
}

.container {
  max-width: 600px;
  margin: 0 auto;
  background-color: #fff;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
}

h3 {
  text-align: center;
  margin-bottom: 20px;
}

label {
  font-weight: bold;
}

input[type="text"],
textarea,
select {
  width: 100%;
  padding: 10px;
  margin-bottom: 15px;
  border: 1px solid #ccc;
  border-radius: 4px;
  box-sizing: border-box;
}
```

```
        resize: vertical;
    }

    textarea {
        height: 150px;
    }

    select {
        height: 40px;
    }

    input[type="submit"] {
        background-color: blue;
        color: #fff;
        border: none;
        padding: 10px 20px;
        border-radius: 4px;
        cursor: pointer;
        transition: background-color 0.3s;
    }

    input[type="submit"]:hover {
        background-color: navy;
    }

</style>

</head>

<body>

<div class="container">

    <h3>Edit Category</h3>
```

```

<form action="" method="post">

    <label for="category_title">Category Title:</label><br>

    <input type="text" id="category_title"
name="category_title" value="<?php echo
isset($category['category_title']) ? $category['category_title'] : '';
?>"><br>

    <input type="submit" name="update_category" value="Update
Category">

</form>

</div>

</body>

</html>

```

admin/edit_product.php

```

<?php

// Include the database connection file

include('../include/connect.php');

// Check if product ID is provided in the URL

if(isset($_GET['id'])) {

    $product_id = $_GET['id'];

    // Select the product details from the database

    $select_query = "SELECT * FROM products WHERE product_id =
$product_id";

    $result = mysqli_query($con, $select_query);

    // Check if the product exists

```

```

if(mysqli_num_rows($result) > 0) {

    $product = mysqli_fetch_assoc($result);

} else {

    // Redirect to view products page if product not found

    header("Location: index.php?view_products");

    exit();

}

} else {

    // Redirect to view products page if product ID is not provided

    header("Location: index.php?view_products");

    exit();

}

// Select all categories from the database

$category_query = "SELECT * FROM categories";

$category_result = mysqli_query($con, $category_query);

// Check if form is submitted

if(isset($_POST['update_product'])) {

    $product_title = $_POST['product_title'];

    $product_price = $_POST['product_price'];

    $product_description = $_POST['product_description'];

    $category_id = $_POST['category_id']; // Newly added

    // Update the product in the database

    $update_query = "UPDATE products SET
product_title='$product_title', product_price='$product_price',
product_description='$product_description', category_id='$category_id'
WHERE product_id=$product_id";

    $result = mysqli_query($con, $update_query);
}

```

```

if($result) {

    // Redirect to view products page after successful update

    header("Location: index.php?view_products");

    exit();

} else {

    echo "Error updating product.';

}

?>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Edit Product</title>

    <style>

        body {

            font-family: Arial, sans-serif;

            background-color: #f4f4f4;

            padding: 20px;

        }

        .container {

            max-width: 600px;

            margin: 0 auto;

            background-color: #fff;

```

```
padding: 20px;  
border-radius: 8px;  
box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);  
}  
  
h3 {  
text-align: center;  
margin-bottom: 20px;  
}  
  
label {  
font-weight: bold;  
}  
  
input[type="text"],  
textarea,  
select {  
width: 100%;  
padding: 10px;  
margin-bottom: 15px;  
border: 1px solid #ccc;  
border-radius: 4px;  
box-sizing: border-box;  
resize: vertical;  
}  
  
textarea {  
height: 150px;  
}
```

```

        select  {

            height: 40px;
        }

        input[type="submit"]  {

            background-color: blue;
            color: #fff;
            border: none;
            padding: 10px 20px;
            border-radius: 4px;
            cursor: pointer;
            transition: background-color 0.3s;
        }

        input[type="submit"]:hover  {

            background-color: navy;
        }
    
```

</style>

</head>

<body>

```

        <div class="container">

            <h3>Edit Product</h3>

            <form action="" method="post">

                <label for="product_title">Product Title:</label><br>

                <input type="text" id="product_title" name="product_title"
value=<?php echo $product['product_title']; ?>><br>

                <label for="product_price">Product Price (MYR) :</label><br>

```

```

<input type="text" id="product_price" name="product_price"
value="php echo $product['product_price']; ?&gt;"&gt;&lt;br&gt;

&lt;label for="product_description"&gt;Product
Description:&lt;/label&gt;&lt;br&gt;

&lt;textarea id="product_description"
name="product_description"&gt;<?php echo $product['product_description'];
?&gt;&lt;/textarea&gt;&lt;br&gt;

&lt;!-- Newly added: Dropdown list for categories --&gt;
&lt;label for="category_id"&gt;Category:&lt;/label&gt;&lt;br&gt;
&lt;select id="category_id" name="category_id"&gt;

    &lt;?php while($category =
mysqli_fetch_assoc($category_result)) : ?&gt;
        &lt;option value="<?php echo $category['category_id'];
?&gt;" &lt;?php if($category['category_id'] == $product['category_id']) echo
'selected'; ?&gt;
            &lt;?php echo $category['category_title']; ?&gt;
        &lt;/option&gt;
    &lt;?php endwhile; ?&gt;
&lt;/select&gt;&lt;br&gt;

&lt;input type="submit" name="update_product" value="Update
Product"&gt;

&lt;/form&gt;
&lt;/div&gt;
&lt;/body&gt;
&lt;/html&gt;
</pre

```

admin/edit_user.php

```
<?php

// Include the database connection file
include("../include/connect.php");

// Check if user ID is provided in the URL
if(isset($_GET['id'])) {
    $user_id = $_GET['id'];

    // Select the user details from the database
    $select_query = "SELECT * FROM users WHERE id = $user_id";
    $result = mysqli_query($con, $select_query);

    // Check if the user exists
    if(mysqli_num_rows($result) == 1) {
        $user = mysqli_fetch_assoc($result);
    } else {
        // Redirect to user list page if user not found
        header("Location: index.php?user_list");
        exit();
    }
} else {
    // Redirect to user list page if user ID is not provided
    header("Location: index.php?user_list");
    exit();
}

// Check if form is submitted
if(isset($_POST['update_user'])) {
```

```

$username = $_POST['username'];
$firstName = $_POST['firstName'];
$lastName = $_POST['lastName'];
$email = $_POST['email'];

// Update the user in the database

$update_query = "UPDATE users SET firstname='$firstName',
lastname='$lastName', email='$email' WHERE id=$user_id";
$result = mysqli_query($con, $update_query);

if($result) {
    // Redirect to user list page after successful update
    header("Location: index.php?user_list");
    exit();
} else {
    echo "Error updating user.";
}

?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Edit User</title>
    <style>
        body {
            font-family: Arial, sans-serif;

```

```
background-color: #f4f4f4;  
padding: 20px;  
}  
  
.container {  
max-width: 600px;  
margin: 0 auto;  
background-color: #fff;  
padding: 20px;  
border-radius: 8px;  
box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);  
}  
  
h3 {  
text-align: center;  
margin-bottom: 20px;  
}  
  
label {  
font-weight: bold;  
}  
  
input[type="text"],  
input[type="email"] {  
width: 100%;  
padding: 10px;  
margin-bottom: 15px;  
border: 1px solid #ccc;  
border-radius: 4px;
```

```

    box-sizing: border-box;
    resize: vertical;
}

input[type="submit"] {
    background-color: blue;
    color: #fff;
    border: none;
    padding: 10px 20px;
    border-radius: 4px;
    cursor: pointer;
    transition: background-color 0.3s;
}

input[type="submit"]:hover {
    background-color: navy;
}

</style>

</head>

<body>

<div class="container">

    <h3>Edit User</h3>

    <form action="" method="post">

        <label for="username">Username:</label><br>
        <input type="text" id="username" name="username" value="php echo $user['username']; ?&gt;" readonly&gt;&lt;br&gt;

        &lt;label for="firstName"&gt;First Name:&lt;/label&gt;&lt;br&gt;
        &lt;input type="text" id="firstName" name="firstName" value="<?php echo $user['firstname']; ?&gt;"&gt;&lt;br&gt;
</pre

```

```

<label for="lastName">Last Name:</label><br>

<input type="text" id="lastName" name="lastName"
value=<?php echo $user['lastname']; ?>><br>

<label for="email">Email:</label><br>

<input type="email" id="email" name="email" value=<?php
echo $user['email']; ?>><br>

<input type="submit" name="update_user" value="Update
User">

</form>

</div>

</body>

</html>

```

admin/inbox.php

```

<?php

// Include database connection file

include("../include/connect.php");




// Fetch all messages from the contact_responses table

$query = "SELECT * FROM contact_responses";

$result = mysqli_query($con, $query);

?


<!DOCTYPE html>

<html lang="en">

```

```

<head>

  <meta charset="UTF-8">

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

  <title>Inbox</title>

  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.m
in.css" rel="stylesheet"
integrity="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhj
Y6hW+ALEwIH" crossorigin="anonymous">

</head>

<body>

  <div class="container mt-5">

    <h2>Inbox</h2>

    <table class="table table-striped">

      <thead>

        <tr>

          <th>ID</th>

          <th>Name</th>

          <th>Email</th>

          <th>Created At</th>

        </tr>

      </thead>

      <tbody>

        <?php while ($row = mysqli_fetch_assoc($result)) { ?>

          <tr>

            <td><a href="message_details.php?id=<?php echo
$row['id']; ?>"><?php echo $row['id']; ?></a></td>

            <td><?php echo $row['name']; ?></td>

            <td><?php echo $row['email']; ?></td>

            <td><?php echo $row['created_at']; ?></td>

          </tr>

        <?php } ?>

      </tbody>

    </table>

  </div>

</body>

```

```
        </tr>

        <?php } ?>

    </tbody>

</table>

</div>

</body>

</html>
```

admin/index.php

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Admin</title>

    <link
        href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet"
        integrity="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pN1yT2bRjXh0JMhjY6hW+ALEwIH" crossorigin="anonymous">

    <style>

        body {

            background-color: #f8f9fa;
            font-family: Arial, sans-serif;
        }

        .header {

            background-color: #007bff;
```

```
        color: #fff;
        padding: 20px 0;
        text-align: center;
    }

.button-container {
    text-align: center;
    margin-top: 20px;
}

.button-container .btn {
    margin: 10px;
}

.container {
    margin-top: 30px;
    padding: 20px;
    background-color: #fff;
    border-radius: 5px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

</style>

</head>

<body>

<!-- Header -->

<div class="header">
    <h3>Manage Listing</h3>
</div>


```

```

<!-- Buttons -->

<div class="button-container">

    <a href="index.php?insert_product" class="btn btn-primary">Insert Products</a>

    <a href="index.php?view_products" class="btn btn-primary">View Products</a>

    <a href="index.php?insert_category" class="btn btn-primary">Insert Categories</a>

    <a href="index.php?view_categories" class="btn btn-primary">View Categories</a>

    <a href="index.php?view_orders" class="btn btn-primary">View Orders</a>

    <a href="index.php?all_payments" class="btn btn-primary">All Payments</a>

    <a href="index.php?user_list" class="btn btn-primary">List Users</a>

    <a href="index.php?inbox" class="btn btn-primary">Inbox</a>

</div>

<!-- Content -->

<div class="container mt-5">

    <?php

    // Include the appropriate file based on the URL parameters

    if(isset($_GET['insert_product'])){

        include('insert_product.php');

    }

    if(isset($_GET['insert_category'])){

        include('insert_category.php');

    }

    if(isset($_GET['view_products'])){

        include('view_products.php');

    }


```

```

    }

    if(isset($_GET['view_categories'])){

        include('view_categories.php');

    }

    if(isset($_GET['view_orders'])){

        include('view_orders.php');

    }

    if(isset($_GET['all_payments'])){

        include('all_payments.php');

    }

    if(isset($_GET['user_list'])){

        include('user_list.php');

    }

    if(isset($_GET['inbox'])){

        include('inbox.php');

    }

    ?>

    </div>

</body>

</html>

```

admin/insert_category.php

```

<?php

include('../include/connect.php');

if(isset($_POST['insert_category'])) {

```

```

$category_title=$_POST['category'];

// select data from database

$select_query="SELECT * FROM `categories` WHERE category_title
='$category_title';

$result_select=mysqli_query($con,$select_query);

$number=mysqli_num_rows($result_select);

if($number>0){

    echo "<script>alert('This category is already
existed.')</script>";

} else{

    $insert_query="insert into `categories` (category_title) VALUES
('{$category_title}');

$result = mysqli_query($con, $insert_query);

if($result){

    echo "<script>alert('Category has been inserted
successfully!')</script>";

}

}

?>

<form action="" method="post">

<label for="category">Enter Category: </label>

<br>

<input type="text" id="category" name="category"
placeholder="Insert category here">

<br>

<input type="submit" id="insert_category" name="insert_category">

</form>

```

admin/insert_product.php

```
<?php

include('..../include/connect.php');

?>

<form action="" method="post" enctype="multipart/form-data">

<!--title-->

<label for="product_title">Enter Product Title: </label>

<br>

<input type="text" id="product_title" name="product_title"
placeholder="Insert Product Title here" required="required"
class="form-control">

<br>

<!--description-->

<label for="product_description">Enter Product Description:
</label>

<br>

<textarea id="product_description" name="product_description"
placeholder="Insert Product description here" required="required"
class="form-select"></textarea>

<br>

<!--category-->

<label for="product_category">Select a Product Category: </label>

<select name="product_category" id="product_category"
class="form-select">

<option value="">Select a category</option>

<?php

$select_query="SELECT * FROM `categories`";

$result_query=mysqli_query($con,$select_query);
```

```

while($row=mysqli_fetch_assoc($result_query))

{
    $category_titles=$row['category_title'];
    $category_id=$row['category_id'];
    echo "<option
value='".$category_id'>$category_titles</option>";
}

?>

</select>
<br>

<!--image-->
<label for="product_image">Product Image: </label>
<input type="file" name="product_image" id="product_image"
required="required" class="form-control">
<br>

<!--price-->
<label for="product_price">Product Price: </label>
<input type="text" name="product_price" id="product_price"
placeholder="RM 0.00" required="required" class="form-control" >
<br>

<!--input-->
<input type="submit" name="insert_product" id="insert_product"
value="Insert Product">

</form>

<?php
if(isset($_POST['insert_product'])){


```

```

$product_title=$_POST['product_title'];
$product_description=$_POST['product_description'];
$product_category=$_POST['product_category'];
$product_price=$_POST['product_price'];

//accessing images
$product_image=$_FILES['product_image']['name'];

// accessing images tmp name
$temp_image=$_FILES['product_image']['tmp_name'];

// checking empty condition
if($product_title=='' or $product_description=='' or
$product_category=='' or $product_price=='' or $product_image=='') {
    echo "<script>alert('Please fill all the available
fields.')</script>";
    exit();
} else{

move_uploaded_file($temp_image,"./product_images/$product_image");

//insert query
$insert_products="INSERT INTO `products` (product_title,
product_description, category_id, product_image, product_price)

VALUES ('$product_title', '$product_description', '$product_category', '$product_image', '$product_price')";

$result_query=mysqli_query($con,$insert_products);

if($result_query) {
}
}

```

```
        echo "<script>alert('Successfully inserted  
product!')</script>";  
    }  
}  
}  
?>
```

admin/message_details.php

```
<?php  
  
// Include database connection file  
include("../include/connect.php");  
  
// Check if ID parameter is set  
  
if(isset($_GET['id'])) {  
    $id = $_GET['id'];  
  
    // Fetch message details from the database based on the ID  
    $query = "SELECT * FROM contact_responses WHERE id = $id";  
    $result = mysqli_query($con, $query);  
  
    // Check if message exists  
    if(mysqli_num_rows($result) == 1) {  
        $row = mysqli_fetch_assoc($result);  
        $name = $row['name'];  
        $email = $row['email'];  
        $message = $row['message'];  
        $created_at = $row['created_at'];  
  
        // Display message details
```

```

?>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Message Details</title>

<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pN1yT2bRjXh0JMhjY6hW+ALEwIH" crossorigin="anonymous">

</head>

<body>

<div class="container mt-5">

<h2>Message Details</h2>

<div class="mb-3">

<label for="name"
class="form-label">Name:</label>

<input type="text" class="form-control"
id="name" name="name" value="<?php echo $name; ?>" readonly>

</div>

<div class="mb-3">

<label for="email"
class="form-label">Email:</label>

<input type="email" class="form-control"
id="email" name="email" value="<?php echo $email; ?>" readonly>

</div>

<div class="mb-3">

<label for="message"
class="form-label">Message:</label>

```

```

        <textarea class="form-control" id="message"
name="message" rows="5" readonly><?php echo $message; ?></textarea>

        </div>

        <div class="mb-3">

            <label for="created_at"
class="form-label">Received At:</label>

            <input type="text" class="form-control"
id="created_at" name="created_at" value="<?php echo $created_at; ?>" readonly>

        </div>

        </div>

        <!-- Reply form -->

        <div class="container mt-5">

            <h2>Reply</h2>

            <div class="mb-3">

                <a href="mailto:<?php echo $email;
?>?subject=Reply%20to%20Your%20Inquiry&body=" class="btn
btn-primary">Send Reply</a>

            </div>

            </div>

            <!-- Back to Home button -->

            <div class="container mt-3">

                <a href="index.php?inbox" class="btn
btn-secondary">Back to Home</a>

            </div>

        </body>

        </html>

        <?php

    } else {

```

```
        echo "Message not found.";  
    }  
} else {  
    echo "Invalid request.";  
}  
?  
?
```

admin/order_details.php

```
<?php  
  
// Include the database connection file  
include('../include/connect.php');  
  
// Check if order ID is provided in the URL  
  
if(isset($_GET['order_id'])) {  
    $orderId = $_GET['order_id'];  
  
    // Select the order details from the database  
    $select_query = "SELECT * FROM orders WHERE order_id = '$orderId';  
    $result = mysqli_query($con, $select_query);  
  
    // Check if the order exists  
    if(mysqli_num_rows($result) > 0) {  
        $order = mysqli_fetch_assoc($result);  
    } else {  
        // Redirect to view orders page if order not found  
        header("Location: view_orders.php");  
        exit();  
    }  
}
```

```

} else {

    // Redirect to view orders page if order ID is not provided

    header("Location: view_orders.php");
    exit();
}

// Fetch order details from the order_details table based on the order
ID

$orderDetailsQuery = "SELECT * FROM order_details WHERE order_id =
'$orderId'";

$orderDetailsResult = mysqli_query($con, $orderDetailsQuery);

// Check if the query was successful

if ($orderDetailsResult) {

    // Fetch and store order details data

    $orderDetails = mysqli_fetch_all($orderDetailsResult,
    MYSQLI_ASSOC);

} else {

    // Handle database error

    $error = "Failed to fetch order details";
}

// Calculate total price

$totalPrice = 0;

foreach ($orderDetails as $orderDetail) {

    $totalPrice += $orderDetail['quantity'] *
    $orderDetail['price_per_unit'];
}

?>

```

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Order Details</title>

    <style>

        body {

            font-family: Arial, sans-serif;

            background-color: #f4f4f4;

            margin: 0;

            padding: 0;

        }

        .container {

            max-width: 800px;

            margin: 20px auto;

            background-color: #fff;

            border-radius: 5px;

            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

            padding: 20px;

        }

        h2 {

            margin-top: 0;

            color: #333;

        }

    </style>

</head>

<body>

    <div class="container">

        <h2>Order Details</h2>

        <table border="1">

            <thead>

                <tr>

                    <th>Product</th>

                    <th>Quantity</th>

                    <th>Price</th>

                </tr>

            </thead>

            <tbody>

                <tr>

                    <td>Laptop</td>

                    <td>1</td>

                    <td>$1200</td>

                </tr>

                <tr>

                    <td>Monitor</td>

                    <td>1</td>

                    <td>$300</td>

                </tr>

                <tr>

                    <td>Keyboard</td>

                    <td>1</td>

                    <td>$100</td>

                </tr>

                <tr>

                    <td>Mouse</td>

                    <td>1</td>

                    <td>$50</td>

                </tr>

            </tbody>

        </table>

        <div>

            <strong>Total: $1650</strong>

        </div>

    </div>

</body>

</html>
```

```
table {
    width: 100%;

    border-collapse: collapse;
    margin-bottom: 20px;
}

th, td {
    padding: 10px;
    text-align: left;
    border-bottom: 1px solid #ddd;
}

th {
    background-color: #f2f2f2;
}

.orders {
    margin-bottom: 20px;
}

p {
    margin: 0;
}

</style>

</head>

<body>

    <!-- Add your admin header/navbar here -->

    <div class="container">
```

```

<h2>Order Details</h2>

<!-- Display order information -->

<p><strong>Order ID:</strong> <?php echo $order['order_id'];
?></p>

<p><strong>User ID:</strong> <?php echo $order['user_id'];
?></p>

<p><strong>Shipping Address:</strong> <?php echo
$order['shipping_address']; ?></p>

<p><strong>Payment Method:</strong> <?php echo
$order['payment_method']; ?></p>

<p><strong>Date Created:</strong> <?php echo
$order['date_created']; ?></p>

<!-- Display order details table -->

<h3>Order Items</h3>

<table>

<thead>

<tr>

<th>Product Name</th>

<th>Quantity</th>

<th>Price Per Unit (MYR)</th>

<th>Total Price (MYR)</th>

</tr>

</thead>

<tbody>

<?php foreach ($orderDetails as $orderDetail) { ?>

<tr>

<td><?php echo $orderDetail['product_name'];
?></td>

<td><?php echo $orderDetail['quantity'];
?></td>

```

```

                <td><?php echo
number_format($orderDetail['price_per_unit'], 2); ?></td>

                <td><?php echo
number_format($orderDetail['quantity'] *
$orderDetail['price_per_unit'], 2); ?></td>

            </tr>

        <?php } ?>

    </tbody>

</table>

<p><strong>Total Payment (MYR) :</strong> MYR <?php echo
number_format($totalPrice, 2); ?></p>

</div>

<!-- Back to Admin Home Page button --&gt;

&lt;div class="container"&gt;

    &lt;a href="index.php?view_orders" class="btn"&gt;Back to Admin Home
Page&lt;/a&gt;

&lt;/body&gt;

&lt;/html&gt;
</pre>

```

admin/refund.php

```

<?php

// Include the database connection file

include('../include/connect.php');

// Function to delete an order and its details

function deleteOrder($orderId, $connection) {

    // Delete order details first

    $deleteOrderDetailsQuery = "DELETE FROM order_details WHERE
order_id = '$orderId';

```

```

$deleteOrderDetailsResult = mysqli_query($connection,
$deleteOrderDetailsQuery);

// Delete the order

$deleteOrderQuery = "DELETE FROM orders WHERE order_id =
'$orderId'";

$deleteOrderResult = mysqli_query($connection, $deleteOrderQuery);

// Return true if both queries executed successfully, otherwise
return false

return $deleteOrderResult && $deleteOrderDetailsResult;

}

// Check if delete request is submitted

if(isset($_GET['order_id'])) {

$orderIdToDelete = $_GET['order_id'];

$deleteSuccess = deleteOrder($orderIdToDelete, $con);

if($deleteSuccess) {

// Order and its details deleted successfully

echo "<script>alert('Order and its details refunded
successfully.')</script>";

} else {

// Error deleting order

echo "<script>alert('Error deleting order.')</script>";

}

// Redirect to view orders page

echo

"<script>window.location.replace('index.php?view_orders');</script>";

exit();

```

```

} else {
    // Redirect to view orders page if order ID is not provided
    header("Location: index.php?view_orders");
    exit();
}

?>

```

admin/user_list.php

```

<?php

// Include the database connection file
include('../include/connect.php');

// Select all users from the database
$select_query = "SELECT * FROM users";
$result = mysqli_query($con, $select_query);

?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>User List</title>
    <!-- Font Awesome CDN -->
    <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css">
    <style>

```

```
/* Table styles (same as before) */

table {
    width: 100%;
    border-collapse: collapse;
    margin-top: 20px;
}

th, td {
    padding: 10px;
    text-align: left;
    border-bottom: 1px solid #ddd;
}

th {
    background-color: #f2f2f2;
}

tr:hover {
    background-color: #f5f5f5;
}

/* Icon styles */

.icon {
    font-size: 18px;
    color: #007bff; /* Blue color */
    margin-right: 5px;
    cursor: pointer;
}

</style>
```

```

</head>

<body>

<h3>User List</h3>

<table>

<tr>

<th>ID</th>

<th>Username</th>

<th>Email</th>

<th>Name</th>

<th>Edit</th>

<th>Delete</th>

</tr>

<?php

// Fetch and display users

while ($row = mysqli_fetch_assoc($result)) {

    echo "<tr>";

    echo "<td>" . $row['id'] . "</td>";

    echo "<td>" . $row['username'] . "</td>";

    echo "<td>" . $row['email'] . "</td>";

    // Concatenate first name and last name

    echo "<td>" . $row['firstname'] . " " . $row['lastname'] . "</td>";

    // Edit icon

    echo "<td><a href='edit_user.php?id=" . $row['id'] . "'><i></i></a></td>";

    // Delete icon

    echo "<td><a href='delete_user.php?id=" . $row['id'] . "' onclick='return confirm(\"Are you sure you want to delete this user?\")'><i class='fas fa-trash-alt icon'></i></a></td>";

    echo "</tr>";

}

```

```
    }

    ?>

</table>

</body>

</html>
```

admin/view_categories.php

```
<?php

// Include the database connection file

include('../include/connect.php');

// Select all categories from the database

$select_query = "SELECT * FROM categories";

$result = mysqli_query($con, $select_query);

?>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>View Categories</title>

<!-- Font Awesome CDN -->

<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css">

<style>

/* Table styles (similar to view_products.php) */
```

```

table {
    width: 100%;

    border-collapse: collapse;
    margin-top: 20px;
}

th, td {
    padding: 10px;
    text-align: left;
    border-bottom: 1px solid #ddd;
}

th {
    background-color: #f2f2f2;
}

tr:hover {
    background-color: #f5f5f5;
}

/* Icon styles (same as view_products.php) */

.icon {
    font-size: 18px;
    color: #007bff; /* Blue color */
    margin-right: 5px;
    cursor: pointer;
}

</style>

</head>

```

```

<body>

    <h3>View Categories</h3>

    <table>

        <tr>

            <th>ID</th>

            <th>Category Name</th>

            <th>Edit</th>

            <th>Delete</th>

        </tr>

        <?php

            // Fetch and display categories

            while ($row = mysqli_fetch_assoc($result)) {

                echo "<tr>";

                echo "<td>" . $row['category_id'] . "</td>";

                echo "<td>" . $row['category_title'] . "</td>";

                // Edit icon

                echo "<td><a href='edit_category.php?id=" .
                $row['category_id'] . "'><i class='fas fa-edit icon'></i></a></td>";

                // Delete icon

                echo "<td><a href='delete_category.php?id=" .
                $row['category_id'] . "' onclick='return confirm(\"Are you sure you
                want to delete this category?\")'><i class='fas fa-trash-alt
                icon'></i></a></td>";

                echo "</tr>";

            }

        ?>

    </table>

</body>

</html>

```

admin/view_orders.php

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>View Orders</title>

    <style>

        /* Table styles (similar to previous examples) */

        table {

            width: 100%;

            border-collapse: collapse;

            margin-top: 20px;

        }

        th, td {

            padding: 10px;

            text-align: left;

            border-bottom: 1px solid #ddd;

        }

        th {

            background-color: #f2f2f2;

        }

        tr:hover {

            background-color: #f5f5f5;

        }

    </style>

</head>

<body>

    <h1>View Orders</h1>

    <table>

        <thead>

            <tr>

                <th>Order ID</th>

                <th>Product Name</th>

                <th>Quantity</th>

                <th>Price</th>

            </tr>

        </thead>

        <tbody>

            <tr>

                <td>12345678901234567890</td>

                <td>Product A</td>

                <td>1</td>

                <td>$10.99</td>

            </tr>

            <tr>

                <td>12345678901234567891</td>

                <td>Product B</td>

                <td>2</td>

                <td>$20.99</td>

            </tr>

            <tr>

                <td>12345678901234567892</td>

                <td>Product C</td>

                <td>1</td>

                <td>$15.99</td>

            </tr>

            <tr>

                <td>12345678901234567893</td>

                <td>Product D</td>

                <td>3</td>

                <td>$30.99</td>

            </tr>

        </tbody>

    </table>

</body>

</html>
```

```

    }

.link {
    color: blue;
    text-decoration: underline;
    cursor: pointer;
}

</style>

</head>

<body>

<h3>View Orders</h3>

<table>

<tr>
    <th>Order ID</th>
    <th>User ID</th>
    <th>Total Payment (MYR)</th>
    <th>Shipping Address</th>
    <th>Payment Method</th>
    <th>Date Created</th>
    <th>Action</th>
</tr>

<?php

// Include the database connection file

include('../include/connect.php');

// Select all orders from the database

$select_query = "SELECT * FROM orders";

$result = mysqli_query($con, $select_query);

```

```

// Fetch and display orders

while ($row = mysqli_fetch_assoc($result)) {

    echo "<tr>";

    echo "<td><a href='order_details.php?order_id=" .
$row['order_id'] . "' class='link'>" . $row['order_id'] . "</a></td>";

    echo "<td>" . $row['user_id'] . "</td>";

    echo "<td>MYR " . number_format($row['total_payment'], 2) . "</td>";

    echo "<td>" . $row['shipping_address'] . "</td>";

    echo "<td>" . $row['payment_method'] . "</td>";

    echo "<td>" . $row['date_created'] . "</td>";

    echo "<td><a href='refund.php?order_id=" . $row['order_id'] .
. "' onclick='return confirm(\"Are you sure you want to refund this
order?\")'>Refund</a></td>";

    echo "</tr>";

}

?>

</table>

</body>

</html>

```

admin/view_products.php

```

<?php

// include the database connection file

include('../include/connect.php');

// select all products from the database with their categories

$select_query = "SELECT p.product_id, p.product_title, p.product_price,
c.category_title FROM products p JOIN categories c ON p.category_id =
c.category_id";

```

```
$result = mysqli_query($con, $select_query);  
?  
  
<!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>  
    <!-- Font Awesome CDN -->  
    <link rel="stylesheet"  
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all  
.min.css">  
  
<style>  
    /* Table styles (same as before) */  
  
    table {  
        width: 100%;  
        border-collapse: collapse;  
        margin-top: 20px;  
    }  
  
    th, td {  
        padding: 10px;  
        text-align: left;  
        border-bottom: 1px solid #ddd;  
    }  
  
    th {  
        background-color: #f2f2f2;  
    }  
    .
```

```

        }

    tr:hover {
        background-color: #f5f5f5;
    }

    /* Icon styles */
    .icon {
        font-size: 18px;
        color: #007bff; /* Blue color */
        margin-right: 5px;
        cursor: pointer;
    }

</style>

</head>

<body>

<h3>View Products</h3>

<table>

    <tr>
        <th>ID</th>
        <th>Product Name</th>
        <th>Price (MYR)</th>
        <th>Category</th>
        <th>Edit</th>
        <th>Delete</th>
    </tr>

    <?php

        // fetch and display products

        while ($row = mysqli_fetch_assoc($result)) {

```

```

echo "<tr>";

echo "<td>" . $row['product_id'] . "</td>";
echo "<td>" . $row['product_title'] . "</td>";
echo "<td>MYR " . number_format($row['product_price'], 2) . "</td>";
echo "<td>" . $row['category_title'] . "</td>";
// Edit icon

echo "<td><a href='edit_product.php?id=" .
$row['product_id'] . "'><i class='fas fa-edit icon'></i></a></td>";
// Delete icon

echo "<td><a href='delete_product.php?id=" .
$row['product_id'] . "' onclick='return confirm(\"Are you sure you want
to delete this product?\")'><i class='fas fa-trash-alt
icon'></i></a></td>";

echo "</tr>";
}

?>

</table>

</body>

</html>

```

Functions

The common_function.php is use for get product details and each categories of product, it

functions/common_function.php

```

<?php

include ('./include/connect.php');

```

```

// Getting products

function getProducts()
{

    global $con;

    // Condition to check if category is set or not

    if (!isset($_GET['category'])) {

        $default_category_id = 1;

        $select_query = "SELECT * FROM `products` WHERE category_id = $default_category_id";

        $result_query = mysqli_query($con, $select_query);

        while ($row = mysqli_fetch_assoc($result_query)) {

            $product_id = $row['product_id'];

            $product_title = $row['product_title'];

            $product_description = $row['product_description'];

            $category_id = $row['category_id'];

            $product_image = $row['product_image'];

            $product_price = $row['product_price'];

            echo "<div class='col-md-4'>

                <div class='card product' style='width: 18rem;'

data-id='$product_id'>

                    <img

src='./admin/product_images/$product_image' class='card-img-top'

alt='...'

                    <div class='card-body'>

                        <h5 class='card-title'>$product_title</h5>

```



```

$product_id=$row['product_id'];

$product_title=$row['product_title'];

$product_description=$row['product_description'];

$category_id=$row['category_id'];

$product_image=$row['product_image'];

$product_price=$row['product_price'];

echo "<div class='col-md-4'>

    <div class='card product' style='width: 18rem;'

data-id='".$product_id'>

        <img

src='./admin/product_images/$product_image' class='card-img-top'

alt='...'

        <div class='card-body'>

            <h5 class='card-title'>$product_title</h5>

            <h5 class='card-subtitle'>RM

$product_price</h5>

            <a href='#' class='btn btn-primary'>Add to

Cart</a>

            <a href='#' class='btn btn-secondary'>View

More</a>

        </div>

    </div>

</div>";

}

}

}

function getCategories() {

    global $con;

    $select_categories="SELECT * FROM `categories`";

```

```

$result_categories=mysqli_query($con,$select_categories);

while($row=mysqli_fetch_assoc($result_categories)){
    $category_title=$row['category_title'];
    $category_id=$row['category_id'];

    echo "<li class='nav-item'>
        <a href='menu.php?category=$category_id' class='nav-link'>$category_title</a>
    </li>";
}

}

function getProductPreview(){
    global $con;

    // condition to check isset or not
    if(isset($_GET['category'])){
        $category_id = $_GET['category'];

        $select_query = "SELECT * FROM `products` WHERE category_id = $category_id";
    } else {
        // If no specific category is requested, retrieve products from the default category
        $default_category_id = 1;

        $select_query = "SELECT * FROM `products` WHERE category_id = $default_category_id";
    }

    $result_query=mysqli_query($con, $select_query);

    while($row=mysqli_fetch_assoc($result_query)){
        $product_id=$row['product_id'];
        $product_name=$row['product_name'];
        $product_desc=$row['product_desc'];
        $product_price=$row['product_price'];
        $product_image=$row['product_image'];
    }
}

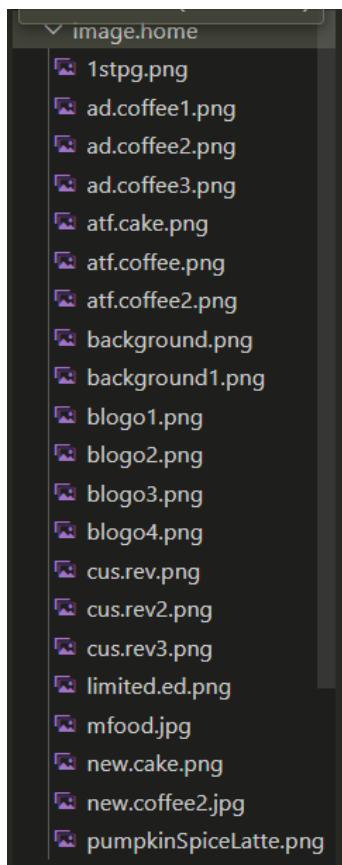
```

```
$product_title=$row['product_title'];
$product_description=$row['product_description'];
$category_id=$row['category_id'];
$product_image=$row['product_image'];
$product_price=$row['product_price'];

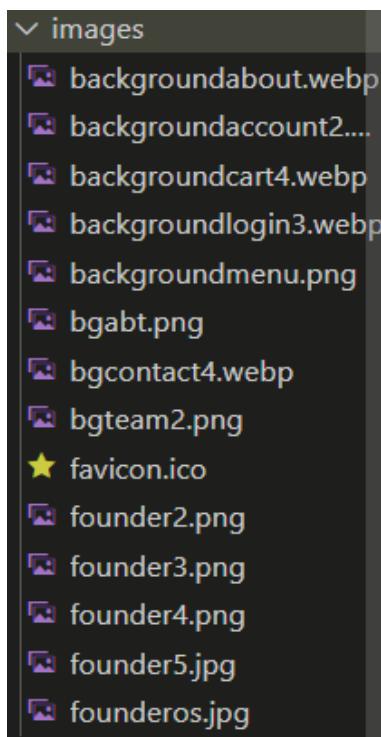
echo "<div class='preview' data-target='".$product_id.'>
    <i class='fa fa-times'></i>
    <img src='./admin/product_images/".$product_image.'>
    <h3>$product_title</h3>
    <p>$product_description</p>
    <div class='price'>RM$product_price</div>
</div>";
}

?>
```

Image.home



Images



Include

Includes folders with connect.php, footer.php and navbar.php to easily import functions to other PHP files for use. For example, use footer and navbar.php in menu.php and be able to connect mySQL database at any time.

include/connect.php

```
<?php

$con=mysqli_connect('localhost','root','','cup_corner');

if(!$con) {

    die(mysqli_error($con));

}

?>
```

include/footer.php

```
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css">

<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.2.0/css/all.min.css"
integrity="sha512-xh6O/CkQoPOWDdYTDqeRdPCVd1SpvCA9XXcUnZS2FmJNp1coAFzvtCN9BmamE+4aHK8yyUHUSCcJHgXloTyT2A=="
crossorigin="anonymous"
referrerpolicy="no-referrer" />

<div class="footer">

    <div class="footercontainer">

        <div class="row">

            <div class="footer-col-1">

                <a href="https://www.facebook.com/"><i
class="fa-brands fa-facebook"></i></a>
```

```

        <a href="https://www.instagram.com/"><i
class="fa-brands fa-instagram"></i></a>

        <a href="https://twitter.com/?lang=en"><i
class="fa-brands fa-twitter"></i></a>

        <a href="https://www.youtube.com/"><i
class="fa-brands fa-youtube"></i></a>

    </div>

    <div class="footer-col-2">

        <ul>

            <li><a href="ourteam.php">Our Team</a></li>

            <li><a href="about.php">About</a></li>

            <li><a href="contact.php">Contact Us</a></li>

        </ul>

    </div>

    <div class="footerBottom">

        <p>&copy; 2024 Cup Corner. All rights reserved.</p>

    </div>

</div>

</div>

<style>

.footer {

background-color: #111;
bottom: 0;
left: 0;

}

.footercontainer{

width: 100%;
```

```
padding: 70px 30px 20px;  
}  
  
.footer-col-1{  
    display: flex;  
    justify-content: center;  
}  
  
.footer-col-1 a{  
    text-decoration: none;  
    padding: 10px;  
    background-color: white;  
    margin: 10px;  
    border-radius: 50%;  
}  
  
.footer-col-1 a i{  
    font-size: 2em;  
    color: black;  
    opacity: 0.9;  
}  
  
.footer-col-1 a:hover{  
    background-color: #111;  
    transition: 0.5s  
}  
  
.footer-col-1 a:hover i{  
    color: white;
```

```
    transition: 0.5s;
}

.footer-col-2{
    margin: 30px 0;
}

.footer-col-2 ul{
    display: flex;
    justify-content: center;
    list-style-type: none;
}

.footer-col-2 ul li a{
    color: white;
    margin: 20px;
    text-decoration: none;
    font-size: 1.3em;
    opacity: 0.7;
    transition: 0.5s;
}

.footer-col-2 ul li a:hover{
    opacity: 1;
}

.footerBottom{
    padding: 20px;
    text-align: center;
}
```

```
.footerBottom p{  
    color: white;  
}  
</style>
```

include/navbar.php

```
<!DOCTYPE html>  
  
<html lang="en">  
  
<head>  
  
    <meta charset="UTF-8">  
  
    <meta http-equiv="X-UA-Compatible" content="IE=edge">  
  
    <meta name="viewport" content="width=device-width,  
initial-scale=1.0">  
  
    <title>Navbar</title>  
  
  
    <link  
    href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500  
;600;700&display=swap" rel="stylesheet">  
  
    <link href='https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css'  
rel='stylesheet'>  
  
</head>  
  
<body>  
  
<header class="header">  
    <a href="#" class="logo">Cup Corner</a>  
    <input type="checkbox" id="check">  
    <label for="check" class="icons">
```

```

        <i class='bx bx-menu' id="menu-icon"></i>

        <i class='bx bx-x' id="close-icon"></i>

    </label>

<nav class="navbar">

    <a href="index.php" style="--i:0;">Home</a>

    <a href="about.php" style="--i:1;">About</a>

    <a href="menu.php" style="--i:2;">Menu</a>

    <a href="contact.php" style="--i:3;">Contact</a>

    <a href="cart.php" style="--i:4;">Cart</a>

    <a href="account.php" style="--i:5;">Account</a>

</nav>

</header>

<style>

    * {

        margin: 0;
        padding: 0;
        box-sizing: border-box;
        font-family: "Poppins", sans-serif;
    }

.header{
    position: sticky;
    top: 0;
    left: 0;
    width: 100%;
    padding: 1.3rem 10%;
    display: flex;
    justify-content: space-between;
    align-items: center;
}

```

```
z-index: 100;  
  
background-image: url("bg2.png") ;  
}  
  
  
.header::before{  
  
content: '';  
  
position: absolute;  
  
top: 0;  
  
left: 0;  
  
width: 100%;  
  
height: 100%;  
  
background: rgba(0,0,0, .1);  
  
backdrop-filter: blur(50px);  
  
z-index: -1;  
}  
  
  
header::after{  
  
content: '';  
  
position: absolute;  
  
top: 0;  
  
left: -100%;  
  
width: 100%;  
  
height: 100%;  
  
background: linear-gradient(90deg, transparent, rgba(255,255,255,.4), transparent);  
  
z-index: -1;  
  
transition: .5s;  
}
```

```
.header:hover::after{
  left: 100%;
}

.logo{
  font-size: 1.5rem;
  color: #fff;
  text-decoration: none;
  font-weight: 700;
}

.navbar a{
  font-size: 1.15rem;
  color: #7B241C;
  text-decoration: none;
  font-weight: 500;
  margin-left: 2.5rem;
}

#check{
  display: none;
}

.icons{
  position: absolute;
  right: 5%;
  font-size: 2.8rem;
  color: #7B241C;
  cursor: pointer;
}
```

```
display: none;
}

@media (max-width: 992px) {

  .header{
    padding: 1.3rem 5%;

  }
}

@media (max-width: 768px) {

  .icons{
    display: inline-flex;
  }

  #check:checked~.icons #menu-icon{
    display: none;
  }

  .icons #close-icon{
    display: none;
  }

  #check:checked~.icons #close-icon{
    display: block;
  }

}

.navbar{
  position: absolute;
  top: 100%;
  left: 0;
  width: 100%;
```

```
height: 0;  
  
background: rgba(0, 0, 0, .1);  
  
backdrop-filter: blur(50px);  
  
box-shadow: 0.5rem 1rem rgba(0,0,0, .1);  
  
overflow: hidden;  
  
transition: .3s ease;  
}  
  
  
#check:checked~.navbar{  
  
height: 17.7rem;  
}  
  
  
.navbar a{  
  
display: block;  
  
font-size: 1.1rem;  
  
margin: 1.5rem 0;  
  
text-align: center;  
  
transform: translateY(-50px);  
  
opacity: 0;  
  
transition: color.3s ease;  
}  
  
  
#check:checked~.navbar a{  
  
transform: translateY(0);  
  
opacity: 1;  
  
transition-delay: calc(.15s * var(--i));  
}  
}
```

```
</style>

</body>

</html>
```

General

Within the general section, files such as config.php and check_login_status.php are instrumental in establishing a secure environment, where administrative credentials are verified and sessions are managed to maintain the integrity of the backend processes. The front-facing components like about.php, contact.php, login.php, menu.php, and their corresponding CSS files (aboutus.css, contactus.css, login.css, menu.css) ensure that the website remains aesthetically consistent and functionally responsive to provide administrators with an intuitive user interface for performing their duties.

about.php

```
<!DOCTYPE html>

<html lang="en">

<head>

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

    <title>About Us</title>

    <link rel="stylesheet" href="include/navbar.css">

    <link rel="stylesheet" type="text/css" href="aboutUs.css">

    <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500
;600;700&display=swap" rel="stylesheet">

</head>

<body>

<!--navbar-->

<?php include("include/navbar.php") ?>

<!--content-->

<section class="about" id="about">
```

```
<div class="section_container about_container">

  <div class="about_content">

    <h2 class="section_header">About us</h2>

    <p class="section_subheader">

      Cup Corner is founded by a group of coffee enthusiasts and has grown from humble beginnings to become a beloved destination for coffee lovers all over the world. Our journey began with a simple desire: to create a space where the rich aroma of freshly brewed coffee and the warmth of community could coexist seamlessly. Cup Corner stands today as a testament to our passion for coffee, serving a diverse selection of meticulously sourced and expertly crafted beverages. At the heart of our brand is a dedication to quality and a strong belief in building relationships over coffee. Join us at Cup Corner, where each sip reveals a story of passion and excellence.

    </p>

    <div class="about_flex">

      <div class="about_card">

        <h4>No.1</h4>

        <p>Coffee Brand</p>

      </div>

      <div class="about_card">

        <h4>2094</h4>

        <p>Stores worldwide</p>

      </div>

      <div class="about_card">

        <h4>3</h4>

        <p>Michelin Stars</p>

      </div>

    </div>

  </div>

  <div class="about_image">

  </div>


```

```
</div>

</div>

</section>

</body>

<footer>

<?php

include('include/footer.php');

?>

</footer>

</html>
```

aboutus.css

```
* {

margin: 0;
padding: 0;
font-family: 'Poppins', sans-serif;
box-sizing: border-box;
}

:root {

--primary-color: #8f6b4c;
--dark-color: #5C4033;
--accent-color: #C19A6B;
--text-color: #333;
--background-color: #FFF8DC;
}

.about {

position: relative;
```

```
min-height: 100vh;  
padding: 50px 100px;  
display: flex;  
justify-content: center;  
align-items: center;  
flex-direction: column;  
background-image: url("images/bgabt.png");  
background-size: cover;  
}  
  
.section_header {  
color: var(--dark-color);  
font-size: 2.8em;  
font-weight: 800;  
margin-bottom: 0.5em;  
}  
  
.section_subheader {  
color: var(--text-color);  
line-height: 1.6;  
margin-bottom: 1em;  
font-size: 1.1em;  
max-width: 60ch;  
font-weight: 600;  
font-family: 'Open Sans', sans-serif;  
text-align: justify;  
}  
  
.about_container {
```

```
display: grid;
grid-template-columns: repeat(2, 1fr);
gap: 2rem;
align-items: center;
}

.about_flex{
margin: 2rem 0;
padding: 2rem 1rem;
display: flex;
align-items: center;
gap: 1rem;
flex-wrap: nowrap;
background-color: var(--light-brown);
}

.about_card{
flex: 0 0 auto;
background-color: var(--background-color);
padding: 1.5em;
border-radius: 8px;
box-shadow: 0px 8px 16px 0px rgba(0,0,0,0.1);
margin-bottom: 1em;
text-align: center;
}

.about_card:last-child{
border: none;
}
```

```
.about_card h4{  
  color: var(--primary-color);  
  font-size: 1.8em;  
  margin-bottom: 0.25em;  
}  
  
.about_card p{  
  color: var(--text-color);  
  font-size: 1em;  
}  
  
.about_image img{  
  max-width: 450px;  
  margin: auto;  
  margin-left: 80px;  
  margin-bottom: 100px;  
  border-radius: 50%;  
  box-shadow: 0 0 25px var(--accent-color);  
  cursor: pointer;  
  transition: 0.4s ease-in-out;  
}  
  
.about_image img:hover{  
  box-shadow: 0 0 25px var(--accent-color),  
              0 0 50px var(--accent-color),  
              0 0 100px var(--accent-color);  
}
```

```
@media (max-width: 768px) {  
  
    .about_container {  
  
        grid-template-columns: 1fr;  
    }  
  
    .about_image img {  
  
        margin-left: 0;  
        max-width: 100%;  
        display: block;  
    }  
  
    .about {  
  
        padding: 20px;  
    }  
  
    .section_subheader {  
  
        max-width: none;  
    }  
}
```

account.php

```
<?php  
  
    session_start();  
  
    include("config.php");  
  
  
    // Check if the user is logged in  
  
    if (!isset($_SESSION['userId']) || empty($_SESSION['userId'])) {  
  
        header("Location: login.php");  
  
        exit();  
    }  
  
  
    // Fetch user's personal information from the database
```

```

$userId = $_SESSION['userId'];

$query = "SELECT * FROM users WHERE id = $userId";

$result = mysqli_query($con, $query);

// Check if the query was successful

if ($result) {

    $userData = mysqli_fetch_assoc($result);

} else {

    // Handle database error

    $error = "Failed to fetch user data";

}

// Fetch user's orders from the database and sort by date

$orderQuery = "SELECT * FROM orders WHERE user_id = $userId ORDER
BY date_created DESC";

$orderResult = mysqli_query($con, $orderQuery);

// Check if the query was successful

if ($orderResult) {

    // Fetch and store orders data

    $orders = mysqli_fetch_all($orderResult, MYSQLI_ASSOC);

} else {

    // Handle database error

    $error = "Failed to fetch user orders";

}

// Logout function

if (isset($_POST['logout'])) {

    // Unset all session variables

```

```

$_SESSION = array();

// Destroy the session

session_destroy();

// Redirect to login page after logout

header("Location: login.php");

exit();

}

?>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>My Account</title>

<link rel="stylesheet" href="reAccount.css">

</head>

<body>

<!-- Include navbar -->

<?php include("include/navbar.php") ?>

<div class="container">

<div class="box">

<div class="form-box">

<h1>My Account</h1>

<!-- Display user's personal information -->

```

```

<div class="user-info">

    <p><strong>First Name:</strong> <?php echo
$userData['firstname']; ?></p>

    <p><strong>Last Name:</strong> <?php echo
$userData['lastname']; ?></p>

    <p><strong>Username:</strong> <?php echo
$userData['username']; ?></p>

    <p><strong>Email:</strong> <?php echo
$userData['email']; ?></p>

    <!-- You can display more information here if
needed -->

</div>

<!-- Logout button -->

<form action="" method="post">

    <input type="submit" name="logout" value="Logout"
class="logout-button">

</form>

</div>

</div>

<!-- Display user's orders -->

<div class="container">

    <div class="box">

        <div class="form-box">

            <h2>My Orders</h2>

            <div class="orders">

                <?php if (isset($orders) && !empty($orders)) { ?>

                    <table>

                        <thead>

                            <tr>

```

```

<th>Order ID</th>
<th>Total Payment</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<?php foreach ($orders as $order) { ?>
<tr>
<td><a href="order_details.php?order_id=<?php echo $order['order_id']; ?><?php echo $order['order_id']; ?>></a></td>
<td><?php echo $order['total_payment']; ?></td>
<td><?php echo $order['date_created']; ?></td>
</tr>
<?php } ?>
</tbody>
</table>
<?php } else { ?>
<p>No orders found.</p>
<?php } ?>
</div>
</div>
</div>
<footer>
<?php include('include/footer.php'); ?>
</footer>

```

```
</body>
```

```
</html>
```

add_to_cart.php

```
<?php

session_start();

if (!isset($_SESSION['cart'])) {
    $_SESSION['cart'] = array();
}

$data = json_decode(file_get_contents('php://input'), true);

if (isset($data['productId'])) {
    $productId = $data['productId'];
    $increment = $data['increment'] ?? false;

    if (!isset($_SESSION['cart'][$productId])) {
        $_SESSION['cart'][$productId] = 1;
    } else if ($increment) {
        $_SESSION['cart'][$productId]++;
    }
}

echo json_encode(['message' => 'Product added to cart
successfully.']);
} else {
    echo json_encode(['message' => 'Product ID is missing.']);
}
```

```
?>
```

cart.css

```
body {
  height: 100vh;
  display: flex;
  flex-direction: column;
  font-family: 'Poppins', sans-serif;
  background-image: url("images/backgroundcart4.webp");
}

.container {
  width: 80%;
  margin: auto;
  padding: 20px;
  background-image: url("image.home/background.png");
  border-radius: 8px;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
}

h1, h2 {
  color: #333;
  text-align: center;
  margin-top: 1em;
  margin-bottom: 1em;
}

.cart-table {
```

```
width: 100%;

margin-bottom: 2em;
}

.cart-table th, .cart-table td {

border: 1px solid #ddd;
padding: 15px;
text-align: left;
}

.cart-table th {

background-color: burlywood;
color: black;
}

.cart-item-image {

width: 130px;
height: auto;
border-radius: 5px;
margin-right: 30px;
}

.cart-item-name {

font-size: 1.2rem;
font-weight: 600;
color: #333;
margin-left: 15;
}
```

```
.btn-remove {  
  color: #f44336;  
  cursor: pointer;  
  font-size: 1rem;  
  font-weight: 500;  
  transition: color 0.3s ease;  
}  
  
.btn-remove:hover {  
  color: #cc0000;  
  text-decoration: underline;  
}  
  
/* Enhancements for the Checkout and Remove buttons */  
  
.btn-success, .btn-remove {  
  padding: 10px 20px;  
  border: none;  
  border-radius: 4px;  
  text-transform: uppercase;  
  letter-spacing: 0.5px;  
}  
  
.btn-success {  
  background-color: #4CAF50;  
  color: white;  
}  
  
.btn-success:hover {  
  background-color: #43a047;
```

```
}

/* Responsive table */

@media screen and (max-width: 768px) {

    .cart-table, .cart-table thead, .cart-table tbody, .cart-table th,
    .cart-table td, .cart-table tr {

        display: block;

        width: 100%;

    }

    .cart-table th, .cart-table td {

        text-align: right;

    }

    .cart-table td::before {

        content: attr(data-label);

        float: left;

        font-weight: bold;

    }

    .cart-table td {

        text-align: right;

    }

    .container {

        padding: 10px;

    }

}
```

```
/* Additional styles for buttons and inputs */

input[type="text"], select.form-select {
    width: 100%;

    padding: 12px 20px;

    margin: 8px 0;

    display: inline-block;

    border: 1px solid #ccc;

    border-radius: 4px;

    box-sizing: border-box;
}

button:hover, input[type="submit"]:hover {
    opacity: 0.8;
}
```

cart.php

```
<?php

session_start();

include('./include/connect.php');

include('./functions/common_function.php');

include("config.php");

if (!isset($_SESSION['valid'])) {
    header('Location: login.php');
    exit;
}
```

```

function getUserInfo() {
    if (isset($_SESSION['userId']) && isset($_SESSION['username'])) {
        return [
            'userId' => $_SESSION['userId'],
            // 'username' => $_SESSION['username']
        ];
    }
    return null;
}

if (!isset($_SESSION['cart'])) {
    $_SESSION['cart'] = [];
}

$cart_products = [];
$total_price = 0;

foreach ($_SESSION['cart'] as $product_id => $quantity) {
    $query = "SELECT * FROM products WHERE product_id = '$product_id'";
    $result = mysqli_query($con, $query);
    if ($row = mysqli_fetch_assoc($result)) {
        $row['quantity'] = $quantity;
        $cart_products[] = $row;
        $total_price += $row['product_price'] * $quantity;
    }
}

?>

```

```

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Cart</title>

    <link rel="stylesheet" href="Cart.css">

</head>

<body>

    <?php include("include/navbar.php") ?>

    <div class="container mt-5">

        <h1 class="mb-4" style="text-align: center;
padding:20px;">Cart</h1>

        <?php if(count($cart_products) > 0): ?>

        <div class="row">

            <div class="col-md-8">

                <table class="table cart-table">

                    <thead>

                        <tr>

                            <th scope="col">Item</th>

                            <th scope="col">Quantity</th>

                            <th scope="col">Price</th>

                            <th scope="col"></th>

                        </tr>

                    </thead>

                    <tbody>

</div>

```

```

<?php foreach ($cart_products as $product): ?>

<tr>

    <td>

        <span class="cart-item-name"><?php echo
htmlspecialchars($product['product_title']); ?></span>

    </td>

    <td>

        <?php echo $product['quantity']; ?>

    </td>

    <td>$<?php echo
number_format($product['product_price'], 2); ?></td>

    <td>

        <span class="btn-remove"
data-productid="<?php echo $product['product_id']; ?>">Remove</span>

    </td>

</tr>

<?php endforeach; ?>

</tbody>

</table>

</div>

<div class="col-md-4">

    <div class="border p-3 mb-3">

        <h2
style:text-align="right";padding:30px;font-size:50;>Total: $<?php echo
number_format($total_price, 2); ?></h2>

    </div>

    <br>

```

```

<?php

    $userId = $_SESSION['userId'] ?? 'Not logged
in';

    // $username = $_SESSION['username'] ?? 'Not
available';

    // Ensure this key matches exactly

    echo "UserID: " . htmlspecialchars($userId) .
"<br>";

    //echo "Username: " .
htmlspecialchars($username) . "<br>";


?>

<!-- Checkout form -->

<div class="border p-3">

    <h2 style="padding-top:50px">Shipping &
Payment</h2>

    <form action="checkout.php" method="post">

        <div class="mb-3">

            <label for="shippingAddress"
class="form-label" style="font-size:18px; padding:30px;">Shipping
Address</label>

            <input type="text" class="form-control"
id="shippingAddress" name="shipping_address" required>

        </div>

        <div class="mb-3">

            <label for="paymentMethod"
class="form-label" style="font-size:18px; padding:30px;">Payment
Method</label>

            <select class="form-select"
id="paymentMethod" name="payment_method" required>


```

```

        <option value="Credit Card">Credit
Card</option>

        <option
value="PayPal">PayPal</option>

        <option value="Bank Transfer">Bank
Transfer</option>

    </select>

</div>

<button type="submit" class="btn
btn-success" style="font-size: 16px; padding: 5px 30px; text-align:
center;">Checkout</button>

</form>

</div>

</div>

<?php else: ?>

<p style="text-align: center; font-weight: 600;
padding: 50px;">Your cart is empty.</p>

<?php endif; ?>

</div>

<script>

document.addEventListener('DOMContentLoaded', function() {
    const removeButtons = document.querySelectorAll('.btn-remove');

    removeButtons.forEach(button => {
        button.addEventListener('click', function() {
            const productId = this.getAttribute('data-productid');
            fetch('remove_from_cart.php', {
                method: 'POST',
                body: JSON.stringify({ productId: productId }),
            })
        })
    })
})

```

```

        headers: {
            'Content-Type': 'application/json'
        }
    })
    .then(response => response.json())
    .then(data => {
        if(data.status === 'success') {
            window.location.reload();
        } else {
            alert(data.message);
        }
    })
    .catch((error) => {
        console.error('Error:', error);
    });
}
);
}

</script>

<script
src="https://code.jquery.com/jquery-3.6.0.slim.min.js"></script>

<script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.1/dist/umd/popper.min.js"></script>

<footer>
    <?php include('include/footer.php'); ?>
</footer>
</body>
</html>

```

check_login_status.php

```
<?php

session_start();

$response = ['loggedIn' => false];

if (isset($_SESSION['valid'])) {

    $response['loggedIn'] = true;

}

echo json_encode($response);

?>
```

checkout.php

```
<?php

session_start();

include('config.php');

if (!isset($_SESSION['userId']) || $_SESSION['userId'] == null) {

    header('Location: login.php');

    exit;

}

//error_log("All session variables: " . print_r($_SESSION, true));

if (!isset($_SESSION['userId'])) {

    error_log("Session userId not set. Redirecting to login.");
}
```

```

        header('Location: login.php');

        exit;

} else {

    error_log("User is logged in with userId=" . $_SESSION['userId']);

}

// Process the checkout

if ($_SERVER["REQUEST_METHOD"] == "POST") {

    // Generate unique order ID

    $order_id = uniqid($_SESSION['userId'] . '_');

    // Retrieve form data

    $user_id = $_SESSION['userId'];

    $shipping_address = $_POST['shipping_address'];

    $payment_method = $_POST['payment_method'];

    $date_created = date('Y-m-d H:i:s');

    $total_payment = 0;

    // Begin database transaction

    mysqli_begin_transaction($con);

    try {

        // Calculate total payment

        foreach ($_SESSION['cart'] as $product_id => $quantity) {

            // Retrieve product price

            $product_query = "SELECT product_price FROM products WHERE
product_id = ?";

            $product_stmt = $con->prepare($product_query);

            $product_stmt->bind_param("i", $product_id);

            $product_stmt->execute();

            $product_stmt->bind_result($product_price);

            $product_stmt->fetch();

            $product_stmt->close();

            $total_payment += $product_price * $quantity;

        }

        // Insert into database

        $insert_query = "INSERT INTO orders (user_id, order_id, total_payment, date_created, payment_method, shipping_address) VALUES (?, ?, ?, ?, ?, ?)";

        $insert_stmt = $con->prepare($insert_query);

        $insert_stmt->bind_param("iissss", $user_id, $order_id, $total_payment, $date_created, $payment_method, $shipping_address);

        $insert_stmt->execute();

        $insert_stmt->close();

    }

}

// Redirect to success page

header('Location: success.php');

exit;

```

```

$product_stmt->execute();

$product_result = $product_stmt->get_result();

$product_data = $product_result->fetch_assoc();

$price_per_unit = $product_data['product_price'];

$total_payment += $quantity * $price_per_unit;

}

// Insert order into orders table

$insert_order_query = "INSERT INTO orders (order_id, user_id,
total_payment, shipping_address, payment_method, date_created)

VALUES (?, ?, ?, ?, ?, ?)";

$insert_order_stmt = $con->prepare($insert_order_query);

$insert_order_stmt->bind_param("siisss", $order_id, $user_id,
$total_payment, $shipping_address, $payment_method, $date_created);

$insert_order_stmt->execute();

$insert_order_stmt->close();

// Insert order details for each item in cart

foreach ($_SESSION['cart'] as $product_id => $quantity) {

// Retrieve product details

$product_query = "SELECT product_title, product_price FROM
products WHERE product_id = ?";

$product_stmt = $con->prepare($product_query);

$product_stmt->bind_param("i", $product_id);

$product_stmt->execute();

$product_result = $product_stmt->get_result();

$product_data = $product_result->fetch_assoc();

$product_name = $product_data['product_title'];

$price_per_unit = $product_data['product_price'];

$total_payment += $quantity * $price_per_unit;
}

```

```

        // Insert order details into order_details table

        $insert_detail_query = "INSERT INTO order_details
(order_id, product_name, price_per_unit, quantity)
VALUES (?, ?, ?, ?)";

$insert_detail_stmt = $con->prepare($insert_detail_query);

$insert_detail_stmt->bind_param("ssdi", $order_id,
$product_name, $price_per_unit, $quantity);

$insert_detail_stmt->execute();

$insert_detail_stmt->close();

}

// Commit transaction

mysqli_commit($con);

mysqli_commit($con);

echo "Order placed successfully. Redirecting back to cart in 3
seconds...";

unset($_SESSION['cart']);

echo "<meta http-equiv='refresh' content='3;url=cart.php'>";

exit;

} catch (mysqli_sql_exception $exception) {

mysqli_rollback($con);

throw $exception;

}

?>

```

config.php

```
<?php

$servername = "localhost";
$username = "root";
$password = "";
$dbname = "cup_corner";

$con = mysqli_connect($servername, $username, $password, $dbname);

if (!$con) { // Change $conn to $con
    die("Connection failed: " . mysqli_connect_error());
}

?>
```

contact.php

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Contact Us</title>

    <link rel="stylesheet" href="include/navbar.css">

    <link rel="stylesheet" type="text/css" href="contactus.css">

    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css">

    <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500
;600;700&display=swap" rel="stylesheet">

</head>

<body>
```

```

<?php

session_start();

include("include/navbar.php");

if(isset($_SESSION['feedback'])): ?>

<div class="feedback"><?= $_SESSION['feedback']; ?></div>

<?php unset($_SESSION['feedback']); // Remove message after
displaying

endif;

?>

<section class="contact">

<div class="content">

<h2>Contact Us</h2>

<p>Have a question or feedback? Feel free to reach out to us - we'd love to hear from you! Contact Cup Corner today for any inquiries or assistance.</p>

</div>

<div class="container">

<div class="contactInfo">

<div class="box">

<div class="icon"><i class="fa
fa-map-marker"></i></div>

<div class="text">

<h3>Address</h3>

<p>1, Jalan Sungai Long, <br>43200 Kajang, Selangor</p>

</div>

</div>

<div class="box">

```

```

<div class="icon"><i class="fa fa-phone"></i></div>

<div class="text">
    <h3>Phone</h3>
    <p>012-1232094</p>
</div>

</div>

<div class="box">
    <div class="icon"><i class="fa fa-envelope"></i></div>
    <div class="text">
        <h3>Email</h3>
        <p>cupcorner@gmail.com</p>
    </div>
</div>

</div>

<div class="contactform">
    <form action="submit_contact.php" method="post">
        <h2>Feedback</h2>
        <div class="inputBox">
            <input type="text" name="name" required="required">
            <span>Full Name</span>
        </div>
        <div class="inputBox">
            <input type="text" name="email" required="required">
            <span>Email</span>
        </div>
        <div class="inputBox">
            <textarea name="message" required="required"></textarea>
            <span>Type your Message...</span>
        </div>
    </form>
</div>

```

```

        </div>

        <div class="inputBox">
            <input type="submit" name="submit" value="Send">
        </div>
    </form>
</div>
</section>
</body>

<footer>
<?php include('include/footer.php'); ?>
</footer>
</html>

```

contactus.css

```

* {
    margin: 0;
    padding: 0;
    font-family: 'Poppins', sans-serif;
    box-sizing: border-box;
}

.contact{
    position: relative;
    min-height: 100vh;
    padding: 50px 100px;
    display: flex;
    justify-content: center;
    align-items: center;
    flex-direction: column;
}

```

```
background: url("images/bgcontact4.webp") no-repeat center center
fixed;

background-size: cover;

}

.contact .content{

max-width: 800px;

text-align: center;

}

.contact .content h2{

font-size: 36px;

font-weight: 700;

color: #5C4033;

margin-bottom: 10px;

}

.contact .content p{

color: #333;

font-weight: 500;

font-size: 18px;

line-height: 1.6;

margin-bottom: 20px;

}

.container{

width: 100%;

display: flex;

justify-content: center;
```

```
  align-items: center;
  margin-top: 30px;
}

.container .contactInfo{
  width: 50%;
  display: flex;
  flex-direction: column;
}

.container .contactInfo .box{
  position: relative;
  padding: 20px 0;
  display: flex;
}

.container .contactInfo .box .icon{
  color: #5C4033;
  min-width: 60px;
  height: 60px;
  background: #FFF8DC;
  display: flex;
  justify-content: center;
  align-items: center;
  border-radius: 50%;
  font-size: 25px;
  margin-left: 25px;
}

.container .contactInfo .box .text{
```

```
display: flex;
margin-left: 20px;
font-size: 20px;
color: #5C4033;
flex-direction: column;
font-weight: 500;
}

.container .contactInfo .box .text h3{
font-weight: 700;
color: #8B5A2B;
}

.contactform{
width: 40%;
padding: 40px;
background: #ffffff;
box-shadow: 0 5px 25px rgba(0,0,0,0.1);
border-radius: 8px;
}

.contactform h2{
font-size: 30px;
color: #5C4033;
font-weight: 500;
margin-bottom: 20px;
}

.contactform .inputBox{
```

```
position: relative;
margin-top: 10px;
}

.contactform .inputBox input, .contactform .inputBox textarea{
width: 100%;
padding: 5px 0;
font-size: 16px;
margin: 10px 0;
border: none;
border-bottom: 2px solid #333;
outline: none;
resize: none;
}

.contactform .inputBox span{
position: absolute;
left: 0;
padding: 5px 0;
font-size: 16px;
margin: 10px 0;
pointer-events: none;
transition: 0.5s;
color: #666;
}

.contactform .inputBox input:focus ~span,
.contactform .inputBox input:valid ~span,
.contactform .inputBox textarea:focus ~span,
```

```
.contactform .inputBox textarea:valid ~span{  
  color: #e91e63;  
  font-size: 12px;  
  transform: translateY(-20px);  
}  
  
.contactform .inputBox input[type="submit"]{  
  width: 100px;  
  background: #A0522D;  
  color: #fff;  
  border: none;  
  cursor: pointer;  
  padding: 10px;  
  font-size: 18px;  
}  
  
@media (max-width: 991px){  
  .contact{  
    padding: 50px;  
  }  
  .container{  
    flex-direction: column;  
  }  
  .container .contactInfo{  
    margin-bottom: 40px;  
  }  
  .container .contactInfo, .contactform{  
    width: 100%;  
  }  
}
```

index.php

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Cup Corner</title>

    <link rel="icon" type="image/x-icon" href="images/favicon.ico">

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

    <link
        href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500
        ;600;700&display=swap" rel="stylesheet">

    <link rel="stylesheet"
        href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-aw
        esome.min.css">

    <link rel="stylesheet"
        href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.2.0/css/all
        .min.css"
        integrity="sha512-xh6O/CkQoPOWDdYTDqeRdPCVd1SpvCA9XXcUnZS2FmJNp1coAFzvt
        CN9BmamE+4aHK8yyUHUSCcJHgXloTyT2A==" crossorigin="anonymous"
        referrerPolicy="no-referrer" />

</head>

<body>

<!--navbar-->

<?php
    include("include/navbar.php")
?>

<!--content-->
```

```

<div class="content">

    <div class="container">

        <div class="row">
            <div class="col-6">
                <h1>Embrace the Aroma: <br>Elevate Your Coffee Experience Today!</h1>
                <p>Savoring a perfect cup of coffee isn't just about that initial burst of flavor,<br>it's about consistently delivering an unparalleled experience with every sip.</p>
                <a href="menu.php" class="btn">Explore Now &#10172;</a>
            </div>
            <div class="col-6">
                
            </div>
        </div>
    </div>

<!-------featured categories----->

<div class="categories">
    <div class="small-container">
        <div class="row">
            <div class="col-3">
                
            </div>
            <div class="col-3">
                
            </div>
            <div class="col-3">
                
            </div>
        </div>
    </div>
</div>

```

```

        </div>

        </div>

        </div>

        </div>

<!--featured menu-->



## Seasonal Flavors





#### Pumpkin Spice Latte



RM14.90





#### Cranberry Orange Cake


```

```

<i class="fa fa-star"></i>

</div>

<p>RM18.90</p>

</div>

<div class="col-4">



<h4>Maple Pecan Scones</h4>

<div class="rating">

<i class="fa fa-star"></i>

<i class="fa fa-star"></i>

<i class="fa fa-star"></i>

<i class="fa fa-star"></i>

<i class="fa fa-star-o"></i>

</div>

<p>RM9.90</p>

</div>

<div class="col-4">



<h4>Caramel Apple Latte</h4>

<div class="rating">

<i class="fa fa-star"></i>

</div>

<p>RM15.90</p>

</div>

</div>

```

```

<h2 class="title">All Time Favourites</h2>

<div class="row">

  <div class="col-4">

    <h4>Cupa Cozy</h4>

    <div class="rating">

      <i class="fa fa-star"></i>

      <i class="fa fa-star"></i>

      <i class="fa fa-star"></i>

      <i class="fa fa-star"></i>

      <i class="fa fa-star"></i>

    </div>

    <p>RM11.90</p>

  </div>

  <div class="col-4">

    <h4>Tiramisu Cake</h4>

    <div class="rating">

      <i class="fa fa-star"></i>

      <i class="fa fa-star"></i>

      <i class="fa fa-star"></i>

      <i class="fa fa-star"></i>

      <i class="fa fa-star"></i>

    </div>

    <p>RM12.90</p>

  </div>

  <div class="col-4">

    <h4>Americano</h4>
  
```

```

<div class="rating">

    <i class="fa fa-star"></i>

    <i class="fa fa-star"></i>

    <i class="fa fa-star"></i>

    <i class="fa fa-star"></i>

    <i class="fa fa-star"></i>

</div>

<p>RM7.90</p>

</div>

</div>

<!--offer-->

<div class="offer">

    <div class="small container">

        <div class="row">

            <div class="col-2">

            </div>

            <div class="col-2">

                <p>Spooktacular saving</p>

                <h1> Pumpkin Patch Pals</h1>

                <small>Add some festive flair to your Halloween
celebrations with our adorable Pumpkin Patch Pals plush sponsored by
Disney! This cuddly companion is the perfect addition to your spooky
decor, featuring soft, plush fabric and charming pumpkin design.
Receive this delightful plush as a free gift with the purchase of any
of our seasonal drinks, and bring a touch of whimsy to your
festivities. Hurry, while supplies last! </small>

            </div>

        </div>

    </div>

</div>

```

```

        </div>

        </div>

<!--customer review-->

<div class="cusreview">

    <div class="small-container">

        <div class="row">

            <div class="col-3">

                <i class="fa fa-quote-left"></i>

                    <p>As a Muslim, I truly appreciate Cup Corner's unwavering commitment to offering Halal-certified coffee. Their rich blends never fail to uplift my mornings with their smooth and aromatic flavors.</p>

                <div class = "rating">

                    <i class="fa fa-star"></i>

                    <i class="fa fa-star"></i>

                    <i class="fa fa-star"></i>

                    <i class="fa fa-star"></i>

                    <i class="fa fa-star-half"></i>

                </div>

                <h3>Nur Fazira, Malaysian Actress</h3>

            </div>

            <div class="col-3">

                <i class="fa fa-quote-left"></i>

                    <p>Cup Corner's coffee is a delightful blend of tradition and innovation. I appreciate the balance of bold flavors and subtle nuances in every cup, making it a perfect companion for any occasion.</p>

                <div class = "rating">

                    <i class="fa fa-star"></i>


```

```

<i class="fa fa-star"></i>
<i class="fa fa-star"></i>
<i class="fa fa-star"></i>
<i class="fa fa-star-o"></i>
</div>

<h3>Chong Yu Tou, UTAR professor</h3>
</div>
<div class="col-3">
<i class="fa fa-quote-left"></i>
<p>Cup Corner's coffee transports me back to the bustling streets of India, where coffee is more than just a drink – it's a ritual. Their robust brews provide the perfect pick-me-up during hectic days.</p>
<div class = "rating">
<i class="fa fa-star"></i>
</div>

<h3>Kumar A/L Aditya, CEO of Microhard </h3>
</div>
</div>
</div>
<div class="brands">
<div class="small-container">

```

```

<div class="row">

    <div class="col-5">
        
    </div>

    <div class="col-5">
        
    </div>

    <div class="col-5">
        
    </div>

    <div class="col-5">
        
    </div>

</div>

<!--footer-->

</body>

<footer>

<?php
    include('include/footer.php');
?>

</footer>

</html>

```

login.css

```

* {

```

```
padding:0;
margin: 0;
box-sizing: border-box;
font-family:'Poppins', sans-serif;

}

:root{
--primary-color: #d17d00;
}

body{
height: 100%;
margin: 0;
}

.container{
display: flex;
padding: 50px;
flex-direction:column;
align-items: center;
justify-content: center;
background: url ("images/backgroundlogin3.webp") no-repeat center center fixed;
background-size: cover;
}

.box{
background: #ffe8d6;
```

```
background: linear-gradient(135deg, #ffe8d6, #dbb180);  
padding: 25px 25px;  
border-radius: 20px;  
box-shadow: 0 4px 8px rgba(0,0,0,0.1);  
color: #5C4033;  
margin-bottom: 70px;  
}  
  
.form-box{  
width:400px;  
margin: 0px 10px;  
}  
  
.form-box h1{  
font-size:25px;  
font-weight: 600;  
padding-bottom: 10px;  
border-bottom: 1px solid white;  
margin-bottom: 10px;  
text-align: center;  
}  
  
.form-box .input{  
display:flex;  
margin-bottom: 10px;  
flex-direction: column;  
}
```

```
.form-box .input input{  
  height: 45px;  
  width: 100%;  
  font-size: 16px;  
  padding: 0 10px;  
  border-radius: 5px;  
  border: 2px solid #eee;  
  outline: none;  
  box-shadow: inset 0 2px 4px rgba(0,0,0,0.1);  
  transition: border-color 0.3s;  
}  
  
.btn-field{  
  width: 70%;  
  margin: 0 auto;  
}  
  
.btn{  
  height: 70px;  
  background: #d17d00;  
  border: 0;  
  border-radius: 8px;  
  color: white;  
  font-size: 40px;  
  cursor: pointer;  
  transition: all .3s;  
  margin-top: 25px;  
  padding: 0px 10px;  
}
```

```
  transition: background-color 0.3s, transform 0.2s;
}

.btn:hover{
  opacity: 0.7;
}

.submit{
  width: 100%;
}

.link{
  margin-bottom: 20px;
  text-align: center;
}

.link a{
  text-decoration:none;
  color: blue;
}

.link a:hover{
  color: #d17d00;
  text-decoration: none;
}

.message {
  font-size: 20px;
  text-align: center;
  background: #fff3cd;
```

```
padding: 15px;  
border: 1px solid #000000;  
border-color: #fffeeba;  
margin-bottom: 10px;  
color: #856404;  
  
}  
  
@media screen and (max-width: 768px) {  
    .form-box {  
        width: 90%;  
    }  
}  
  
@media screen and (max-width: 576px) {  
    .form-box {  
        width: 100%;  
    }  
}
```

login.php

```
<?php  
    session_start();  
    include("config.php");  
  
    $error = "";
```

```

if(isset($_POST['submit'])) {

    $username = $_POST['username'];

    $password = $_POST['password'];

    $query = "SELECT * FROM users WHERE Username='$username' AND
Password='$password';

    $result = mysqli_query($con, $query);

    if(mysqli_num_rows($result) > 0) {

        $row = mysqli_fetch_assoc($result);

        $_SESSION['valid'] = true;

        $_SESSION['userId'] = $row['id'];

        header("Location: menu.php");

        exit();

    } else {

        $error = "Incorrect username or password. Please try
again.';

    }

}

?>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Login</title>

    <link rel="stylesheet" href="login.css">

</head>

```

```
<body>

<!--navbar-->

<?php include("include/navbar.php") ?>

<div class="container">

    <div class="box">

        <div class="form-box">

            <!-- Display error message -->

            <?php if(!empty($error)) { ?>

                <div class="message">

                    <p><?php echo $error; ?></p>

                </div>

            <?php } ?>

            <h1>Login</h1>

            <form action="" method="post">

                <div class="input">

                    <label for="username">Username</label>

                    <input type="text" placeholder="Enter your
username" name="username" id="username" required>

                </div>

                <div class="input">

                    <label for="password">Password</label>

                    <input type="password" placeholder="Enter your
password" name="password" id="password" required>

                </div>

                <div class="input">

                    <div class="btn-field">

                        <input type="submit" class="btn" name="submit"
value="Login" required>

                    </div>

                </div>

            </form>

        </div>

    </div>

</div>
```

```

        <div class="link">

            Don't have an account yet ? <a
            href="register.php">Sign up now !</a>

        </div>

    </form>

</div>

</div>

</body>

<footer>

    <?php include('include/footer.php'); ?>

</footer>

</html>

```

menu.css

```

* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
    font-family: 'Open Sans', sans-serif;
}

:root {
    --primary-brown: #8f6b4c;
    --light-brown: #A0522D;
    --dark-brown: #5C4033;
    --text-color: #ffffff;
}

```

```
--button-hover-color: #C19A6B;
--button-text-color: #FFF8DC;
}

body{
background-image: url ("images/backgroundmenu.png");
}

html {
color: var(--text-color);
}

.card {
background-color: #ffffff;
border: none;
transition: transform 0.3s ease, box-shadow 0.3s ease;
border-radius: 8px;
overflow: hidden;
}

.card:hover {
transform: translateY(-5px);
box-shadow: 0 5px 15px rgba(0, 0, 0, 0.3);
}

.card-img-top{
width: 100%;
height: 200px;
object-fit: contain;
```

```
}

.side-nav{
  padding: 0;
  color:white ;
  text-align: center;
  background-color: var(--dark-brown);
}

.navbar-nav {
  background-color: var(--dark-brown);
}

.logo{
  width: auto;
  height: 30px;
}

.admin-listing-header{
  text-align: center;
  padding: 20px;
}

.admin-name{
  text-align: center;
}

.admin-img{
  width: auto;
```

```
height: 100px;  
}  
  
.admin-panel-button{  
    text-align: center;  
    color: white;  
    margin: 10px 50px;  
    padding: 10px;  
    background-color: var(--primary-brown);  
    color: var(--text-color);  
}  
  
.third-child{  
    align-items: center;  
}  
  
.col-md-4{  
    justify-content: center;  
    display: flex;  
    margin-top: 20px;  
}  
  
.card{  
    box-shadow: 5px 10px 8px 10px rgb(0,0,0,0.1);  
}
```

```
.products-preview{  
  position: fixed;  
  top:0; left:0;  
  min-height: 100vh;  
  width: 100%;  
  background: rgba(0,0,0,.8);  
  display: none;  
  align-items: center;  
  justify-content: center;  
  z-index: 1000;  
  
}  
  
.products-preview .preview{  
  display: none;  
  padding:1rem;  
  text-align: center;  
  background: var(--text-color);  
  position: relative;  
  margin:1.5rem;  
  width: 40rem;  
  border: 2px solid var(--primary-brown);  
  box-shadow: 0 4px 8px var(--dark-brown);  
  
}  
  
.products-preview .preview.active{  
  display: inline-block;  
}
```

```
.products-preview .preview img{  
  height: 15rem;  
}  
  
.products-preview .preview .fa-times{  
  position: absolute;  
  top:1rem; right:1.5rem;  
  cursor: pointer;  
  color: var(--dark-brown);  
  font-size: 2rem;  
}  
  
.products-preview .preview .fa-times:hover{  
  transform: rotate(90deg);  
  color: var(--light-brown);  
}  
  
.products-preview .preview h3{  
  color:#444;  
  padding:.5rem 0;  
}  
  
.products-preview .preview p{  
  line-height: 1.5;  
  padding:0.5rem 0;  
  color:#777;  
  font-size: 1em;  
}
```

```
.products-preview .preview .price{  
  padding: 0.5rem 0;  
  font-size: 1rem;  
  color: #27ae60;  
}  
  
.products-preview .preview .buttons{  
  display: flex;  
  gap: 1.5rem;  
  flex-wrap: wrap;  
  margin-top: 1rem;  
}  
  
.products-preview .preview .buttons a{  
  flex: 1 1 16rem;  
  padding: 1rem;  
  font-size: 1.5rem;  
  color: #444;  
  border: 1rem solid #444;  
  border-color: var(--primary-brown);  
}  
  
.products-preview .preview .buttons a.cart{  
  background: #444;  
  color: #fff;  
  text-decoration: none;  
  background-color: var(--dark-brown);  
}
```

```
.products-preview .preview .buttons a.cart:hover{  
  background-color: var(--light-brown);  
}  
  
.btn {  
  background-color: var(--primary-brown);  
  color: var(--button-text-color);  
  border: 1px solid var(--dark-brown);  
  transition: all 0.3s ease;  
  text-shadow: 1px 1px 2px black;  
  border-radius: 4px; /* Rounded buttons */  
}  
  
.btn:hover {  
  background-color: var(--button-hover-color);  
  color: var(--text-color);  
  border-color: var(--button-hover-color);  
}  
  
.btn-outline-primary {  
  border-color: var(--primary-brown);  
  color: var(--primary-brown);  
}  
  
.btn-outline-primary:hover {  
  background-color: var(--primary-brown);  
  color: var(--text-color);  
}
```

menu.php

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Cup Coffee Menu</title>

    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pN1yT2bRjXh0JMhjY6hW+ALEwIH" crossorigin="anonymous">

    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-YvpcrYf0tY3lHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcIdslK1eN7N6jIeHz" crossorigin="anonymous"></script>

    <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.1/css/all.min.css">

    <link rel="stylesheet" href="menu.css">

    <script src='menu.js'></script>

</head>

<body>

<?php include("include/navbar.php") ?>

<?php

include('../include/connect.php');

include('../functions/common_function.php');

?>
```

```

<!-----contents----->

<div class="row products-container">

    <div class="col-md-10">

        <!-----products----->

        <div class="row">

            <!-----fetching products----->

            <?php

                getProducts();

                getUniqueCategory();

            ?>

        </div>

    </div>

    <div class="side-nav col-md-2">

        <!-----side nav----->

        <ul class="navbar-nav me-auto">

            <li class="nav-item">

                <a href="#" class="nav-link"><h4>Categories</h4></a>

            </li>

            <?php

                getCategories();

            ?>

        </ul>

    </div>

</div>

<div class='products-preview'>

    <?php

        getProductPreview();

    ?>

```

```
</div>

<footer>

<?php
include('include/footer.php');

?>

</footer>

</body>

</html>
```

order_details.css

```
body {
    font-family: Arial, sans-serif;
    background-color: #f4f4f4;
    margin: 0;
    padding: 0;
}

.container {
    max-width: 800px;
    margin: 20px auto;
    background-color: #fff;
    border-radius: 5px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    padding: 20px;
}

h2 {
```

```
margin-top: 0;  
color: #333;  
}  
  
  
table {  
width: 100%;  
border-collapse: collapse;  
margin-bottom: 20px;  
}  
  
  
th, td {  
padding: 10px;  
text-align: left;  
border-bottom: 1px solid #ddd;  
}  
  
  
th {  
background-color: #f2f2f2;  
}  
  
  
.orders {  
margin-bottom: 20px;  
}  
  
  
p {  
margin: 0;  
}
```

order_details.php

```
<link rel="stylesheet" href="order_details.css">

<?php

    session_start();

    include("config.php");

    // Check if the order ID is provided in the URL

    if (!isset($_GET['order_id']) || empty($_GET['order_id'])) {

        // Redirect the user back to the account page or display an
        error message

        header("Location: account.php");

        exit();
    }

    // Retrieve the order ID from the URL

    $orderId = $_GET['order_id'];

    // Fetch order details from the database based on the order ID

    $orderQuery = "SELECT * FROM order_details WHERE order_id =
    '$orderId';

    $orderResult = mysqli_query($con, $orderQuery);

    // Check if the query was successful

    if ($orderResult) {

        // Fetch and store order details data

        $orderDetails = mysqli_fetch_all($orderResult, MYSQLI_ASSOC);

    } else {

        // Handle database error

        $error = "Failed to fetch order details";
    }
}
```

```

// Fetch shipping address and payment method from the orders table

$orderInfoQuery = "SELECT shipping_address, payment_method,
date_created FROM orders WHERE order_id = '$orderId';

$orderInfoResult = mysqli_query($con, $orderInfoQuery);

// Check if the query was successful

if ($orderInfoResult) {

    // Fetch shipping address and payment method

    $orderInfo = mysqli_fetch_assoc($orderInfoResult);

} else {

    // Handle database error

    $error = "Failed to fetch order information";

}

// Calculate total price

$totalPrice = 0;

foreach ($orderDetails as $orderDetail) {

    $totalPrice += $orderDetail['quantity'] *
$orderDetail['price_per_unit'];

}

?>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Order Details</title>

```

```

<link rel="stylesheet" href="styles.css">

</head>

<body>

<!-- Include navbar -->

<?php include("include/navbar.php") ?>

<div class="container">

<div class="box">

<div class="form-box">

<h2>Order Details</h2>

<div class="orders">

<?php if (isset($orderDetails) &&
!empty($orderDetails)) { ?>

<table>

<thead>

<tr>

<th>Product Name</th>

<th>Quantity</th>

<th>Price Per Unit (MYR)</th>

<th>Total Price (MYR)</th>

</tr>

</thead>

<tbody>

<?php foreach ($orderDetails as
$orderDetail) { ?>

<tr>

<td><?php echo
$orderDetail['product_name'] ; ?></td>

<td><?php echo
$orderDetail['quantity'] ; ?></td>

```

```

                <td><?php echo
number_format($orderDetail['price_per_unit'], 2); ?></td>

                <td><?php echo
number_format($orderDetail['quantity'] *
$orderDetail['price_per_unit'], 2); ?></td>

            </tr>

        <?php } ?>

        <!-- Display total price -->

        <tr>

            <td colspan="3" style="text-align:
right;"><strong>Total Price (MYR) :</strong></td>

            <td><strong><?php echo
number_format($totalPrice, 2); ?></strong></td>

        </tr>

    </tbody>

</table>

<?php } else { ?>

    <p>No order details found.</p>

    <?php } ?>

</div>

</div>

</div>

<!-- Display shipping address and payment method -->

<div class="container">

    <div class="box">

        <div class="form-box">

            <h2>Order Date</h2>

            <p><?php echo $orderInfo['date_created']; ?></p>

            <br>

```

```

        <h2>Shipping Address</h2>

        <p><?php echo $orderInfo['shipping_address']; ?></p>

        <br>

        <h2>Payment Method</h2>

        <p><?php echo $orderInfo['payment_method']; ?></p>

    </div>

</div>

<!-- Back to Home button -->

<div class="container">

    <div class="box">

        <div class="form-box">

            <a href="account.php" class="btn">Back to Account</a>

        </div>

    </div>

</div>

<footer>

    <?php include('include/footer.php'); ?>

</footer>

</body>

</html>

```

our.css

```

*{
    margin: 0;
    padding: 0;
}

```

```
  font-family: 'Poppins' , sans-serif;
}

body {
  background-image: url("images/bgteam2.png");
  background-size: cover;
  background-attachment: fixed;
  font-family: 'Poppins', sans-serif;
}

section {
  padding: 50px;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
}

.container{
  position: relative;
  z-index: 1;
  display: flex;
  justify-content: center;
  align-items: center;
  flex-wrap: wrap;
  margin: 40px 0;
}

.container .card{
```

```
position: relative;
width: 300px;
height: 400px;
background: rgba(255, 255, 255, 0.05);
margin: 20px;
box-shadow: 0 15px 35px rgba(0,0,0,0.2);
border-radius: 15px;
display: flex;
justify-content: center;
align-items: center;
backdrop-filter: blur(10px);
}

.container .card .content{
position: relative;
display: flex;
justify-content: center;
align-items: center;
flex-direction: column;
opacity: 0.5;
transition: 0.5s;
}

.container .card:hover .content{
opacity: 1;
transform: translateY(-20px);
}

.container .card .content .imgBx{
```

```
position: relative;  
width: 150px;  
height: 150px;  
border-radius: 50%;  
overflow: hidden;  
border: 10px solid rgba(0,0,0,0.25);  
}  
  
.container .card .content .imgBx img{  
position: absolute;  
top: 0;  
left: 0;  
width: 100%;  
height: 100%;  
object-fit: cover;  
}  
  
.container .card .content .contentBx h3{  
color: #fff;  
text-transform: uppercase;  
letter-spacing: 2px;  
font-weight: 700;  
font-size: 18px;  
text-align: center;  
margin: 20px 0 10px;  
line-height: 1.1em;  
}  
  
.container .card .content .contentBx h3 span{
```

```

    font-size: 12px;
    font-weight: 300;
    text-transform: initial;
}

.container .card .sci{
    position: absolute;
    bottom: 50px;
    display: flex;
}

```

ourteam.php

```

<!DOCTYPE html>

<html lang="en">

<head>

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

    <title>Our Team</title>

    <link rel="stylesheet" href="include/navbar.css">

    <link rel="stylesheet" type="text/css" href="our.css">

    <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500
;600;700&display=swap" rel="stylesheet">

    <link rel="stylesheet"
href="path/to/font-awesome/css/font-awesome.min.css">

    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-aw
esome.min.css">

</head>

<body>

```

```

<!-----navbar----->

<?php include("include/navbar.php") ?>

<!-----content----->

<section>

    <div class="container">

        <div class="card">

            <div class="content">

                <div class="imgBx"></div>

                <div class="contentBx">

                    <h3>CEO of Cup
Corner<br><span>Founder</span></h3>

                </div>

            </div>

        </div>

        <div class="card">

            <div class="content">

                <div class="imgBx"></div>

                <div class="contentBx">

                    <h3>Head of Cup
Corner<br><span>Manager</span></h3>

                </div>

            </div>

        </div>

        <div class="card">

            <div class="content">

                <div class="imgBx"></div>

                <div class="contentBx">

```

```

<h3>Head of barista<br><span>Coffee
Innovator</span></h3>
</div>
</div>
</div>
</div>
</section>
</body>
<footer>
<?php
include('include/footer.php');

?>
</footer>
</html>

```

reAccount.css

```

body {
    font-family: 'Poppins', sans-serif;
    background: url("images/backgroundaccount2.webp") no-repeat center
    center fixed;
    background-size: cover;
    margin: 0;
    padding: 0;
}

.container {
    max-width: 800px;
    margin: 40px auto;
    background: rgba(250, 245, 235, 0.95);

```

```
border-radius: 8px;  
box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);  
padding: 30px;  
overflow: hidden;  
}  
  
.box {  
margin-bottom: 30px;  
}  
  
.form-box {  
background: rgba(255, 255, 255, 0.9);  
padding: 25px;  
border-radius: 8px;  
box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);  
margin-bottom: 20px;  
}  
  
h1, h2, th, .btn {  
color: #5C4033;  
}  
  
h1 {  
font-size: 2.5rem;  
margin-bottom: 0.8em;  
}  
  
h2 {  
font-size: 2rem;
```

```
margin-top: 20px;  
margin-bottom: 0.8em;  
}  
  
p, th, td {  
margin: 0.5em 0;  
}  
  
table {  
width: 100%;  
border-collapse: collapse;  
margin: 20px 0;  
}  
  
th, td {  
padding: 12px 15px;  
text-align: left;  
border-bottom: 1px solid #ddd;  
}  
  
th {  
background-color: #eae0d6;  
}  
  
.btn {  
background-color: #d17d00;  
padding: 12px 18px;  
border-radius: 4px;  
font-weight: bold;
```

```
text-transform: uppercase;
transition: background 0.3s ease;
}

.btn:hover {
background-color: #b15d00;
}

.logout-button {
background-color: #C19A6B;
color: white;
padding: 10px 20px;
border: none;
border-radius: 4px;
cursor: pointer;
font-weight: bold;
text-transform: uppercase;
transition: all 0.3s ease;
text-decoration: none;
display: inline-block;
margin-top: 20px;
}

.logout-button:hover {
background-color: #a93226;
box-shadow: 0 2px 4px rgba(0, 0, 0, 0.2);
}

@media screen and (max-width: 768px) {
```

```
.container {  
    padding: 20px;  
}  
  
h1, h2 {  
    font-size: 1.8rem;  
}  
  
table, th, td {  
    display: block;  
}  
  
th, td {  
    text-align: right;  
}  
  
th::before, td::before {  
    content: attr(data-label);  
    float: left;  
    font-weight: bold;  
    text-transform: uppercase;  
}  
}
```

register.php

```
<?php  
    session_start();  
    include ("config.php");  
?>
```

```

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Sign Up</title>

    <link rel="stylesheet" href="registerme.css">

</head>

<body>

    <header>

        <?php include("include/navbar.php"); ?>

    </header>

    <div class="container">

        <div class="box">

            <div class="form-box">

                <?php

                    if (isset($_POST["submit"])) {

                        $firstname = $_POST['first-name'];

                        $lastname = $_POST['last-name'];

                        $username = $_POST['username'];

                        $email = $_POST['email'];

                        $password = $_POST['password'];



                        // Check if the email is already used

                        $verify_email = mysqli_query($con, "SELECT Email
FROM users WHERE Email='$email'");



                        if (mysqli_num_rows($verify_email) != 0) {

```

```

        echo "<div class='message'><p>Email is already
used! Please try another one.</p></div><br>";

        echo "<a
href='javascript:self.history.back()'><button class='btn'>Go
Back</button></a>";

    } else {

        // Check if the username is already used

        $verify_username = mysqli_query($con, "SELECT
Username FROM users WHERE Username='{$username}'");

        if (mysqli_num_rows($verify_username) != 0) {

            echo "<div class='message'><p>Username is
already used! Please try another one.</p></div><br>";

            echo "<a
href='javascript:self.history.back()'><button class='btn'>Go
Back</button></a>";

        } else {

            // Insert user data into the database

            $query = "INSERT INTO users(Firstname,
Lastname, Username, Email, Password) VALUES ('{$firstname}', '{$lastname}', '{$username}', '{$email}', '{$password}')";

            if (mysqli_query($con, $query)) {

                echo "<div
class='message2'><p>Registration successful!</p></div><br>";

                echo "<a href='login.php'><button
class='btn'>Login Now</button></a>";

            } else {

                echo "<div class='message'><p>Error
Occurred: " . mysqli_error($con) . "</p></div>";

            }

        }

    } else {

    ?>

```

```
<h1>Sign Up</h1>

<form action="" method="post">

    <div class="input">

        <label for="first-name">First Name</label>

        <input type="text" placeholder="Enter your
first name" name="first-name" id="first-name" required>

    </div>

    <div class="input">

        <label for="last-name">Last Name</label>

        <input type="text" placeholder="Enter your last
name" name="last-name" id="last-name" required>

    </div>

    <div class="input">

        <label for="username">Username</label>

        <input type="text" placeholder="Enter your
username" name="username" id="username" required>

    </div>

    <div class="input">

        <label for="email">Email</label>

        <input type="email" placeholder="Enter your
email" name="email" id="email" required>

    </div>

    <div class="input">

        <label for="password">Password</label>

        <input type="password" placeholder="Enter your
password" name="password" id="password" required>

    </div>
```

```

<div class="input">

    <div class="btn-field">

        <input type="submit" class="btn"
name="submit" value="Register" required>

    </div>

</div>

<div class="link">

    Already have an account? <a
href="login.php">Login now!</a>

</div>

</form>

<?php } ?>

</div>

</div>

<div>

    <?php include('include/footer.php'); ?>

</div>

</body>

</html>

```

registerme.css

```

* {
    padding: 0;
    margin: 0;
    box-sizing: border-box;
}

```

```
  font-family: 'Poppins', sans-serif;
}

body{
  height: 100vh;
  margin: 0;
  display: flex;
  flex-direction: column;
  background: url("images/backgroundlogin3.webp") no-repeat center
  center fixed;
  background-size: cover;
}

header, main, footer {
  width: 100%;
}

.container{
  flex: 1;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
}

.box{
  background: linear-gradient(135deg, #ffe8d6, #dbb180);
  padding: 25px 25px;
  border-radius: 20px;
}
```

```
  box-shadow: 0 4px 8px rgba(0,0,0,0.1);  
  color: #5C4033;  
  margin: 70px 0;  
}  
  
.form-box{  
  width:400px;  
}  
  
.form-box h1{  
  font-size:25px;  
  font-weight: 600;  
  margin-bottom: 10px;  
  text-align: center;  
  border-bottom: 1px solid white;  
}  
  
}  
  
.form-box .input{  
  display:flex;  
  margin-bottom: 10px;  
  flex-direction: column;  
}  
  
.form-box .input input{  
  height: 45px;  
  width: 100%;  
  font-size: 16px;  
  padding: 0 10px;  
}
```

```
border-radius: 5px;  
border: 2px solid #eee;  
outline: none;  
box-shadow: inset 0 2px 4px rgba(0,0,0,0.1);  
}  
  
.btn{  
height: 70px;  
background: #d17d00;  
border: none;  
border-radius: 8px;  
color: white;  
font-size: 40px;  
cursor: pointer;  
transition: background-color 0.3s, opacity 0.3s, transform 0.2s;  
margin-top: 25px;  
width: calc(100% - 20px);  
}  
  
.btn:hover{  
opacity: 0.7;  
}  
  
.submit{  
width: 100%;  
}  
  
.link{
```

```
margin-bottom: 20px;  
text-align: center;  
}  
  
.link a{  
text-decoration:none;  
color: blue;  
}  
  
.link a:hover{  
color: #d17d00;  
text-decoration: none;  
}  
  
.message, .message2 {  
font-size: 20px;  
text-align: center;  
padding: 15px;  
border-radius: 5px;  
margin-bottom: 10px;  
border: 1px solid #000;  
}  
  
.message {  
background: #fff3cd;  
color: #856404;  
border-color: #ffeeba;  
}  
  
.message2 {
```

```

background: #161515;

color: rgb(0, 248, 95);

}

@media screen and (max-width: 768px) {

.form-box {

width: 90%;

}

}

@media screen and (max-width: 576px) {

.form-box {

width: 100%;

}

}

```

remove_from_cart.php

```

<?php

session_start();


if (!isset($_SESSION['cart'])) {

    echo json_encode(['status' => 'error', 'message' => 'Cart not
initialized']);

    exit;

}

$inputJSON = file_get_contents('php://input');

$input = json_decode($inputJSON, TRUE); // Convert it into an
associative array

```

```

if(isset($input['productId']) && is_numeric($input['productId'])) {
    $product_id = (int)$input['productId'];

    if (isset($_SESSION['cart'][$product_id])) {
        if ($_SESSION['cart'][$product_id] > 1) {
            $_SESSION['cart'][$product_id] -= 1; // Decrement quantity
        } else {
            unset($_SESSION['cart'][$product_id]); // Remove item if
            quantity is 1
        }
        echo json_encode(['status' => 'success', 'message' => 'Item
updated successfully']);
    } else {
        echo json_encode(['status' => 'error', 'message' => 'Item not
found in cart']);
    }
} else {
    echo json_encode(['status' => 'error', 'message' => 'Invalid
product ID']);
}
?>

```

style.css

```

* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

```

```
body {
  font-family: "Poppins", sans-serif;
}

.container {
  max-width: 1300px;
  margin: auto;
  padding-left: 25px;
  padding-right: 25px;
  background-image: url("image.home/background.png");
}

.small-container {
  max-width: 1300px;
  margin: auto;
  padding-left: 25px;
  padding-right: 25px;
}

.row {
  display: flex;
  align-items: center;
  flex-wrap: wrap;
  justify-content: space-around;
}

.col-6{
  flex-basis: 50%;
  min-width: 30px;
}
```

```
}

.col-6 h1 {
  font-size: 50px;
  line-height: 60px;
  margin: 25px 0;
  color: black;
}

.col-6 p {
  color: #5C4033;
  font-weight: 400;
}

.col-6 img {
  margin-left: 180px;
}

.col-2 {
  flex-basis: 50%;
  min-width: 300px;
}

.col-2 p {
  color: #A0522D;
  font-weight: 600;
}

.col-2 img {
```

```
margin-left: 80px;  
}  
  
.col-2 h1 {  
    font-size: 50px;  
    line-height: 60px;  
    margin: 25px 0;  
    color: #5C4033;  
}  
  
.col-2 small{  
    color: #8f6b4c;  
    font-weight: 600;  
}  
  
.btn {  
    display: inline-block;  
    background: #7B241C;  
    color: #fff;  
    padding: 8px 30px;  
    margin: 30px 0;  
    border-radius: 30px;  
    border: none;  
    cursor: pointer;  
    transition: background 0.5s;  
}  
  
.btn:hover {  
    background: #5a1a14;
```

```
}

.content {
  background: radial-gradient(#f5f5dc, #ecd9c6);
  padding: 40px 0;
  margin-top: -50px;
}

.content .row {
  margin-top: 70px;
}

.categories {
  margin: 70px 0;
}

.col-3 {
  flex-basis: 30%;
  min-width: 250px;
  margin-bottom: 30px;
  margin-right: 20px;
}

.col-3 img {
  width: 100%;
}

.col-4 {
  flex-basis: 25%;
```

```
padding: 10px;  
min-width: 200px;  
margin-bottom: 50px;  
transition: transform 0.5s;  
}  
  
.col-4 img {  
width: 100%;  
height: auto;  
}  
  
.title {  
text-align: center;  
margin: 0 auto 80px;  
position: relative;  
line-height: 60px;  
color: #555;  
}  
  
.title::after {  
content: '';  
background: #7B241C;  
width: 80px;  
height: 5px;  
border-radius: 5px;  
position: absolute;  
bottom: 0;  
left: 50%;  
transform: translateX(-50%);
```

```
}

.h4 {
  color: #5C4033;
  font-weight: bold;
}

.col-4 p {
  font-size: 14px;
}

.rating .fa {
  color: #7B241C;
}

.col-4p:hover {
  transform: translateY(-5px);
}

.offer {
  background: radial-gradient(#f5f5dc, #ecd9c6);
  margin-top: 80px;
  padding: 60px 0;
}

.col-2 .limited-img {
  padding: 50px;
}
```

```
small {
  color: #555;
}

.cusreview {
  padding-top: 100px;
}

.cusreview .col-3 {
  text-align: center;
  padding: 40px 20px;
  box-shadow: 0 0 20px 0px rgba(0, 0, 0, 0.1);
  cursor: pointer;
  transition: transform 0.5s;
}

.cusreview .col-3 img {
  width: 50px;
  margin-top: 20px;
  border-radius: 50%;
}

.cusreview .col-3:hover {
  transform: translateY(-10px);
}

.fa.fa-quote-left {
  font-size: 34px;
  color: darkred;
}
```

```
}

.col-3 p {
    font-size: 12px;
    margin: 12px 0;
}

.cusreview .col-3 h3 {
    font-weight: 600;
    color: darkred;
    font-size: 16px;
}

.brands {
    margin: 100px auto;
}

.col-5 {
    width: 160px;
}

.col-5 img {
    width: 100%;
    cursor: pointer;
    filter: grayscale(100%);
}

.col-5 img:hover {
    filter: grayscale(0);
}
```

```
}
```

submit_contact.php

```
<?php

include('include/connect.php'); // Ensure this path is correct


if ($_SERVER["REQUEST_METHOD"] == "POST") {

    $name = $_POST['name'];
    $email = $_POST['email'];
    $message = $_POST['message'];

    $stmt = $con->prepare("INSERT INTO contact_responses (name, email, message) VALUES (?, ?, ?)");
    $stmt->bind_param("sss", $name, $email, $message);

    if ($stmt->execute()) {
        echo "<script>alert('Your message has been sent, please wait till our team to contact you shortly.');" . "window.location.href='contact.php';</script>";
    } else {
        echo "<script>alert('Error sending message.');" . "window.location.href='contact.php';</script>";
    }

    $stmt->close();
    $con->close();
}

?>
```

JavaScript

The menu.js JavaScript file is pivotal for adding interactive features to the product menu, such as dynamic filters or live updates, which enhance the user experience for both customers and administrators. On the database side, MySQL plays a crucial role in storing and retrieving data efficiently, supporting the administrative functionality through structured and secure data handling, reflecting the changes made by the admins in real-time across the website.

menu.js

```
document.addEventListener("DOMContentLoaded", function() {  
  // Add event listener to all "View More" buttons  
  
  const viewMoreButtons =  
document.querySelectorAll('.btn-secondary');  
  
  viewMoreButtons.forEach(button => {  
  
    button.addEventListener('click', function(event) {  
  
      event.preventDefault(); // Prevent the default action of  
the link  
  
      // Get the product ID from the data-id attribute  
  
      const productId =  
button.closest('.product').getAttribute('data-id');  
  
      // Show the corresponding product preview  
  
      showProductPreview(productId);  
  
    }) ;  
  }) ;  
  
  // Function to show product preview  
  
  function showProductPreview(productId) {  
  
    // Hide any currently active previews
```

```

    const activePreviews =
document.querySelectorAll('.products-preview .preview.active');

    activePreviews.forEach(preview => {
        preview.classList.remove('active');
    });

    // Show the preview corresponding to the product ID

    const targetPreview = document.querySelector(`.products-preview
.preview[data-target='${productId}']`);

    if (targetPreview) {
        targetPreview.classList.add('active');
    }

    // Show the products preview container

    const productsPreviewContainer =
document.querySelector('.products-preview');

    productsPreviewContainer.style.display = 'flex';

}

    // Close preview when clicking on the close button

    const closeButton = document.querySelectorAll('.products-preview
.fa-times');

    closeButton.forEach(button => {
        button.addEventListener('click', function() {
            const productsPreviewContainer =
document.querySelector('.products-preview');

            productsPreviewContainer.style.display = 'none';
        });
    });

    // Add event listener to all "Add to Cart" buttons

```

```

const addToCartButtons = document.querySelectorAll('.btn-primary');

addToCartButtons.forEach(button => {
    button.addEventListener('click', function(event) {
        event.preventDefault(); // Prevent the default form
        submission

        // Get the product ID from the closest product container
        const productId =
button.closest('.product').getAttribute('data-id');

        fetch('add_to_cart.php', {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json',
            },
            body: JSON.stringify({ productId: productId, increment:
true })
        })
        .then(response => response.json())
        .then(data => alert(data.message))
        .catch(error => console.error('Error adding product to
cart:', error));
    });
});

```

MySQL

This is to create a database called cup_corner and generate 5 tables to save related data into its own table.

cup_corner.sql

```
-- phpMyAdmin SQL Dump

-- version 5.2.1

-- https://www.phpmyadmin.net/

-- 

-- Host: 127.0.0.1

-- Generation Time: Apr 19, 2024 at 08:58 AM

-- Server version: 10.4.32-MariaDB

-- PHP Version: 8.2.12


SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";

START TRANSACTION;

SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;

/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;

/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;

/*!40101 SET NAMES utf8mb4 */;

-- 

-- Database: `cup_corner`


CREATE DATABASE IF NOT EXISTS `cup_corner` DEFAULT CHARACTER SET utf32
COLLATE utf32_general_ci;

USE `cup_corner`;

-- -----
```

```

-- 
-- Table structure for table `categories` 

-- 

CREATE TABLE `categories` (
  `category_id` int(11) NOT NULL,
  `category_title` varchar(100) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf32 COLLATE=utf32_general_ci;

-- 

-- Dumping data for table `categories` 

-- 

INSERT INTO `categories` (`category_id`, `category_title`) VALUES
(1, 'Coffee'),
(3, 'Pastries'),
(4, 'Hot meals'),
(5, 'Non-Coffee'),
(7, 'Seasonal');

----- 

-- 
-- Table structure for table `contact_responses` 

-- 

CREATE TABLE `contact_responses` (
  `id` int(11) NOT NULL,
  `name` varchar(255) NOT NULL,

```



```

CREATE TABLE `orders` (
  `order_id` varchar(255) NOT NULL,
  `user_id` int(11) DEFAULT NULL,
  `total_payment` decimal(10,2) DEFAULT NULL,
  `shipping_address` varchar(255) DEFAULT NULL,
  `payment_method` varchar(50) DEFAULT NULL,
  `date_created` datetime DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf32 COLLATE=utf32_general_ci;

-- 
-- Dumping data for table `orders`
-- 

INSERT INTO `orders` (`order_id`, `user_id`, `total_payment`, `shipping_address`, `payment_method`, `date_created`) VALUES
('4_6620d4eec6cf3', 4, 232.00, 'bubu home', 'Bank Transfer', '2024-04-18 10:08:14'),
('4_6620d50a58bb8', 4, 92.00, 'utar', 'Bank Transfer', '2024-04-18 10:08:42'),
('6_6620e148738d8', 6, 236.00, '1, Jalan Besar, KL', 'Credit Card', '2024-04-18 11:00:56'),
('6_6620e18251b29', 6, 278.00, 'Shell Mahkota Cheras, Selangor', 'PayPal', '2024-04-18 11:01:54'),
('7_6620e1dda847a', 7, 138.00, 'Halloween Ground, Putrajaya', 'Credit Card', '2024-04-18 11:03:25'),
('8_66212a2353c44', 8, 50.00, 'Oscar home', 'PayPal', '2024-04-18 16:11:47');

-- 
-- 
-- 
```

```
-- Table structure for table `order_details`  
--  
  
CREATE TABLE `order_details` (  
    `detail_id` int(11) NOT NULL,  
    `order_id` varchar(255) DEFAULT NULL,  
    `product_name` varchar(255) DEFAULT NULL,  
    `price_per_unit` decimal(10,2) DEFAULT NULL,  
    `quantity` int(11) DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf32 COLLATE=utf32_general_ci;  
  
--  
-- Dumping data for table `order_details`  
--  
  
INSERT INTO `order_details`(`detail_id`, `order_id`, `product_name`,  
    `price_per_unit`, `quantity`) VALUES  
(36, '4_6620d4eec6cf3', 'Asian Dolce Latte', 18.00, 1),  
(37, '4_6620d4eec6cf3', 'Caffe Latte', 14.00, 1),  
(38, '4_6620d4eec6cf3', 'Cappuccino', 14.00, 1),  
(39, '4_6620d4eec6cf3', 'Cold Brew', 18.00, 1),  
(40, '4_6620d4eec6cf3', 'Coffee by the Press', 18.00, 1),  
(41, '4_6620d4eec6cf3', 'Cocoa Cappuccino', 16.00, 1),  
(42, '4_6620d4eec6cf3', 'Matcha Latte', 18.00, 2),  
(43, '4_6620d4eec6cf3', 'Spaghetti Carbonara', 18.00, 2),  
(44, '4_6620d4eec6cf3', 'Spaghetti Bolognese', 18.00, 2),  
(45, '4_6620d4eec6cf3', 'Cinnamon Roll', 16.00, 1),  
(46, '4_6620d4eec6cf3', 'Wholemeal Tuna Sandwich', 10.00, 1),  
(47, '4_6620d50a58bb8', 'Asian Dolce Latte', 18.00, 1),
```

```

(48, '4_6620d50a58bb8', 'Caffe Latte', 14.00, 1),
(49, '4_6620d50a58bb8', 'Cappuccino', 14.00, 1),
(50, '4_6620d50a58bb8', 'Cinnamon Roll', 16.00, 1),
(51, '4_6620d50a58bb8', 'Croissant', 10.00, 1),
(52, '4_6620d50a58bb8', 'Pumpkin Spice Latte', 20.00, 1),
(53, '6_6620e148738d8', 'Americano', 10.00, 1),
(54, '6_6620e148738d8', 'Asian Dolce Latte', 18.00, 2),
(55, '6_6620e148738d8', 'Coffee by the Press', 18.00, 1),
(56, '6_6620e148738d8', 'Matcha Cold Foam Iced Americano', 20.00, 1),
(57, '6_6620e148738d8', 'Cocoa Cappuccino', 16.00, 1),
(58, '6_6620e148738d8', 'Wholemeal Tuna Sandwich', 10.00, 1),
(59, '6_6620e148738d8', 'Apple Turnover', 16.00, 2),
(60, '6_6620e148738d8', 'Spaghetti Bolognese', 18.00, 2),
(61, '6_6620e148738d8', 'Spaghetti Carbonara', 18.00, 1),
(62, '6_6620e148738d8', 'Classic Mac & Cheese', 20.00, 1),
(63, '6_6620e148738d8', 'Pumpkin Spice Latte', 20.00, 1),
(64, '6_6620e18251b29', 'Cinnamon Roll', 16.00, 1),
(65, '6_6620e18251b29', 'Spaghetti Bolognese', 18.00, 4),
(66, '6_6620e18251b29', 'Signature Chicken Lasagna', 20.00, 4),
(67, '6_6620e18251b29', 'Signature Hot Chocolate', 18.00, 2),
(68, '6_6620e18251b29', 'Brewed Tea', 10.00, 2),
(69, '6_6620e18251b29', 'Iced Shaken Lemon Tea', 14.00, 1),
(70, '6_6620e18251b29', 'Strawberry Açai With Lemonade Starbucks Refreshers™', 20.00, 1),
(71, '6_6620e18251b29', 'Pumpkin Spice Latte', 20.00, 1),
(72, '7_6620e1dda847a', 'Pumpkin Spice Latte', 20.00, 4),
(73, '7_6620e1dda847a', 'Cinnamon Roll', 16.00, 1),
(74, '7_6620e1dda847a', 'Apple Turnover', 16.00, 1),
(75, '7_6620e1dda847a', 'Croissant', 10.00, 1),

```

```
(76, '7_6620e1dda847a', 'Salted Caramel Nut Danish', 16.00, 1),
(81, '8_66212a2353c44', 'Asian Dolce Latte', 18.00, 1),
(82, '8_66212a2353c44', 'Caffe Latte', 14.00, 1),
(83, '8_66212a2353c44', 'Spaghetti Carbonara', 18.00, 1);

-- 
-- Table structure for table `products` 

CREATE TABLE `products` (
  `product_id` int(11) NOT NULL,
  `product_title` varchar(100) NOT NULL,
  `product_description` text NOT NULL,
  `category_id` int(11) NOT NULL,
  `product_image` varchar(255) NOT NULL,
  `product_price` decimal(10,2) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf32 COLLATE=utf32_general_ci;

-- 
-- Dumping data for table `products` 

INSERT INTO `products` (`product_id`, `product_title`,
`product_description`, `category_id`, `product_image`, `product_price`)
VALUES
(1, 'Americano', 'Rich, full-bodied espresso with hot water in true
European style. While the americano is similar in strength and taste to
American-style brewed coffee, there are subtle differences achieved by
pulling a fresh shot of espresso for the beverage base. The best way to
```

```
discover these nuances, of course, is to try a cup yourself.', 1,  
'americano.jpg', 10.00),  
  
(2, 'Asian Dolce Latte', 'Introducing Asian Dolce Latte, the new smooth  
and velvety textured latte with an Asian twist. Inspired by how coffee  
is drank in many parts of Asia, strong yet smooth and flavored, Asian  
Dolce Latte has a delicious local flavor in every sip.', 1,  
'asianDolceLatte.jpg', 18.00),  
  
(3, 'Caffe Latte', 'Rich, full-bodied espresso in steamed milk, lightly  
topped with foam. This is the original coffeehouse classic. And like  
most classics, part of its appeal comes from its simplicity. A caffe  
latte is simply a shot or two of bold, tasty espresso with fresh, sweet  
steamed milk over it. Some prefer to add syrup or extra espresso to the  
recipe. Some maintain that it is entirely perfect as is!', 1,  
'caffelatte.jpg', 14.00),  
  
(4, 'Caffe Mocha', 'Espresso with bittersweet mocha sauce and steamed  
milk. Topped with sweetened whipped cream. There's no question  
chocolate and coffee are flavors that meant for each other. Both are  
rich and full of depth. Where one is creamy, the other is roasty. They  
complement each other perfectly. And when they come together under a  
fluffy cloud of sweetened whipped cream, you'll wish their union would  
last forever.', 1, 'caffemocha.jpg', 16.00),  
  
(5, 'Cappuccino', 'Espresso with steamed milk, topped with a deep layer  
of foam. With less milk than a latte, cappuccino offers a stronger  
espresso flavor and a luxurious texture. To make it properly requires  
much skill and attentiveness. Arguably the most important part is  
frothing the foam to velvety perfection as the milk steams – something  
our baristas take great care to achieve. The milky moustache that  
clings to your upper lip is proof we've made yours right. And may we  
say, you wear it well.', 1, 'cappucino.jpg', 14.00),  
  
(6, 'Caramel Macchiato', 'Freshly steamed milk with vanilla-flavored  
syrup is marked with espresso, and finished with caramel sauce. Scores  
of people are passionate devotees of this signature beverage. So  
bewitched are they, you'd think it was some kind of magical elixir.  
Well there's no hocus pocus here. We'll tell you exactly what goes into  
it: creamy vanilla-flavored syrup, freshly steamed milk with a topping  
of velvety-rich foam, an intense hit of our Espresso Roast, a finishing  
of buttery caramel drizzle ... okay, we take it back. That does sounds  
like magic to us. (And it tastes even better.)', 1,  
'caramelMacchiato.jpg', 16.00),
```

(7, 'Cocoa Cappuccino', 'Dark, rich espresso with bittersweet mocha sauce lies in wait under a smoothed and stretched layer of thick foam. With less milk than a latte, cappuccino offers a stronger espresso flavor and a luxurious texture. To make it properly requires much skill and attentiveness. Arguably the most important part is frothing the foam to velvety perfection as the milk steams - something our baristas take great care to achieve. The milky moustache that clings to your upper lip is proof we've made yours right. And may we say, you wear it well.', 1, 'cocoaCappucino.jpg', 16.00),

(8, 'Coffee by the Press', 'The coffee press is a classic, straightforward brewing method that produces a boldly flavorful cup. To brew, fresh coffee grounds are fully immersed in hot water. The mesh filter encourages an even extraction that releases flavorful oils into the cup and creates rich, full-bodied cup.', 1, 'caffepress.jpg', 18.00),

(9, 'Cold Brew', 'Slow-steeped, small-batch and super smooth. We use a unique craft-brewing process to create a super smooth tasting coffee. While making our Cold Brew, the coffee never comes into contact with hot water. Instead, the coffee is slow-steeped in cool water for more than 10 hours and is handcrafted in small batches each day. To create our signature recipe, our team spent months experimenting with different brew times and coffee varietals. We specifically developed the Starbucks® Cold Brew Blend to heighten the rich, naturally sweet flavor created during the cold brewing process. The blend incorporates African and Latin American coffees.', 1, 'coldBrew.jpg', 18.00),

(10, 'Matcha Cold Foam Iced Americano', 'A dream come true for coffee and tea lovers - Cup Corner is reimagining the classic Iced Americano with the infusion of matcha - in the format of meringue-like, earthy Matcha Cold Foam. Made with nonfat milk, this layer of smooth Matcha Cold Foam atop the classic Iced Americano instantly intrigues the palate with the contrasting yet complementing flavors of Americano and Matcha. The Matcha Cold Foam Iced Americano is available cold only.', 1, 'matchaFoamAmericano.jpg', 20.00),

(11, 'DoubleShot® Iced Shaken Espresso', 'Two fresh shots of espresso, hand shaken with classic syrup and ice, finished with low fat milk mixed with sweetened whipped cream. Our only hand-shaken espresso drink, it melds the flavors of the rich, full-bodied espresso you love and is chilled and mellowed with a touch of milk and then lightly sweetened. All with tiny bubbles of frothy foam.', 1, 'doubleShot.jpg', 12.00),

(21, 'Brewed Tea', 'Discover the vibrant flavors of full-leaf tea. From light and invigorating to a bold and energizing blend, we have something for everyone.', 5, 'brewedTea.jpg', 10.00),
(22, 'Signature Hot Chocolate', 'Steamed milk with a generous amount of mocha-syrup, topped with fluffy whipped cream and cocoa powder.\r\n\r\n*Only available in hot*', 5, 'hotChocolate.jpg', 18.00),
(23, 'Matcha Latte', 'Sweetened matcha green tea with steamed milk. The Japanese tea ceremony emphasizes the virtues of humility, restraint and simplicity, its practice governed by a set of highly ritualized actions. But this smooth and creamy matcha-based beverage can be enjoyed any way you like. So by all means, slurp away if you want to.', 5, 'matchaLatte.jpg', 18.00),
(24, 'Iced Shaken Lemon Tea', 'Dark black tea. Sunny lemonade. Can such contradictory figures truly coexist in one delightfully refreshing drink? Thankfully, in this case opposites not only attract, they form the perfect marriage. Better yet, no one has to make any compromises - especially not in the flavor department. Did you know? The black and yellow stripes on the common bumblebee may have been the inspiration for this blend of black tea and lemonade. At least, no bumblebee has ever told us otherwise.', 5, 'lemonTea.jpg', 14.00),
(25, 'Strawberry Açaí With Lemonade Starbucks Refreshers™', 'This ice-cold, fruity, and flavorful beverage bursts with berry goodness. Handcrafted with a delicious blend of fruit juice accented by strawberry and açaí notes, delightful zing of citrusy lemonade, and green coffee extract, this hand-shaken beverage surprises thanks to blissful strawberry pieces in every sip. The Strawberry Açaí with Lemonade Starbucks Refreshers™ is only available as an iced handcrafted beverage.', 5, 'strawberryLemonade.jpg', 20.00),
(26, 'Pumpkin Spice Latte', '\r\nPumpkin Spice Latte is a popular seasonal beverage featuring espresso, steamed milk, pumpkin pie spice, and often topped with whipped cream, enjoyed during autumn months.', 7, 'pumpkinSpiceLatte.png', 20.00),
(31, 'Maple Pecan Scones', 'Enjoy the delectable combination of rich maple and crunchy pecans in our Maple Pecan Scones. Each buttery scone is expertly crafted, with a delicate balance of sweet maple syrup and nutty pecans in every bite. Our Maple Pecan Scones are ideal for pairing with your morning coffee or as a delicious afternoon treat, and they provide a comforting taste of indulgence that will leave you wanting more.', 7, 'mfood.jpg', 9.90),

```

(32, 'Cranberry Orange Cake', 'Our Cranberry Orange Cake combines tangy
cranberries and zesty orange to create a delightful harmony of
flavours. Each moist and flavorful bite is a symphony of sweet and
citrus notes, expertly balanced to tantalise your palate.', 7,
'new.cake.png', 18.90),

(33, 'Caramel Apple Latte', 'Enjoy the warm embrace of autumn with our
Caramel Apple Latte. Enjoy the comforting warmth of freshly brewed
espresso, perfectly blended with creamy caramel and the crisp sweetness
of ripe apples. Each sip immerses your senses in a symphony of autumnal
flavours, transporting you to orchard-filled landscapes and cosy
fireside gatherings.', 7, 'new.coffee2.jpg', 15.90),

(34, 'Tiramisu Cake', 'Embark on a decadent journey with our Tiramisu
Cake, a divine creation that combines layers of delicate sponge cake
soaked in rich espresso, luscious mascarpone cream, and cocoa powder.
Each forkful is a symphony of flavours and textures, from the strong
kick of coffee to the creamy indulgence of mascarpone, culminating in a
divine dessert experience', 3, 'atf.cake.png', 12.90);

-- -----
-- 
-- 
-- Table structure for table `users` 

-- 
-- 

CREATE TABLE `users` (
  `id` int(10) UNSIGNED NOT NULL,
  `firstname` varchar(30) NOT NULL,
  `lastname` varchar(30) NOT NULL,
  `username` varchar(50) NOT NULL,
  `email` varchar(50) NOT NULL,
  `password` varchar(20) NOT NULL
) ENGINE=MyISAM DEFAULT CHARSET=utf32 COLLATE=utf32_general_ci;

-- 

```

```
-- Dumping data for table `users`
--


INSERT INTO `users` (`id`, `firstname`, `lastname`, `username`,
`email`, `password`) VALUES
(4, 'Jien Weng', 'Lai', 'Jr', 'reallyhat@gmail.com', '12345678'),
(5, 'Oscar', 'Chong', 'Oscar', 'oscar@gmail.com', 'Oscar1234'),
(6, 'En Yi', 'Liew', 'enyi', 'liewenyi@yahoo.com', '12345678'),
(7, 'John', 'Max', 'max', 'johnmax@hotmail.com', '12345678'),
(8, 'Oscar', 'Lai', 'Oscarr', 'oscar@hotmail.com', '12345678');

--


-- Indexes for dumped tables

--


-- Indexes for table `categories`


--


ALTER TABLE `categories`
ADD PRIMARY KEY (`category_id`);


--


-- Indexes for table `contact_responses`


--


ALTER TABLE `contact_responses`
ADD PRIMARY KEY (`id`);


--


-- Indexes for table `orders`
```

```
--  
ALTER TABLE `orders`  
  ADD PRIMARY KEY (`order_id`);  
  
--  
-- Indexes for table `order_details`  
--  
  
ALTER TABLE `order_details`  
  ADD PRIMARY KEY (`detail_id`),  
  ADD KEY `order_id` (`order_id`);  
  
--  
-- Indexes for table `products`  
--  
  
ALTER TABLE `products`  
  ADD PRIMARY KEY (`product_id`);  
  
--  
-- Indexes for table `users`  
--  
  
ALTER TABLE `users`  
  ADD PRIMARY KEY (`id`),  
  ADD UNIQUE KEY `username` (`username`),  
  ADD UNIQUE KEY `email` (`email`);  
  
--  
-- AUTO_INCREMENT for dumped tables  
--
```

```
--  
-- AUTO_INCREMENT for table `categories`  
  
--  
  
ALTER TABLE `categories`  
  
    MODIFY `category_id` int(11) NOT NULL AUTO_INCREMENT,  
AUTO_INCREMENT=10;  
  
--  
  
-- AUTO_INCREMENT for table `contact_responses`  
  
--  
  
ALTER TABLE `contact_responses`  
  
    MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=10;  
  
--  
  
-- AUTO_INCREMENT for table `order_details`  
  
--  
  
ALTER TABLE `order_details`  
  
    MODIFY `detail_id` int(11) NOT NULL AUTO_INCREMENT,  
AUTO_INCREMENT=90;  
  
--  
  
-- AUTO_INCREMENT for table `products`  
  
--  
  
ALTER TABLE `products`  
  
    MODIFY `product_id` int(11) NOT NULL AUTO_INCREMENT,  
AUTO_INCREMENT=35;  
  
--  
  
-- AUTO_INCREMENT for table `users`  
  
--
```

```

ALTER TABLE `users`

    MODIFY `id` int(10) UNSIGNED NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=9;

-- 
-- Constraints for dumped tables
-- 
-- 
-- 
-- Constraints for table `order_details`
-- 

ALTER TABLE `order_details`

    ADD CONSTRAINT `order_details_ibfk_1` FOREIGN KEY (`order_id`)
REFERENCES `orders` (`order_id`);

```

Conclusion

In conclusion, by effectively combining functionality with user-centric design concepts, the Cup Corner website has emerged as a leading platform in the online coffee shop business. The website, which was created with PHP, HTML, JavaScript and MySQL, is a prime example of how combined web technologies can produce a dynamic and engaging user experience. Because of the site's sophisticated interaction and easy navigation, consumers may explore a wide variety of coffee products, discover more about coffee culture, and make purchases easily.

The website's strong security standards and dynamic content management system, which guarantee safe and effective handling of user data, serve as features of its technical expertise. The responsive design seamlessly transitions across different devices, guaranteeing that the visual attractiveness doesn't detract from the ease of navigation. One of the most important parts of the backend is the administrative dashboard, which makes operations much more

efficient. It offers complete capabilities for user administration, product listing changes, real-time content management, and sales analytics. Administrators need this dashboard in order to effectively manage site operations and respond swiftly to changes in the market or in strategy.

Moreover, the site's performance and scalability are improved by the use of MySQL databases for backend data handling, making it easier to manage large volumes of transactions and data exchanges. A sophisticated yet user-friendly checkout and shopping cart system, developed to manage a wide variety of consumer interactions from various geographic and demographic categories, is supported by this robust infrastructure.

The fundamental technology and design decisions employed in this project provide a strong basis for future improvements as Cup Corner grows and changes. Upgrades are in the works, including improved payment methods to improve accessibility and convenience for all users, deeper interaction with mobile platforms, and sophisticated machine learning algorithms for individualized user experiences. The admin page itself is ready for future development to add predictive analytics and more granular controls, which will improve the capacity to successfully forecast market trends and client wants.

This website not only serves as a testament to the skill and dedication of its development team but also as a model for future innovations in the digital e-commerce landscape. In light of these accomplishments, Cup Corner is well-positioned to expand its market presence and continue to delight and engage coffee lovers around the world. The journey of Cup Corner from concept to realization reflects a meticulous and thoughtful approach to digital retail, setting a high standard for future projects within the industry.

Workload Summary

The objective of the Cup Corner project is to provide a responsive and engaging online coffee shop platform that will improve user experience and enable simple commerce and navigation. The project's full development process took approximately thirteen weeks, beginning with a two-week planning and design phase during which demand was gathered, website architecture and database program design were created, and a rough thread framework was created using tools. A development environment which utilized PHP, HTML, CSS, and JavaScript to encode both the front and back ends of the website and integrated with MySQL for dynamic content management followed the eight-week implementation phase. Furthermore, an innovative activity has been launched to create a management dashboard providing thorough sales analysis, improves backend management capabilities, and facilitates the efficient handling of content, product listings, and user data. The development team consists of four people as following table:

Members Name	Responsibilities
Lai Jien Weng	handles the website's administrative area
Chow Ping Ching	handle security and the back ends of the website
Wong Siew Kuan	handle security and the back ends of the website
Janice Ng Zhi Yan	handle the front ends and the css design of the website

Following the process of encoding, a week-long testing and launch phase ensures the website's usability, security, and functionality. Strict testing procedures, such as availability, security, and functional audits, result in the website's successful launch. The website has entered a phase of ongoing maintenance since its launch, which includes frequent content updates, performance monitoring, security enhancements, and functional upgrades to keep up with changing user demands and technological developments.

References

- 30 Bootstrap Navbars - free examples & easy customization.* (n.d.). MDB - Material Design for Bootstrap.
<https://mdbootstrap.com/docs/standard/navigation/navbar/examples-and-customization/>
- Hovhannisyan, A. (2021, September 18). Creating a Responsive Navbar with HTML, CSS, and JavaScript. *Aleksandr Hovhannisyan*.
<https://www.aleksandrhovhannisyan.com/blog/responsive-navbar-tutorial/>
- Admin and user login in php and mysql database.* (n.d.). CodeWithAwa.
<https://codewithawa.com/posts/admin-and-user-login-in-php-and-mysql-database>
- Faraz. (2023, September 15). A Collection of 20+ Restaurant Website with HTML, CSS, and JavaScript. *Code with Faraz - Best Front-End Components and Blogs*.
<https://www.codewithfaraz.com/article/102/a-collection-of-20-restaurant-website-with-html-css-and-javascript>
- Simple PHP Shopping Cart - PHPPot.* (n.d.). Phppot.
<https://phppot.com/php/simple-php-shopping-cart/>
- Contributor, G. (2024, January 11). *How to connect to MySQL database in PHP*. Sling Academy.
<https://www.slingacademy.com/article/how-to-connect-to-mysql-database-in-php/>
- PHP: MySQL Functions - manual.* (n.d.). <https://www.php.net/manual/en/ref.mysql.php>
- Creating a User Login System with PHP and MySQL - Tutorial Republic.* (n.d.).
<https://www.tutorialrepublic.com/php-tutorial/php-mysql-login-system.php>