A red and black sign

AI-generated content may be incorrect.

Figure 1: Converted enquiry.html to enquiry.php with PHP session and form handling.

**COS10026 WEB TECHNOLOGY PROJECT**

**Assignment 2 Individual Report**

**Done by: Kelvin Sze Hoon LAU (104391559)**

Table of Contents

# 1.0 Introduction

The purpose of this report is to document my individual contributions to the enhancement of the Brew & Go Coffee website through the integration of PHP and MySQL. This project builds upon Assignment 1 by transforming a static coffee shop website into a dynamic, database-driven web application with server-side features.

This report includes an overview of the website, my specific tasks involving PHP scripting and database design, reflections on teamwork and professional learning, and a discussion on privacy in web development. Screenshots of implemented features are also included to support technical explanations.

# 2.0 The Website

The Brew & Go Coffee website serves as a digital platform for a local coffee brand. Initially designed with HTML and CSS, it was extended in this project to support backend functionality. Key pages allow customers to browse products, apply for jobs, sign up for memberships, and submit enquiries. Admin users can now manage these submissions through secure login pages.

The website’s users include general customers and internal admin staff. Customers interact with the site through public forms, while admin users have exclusive access to backend dashboards. The primary goal of the site is to strengthen Brew & Go’s online presence and streamline user engagement.

# 3.0 My contribution

## 3.1 HTML to PHP Migration and Bug Fixes

As part of the website enhancement process, I identified and resolved several functional issues that arose from the use of .html files after we migrated the entire website to PHP.

Previously, pages like enquiry.html, products.html, and hardcoded footers were static and unable to process PHP logic. This caused problems such as:

* Broken links or form submissions.
* Inability to include server-side validation or session handling.
* Missing or inconsistent UI elements across pages.

To resolve these issues, I:

* Converted enquiry.html to enquiry.php so it could submit data to enquiry\_process.php and display validation/confirmation messages dynamically.
* Rebuilt the products page using product.php, allowing real-time search, member price checks, and secure product display using data from the MySQL database.
* Modularised the footer using footerbare.inc, then included it across all pages with <?php include("inc/footerbare.inc"); ?> to ensure consistency and easier maintenance.
* Reviewed and fixed all hardcoded .html links in navigation menus, forms, and footers, replacing them with .php equivalents to support session-based content (e.g., showing admin-only links or user login state).

This migration not only resolved the file loading errors but also allowed dynamic features like:

* Session-aware footers and navigation.
* PHP validation in forms (e.g., required fields in enquiry).
* Secure database integration for all key pages.

By addressing these bugs and standardising the file format, the entire website became fully PHP-compatible and scalable, with consistent styling and logic across every page.

## 3.2 Anti-Spam Feature

To prevent spam, I developed a robust anti-spam script (anti\_spam\_check.php) that tracks submissions using an IP-based MySQL table. Each user is limited to 5 submissions per 60 seconds. If exceeded, they are blocked for 5 minutes.

* Table: spam\_control with fields ip\_address, attempt\_count, first\_attempt, blocked\_until.
* Blocking Logic: If user exceeds limits, they are redirected to a styled blocking page showing the remaining block time.
* Security: All queries use mysqli prepared statements to avoid SQL injection.

This goes beyond the basic requirement by offering persistent spam protection even across sessions, tabs, and browsers.

A screen shot of a computer

AI-generated content may be incorrect.

Figure 2: Modular footer included using `footerbare.inc` across all pages.

A screen shot of a computer program

AI-generated content may be incorrect.

Figure 3: Anti-spam table structure using IP-based tracking.

A screenshot of a computer program

AI-generated content may be incorrect.

Figure 4: PHP logic to block excessive submissions using session-aware logic.

A screen shot of a computer program

AI-generated content may be incorrect.

Figure 5: Styled blocking page shown after spam threshold is exceeded.

## 3.3 Product Search Feature

I implemented a Product Search Feature that allows users to search for available coffee products using keywords. This enhancement improves user experience by making it easier to navigate through the product list, especially as the inventory grows.

How It Works

* A search bar is provided at the top of the product page (products.php).
* When a user enters a keyword and clicks the search button, a PHP script captures the input and performs a LIKE query on the products table in MySQL.
* The page dynamically displays only the matching results.
* If no matches are found, a “No products found” message is shown.

Technical Implementation

* The search form uses the GET method to send user input.
* The SQL statement is securely written using mysqli prepared statements to avoid SQL injection.
* Example PHP snippet:

if (isset($\_GET['search'])) {

$keyword = '%' . $\_GET['search'] . '%';

$stmt = $conn->prepare("SELECT \* FROM products WHERE name LIKE ?");

$stmt->bind\_param("s", $keyword);

$stmt->execute();

$result = $stmt->get\_result();

}

* The search results are looped and displayed dynamically using a while loop.

Why It’s Valuable

* Improves usability by reducing the time users need to find what they want.
* Encourages engagement by making exploration of the menu feel interactive.
* Provides a scalable solution that adapts as more products are added.

Innovative Aspects

* Real-time product filtering using a simple but powerful SQL LIKE operation.
* Dynamic result rendering that maintains consistent site styling.
* Graceful fallback handling when no matches are returned.

A screen shot of a computer program

AI-generated content may be incorrect.

Figure 6: Product search bar allowing keyword-based filtering.

A screen shot of a computer code

AI-generated content may be incorrect.

Figure 7: Backend PHP `LIKE` query for fetching matching products.

A screen shot of a computer code

AI-generated content may be incorrect.

Figure 8: Displayed results or fallback message when no products found.

## 3.4 Member top-up Module

The Member Top-Up Module allows registered users to add credit to their accounts, simulating a prepaid payment system. This feature improves convenience and supports future functionality such as order payments or reward tracking.

How It Works

* A form in the user dashboard (user\_dashboard.php) links to member\_topup.php, where members can select a top-up amount and method.
* Upon submission, the form data is validated, and the amount is inserted into the topup table.
* The system records the amount, method (e.g., TNG, Boost), timestamp, and user ID.
* After successful top-up, users receive a confirmation message.

Technical Implementation

* The topup table includes:
  + id (primary key)
  + member\_id (foreign key)
  + amount (DECIMAL)
  + method (ENUM: 'TNG', 'Boost', 'GrabPay', etc.)
  + created\_at (timestamp)
* Secure form handling with mysqli prepared statements prevents SQL injection.
* Example PHP snippet from member\_topup.php:

php

CopyEdit

$member\_id = $\_SESSION['user\_id'];

$amount = $\_POST['amount'];

$method = $\_POST['method'];

$stmt = $conn->prepare("INSERT INTO topup (member\_id, amount, method) VALUES (?, ?, ?)");

$stmt->bind\_param("ids", $member\_id, $amount, $method);

$stmt->execute();

* Users can view their top-up history from their dashboard.

Why It’s Valuable

* Mimics real-world credit systems like Starbucks wallet or Touch 'n Go eWallet.
* Adds interactivity and realism to the user experience.
* Sets the foundation for future features like balance checking or payment deduction.

Innovative Aspects

* Clean, intuitive form design with dropdown for payment methods.
* Stores all top-up data in a structured and searchable format.
* Can be extended easily to add receipt generation or credit usage tracking.

A screen shot of a computer

AI-generated content may be incorrect.

Figure 9: Top-up form with payment method and amount fields.

A screen shot of a computer program

AI-generated content may be incorrect.

Figure 10: PHP code for balance update and method tracking.

A screen shot of a computer program

AI-generated content may be incorrect.

Figure 11: Synced topup table design from members.

A screen shot of a computer

AI-generated content may be incorrect.

Figure 12: Latest update fetched and shown dynamically in footer.

# 3.5 Promotion and News update Module

This enhancement allows the admin to post the latest news, events, or promotions, which are displayed dynamically in the website footer. It enables better customer engagement and gives the website a more dynamic, updated appearance.

How It Works

* The system retrieves the most recent update from the updates table and displays it in the “Latest Update” section of the footer.
* Admins see an embedded form in the footer when logged in, allowing them to add or update messages and optionally attach an image.
* Upon submission, the edit\_update.php script handles the upload and stores the update in the database.

Technical Implementation

* MySQL table updates includes:
  + id (primary key)
  + update\_message (TEXT)
  + photofile (VARCHAR for image path)
  + created\_at (timestamp)
* PHP fetches the latest update with:

php

CopyEdit

$sql = "SELECT update\_message, photofile FROM updates ORDER BY id DESC LIMIT 1";

* When the admin logs in, a conditional PHP block displays a form:

php

CopyEdit

if (isset($\_SESSION['role']) && $\_SESSION['role'] === 'admin') {

// Show editable form with textarea and image upload

}

* The form uses POST with multipart/form-data to allow image upload.

Why It’s Valuable

* Gives the website real-time promotion functionality.
* Keeps content fresh and relevant to visitors.
* Admins can update content without editing source code.

Innovative Aspects

* Dynamic content refresh without full page reload.
* Integrated directly into the footer to maintain visibility on every page.
* Admin-only access with role-based session protection.



Figure 13: Admin-only form for submitting promotions with images.

A computer code on a black background

AI-generated content may be incorrect.

Figure 14: Forgot password form for users to request reset link.

A screen shot of a computer code

AI-generated content may be incorrect.

Figure 15: Token and expiry inserted into password\_resets table.

## 3.6 Forgot Password and Email Reset Feature

I implemented a secure Forgot Password feature that allows users to reset their password via email. This enhancement improves user experience, adds a layer of account recovery, and demonstrates proper use of token-based authentication.

How It Works

* Users click “Forgot Password” on the login page and are directed to a reset request form.
* After submitting their email, the system checks if the email exists in the members table.
* A secure token and expiry timestamp are generated and stored in a password\_resets table.
* The user receives an email with a reset link containing the token.
* Clicking the link brings them to a reset page where they can enter a new password.
* The system validates the token and expiry before updating the password (hashed securely).

Technical Implementation

* I created a new table called password\_resets with fields:
  + email (VARCHAR)
  + token (CHAR(100))
  + expires\_at (DATETIME)
* On form submission:

php

CopyEdit

$token = bin2hex(random\_bytes(50));

$expires = date('Y-m-d H:i:s', time()+3600); // 1 hour validity

* The reset email is sent using XAMPP's Sendmail setup, with the token link:

php

CopyEdit

http://localhost/your\_project/reset\_password.php?token=XYZ

* When the link is opened, the token is validated, and password\_hash() is used to securely update the user's password.

Why It’s Valuable

* Ensures users are not locked out if they forget their password.
* Simulates real-world email-based password recovery.
* Improves professionalism and credibility of the website system.

Innovative Aspects

* Implements secure, expiring tokens to prevent unauthorized resets.
* No hard-coded logic—fully dynamic and database-driven.
* Works seamlessly with the login system, showing strong backend integration.