



**UNIVERSITI KUALA LUMPUR KAMPUS KOTA
MALAYSIAN INSTITUTE OF INFORMATION TECHNOLOGY**

COURSE DETAILS	
CAMPUS	MALAYSIAN INSTITUTE OF INFORMATION TECHNOLOGY
COURSE NAME	PHP WEB PROGRAMMING
COURSE CODE	ITD 31303
LECTURER	SIR HAZMAN BIN BADRUNSHAM

ASSESSMENT DETAILS	
TITLE/NAME	ASSIGNMENT
WEIGHT	15%
DATE	WEEK 6
COURSE OUTCOME	CLO1: Builds knowledge of the PHP language to create a dynamic web page

STUDENT DETAILS	
NAME	IZZAT KZ
STUDENT ID	5210112XXXX
CLASS	L03 – B01

TABLE OF CONTENTS

Pg.3.....	(1) INTRODUCTION
Pg.4.....	(2) PROJECT FLOW
Pg.5.....	(2.1) start.html
Pg.6.....	(2.2) shipping.php & shipping-check
Pg.8.....	(2.3) category.html
Pg.9.....	(2.4) product.php
Pg.11.....	(2.5) cart.php
Pg.13.....	(2.6) form.php & form-check.php
Pg.15.....	(2.7) order-summary.php
Pg.16.....	(2.8) gateway.html
Pg.17.....	(3) IMPLEMENTATION
Pg.18.....	(3.1) Input Validation
Pg.22.....	(3.2) Requirement Logic
Pg.22.....	(3.2.1) if-else statement
Pg.23.....	(3.2.2) calculation
Pg.24.....	(3.2.3) function with a return value
Pg.24.....	(3.2) Multiple Page
Pg.25.....	(3.3) Redux
Pg.26.....	(3.2) Sticky Form

INTRODUCTION

For this assignment, I am required to develop an e-commerce application such as Shopee, Lazada, and etc. The app requirements are as follows:

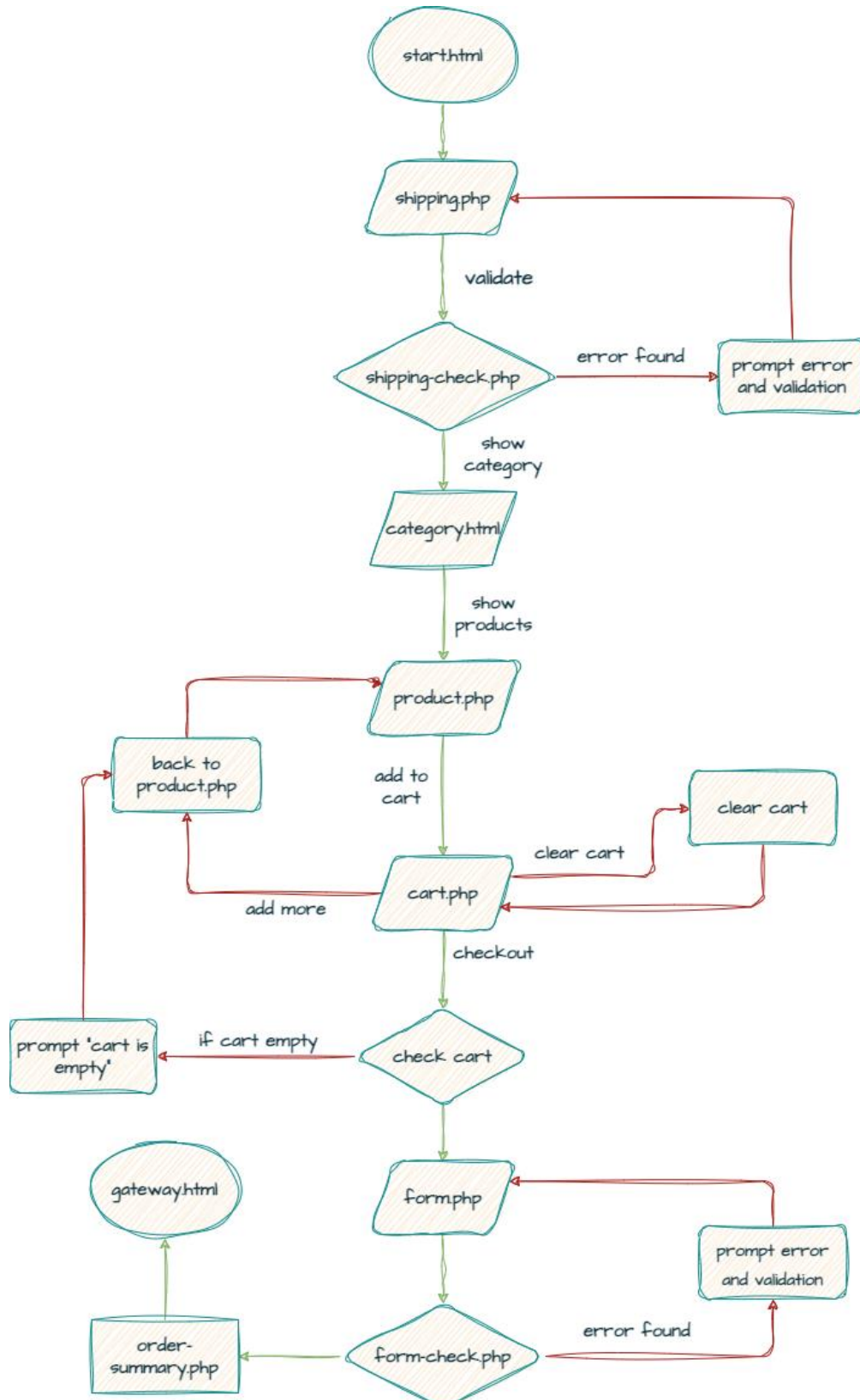
1. Buyers are required to choose their home destination. (address of buyers)
2. Buyes are required to to buy what they want
 - 2.1. Show list Of categories.
 - 2.2. Show list of products for that type of category. (at least 3 products)
3. Buyers are required to choose type of payments.
 - 3.1. Show list of payments. (at least 3 type of payments)
4. Buyers are required to fill form details as buyer. (Full name, phone, notes)
5. Show home destination, total products, quantity, and total payment for the buyer.

I have decided to create an **online watches shop application** named **NtahTime**, taking inspiration from watatime.com and solartime.com websites. NtahTime is a basic user-friendly online store where you can buy watches, wall clocks and accessories. It is a third-party app and does not control the brand of any products. Apart from that, every product is broken down into a variety of categories, such men watches and women watches, to ease users to find what what they're looking for. It has a good UI with good choice of colours making users able to use the app comfortably. Furthermore, this app has the ability to let users put items to cart to make browsing and picking easier. There are also navigation on the left side of the product page so users can move among categories effortlessly. Other than that, NtahTime provides an informative straightforward feedback in a form of validation and errors so the users can solve the problem without frustration.

NtahTime is fully built by using HTML, CSS, and PHP languages and without the use of Javascript, SQL or any other language. This app has a total of 10 pages, excluding 1 stylesheet. The value from the filled form are sent using the post method, which then are stored using session. With session, the code can be written without any complexities as there is no need to repeatedly using post/get method on every page to pass values across pages. Session can store form values on one page, retrieved from on other page. As the app has many pages, it is ideal to use session to store value, including a multidimensional array.

PROJECT FLOW

Flowchart below is a representation of my app's flow:



As mentioned earlier, the app has total of 10 pages. Those pages are;

- start.html
- shipping.php
- shipping-check.php
- category.html
- product.php
- cart.php
- form.php
- form-check.php
- order-summary.php
- gateway.html

START.HTML



this page only has a clickable app title at the center of the page that will direct users to the next page which is shipping.php

```
> < start.html > < html
<html>
<head>
  <link rel="stylesheet" href="stylesheet.css">
  <title>NtahTime</title>
  <style>...
</style>
</head>
<body>
  <div id="bg">
    <div>
      <a href="shipping.php" style="text-decoration: none;">
        <p id="centered-title"><b>NtahTime<br />Watch Shop</b></p>
      </a>
    </div>
  </div>
</body>
</html>
```

SHIPPING.PHP & SHIPPING-CHECK.PHP

shipping.php page contains a shipping form for users to enter their home destination. The form will be requesting users' address, city, state, and postcode of their house. The page includes script from shipping-check.php so that the filled value can be validate. With this, the code can be more organised as two of this code can be executed at the same time.

```
> shipping.php > html > head
<?php require("shipping-check.php") ?>

<html>
  <head>
    <title>Shipping Address</title>

<link rel="stylesheet" href="stylesheet.css">

<style>...

<?php
    if($address_err != null){
        ?> <style>.addressErr{display: block}</style>
    }
    if($city_err != null){
        ?> <style>.cityErr{display: block}</style> <?>
    }
    if($state_err != null){
        ?> <style>.stateErr{display: block}</style> <?>
    }
    if($postcode_err != null){
        ?> <style>.postcodeErr{display: block}</style>
    }
?>

shipping.php > html > head
<p class="name"><i> Shipping Destination </i></p>
</div>
<form action="shipping.php" method="post" autocomplete="off">
<fieldset class="fieldAdd"><legend><b>Shipping Address</b></legend>

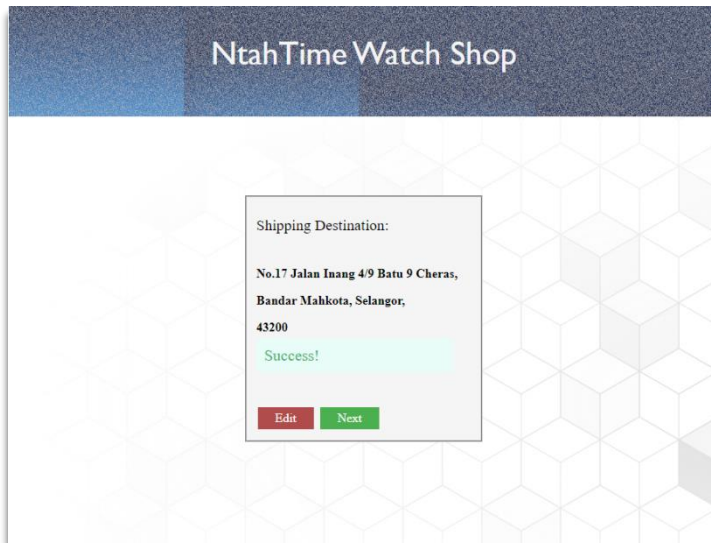
  <p><b>Address</b><b style="color:red;">*</b></p>
  <input type="text" name="shipping_address" placeholder="Address" size="75" value="<?php echo $shipping_address ?>">
  <?php echo $address_err ?>
  </p>

  <p><b>City</b><b style="color:red;">*</b></p>
  <input type="text" name="shipping_city" placeholder="City" size="13" value="<?php echo $shipping_city ?>">
  <p class="error cityErr">
  <?php echo $city_err ?>
  </p>

  <p><b>State</b><b style="color:red;">*</b></p>
  <select name="shipping_state">
    <option value="null">Select A State</option>
    <option value="Johor">Johor</option>
    <option value="Kedah">Kedah</option>
    <option value="Kelantan">Kelantan</option>
    <option value="Melaka">Melaka</option>
    <option value="Negeri Sembilan">Negeri Sembilan</option>
    <option value="Pahang">Pahang</option>
    <option value="Perak">Perak</option>
    <option value="Perlis">Perlis</option>
    <option value="Pulau Pinang">Pulau Pinang</option>
    <option value="Sabah">Sabah</option>
    <option value="Sarawak">Sarawak</option>
    <option value="Selangor">Selangor</option>
    <option value="Terengganu">Terengganu</option>
    <option value="Kuala Lumpur">Kuala Lumpur</option>
  </select>
  <p class="error stateErr">
  <?php echo $state_err ?>
  </p>

  <p><b>Postcode</b><b style="color:red;">*</b></p>
  <input type="number" name="shipping_postcode" placeholder="Postcode" size="6" value="<?php echo $shipping_postcode ?>">
  <p class="error postcodeErr">
  <?php echo $postcode_err ?>
  </p>

  <p><br><input type="checkbox" name="default"> <i>set address as default?</i></p>
  <p><button type="submit" class="button" name="confirmed">Confirm Address</button></p>
</fieldset>
```



The page will then prompt all filled value from the shipping from before once all input has passed the validation. This will let users to double check and edit the information they have submitted. If user want to make adjustments, they can click the EDIT button to be redirected back to shipping.php. If user want to proceed, the NEXT button will direct them to the next page.

The code behind shipping-check.php contains script that sends error message to shipping.php if any field were field incorrectly. This validation process will later be briefly explained on the validation implementation. If no error found, each one values will be stored into each one session.

```
<?php

$shipping_address = null;
$address_err = null;
$shipping_city = null;
$city_err = null;
$shipping_state = null;
$state_err = null;
$shipping_postcode = null;
$postcode_err = null;

if(isset($_POST['confirmed'])) {
    $shipping_address = $_POST['shipping_address'];
    $shipping_city = $_POST['shipping_city'];
    $shipping_state = $_POST['shipping_state'];
    $shipping_postcode = $_POST['shipping_postcode'];

    if(empty(trim($shipping_address))){
        return $address_err = "Address field is empty";
    } else{

        if(empty(trim($shipping_city))){
            return $city_err = "City field is empty";
        } else{

            if(!preg_match("/^[a-zA-Z\s]+$/", $shipping_city)){
                return $city_err = "City can only contain letters";
            } else {

                if($shipping_state=='null'){
                    return $state_err = "Select your state";
                } else{

                    if(empty(trim($shipping_postcode))){
                        return $postcode_err= "Postcode field is empty";
                    }else if (!preg_match('/^\d{5}$/', $shipping_postcode)) {
                        return $postcode_err = "Postcode should be a 5-digit number";
                    }

                    if(empty($address_err) && empty($city_err) && empty($state_err) && empty($postcode_err)) {

                        session_start();

                        $_SESSION['shipping_address'] = $shipping_address;
                        $_SESSION['shipping_city'] = $shipping_city;
                        $_SESSION['shipping_state'] = $shipping_state;
                        $_SESSION['shipping_postcode'] = $shipping_postcode;

                        echo '';
```



```

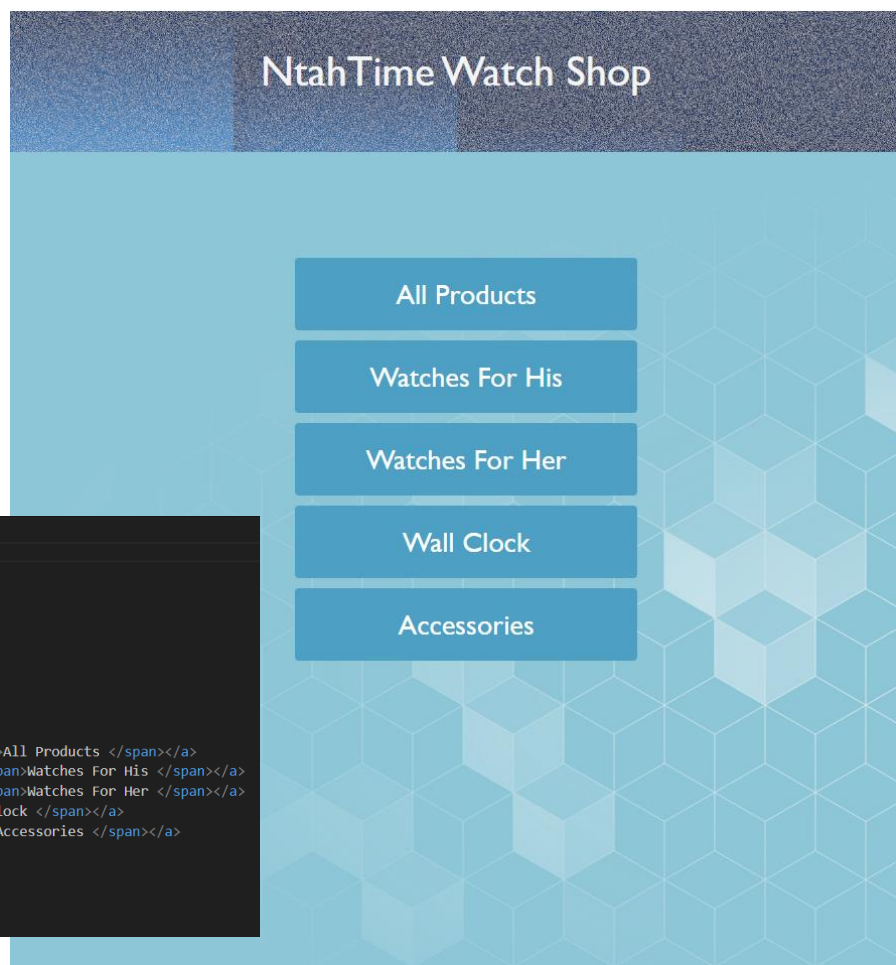
shipping.php  shipping-check.php X
project > shipping-check.php > ...
54
55
56
57 <html><head> <title>success</title>
58 <link rel="stylesheet" href="stylesheet.css">
59 <style>...
81 </head> <body>
82
83 <div id="bg">
84 <div id="header2">NtahTime Watch Shop</div>
85 <fieldset class="fieldAdd">
86 <p style="font-size:larger;">Shipping Destination:<br /><br /></p>
87 <p><b><?php echo $shipping_address . ', ' ?></b></p>
88 <p><b><?php echo $shipping_city . ', ' . $shipping_state . ', ' ?></b></p>
89 <p><b><?php echo $shipping_postcode ?></b><br /></p>
90 <p class="success">Success!</p>
91 <a href="shipping.php" class="buttonRed">Edit</a> <a href="category.html" class="button">Next</a>
92 </fieldset>
93 </div>
94 </body></html>
95
96 <?php
97 exit;

```

Once there are no errors, the code above will be executed, showing the filled data to the users.

CATEGORY.HTML

This page will show list of categories with href link that will lead users to to each category on product.php. This align with the app requirements which is to show list of categories after user filled the destination form.

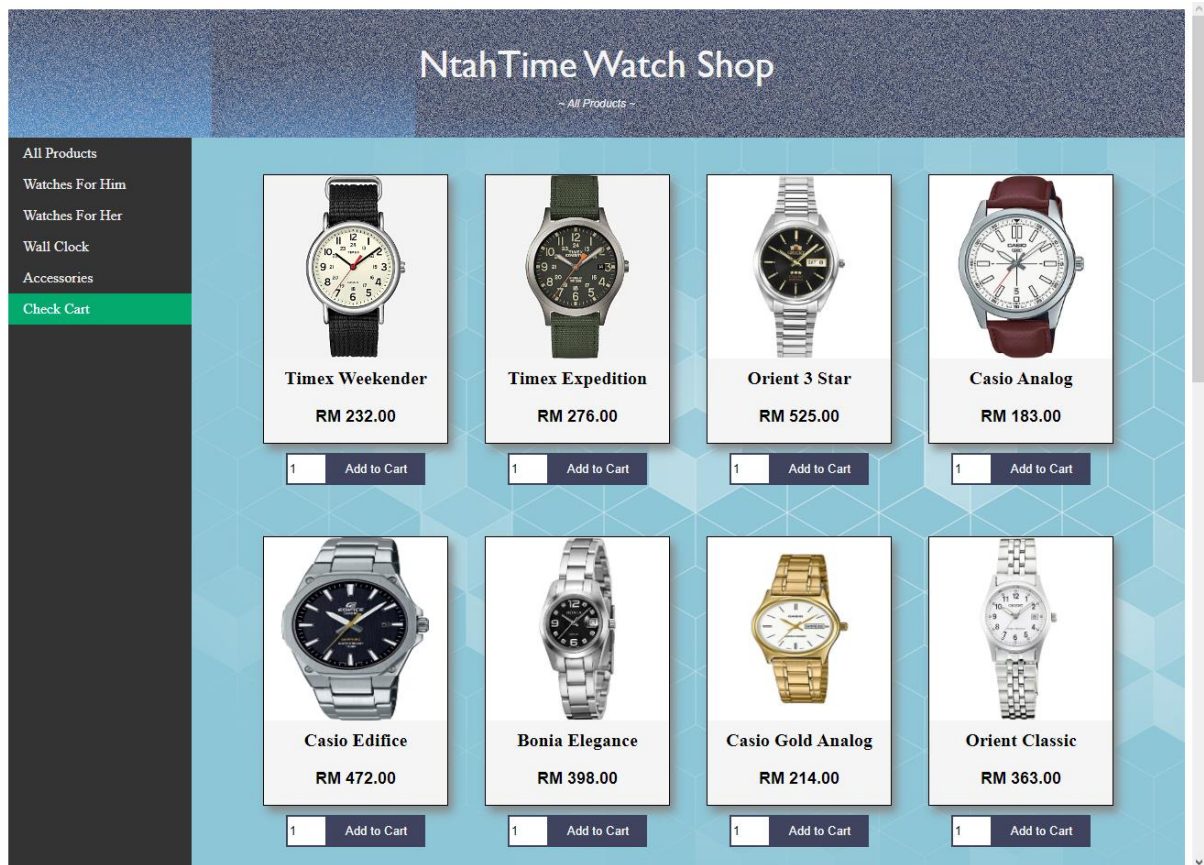


```

> category.html > html
<html>
<head>
<link rel="stylesheet" href="stylesheet.css">
<title>NtahTime Watch Shop</title>
</head>
<body>

<div id="header2">NtahTime Watch Shop</div>
<div id="page">
<div class="cat-container" style="align-items: center;">
<a href="product.php?category=All Products" class="cat"><span>All Products </span></a>
<a href="product.php?category=Watches For Him" class="cat"><span>Watches For His </span></a>
<a href="product.php?category=Watches For Her" class="cat"><span>Watches For Her </span></a>
<a href="product.php?category=Clock" class="cat"><span>Wall Clock </span></a>
<a href="product.php?category=Accessories" class="cat"><span>Accessories </span></a>
</div>
</body>
</html>

```


PRODUCT.PHP

This page will show list of product for user to browse and pick item. It has an add to cart button along with quantity box under each item so users can easily add item and how many they wanted into cart. The quantity box is set as type=number, avoiding conflicts with any accidental typo. It is also being set to only accept 1 to 30, wont allow any lower than 1 and higher than 30. With these, there will be no need for validation in this part. Navigation menu are provided at the left side of the page to ease user navigating among categories, also checking cart. The category's name will be shown below the title on header so the user knows which section is they're on. Once user clicked the add to cart button, all information of the chosen item will be send using method post to cart.php, as well as user will be directed to cart.php. User may come back to this page to add more product into cart. User can also add item from different categories into the same cart.

```
<?php

$products = array(
    "Watches For Him" => array(
        array(
            "name" => "Timex Weekender",
            "picture" => "weekender.jpg",
            "price" => "232.00"
        ),
        array(
            "name" => "Timex Expedition",
            "picture" => "expedition.jpg",
            "price" => "276.00"
        ),
        array(
            "name" => "Orient 3 Star",
            "picture" => "tristar.jpg",
            "price" => "525.00"
        ),
        array(
            "name" => "Casio Analog",
            "picture" => "analog.jpg",
            "price" => "183.00"
        ),
        array(
```

All product information such as name, image, and price are declared in multidimensional array to be broken down into variety of categories.

The function below are using foreach to show list of product for each category along with add to cart button and quantity box.

```
5 references
function displayProducts($category, $products) {
    foreach($products[$category] as $product) {
        echo '<form method="post" action="cart.php">';
        echo '<div class="product">';
        echo '<div class="box">';
        echo '';
        echo '<h3>'. $product["name"] . '</h3>';
        echo '<p>RM '. $product["price"] . '</p>';
        echo '<input type="hidden" name="product_name" value="'. $product['name'] . '">';
        echo '<input type="hidden" name="product_price" value="'. $product['price'] . '">';
        echo '<input type="hidden" name="product_picture" value="'. $product['picture'] . '">';
        echo '</div>';
        echo '<input type="number" name="product_quantity" value="1" min="1" max="30" class="quantityBox"><button type="submit" name="add-to-cart">Add to Cart</button>';
        echo '</div>';
        echo '</form>';
    }
}

$category = isset($_GET['category']) ? $_GET['category'] : 'All Products';
?>
```

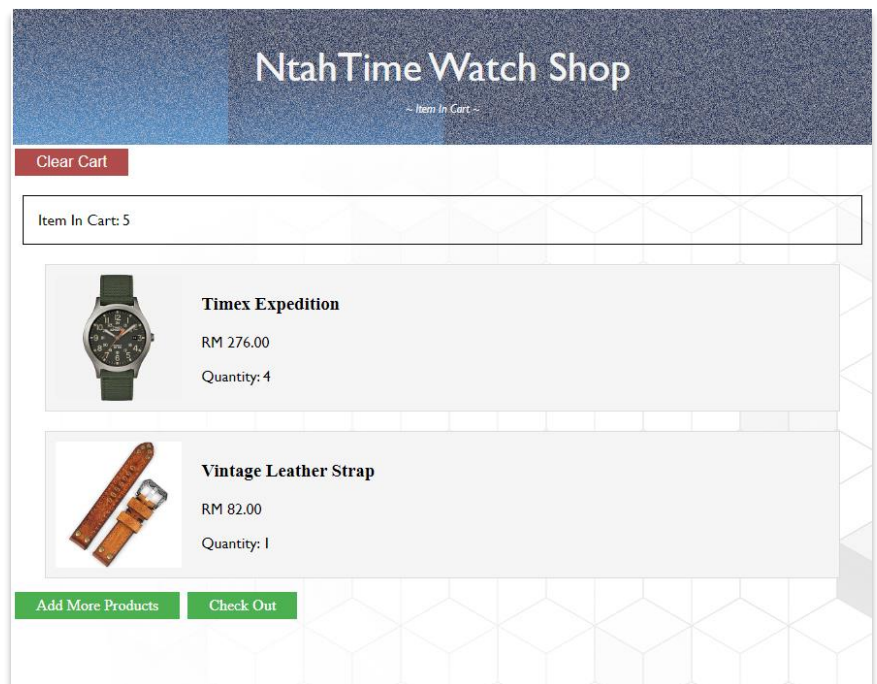
This code will use displayProducts() to show the products depending on categories.

```
if($category == 'All Products') {
    echo '<div id="content">';
    displayProducts('Watches For Him', $products);
    displayProducts('Watches For Her', $products);
    displayProducts('Accessories', $products);
    displayProducts('Clock', $products);
    echo '</div>';
} else {
    echo '<div id="content">';
    displayProducts($category, $products);
    echo '</div>';
}

?>
</div>
```

CART.PHP

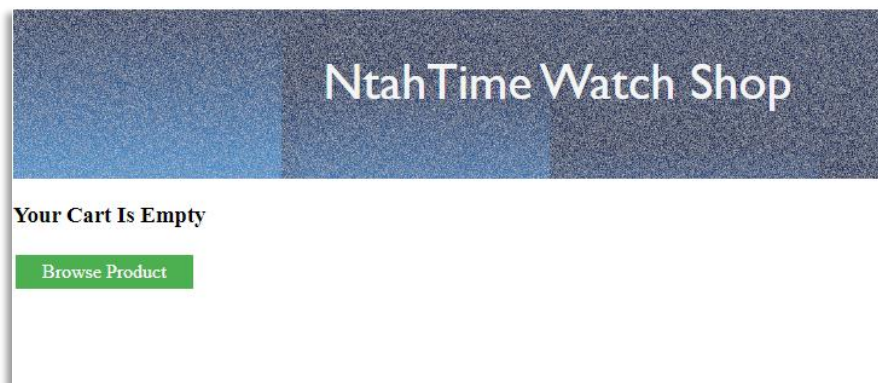
This page will show a list of chosen item and total item in cart. The button Add More Products can redirect users back to product.php to add more items into cart. The values posted from product.php are now being stored into cart multidimensional array. It is then being stored into session[cart] so that when user navigate back to product, the value in cart wont be lost due to the page reloads. User can easily clear cart with the intended button.



```
if(isset($_POST["clear_cart"])){  
    $cart = array();  
    unset($_SESSION["cart"]);  
    header("Location: cart.php");  
    exit;  
}
```

Code for clearing cart array once the clear_cart button were clicked

The checkout button would direct user to the form.php page. However, if user were to checkout with an empty cart, a message will be prompted telling user the cart is empty. Also provides button for user to browse back to product.php



```
// Check if item already exists in the cart
$item_index = -1;
foreach($cart as $index => $item) {
    if($item["product_name"] == $product_name) {
        $item_index = $index;
        break;
    }
}

// If item exists, update its quantity
if($item_index != -1) {
    $cart[$item_index]["product_quantity"] += $product_quantity;
} else {
    // If item does not exist, add it to the cart
    array_push($cart, $cart_item);
}

$_SESSION["cart"] = $cart;
}

if(isset($_POST["clear_cart"])){
    $cart = array();
    unset($_SESSION["cart"]);
    header("Location: cart.php");
    exit;
}

$total_quantity = 0;
foreach($cart as $item) {
    $total_quantity += $item["product_quantity"];
}
```

This code is to avoid any duplication happens when user added the same product at the different time. If the existed item were added, it should only add up the quantity of the item. With this, cart preview wont be messy with a bunch of unnecessary item duplication.

The posted product information from product.php will be stored into array session[cart]. Also same as the product.php, this page will also use foreach to show list of chosen products.

```
session_start();

if(isset($_SESSION["cart"])) {
    $cart = $_SESSION["cart"];
} else {
    $cart = array();
}

if(isset($_POST["add-to-cart"])){
    $product_name = $_POST["product_name"];
    $product_price = $_POST["product_price"];
    $product_picture = $_POST["product_picture"];
    $product_quantity = $_POST["product_quantity"];

    $cart_item = array(
        "product_name" => $product_name,
        "product_price" => $product_price,
        "product_picture" => $product_picture,
        "product_quantity" => $product_quantity,
    );
}
```

FORM.PHP & FORM-CHECK.PHP

NtahTime Watch Shop

~ Buyer's Form ~

Type of Payment

☐ Cash On Delivery

☐ Online Banking

☐ Credit/Debit Card

Please choose one of the payment types

Buyer's Information

Full Name*

Phone Number*

Notes

Confirm Cancel

form.php is a page with buyer's form. The page will requires user to choose a payment type, enter their full name, phone number, and notes. Similar to shipping.php, this page is alos sharing script with form-check.php for validation and errors detection. User can navigate back to cart.php with the Cancel button. Otherwise, user can proceed.

```

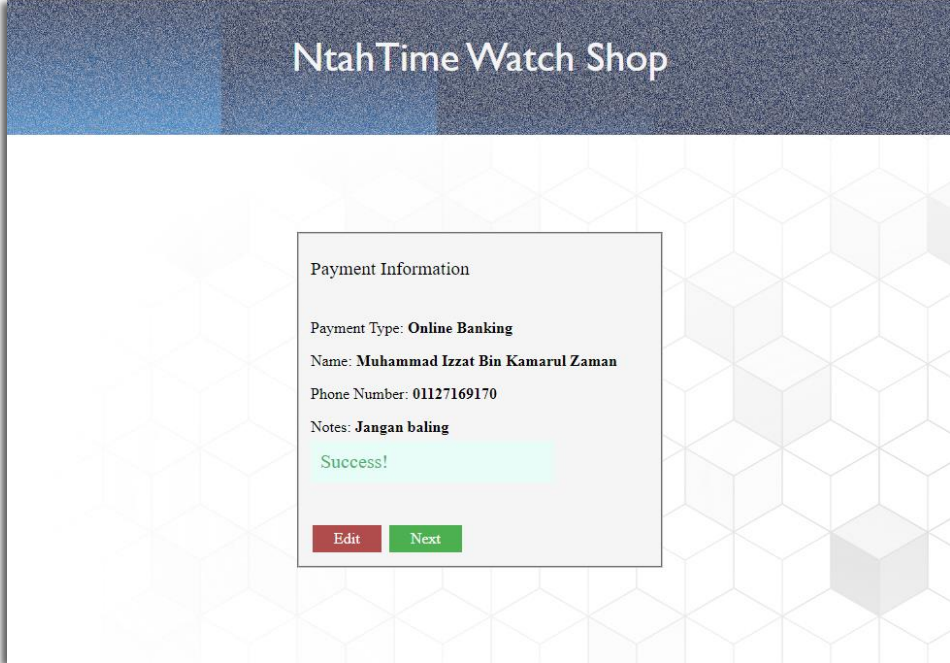
<body>
<div id="header2">
  NtahTime Watch Shop
  <p class="name"><i>~ Buyer's Form ~</i></p>
</div>
<div id="bg">
  <form action="form.php" method="post" autocomplete="off">

<fieldset class="field"><legend class="legend">Type of Payment</legend>
<br>
<p class="radio"><input type="radio" name="paymentType" value="Cash On Delivery" <?php if($paymentType === 'Cash On Delivery') echo 'checked'; ?>> Cash On Delivery</p>
<p class="radio"><input type="radio" name="paymentType" value="Online Banking" <?php if($paymentType === 'Online Banking') echo 'checked'; ?>> Online Banking</p>
<p class="radio"><input type="radio" name="paymentType" value="Credit/Debit Card" <?php if($paymentType === 'Credit/Debit Card') echo 'checked'; ?>> Credit/Debit Card</p>
<?php echo $paymentType_err ?>
</p>
</fieldset>

<fieldset class="field"><legend class="legend">Buyer's Information</legend>
<br>
<p>Full Name<b style="color:red;">*</b></p>
<input type="text" name="buyerName" size="30" value="<?php echo $buyerName ?>">
<p class="error buyerNameErr">
<?php echo $buyerName_err ?>
</p>
<p>Phone Number<b style="color:red;">*</b></p>
<input type="tel" name="phone" size="20" value="<?php echo $phone ?>" placeholder="01XXXXXXXXXX">
<p class="error phoneErr">
<?php echo $phone_err ?>
</p>
<p>Notes</p>
<textarea name="notes"><?php echo $notes ?></textarea>
<p><button type="submit" name="confirmed" class="button">Confirm</button><span>
</span><a href="cart.php" class="buttonRed">Cancel</a></p>
</fieldset>

```


The page will then prompt back the submitted value for users to make changes and double check. Users can then proceed to next page which is order-summary.php



The screenshot shows a web application titled "NtahTime Watch Shop". It features a payment information form with the following details:

- Payment Information**
- Payment Type:** Online Banking
- Name:** Muhammad Izzat Bin Kamarul Zaman
- Phone Number:** 01127169170
- Notes:** Jangan baling
- Success!** (highlighted in green)
- Edit** (red button) and **Next** (green button)

All posted values will then be stored into each session once there are no error found.

```
if(empty($paymentType_err) && empty($buyerName_err) && empty($phone_err)) {  
    $_SESSION['paymentType'] = $paymentType;  
    $_SESSION['buyerName'] = $buyerName;  
    $_SESSION['phone'] = $phone;  
    $_SESSION['notes'] = $notes;  
}
```

ORDER-SUMMARY.PHP

NtahTime Watch Shop

Order Summary

Shipping Destination
 No.17 Jalan Inang 4/9 Batu 9 Cheras
 Bandar Mahkota, Selangor
 43200

Product List
 - Timex Weekender: (RM 232.00) x 1 = RM 232.00
 - Vintage Leather Strap: (RM 82.00) x 3 = RM 246.00
 Total price: RM 478.00

Payment Information
 Payment: Online Banking
 Name: Muhammad Izzat Bin Kamarul Zaman
 Phone: 01127169170
 Notes: jangan baling

This page will view the summary of all submitted information to users before placing orders. Users may cancel and will be redirected back to cart.php. If everything is good, users may place order and be directed to gateway.php.

```
<?php
session_start();

if(isset($_SESSION["cart"])) {
    $cart = $_SESSION["cart"];
} else {
    $cart = array();
}

// Retrieve all session values
$shipping_address = $_SESSION["shipping_address"] ?? "";
$shipping_city = $_SESSION["shipping_city"] ?? "";
$shipping_state = $_SESSION["shipping_state"] ?? "";
$shipping_postcode = $_SESSION["shipping_postcode"] ?? "";

$paymentType = $_SESSION["paymentType"] ?? "";
$buyerName = $_SESSION["buyerName"] ?? "";
$phone = $_SESSION["phone"] ?? "";
$notes = $_SESSION["notes"] ?? "";

$cart = $_SESSION["cart"] ?? [];

// Calculate total price of all products in cart
$total_price = 0;
foreach ($cart["product_price"] ?? [] as $pricekey => $price) {
    $total_price += $price * ($cart["product_quantity"][$pricekey] ?? 0);
}

?>
```

This page will retrieve all information from stored session. Those informations are shipping destination, cart with product array, and payments information. It would also calculate the total price of all product to tell user the amount to pay.

GATEWAY.HTML

Finally, this page will just show a text as there are no payment gateway integrated to this application. Also providing href button to Rollback From Start on start.html

PAYMENT GATEWAY IN PROCESS

End of Coding

author -- izzat kz

[Rollback From Start](#)

```
<html>

<head>
  <link rel="stylesheet" href="stylesheet.css">
  <title>Gateway PROCESS</title>
  <style>...
</style>
</head>
<body>
  <div id="bg">
    <div>
      <p id="centered-title"><b>PAYMENT GATEWAY<br />IN PROCESS</b></p>
      <p id="centered-title2"><b>End of Coding</b></p>
      <p id="centered-title2"><i>author -- izzat kz</i></p>
    </div>
    <div id="centered-title2">
      <a href="start.html" class="buttonRed">Rollback From Start</a>
    </div></div>
</body>
</html>
```

IMPLEMENTATION

The assignment has also requested a few required implementations as follows;

- 1) Input validation
- 2) Requirement logic;
 - Switch-case / if-else
 - Calculation
 - Function with a return value
- 3) Multiple files
- 4) Redux
- 5) Sticky form

INPUT VALIDATION

The page that applying the input validation are shipping destination form (shipping.php & shipping-check.php) and buyer's form (form.php & form-check.php). The two in two of these are sharing script. As below, shipping.php sharing script with shipping-check.php to send the field value so that shipping-check can validate and sending back error message.

```
shipping.php > html > head
<?php require("shipping-check.php") ?>

<html>
  <head>
    <title>Shipping Address</title>
```

shipping.php ↓

```
<form action="shipping.php" method="post" autocomplete="off">
  <div class="fieldAdd"><legend><b>Shipping Address</b></legend>

    <p><b>Address</b><b style="color:red;">*</b></p>
    <input type="text" name="shipping_address" placeholder="Address" size="75" value="<?php echo $shipping_address ?>">
    <p class="error addressErr">
      <?php echo $address_err ?>
    </p>

    <p><b>City</b><b style="color:red;">*</b></p>
    <input type="text" name="shipping_city" placeholder="City" size="13" value="<?php echo $shipping_city ?>">
    <p class="error cityErr">
      <?php echo $city_err ?>
    </p>

    <p><b>State</b><b style="color:red;">*</b></p>
    <select name="shipping_state">...
  </select>
  <p class="error stateErr">
    <?php echo $state_err ?>
  </p>

  <p><b>Postcode</b><b style="color:red;">*</b></p>
  <input type="number" name="shipping_postcode" placeholder="Postcode" size="6" value="<?php echo $shipping_postcode ?>">
  <p class="error postcodeErr">
    <?php echo $postcode_err ?>
  </p>

  <p><br><input type="checkbox" name="default"> <i>set address as default?</i></p>
  <p><button type="submit" class="button" name="confirmed">Confirm Address</button></p>
```

As you can see in above, the error message are echoed under every field. Each error message will executes each own validation. If there are no validation made yet, it wont show the message. Notice that the form action are directing to itself as the script from shipping.php and shipping-check.php can be executed at the same time. The shipping.php will contain the form and the shipping-check.php will contain all handling script for the form. It can be made into one page but with this, code may be organized and easy to maintenance.

```

$shipping_address = null;
$address_err = null;
$shipping_city = null;
$city_err = null;
$shipping_state = null;
$state_err = null;
$shipping_postcode = null;
$postcode_err = null;

if(isset($_POST['confirmed'])) {
    $shipping_address = $_POST['shipping_address'];
    $shipping_city = $_POST['shipping_city'];
    $shipping_state = $_POST['shipping_state'];
    $shipping_postcode = $_POST['shipping_postcode'];

    if(empty(trim($shipping_address))){
        return $address_err = "Address field is empty";
    } else{

        if(empty(trim($shipping_city))){
            return $city_err = "City field is empty";
        } else{

            if(!preg_match("/^[a-zA-Z\s]+$/", $shipping_city)){
                return $city_err = "City can only contain letters";
            } else {

                if($shipping_state=='null'){
                    return $state_err = "Select your state";
                } else{

                    if(empty(trim($shipping_postcode))){
                        return $postcode_err = "Postcode field is empty";
                    }else if (!preg_match("/^\d{5}$/", $shipping_postcode)) {
                        return $postcode_err = "Postcode should be a 5-digit number";
                    }

                }

                if(empty($address_err) && empty($city_err) && empty($state_err) && empty($postcode_err)) {

                    session_start();

                    $_SESSION['shipping_address'] = $shipping_address;
                    $_SESSION['shipping_city'] = $shipping_city;
                    $_SESSION['shipping_state'] = $shipping_state;
                    $_SESSION['shipping_postcode'] = $shipping_postcode;

                    echo '';
                }
            }
        }
    }
}

```

← shipping-check.php

This page would initialize all field and error variables. Using `if(isset)`, each variable will be filled with the value posted earlier. Next, it will use `if else` statements to validate the field. The process should go through one by one to make the form look neat even with error. The `if(empty)` is used to check if the field were empty, returning an error message. The `if(!preg_match)` are then used to check if the value field has any errors or not following the format for the field, also returning an error message. If everything is alright, all field values will be stored into each session.

These are the preview when the validation process are executed every click.

The image displays four overlapping screenshots of a 'Shipping Address' form, illustrating the validation process at different stages:

- First Screenshot (Leftmost):** Shows the form with all fields (Address, City, State, Postcode) empty. A red error message 'Address field is empty' is displayed below the Address field.
- Second Screenshot (Second from left):** Shows the form with 'No. 17 Jalan Inang 4/9' entered in the Address field and 'City' in the City field. A red error message 'City field is empty' is displayed below the City field.
- Third Screenshot (Third from left):** Shows the form with 'No. 17 Jalan Inang 4/9' in Address, '1231' in City, and 'Select A State' in State. A red error message 'City can only contain letters' is displayed below the City field.
- Fourth Screenshot (Rightmost):** Shows the form with 'No. 17 Jalan Inang 4/9' in Address, 'Cheras' in City, 'Select A State' in State, and '43200000' in Postcode. A red error message 'Postcode should be a 5-digit number' is displayed below the Postcode field.

Each form includes a 'Confirm Address' button and a checkbox labeled 'set address as default?'.

```

shipping-check.php > html

<?php
// ...

<html><head> <title>success</title>
<link rel="stylesheet" href="stylesheet.css">
<style>...
</head> <body>

<div id="bg">
  <div id="header2">NtahTime Watch Shop</div>
  <fieldset class="fieldAdd">
    <p style="font-size:larger;">Shipping Destination:<br /><br /></p>
    <p><b><?php echo $shipping_address . ', ' ?></b></p>
    <p><b><?php echo $shipping_city . ', ' . $shipping_state . ', ' ?></b></p>
    <p><b><?php echo $shipping_postcode ?></b><br /></p>
    <p class="success">Success!</p>
    <a href="shipping.php" class="buttonRed">Edit</a>    <a href="category.html" class="button">Next</a>
  </fieldset>
</div>
</body></html>

<?php
exit;
}

```

↑ shipping-check.php

Once all field has passed the validation and stored into session, a success message will show up along with the filled value to the user.



← shipping.php

Shipping Destination:

No.17 Jalan Inang 4/9,
Cheras, Selangor,
43200
 Success!
 Edit Next

The process are similar to form.php and form-check.php, these are the previews for buyer's form validation;

The image displays a sequence of overlapping screenshots of a web form titled 'Buyer's Information' and 'Type of Payment'. The form includes the following fields and validation messages:

- Type of Payment:**
 - Radio buttons for 'Cash On Delivery', 'Online Banking', and 'Credit/Debit Card'.
 - Error message: 'Please choose one of the payment types' (appears when no option is selected).
- Buyer's Information:**
 - Full Name*:** Text input field. Error message: 'Name field is empty' (appears when the field is empty).
 - Phone Number*:** Text input field with a placeholder '01XXXXXXXX'. Error messages: 'Phone number field is empty' (appears when empty), 'Wrong format number' (appears for invalid formats like 'dawood' or '0123').
 - Notes:** Text area.
 - Buttons:** 'Confirm' (green) and 'Cancel' (red).

The screenshots illustrate the validation process for each field, showing the error messages that appear when the input is invalid.

Code in form-check.php →

```
if(empty(trim($paymentType))) {
    return $paymentType_err = "Please choose one of the payment types";
} else {

if(empty(trim($buyerName))){
    return $buyerName_err = "Name field is empty";
} else if (preg_match('/\d/', $buyerName)) {
    return $buyerName_err = "Name cannot contain numbers";
} else{

if(empty(trim($phone))){
    return $phone_err = "Phone number field is empty";
} else if (!preg_match('/^01\d{5,13}$/ ', $phone)) {
    return $phone_err = "Wrong format number";
}
}
```

form.php →

Once all field passed the validation, the success message will show up along with submitted information for user to double check.

REQUIREMENT LOGIC

a) If-else statement

As shown in input validation, if-else were implemented in shipping-check.php and form-check.php to check the value form field value to validate and returning error message. Apart from that, if-else are also used in cart.php to declare cart array and to check item in cart for duplication. It is also exist in order-summary.php to recall the cart array from session. Apart from that, also implemented in product.php

```
if(isset($_SESSION["cart"])) {
    $cart = $_SESSION["cart"];
} else {
    $cart = array();
}
```

← cart.php & order-summary.php
(declaring cart array in session)

cart.php →

(checking cart to avoid any item duplication in cart)

```
// Check if item already exists in the cart
$item_index = -1;
foreach($cart as $index => $item) {
    if($item["product_name"] == $product_name) {
        $item_index = $index;
        break;
    }
}

// If item exists, update its quantity
if($item_index != -1) {
    $cart[$item_index]["product_quantity"] += $product_quantity;
} else {
    // If item does not exist, add it to the cart
    array_push($cart, $cart_item);
}

$_SESSION["cart"] = $cart;
}
```



```

if($category == 'All Products') {
    echo '<div id="content">';
    displayProducts('Watches For Him', $products);
    displayProducts('Watches For Her', $products);
    displayProducts('Accessories', $products);
    displayProducts('Clock', $products);
    echo '</div>';
} else {
    echo '<div id="content">';
    displayProducts($category, $products);
    echo '</div>';
}
?>

```

← product.php
(displaying products for each
different category)

b) Calculation

The calculation implemented on this application is just on cart.php to calculate the total item quantity in cart, and in order-summary.php to calculate the total price of an item and total price for all subtotal.

```

$total_quantity = 0;
foreach($cart as $item) {
    $total_quantity += $item["product_quantity"];
}

```

← cart.php (calculating total quantity)

order-summary.php→
(calculate the total
product of all items)

```

$total_price = 0;
foreach ($cart["product_price"] ?? [] as $pricekey => $price) {
    $total_price += $price * ($cart["product_quantity"][$pricekey] ?? 0);
}

```

```

<?php if (!empty($cart)): ?>
    <h3>Product List</h3>
    <?php
    foreach ($cart as $product) {
        $name = $product['product_name'];
        $price = $product['product_price'];
        $quantity = $product['product_quantity'];
        $subtotal = $price * $quantity;
        $total_price += $subtotal;
    }
?>

```

← order-summary.php
(calculating the subtotal and
total of subtotal)

c) Function with a return value

Function has been implemented in product.php to display product list array depending on category.

product.php →

(function to display different products from array for different categories)

```
function displayProducts($category, $products) {
    foreach($products[$category] as $product) {
        echo '<form method="post" action="cart.php">';
        echo '<div class="product">';
        echo '<div class="box">';
        echo '';
        echo '<h3>' . $product["name"] . '</h3>';
        echo '<p>RM ' . $product["price"] . '</p>';
        echo '<input type="hidden" name="product_name" value="' . $product['name'] . '">';
        echo '<input type="hidden" name="product_price" value="' . $product['price'] . '">';
        echo '<input type="hidden" name="product_picture" value="' . $product['picture'] . '">';
        echo '</div>';
        echo '<input type="number" name="product_quantity" value="1" min="1" max="30" class="quantity">';
        echo '</div>';
        echo '</form>';
    }
}
```

```
if($category == 'All Products') {
    echo '<div id="content">';
    displayProducts('Watches For Him', $products);
    displayProducts('Watches For Her', $products);
    displayProducts('Accessories', $products);
    displayProducts('Clock', $products);
    echo '</div>';
} else {
    echo '<div id="content">';
    displayProducts($category, $products);
    echo '</div>';
}
```

← product.php
(function called for displaying products for each different category)

MULTIPLE PAGE

This application has a total of 10 pages, excluding stylesheet.css

```

v project
  > image
  🐘 cart.php
  <> category.html
  🐘 form-check.php
  🐘 form.php
  <> gateway.html
  🐘 order-summary.php
  🐘 product.php
  🐘 shipping-check.php
  🐘 shipping.php
  <> start.html
  # stylesheet.css
```

REDUX

Redux occurs a lot in three pages which is cart.php, shipping-check.php, and in form-check.php. Occasionally exist when button were clicked for intended purposes. These are some of the examples;

```
if(isset($_POST["add-to-cart"])){
    $product_name = $_POST["product_name"];
    $product_price = $_POST["product_price"];
    $product_picture = $_POST["product_picture"];
    $product_quantity = $_POST["product_quantity"];

    $cart_item = array(
        "product_name" => $product_name,
        "product_price" => $product_price,
        "product_picture" => $product_picture,
        "product_quantity" => $product_quantity,
    );
}
```

← cart.php

(add to cart button clicked for retrieving the product's information from product.php)

```
if(isset($_POST["clear_cart"])){
    $cart = array();
    unset($_SESSION["cart"]);
    header("Location: cart.php");
    exit;
}
```

← cart.php

(clear cart button clicked for clearing cart and resetting cart array to zero)

STICKY FORM

This implementation is used on both form in shipping.php and form.php

```
<p><b>Address</b><b style="color:red;">*</b></p>
<input type="text" name="shipping_address" placeholder="Address" size="75" value="<?php echo $shipping_address ?>">
<p class="error addressErr">
<?php echo $address_err ?>
</p>

<p><b>City</b><b style="color:red;">*</b></p>
<input type="text" name="shipping_city" placeholder="City" size="13" value="<?php echo $shipping_city ?>">
<p class="error cityErr">
<?php echo $city_err ?>
</p>

<p><b>State</b><b style="color:red;">*</b></p>
<select name="shipping_state">...
<p class="error stateErr">
<?php echo $state_err ?>
</p>

<p><b>Postcode</b><b style="color:red;">*</b></p>
<input type="number" name="shipping_postcode" placeholder="Postcode" size="6" value="<?php echo $shipping_postcode ?>">
<p class="error postcodeErr">
<?php echo $postcode_err ?>
</p>

<p><br><input type="checkbox" name="default"> <i>set address as default?</i></p>
<p><button type="submit" class="button" name="confirmed">Confirm Address</button></p>
```

← shipping.php

form.php ↓

```
<fieldset class="field"><legend class="legend">Type of Payment</legend>
<br>
<p class="radio"><input type="radio" name="paymentType" value="Cash On Delivery" <?php if($paymentType === 'Cash On Delivery') echo 'checked'; ?>> Cash On Delivery</p>
<p class="radio"><input type="radio" name="paymentType" value="Online Banking" <?php if($paymentType === 'Online Banking') echo 'checked'; ?>> Online Banking</p>
<p class="radio"><input type="radio" name="paymentType" value="Credit/Debit Card" <?php if($paymentType === 'Credit/Debit Card') echo 'checked'; ?>> Credit/Debit Card</p>
<p class="error paymentTypeErr">
<?php echo $paymentType_err ?>
</p>
</fieldset>

<fieldset class="field"><legend class="legend">Buyer's Information</legend>
<br>
<p>Full Name<b style="color:red;">*</b></p>
<input type="text" name="buyerName" size="30" value="<?php echo $buyerName ?>">
<p class="error buyerNameErr">
<?php echo $buyerName_err ?>
</p>
<p>Phone Number<b style="color:red;">*</b></p>
<input type="tel" name="phone" size="20" value="<?php echo $phone ?>" placeholder="01XXXXXXXX">
<p class="error phoneErr">
<?php echo $phone_err ?>
</p>
<p>Notes</p>
<textarea name="notes"><?php echo $notes ?></textarea>
<p><button type="submit" name="confirmed" class="button">Confirm</button><span>
</span><a href="cart.php" class="buttonRed">Cancel</a></p>
</fieldset>
```

THANK YOU

I've took time and enjoyed a lot during the development of this application.