In class Exercise

Developing a WEBP3001 - E commerce website

Using XAMPP, MYSQL, PHP

Required Database Tables

Create a database called **shopping**. Add the following tables created for our e-commerce site with the following names:

- **`products`**—Stores information about the product that includes product name, weight, price, description, etc.
- **`productfeatures`**—Stores features of the products.
- **`cart`**—Stores information of the products selected in the cart.
- **`customers`**—Stores information about the registered customers.
- **`orders`**—Stores order numbers, order data, and shipping information of the customer who placed the order.
- `orders_details`—Stores the information about all the products that are purchased in a given order.
- **`payment_details`**—Stores the card number and other information about the payment mode selected to pay for any order.

Schema of tables

Products Table

Column	Type	Description
item_code	varchar(20)	Stores unique codes for a product.
item_name	varchar(150)	Stores the product's name.
brand_name	varchar(50)	Stores the brand name of the product.
model_number	varchar(30)	Stores the model number of the product.
weight	varchar(20)	Stores the weight of the product.
dimension	varchar(50)	Stores the dimension of the product.
description	text	Stores the description of the product.
category	varchar(50)	Stores the product category, i.e., whether the product belongs to the Smartphone, Laptop, or Books category.
quantity	SMALLINT	Stores the quantity in hand of the product. That is, the quantity of the product currently in the warehouse.
price	DECIMAL(7,2)	Stores the price of the product.
imagename	varchar(50)	Stores the path and name of the product image.

Productfeatures Tables

Column	Type	Description
item_code	varchar(20)	Stores a unique code for each product.
feature1/ feature2/ feature3/ feature4/ feature5/ feature6	varchar(255)	Stores the features of the product

Cart

Column	Type	Description
cart_sess	char(50)	Stores the session ID of the customer.
cart_itemcode	varchar(20)	Stores a unique code of the product selected by the customer in the cart.
cart_quantity	SMALLINT	Stores a quantity of the product selected in the cart.
cart_item_name	varchar(100)	Stores the name of the product that is selected in the cart.
cart_price	DECIMAL(7,2)	Stores the price of the product that is selected in the cart.

Customers

Column	Type	Description
email_address	varchar(50)	Stores the e-mail address of the registered customer. The e-mail address is considered to be unique for each customer
password	varchar(50)	Stores the password of the registered customer.
complete_name	varchar(50)	Stores the complete name of the registered customer.
address_line1	varchar(255)	Assuming the customer's address is big, this field stores the first address line of the registered customer.
address_line2	varchar(255)	Stores the second address line of the registered customer.
city	varchar(50)	Stores the city name to which the customer belongs.
state	varchar(50)	Stores the state to which the customer belongs.
zipcode	varchar(10)	Stores the ZIP code of the city to which the customer belongs.
country	varchar(50)	Stores the country name of the customer.
cellphone_no	varchar(15)	Stores the cell phone number of the registered customer.

Orders Table

Column	Type	Description
order_no	int(6)	Keeps the order number of the order placed by the customer. The order number is auto generated and is 1 plus the previous order number.
order_date	date	Stores the date on which the customer placed the order.
email_address	varchar(50)	Stores the e-mail address of the customer placing the order.
customer_name	varchar(50)	Stores the complete name of the customer placing the order.
shipping_address_line1	varchar(255)	Stores the first shipping address line where products have to be delivered.
shipping_address_line2	varchar(255)	Stores the second shipping address line where products have to be delivered.
shipping_city	varchar(50)	Stores the city name where products have to be delivered.
shipping_state	varchar(50)	Stores the shipping state.
shipping_country	varchar(50)	Stores the shipping country.
shipping_zipcode	varchar(10)	Stores the ZIP code of the region where products have to be delivered.

orders_details

Column	Type	Description
order_no	int(6)	Keeps the order number of the order placed by the customer.
item_code	varchar(20)	Stores the product code that is selected in the order.
item_name	varchar(100)	Stores the product name that is selected in the order.
quantity	SMALLINT	Stores the quantity of the product that is selected in the order.
price	DECIMAL(7,2)	Stores the price of the product that is selected in the order.

payment_details

Column	Type	Description
order_no	int(6)	Keeps the order number for whom payment is being made.
order_date	date	Keeps the date on which the given order was placed.
amount_paid	DECIMAL(7,2)	Stores the amount that is paid for the given order.
email_address	varchar(50)	Stores the e-mail address of the customer doing the payment.
customer_name	varchar(50)	Stores the name of the customer who is doing the payment.
payment_type	varchar(20)	Stores the mode of payment, i.e. whether the customer is paying through debit card, credit card, net banking, etc.
name_on_card	varchar(30)	Stores the name on the debit/credit card if the customer is paying with a card.
card_number	varchar(20)	Stores the credit card number.
expiration_date	varchar(10)	Stores the expiry date of the card (if customer is paying with a card).

SQL scripts

create database shopping; use shopping;	
(1)	(2)
create table products (item_code varchar(20) not null, item_name varchar(150) not null, brand_name varchar(50) not null, model_number varchar(30) not null, weight varchar(20), dimension varchar(50), description text, category varchar(50), quantity SMALLINT not null, price DECIMAL(7,2), imagename varchar(50));	create table productfeatures (item_code varchar(20) not null, feature1 varchar(255), feature2 varchar(255), feature3 varchar(255), feature4 varchar(255), feature5 varchar(255), feature6 varchar(255));
(3)	(4)
create table cart (cart_sess char(50) not null, cart_itemcode varchar(20) not null, cart_quantity SMALLINT not null, cart_item_name varchar(100), cart_price DECIMAL(7,2));	create table customers (email_address varchar(50) not null, password varchar(50) not null, complete_name varchar(50), address_line1 varchar(255), address_line2 varchar(255), city varchar(50), state varchar(50), zipcode varchar(10), country varchar(50), cellphone_no varchar(15), primary key(email_address));
(5)	(6)
create table orders (order_no int(6) not null auto_increment, order_date date, email_address varchar(50), customer_name varchar(50), shipping_address_line1 varchar(255), shipping_address_line2 varchar(255), shipping_city varchar(50), shipping_state varchar(50), shipping_country varchar(50), shipping_zipcode varchar(10), primary key (order_no));	create table orders_details (order_no int(6) not null, item_code varchar(20) not null, item_name varchar(100) not null, quantity SMALLINT not null, price DECIMAL(7,2));
(7)	
create table payment_details (order_no int(6) not null,	

```
order_date date,
amount_paid DECIMAL(7,2),
email_address varchar(50),
customer_name varchar(50),
payment_type varchar(20),
name_on_card varchar(30),
card_number varchar(20),
expiration_date varchar(10)
);
```

PHP

Sign up form

A sign-up form enables users to register on your site. A sign-up form usually prompts users to enter an e-mail address, password, complete name, address, cell phone number, etc. This information is then stored in a database for future use.
Once their data is stored in a database, users don't have to re-enter it. It will be automatically fetched upon a successful login.

Create a file signup.php

```
<html>
  <head>

  </head>
    <body>
      <form action="addcustomer.php" method="post">
        <table border="0" cellspacing="1" cellpadding="3">
          <tr><td colspan="2" align="center">Enter your information</td></tr>
          <tr><td>Email Address: </td><td><input size="20" type="text" name="emailaddress"></td></tr>
          <tr><td>Password: </td><td><input size="20" type="password" name="password"></td></tr>
          <tr><td>ReType Password: </td><td><input size="20" type="password" name="repassword"></td></tr>
          <tr><td>Complete Name </td><td><input size="50" type="text" name="complete_name"></td></tr>
          <tr><td>Address: </td><td><input size="80" type="text" name="address1"></td></tr>
          <tr><td></td><td><input size="80" type="text" name="address2"> </td></tr>
          <tr><td>City: </td><td><input size="30" type="text" name="city"></td></tr>
          <tr><td>State: </td><td><input size="30" type="text" name="state"></td></tr>
          <tr><td>Country: </td><td><input size="30" type="text" name="country"></td></tr>
          <tr><td>Zip Code: </td><td><input size="20" type="text" name="zipcode"></td></tr>
          <tr><td>Phone No: </td><td><input size="30" type="text" name="phone_no"></td></tr>
          <tr><td><input type="submit" name="submit" value="Submit"></td><td>
          <input type="reset" value="Cancel"></td></tr>
        </table>
      </form>
    </body>
</html>
```

Apply validation check to form

For providing correct input to your application, data must be validated. Data validation is the process of ensuring that data entered into a web form is correct and in the desired format. Data validation includes checking whether:

- Data is entered in the required fields. No essential field is left blank.
- No mistake is made when entering data. For example, no text is entered into a numerical field and vice versa.
- Data is entered in the desired format. For example, a date is entered in the required format.

To achieve this you will be using JavaScript to apply validation checks to the sign-up form. The PHP script, `validatesignup.php`

```
<html>
    <head>
        <script language="JavaScript" type="text/JavaScript" src="checkform.js">
        </script>
    </head>
    <body>
        <form action="addcustomer.php" method="post" onsubmit="return validate(this);">
            <table border="0" cellspacing="1" cellpadding="3">
                <tr><td colspan="2" align="center">Enter your information</td></tr>
                <tr><td>Email Address: </td><td><input size="20" type="text"
name="emailaddress" ><span id="emailmsg"></span></td></tr>
                <tr><td>Password: </td><td><input size="20" type="password" name="password"
><span id="passwdmsg"></span></td></tr>
                <tr><td>ReType Password: </td><td><input size="20" type="password"
name="repassword"><span id="repasswdmsg">
                </span></td></tr>
                <tr><td>Complete Name </td><td><input size="50" type="text"
name="complete_name" ><span id="usrmsg"></span></td></tr>
                <tr><td>Address: </td><td><input size="80" type="text"
name="address1"></td></tr>
                <tr><td></td><td><input size="80" type="text" name="address2"> </td></tr>
                <tr><td>City: </td><td><input size="30" type="text" name="city"></td></tr>
                <tr><td>State: </td><td><input size="30" type="text" name="state"></td></tr>
                <tr><td>Country: </td><td><input size="30" type="text"
name="country"></td></tr>
                <tr><td>Zip Code: </td><td><input size="20" type="text"
name="zipcode"></td></tr>
                <tr><td>Phone No: </td><td><input size="30" type="text"
name="phone_no"></td></tr>
                <tr><td><input type="submit" name="submit" value="Submit">    </td></tr>
                <input type="reset" value="Cancel"></td></tr>
            </table>
        </form>
    </body>
</html>
```

JavaScript is a programming language that is used for extending a web site's functionality by allowing for dynamic pages and implementing validation checks. A few of JavaScript's features are:

• It's a lightweight, interpreted programming language.
• It usually executes on the client machine, hence it consumes less server resources and avoids excessive server traffic.
• It's quite fast in delivering responses. Because it processes and executes on the client's machine, it delivers the response faster than other server-side scripting languages.

```
function validate(userForm) {
    div=document.getElementById("emailmsg"); // #1
    div.style.color="red"; // #2
    if(div.hasChildNodes()) // #3
    {
        div.removeChild(div.firstChild); // #4
    }
    regex=/(^\w+\@\w+\.\w+)/; // #5
    match=regex.exec(userForm.emailaddress.value);
    if(!match)
    {
        div.appendChild(document.createTextNode("Invalid Email")); // #6
        userForm.emailaddress.focus(); // #7
        return false; // #8
    }
    div=document.getElementById("passwdmsg");
    div.style.color="red";
    if(div.hasChildNodes())
    {
        div.removeChild(div.firstChild);
    }

    if(userForm.password.value.length <=5) // #9
    {
        div.appendChild(document.createTextNode("The password should be of at least size
6"));
        userForm.password.focus();
        return false;
    }
    div=document.getElementById("repasswdmsg");
    div.style.color="red";
    if(div.hasChildNodes())
    {
        div.removeChild(div.firstChild);
    }
    if(userForm.password.value != userForm.repassword.value) // #10
    {
        div.appendChild(document.createTextNode("The two passwords don't match"));
        userForm.password.focus();
        return false;
    }
    var div=document.getElementById("usrmsg");
    div.style.color="red";
    if(div.hasChildNodes())
    {
        div.removeChild(div.firstChild);
    }
```

```
    if(userForm.complete_name.value.length ==0) // #11
    {
        div.appendChild(document.createTextNode("Name cannot be blank"));
        userForm.complete_name.focus();
        return false;
    }
    return true;
}
```

Connect PHP with database (MYSQL)

To connect with a MySQL server, you need to execute the mysqli_connect() method with a valid username and password.

```
$variable = mysqli_connect("localhost", $user, $password, $database) or die ("Error
Message.");

$connect = mysqli_connect("localhost", "root", "", "shopping") or die
("Please, check the server connection.");
```

If successful, returns an object that represents the connection to a MySQL server and the specified database.

Executing SQL Commands Through PHP

After establishing the connection with the database, the next task is to execute the required SQL statement on it. For executing required SQL statements on the database, the mysqli_query method.

```
$result = mysqli_query($connect, $sql) or die(mysql_error());
```

The $connect variable represents the connection with the MySQL server and $sql represents the SQL statement that you want to execute on the connected database. The $result variable will store the result of the execution of the SQL statement.

The `checkconnection.php` Script Confirms if the Connection with the MySQL Server is Established

```
<?php
    // Connect to the database server
    $MySQLi = new MySQLi("localhost", "root", "gold", "shopping");
    if ($MySQLi->errno) {
        printf("Unable to connect to the database:<br /> %s", $MySQLi->error);
        exit();
        }
    else
```

```
        printf("Successfully connected with the MySQL server and shopping database is
opened");
?>
```

Storing Information in the Database Table

The PHP script for storing a new user's information in the underlying database table is shown below `addcustomer.php` . This PHP script saves the information entered by the user in the web form that was displayed through the `validatesignup.php` script into the customers table of the shopping database.

```
<?php
    $connect = mysqli_connect("localhost", "root", "", "shopping") or die
    ("Please, check the server connection.");
    $email_address = $_POST['emailaddress'];
    $password = $_POST['password'];
    $repassword = $_POST['repassword'];
    $completename = $_POST['complete_name'];
    $address1 = $_POST['address1'];
    $address2 = $_POST['address2'];
    $city = $_POST['city'];
    $state = $_POST['state'];
    $country = $_POST['country'];
    $zipcode = $_POST['zipcode'];
    $phone_no = $_POST['phone_no'];
    $sql = "INSERT INTO customers (email_address, password, complete_name,
        address_line1, address_line2, city, state, zipcode, country, cellphone_no)
    VALUES ('$email_address',(PASSWORD('$password')), '$completename', '$address1',
    '$address2','$city', '$state', '$zipcode', '$country', '$phone_no')";
    $result = mysqli_query($connect, $sql) or die(mysql_error());
    if ($result)
    {
        ?>
            <p>
            Dear, <?php echo $completename; ?> your account is successfully created
        <?php
    }
    else
    {
        echo "Some error occurred. Please use different email address";
    }
?>
```

Accessing Information from the Database

To access information from the database, the following four methods are used:

- `mysqli_num_rows()`—Returns the count of rows in a given recordset.
- `mysqli_affected_rows()`—Returns the count of rows affected by the specified SQL command.

- `mysqli_fetch_array()`—Returns one row at a time from the given recordset.

- `extract()`—Extracts the columns or fields in the specified row.

Implementing Authentication

Authenticating a user means determining whether the visitor is already registered on the e-commerce site or not.

1. The site has script that enables visitors to sign up and create an account.
   To verify that that the visitor is already registered, they will be provided with a sign-in form that will prompt them to enter a valid e-mail address and password.

2. After entering an e-mail address and password, clicking the Submit button in the sign-in form, calls another script that accesses the customers table and confirms if any customer (row) exists with the supplied e-mail address and password.
   If a customer exists with the specified e-mail address and password, it means the visitor is already registered to your site and a welcome message will be displayed on the screen. If no row exists in the customers table with the supplied e-mail address and password, it means either the visitor is not registered to your site or has entered the wrong information. Hence, the visitor is provided two links to choose from—one will navigate them to create a new account and the other will allow them to try to sign in again.

`signin.php` will implement step 1 of above behavior.

```
<html>
       <head>
       </head>
               <body>
                   <form action="validateuser.php" method="post">
                       <table border="0" cellspacing="1" cellpadding="3">
                           <tr><td>Email Aaddress:</td><td><input type="text"
       name="emailaddress"></td></tr>
                           <tr><td>Password:</td><td><input type="password" name="password">
                           </td></tr>
                           <tr><td colspan=2 align="center"><input type="submit"
       name="submit" value="Login"></td></tr>
                       </table>
                   </form>
        </body>
</html>
```

validateuser.php performs the second step of authentication—it verifies whether the information entered by the visitor is valid.

```
<html>
   <head>
   </head>
        <body>
        <?php
            $connect = mysqli_connect("localhost", "root", "", "shopping") or
            die("Please, check your server connection.");
            $query = "SELECT email_address, password, complete_name FROM customers
            WHERE email_address like '" . $_POST['emailaddress'] . "' " .
                "AND password like (PASSWORD('" . $_POST['password'] . "'))";
            $result = mysqli_query($connect, $query) or die(mysql_error());
            if (mysqli_num_rows($result) == 1) {
                while ($row = mysqli_fetch_array($result, MYSQLI_ASSOC)) {
                    extract($row);
                    echo "Welcome " . $complete_name . " to our WEBP3001 Mall <br>";
                }
        }
        else {
        ?>
            Invalid Email address and/or Password<br>
            Not registered?
            <a href="validatesignup.php">Click here</a> to register.<br><br><br>
            Want to Try again<br>
            <a href="signin.php">Click here</a> to try login again.<br>
            <?php
        }
        ?>
        </body>
</html>
```

Accessing the Database Using PHP

Accessing Products and Displaying Them on Screen

allitemslist.php script for Displaying Items in the products Table

```
<html>
<head>
</head>
   <body>
        <?php
            $connect = mysqli_connect("localhost", "root", "", "shopping") or
            die("Please, check your server connection.");
            $query = "SELECT item_code, item_name, description, imagename, price FROM
            products";
```

```
            $results = mysqli_query($connect, $query) or die(mysqli_error($connect));
            echo "<table border=\"0\">";
            $x=1;
            echo "<tr>";
            while ($row = mysqli_fetch_array($results, MYSQLI_ASSOC)) {
                if ($x <= 3)
                {
                    $x = $x + 1;
                    extract($row);
                    echo "<td style=\"padding-right:15px;\">";
                    echo "<a href=itemdetails.php?itemcode=$item_code>";
                    echo '<img src=' . $imagename . ' style="max-width:220px;max-
        height:240px;
                    width:auto;height:auto;"></img><br/>';
                    echo $item_name .'<br/>';
                    echo "</a>";
                    echo '$'.$price .'<br/>';
                    echo "</td>";
                }
                else
                {
                    $x=1;
                    echo "</tr><tr>";
                }
            }
            echo "</table>";
        ?>
    </body>
</html>
```

Creating a Drop-Down Menu

Assuming the WEBP3001 site sells electronics, home and furniture products, and books, the drop-down menu needs to provide links to these product sections. Also, assuming that the site sells smart phones, laptops, cameras, and televisions, these product categories must be grouped under the Electronics section. The **`menu.php`** file.

```
        <!DOCTYPE html>
        <head>
        <meta charset="utf-8">
        <title>WEBP3001 Online Store</title>
        <link rel="stylesheet" href="css/style.css">
        </head>
        <body>
            <div class="container">
            <nav>
```

```
                <ul class="nav">
                    <li><a href="index.php">Home</a></li>
                    <li class="dropdown">
                <a href="index.php">Electronics</a>
                <ul>
                    <li><a href="itemlist.php?category=CellPhone">Smart
        Phones</a></li>
                    <li><a href="itemlist.php?category=Laptop">Laptops</a></li>
                    <li><a href="index.php">Cameras </a></li>
                    <li><a href="index.php">Televisions</a></li>
                </ul>
                </li>
                <li class="dropdown">
                <a href="index.php">Home & Furniture</a>
                <ul class="large">
                    <li><a href="index.php">Kitchen Essentials</a></li>
                    <li><a href="index.php">Bath Essentials</a></li>
                    <li><a href="index.php">Furniture</a></li>
                    <li><a href="index.php">Dining & Serving</a></li>
                    <li><a href="index.php">Cookware</a></li>
                </ul>
                </li>
                <li><a href="index.php">Books</a></li>
                </ul>
            </nav>
            </div>
    <p>
```

To apply foreground and background colors to the drop-down menu and to the menu option when a mouse pointer hovers over it, the script uses a cascading style sheet called **`style.css`**.

```
img {
    max-width:180px;
    max-height:200px;
    width:auto;
    height:auto;
    }
ol, ul {
    list-style: none;
    }
    nav {
    height: 30px;
    border-bottom: 5px solid white;
    }
.nav {
    margin: 0 auto;
    width: 600px;
    }
.nav a {
    display: block;
    text-decoration: none;
    }
```

```
.nav > li {
        float: left;
        margin-right: 5px;
        }
.nav > li > a {
        height: 34px;
        line-height: 34px;
        padding: 0 20px;
        font-weight: bold;
        color: white;
        text-decoration: none;
        border-radius: 3px 3px 0 0;
        background-color: blue;
        }
.nav > li > a:hover {
        text-decoration: none;
        background: blue;
        background-color: navy;
        }
.nav > li.active > a, .nav > li > a:active, .nav > .dropdown:hover > a {
        background: white;
        color: blue;
        }
        .dropdown {
        position: relative;
        border-bottom: 5px solid white;
        }
        .dropdown:hover ul {
        display: block;
        }
        .dropdown ul {
        display: none;
        position: absolute;
        top: 39px;
        left: -1px;
        z-index: 20000;
        min-width: 160px;
        padding: 0 0 5px;
        background: blue;
        border: 1px solid #dadada;

        border-top: 0;
border-radius: 0 0 3px 3px;
}
.dropdown ul.large {
min-width: 200px;
}
.dropdown li {
display: block;
margin: 0 18px;
overflow: visible;
}
.dropdown li + li {
border-top: 1px solid #eee;
}
.dropdown li a {
color: #FFF;
padding: 8px 18px;
margin: 0 -18px;
}
```

```
.dropdown li a:hover {
background: navy;
}
```

The CSS code gives the drop-down menu a dynamic look.

Sets the height and width of the images

- Hides the list-item markers (circle, square, etc.) for the ordered and unordered lists
- Sets the height and bottom borders of the drop-down menu
- Sets the width and margins of the drop-down menu
- Removes the default underlines from the hyperlinks displayed in the menu and submenus
- Sets the list items, i.e. the submenu options, to float to the left in the drop-down menu, keeping the margin on the right
- Sets the height, padding of the text (from the boundary of the drop-down), foreground, and background color of the text, and removes the default underlines from the hyperlinks
- Sets the background and foreground color of the text when the mouse hovers over the menu and submenu options

The **itemlist.php** script for displaying products whose category is selected from the drop-down menu

```
<?php
include('menu.php');
$connect = mysqli_connect("localhost", "root", "gold", "shopping") or
die("Please, check your server connection.");
$category=$_REQUEST['category'];
$query = "SELECT item_code, item_name, description, imagename, price FROM
products " .
"where category like '$category' order by item_code";
$results = mysqli_query($connect, $query) or die(mysqli_error($connect));
echo "<table border=\"0\">";
$x=1;
echo "<tr>";
while ($row = mysqli_fetch_array($results, MYSQLI_ASSOC)) {
    if ($x <= 3) // #1
    {
    $x = $x + 1;
    extract($row);
    echo "<td style=\"padding-right:15px;\">";
```

```
    echo "<a href=itemdetails.php?itemcode=$item_code>"; // #2
    echo '<img src=' . $imagename . ' style="max-width:220px;max-height:240px;
    width:auto;height:auto;"></img><br/>';
    echo $item_name .'<br/>';
    echo "</a>";
    echo '$'.$price .'<br/>';
    echo "</td>";
    }
    else
    {
    $x=1;
    echo "</tr><tr>";
    }
    }
    echo "</table>";
    ?>
    </body>
    </html>
```

At the top of this file , menu.php is included to make the dropdown menu appear in the web site. Recall that the menu.php displays different categories of products. When a user selects a category, navigation takes place to the `itemlist.php` file and the selected product category is also passed.

Adding the Web Site Header

The *`topmenu.php`* script for displaying headers for the e-commerce site, including the drop-down menu

```
<!DOCTYPE html>
<head>
<meta charset="utf-8">
<title>WEBP3001 Online Store</title>
<link rel="stylesheet" href="css/style.css">
</head>
<body>
<table width="100%" cellspacing="0" cellpadding="2">
<col style="width:30%">
<col style="width:40%">
<col style="width:20%">
<tr><td style="background-color:cyan;color:Blue;"></td><td
style="background-color:cyan;color:Blue;"></td><td style="backgroundcolor:cyan;color:Blue;">
<tr><td style="font-size: 35px;color:blue;background-color:cyan;"><!-- #1 -->
<b>Bintu Online Bazar</b></font></td>
<td bgcolor="cyan">
<form method="post" action="searchitems.php"> <!-- #2 -->
<input size="50" type="text" name="tosearch">
<input type="submit" name="submit" value="Search">
</form></td>
<td bgcolor="cyan" ><a href="cart.php"><img style="max-width:40px;
max-height:40px;width:auto;height:auto;" src="images/cart.png"></img>
<span id="cartcountinfo"></span></a><!-- #3 -->
</td></tr>
</table>
```

```
<div class="container">
<nav>
<ul class="nav">
<li><a href="index.php">Home</a></li>
<li class="dropdown">
<a href="index.php">Electronics</a>
<ul>
<li><a href="itemlist.php?category=CellPhone">Smart Phones</a></li>
<li><a href="itemlist.php?category=Laptop">Laptops</a></li>
<li><a href="index.php">Cameras </a></li>
<li><a href="index.php">Televisions</a></li>
</ul>
</li>
<li class="dropdown">
<a href="index.php">Home & Furniture</a>
<ul class="large">
<li><a href="index.php">Kitchen Essentials</a></li>
<li><a href="index.php">Bath Essentials</a></li>
<li><a href="index.php">Furniture</a></li>
<li><a href="index.php">Dining & Serving</a></li>
<li><a href="index.php">Cookware</a></li>
</ul>
</li>
<li><a href="index.php">Books</a></li>
</ul>
</nav>
</div>
<p>
```

Implementing a Search Feature

Using the Search box to the web site header users can enter text in the Search box and then click the Search button. The entire products table will be searched for the specified text and all the products that contain that text (whether in their name, description, features, and so on) will be displayed on the screen.

`searchitems.php script`

```
<?php
include('topmenu.php');
$connect = mysqli_connect("localhost", "root", "", "shopping") or
die("Please, check your server connection.");
$tosearch=$_POST['tosearch'];
$query = "select * from products where ";
$query_fields = Array();
$sql = "SHOW COLUMNS FROM products"; // #1
$columnlist = mysqli_query($connect, $sql) or die(mysql_error()); // #2
while($arr = mysqli_fetch_array($columnlist, MYSQLI_ASSOC)){ // #3
extract($arr);
$query_fields[] = $Field . " like('%". $tosearch . "%')";
}
$query .= implode(" OR ", $query_fields);
$results = mysqli_query($connect, $query) or die(mysqli_error($connect));
echo "<table border=\"0\" >";
$x=1;
echo "<tr>";
```

```
while ($row = mysqli_fetch_array($results, MYSQLI_ASSOC)) {
if ($x <= 3)
{
$x = $x + 1;
extract($row);
echo "<td style=\"padding-right:15px;\">";
echo "<a href=itemdetails.php?itemcode=$item_code>";
echo '<img src=' . $imagename . ' style="max-width:220px;max-height:240px;
width:auto;height:auto;"></img><br/>';
echo $item_name .'<br/>';
echo "</a>";
echo '$'.$price .'<br/>';
echo "</td>";
}
else
{
$x=1;
echo "</tr><tr>";
}
}
echo "</table>";
?>
```

Elements to be completed for WEBP3001 online shopping ecommerce site.

- Show Product Details
- Session Handling
- Signing in and out
- Home Page of site
- Manage Shopping cart
    - Save items in cart
    - Display cart count using AJAX
    - Check Out
    - Payments