SUBMISSION DATE:
20TH JUNE 2021

SIS2026Y WEBSITE ASSSIGNEMENT

SamShai Italian Restaurant

PREPARED BY:

Shaimah Beebee Bundhoo - 1914779
Sambhav Bholah - 1913983

LECTURER

Mr. Yasser Chuttur

Contents

Table of Figures

Work Distribution

Content	**Sambhav Bholah**	**Shaimah Beebee Bundhoo**
Page Layout Designs	Design mock-up on Photoshop	Searched & tested different color palettes
Information Gathering	Insert all products data in mySQL	Search relevant product's details
index.php	PHP for basic routing	PHP for basic routing
including files from index.php	Separate header from homepage into header.php	Separate footer from homepage into footer.php
Homepage	HTML, CSS, JavaScript for Image Slider	CSS for Services section and JavaScript for Navigation
Products	HTML, CSS, PHP to show products	PHP to retrieve all products data
Product Details	PHP to display only relevant details using POST method	HTML, CSS
Cart	HTML, add to cart using PHP and sessions	CSS, calculation using PHP and sessions
Payment	Forms using POST, JavaScript for validation	HTML, CSS, PHP to insert
Sign up	HTML, CSS, PHP validation	PHP to insert values
Log In	HTML, CSS, PHP validation	PHP to retrieve values
Log out/Change password	PHP to end sessions	PHP to update values
Contact us	HTML, CSS	AJAX
About us	Ajax	HTML, CSS
Frequently Asked Questions	HTML, CSS	JavaScript for dropdown
Review & Rating	HTML, CSS, AJAX, Database	JavaScript star rating
Admin: Input new products	PHP to insert	HTML, CSS

1.0 Introduction

1.1 Purpose of website

A website for a restaurant is created named SamShai. Having a website of its own, SamShai can reduce cost on advertising such as printing of ads and can also give key information on its location, menu, opening times and special offers. For example, SamShai organizes an open Buffet every Saturday as from 5 pm, therefore, SamShai only needs to post precise details instead of printing leaflets which may lack important information. In this way, it also improves awareness of the business. Moreover, it can greatly improve trust and loyalty as the website also provides customers with a 5-star rating to rate their experience with SamShai.

1.2 Flow of website according to web pages

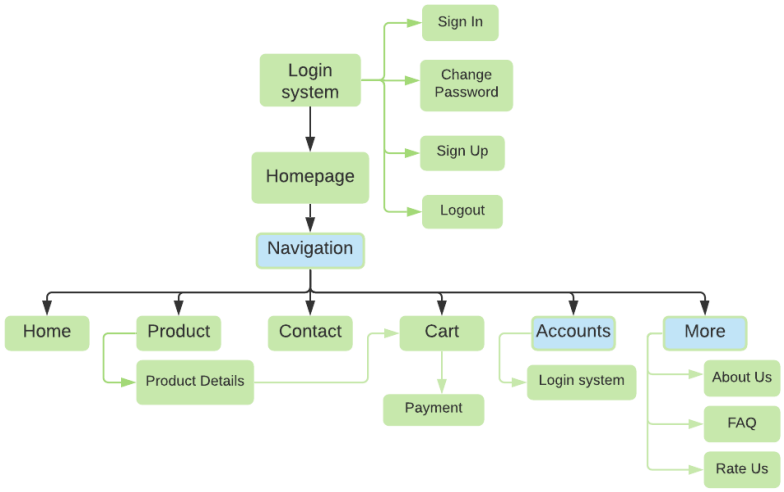


Figure 1Flow of website according to webpages

The user first needs to login in order to place an order. The login system also allows the user to create an account, change password and also log into an existing account. After logging in, he is directed to the homepage where he can also log out. To make a purchase, the user can then browse through the product page where he can view product details for each product and add to cart the product and the quantity he wishes to purchase. The user then reviews his cart details and proceeds to checkout followed by a payment form to be filled.

Furthermore, the user can also view the about us page to know more on SamShai and visit the FAQ page to get an insight of the purchase system and also the user can rate their experience of purchase on SamShai's website. A contact page is therefore created for users to write us a message personally which SamShai receives by mail can reply instantly.

2.0 HTML 5, Navigation, CSS 3 Layout and Design

2.1 Navigation

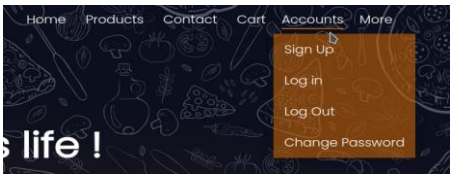
Figure 2Navigation bar

The header contains a logo on the far left, and the navigation on the right side.

There is a hover effect when hovering over the links.

Clicking the links, will redirect the user to the respective page.

Figure 3Navigation bar hover effect

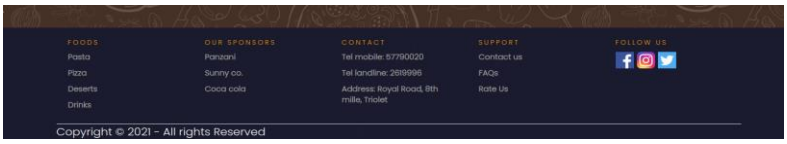


The *Accounts* and *More* sections have a dropdown navigation to get more options.

The navigation is very dynamic and displays all the necessary features that a user need.

Figure 4Dropdown Navigation

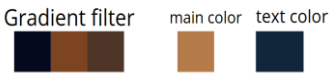
2.2 Footer



The footer contains the different categories, sponsors, more contact information and links to our social media pages.

Figure 5Footer

2.3 Color Palettes



The main color is the orange which symbolizes the orange color of pasta. To provide a good contrast and ease the user's reading, a dark blue is used for text.

Figure 6Color palette

```
body{
    font-family: 'Poppins', sans-serif;
    overflow-x: hidden;
    background-image:  linear-gradient(186deg, rgba(4,9,30,0.7) 10%, rgba(125,68,33,0.7) 46%,
     rgba(77, 54, 40, 0.85) 80%),url("../images/backgrounds/background.jpg");
}
```

The same background image which is a seamless pattern of pizza and pasta is used everywhere besides the sign up and login pages. A font style is also set to be used again.

Figure 7CSS for color gradient

A linear-gradient is applied to the image to provide a more orange tone and friendly appearance. The gradient becoming more orange encourages the user to continue to scroll down upon the page.

Before gradient filter

After gradient filter

2.4 Design Principles used

Repetition - The same color palette is used through all the pages. The nav bar is present through each page as well. The orange tone is present in all the div elements used.

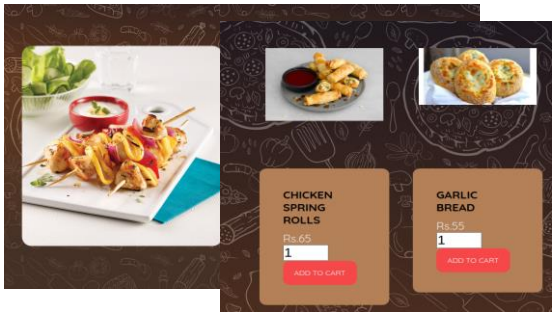


Figure 8Product details

Figure 9Product page

Contrast - A good amount of contrast is shown above where the product info and product image are very much to the front whereby the background has a dark tone to it.
Alignment - All the testimonials are equally aligned next to each other

Proximity - In figure 10, an equal spacing is used and white space serves as the barrier between them. Also, the line spacing used for the text eases the user's eyes while reading.

Figure 10Homepage

3.0 Implementation of JavaScript and/or JQuery

3.1 Dropdown in FAQ

JavaScript has been used for the dropdown in the FAQ page. As shown in the figure below, a '+' sign is seen. However, when the user clicks on it a dropdown occurs which allows the '+' sign to become '-' sign.

Figure 11FAQ dropdown

```
<script>
    $(document).ready(function(){
      $(".card-header").click(function(){
       // self clicking close
       if($(this).next(".card-body").hasClass("active")){
         $(this).next(".card-body").removeClass("active").slideUp()
        $(this).children("span").removeClass("fa-minus").addClass("fa-plus")
       }
      else{
        $(".card .card-body").removeClass("active").slideUp()
        $(".card .card-header span").removeClass("fa-minus").addClass("fa-plus");
        $(this).next(".card-body").addClass("active").slideDown()
         $(this).children("span").removeClass("fa-plus").addClass("fa-minus")
      }
      })
    })

</script>
```

Figure 12JavaScript for dropdown

The code for the JavaScript is as shown in figure 12.

3.2 In Star Review Rating in Review page

JavaScript has been used for the 5-star rating system to function appropriately. For example, the mouseover effect, mouseout effect, the value of the stars, the reaction upon clicks on the star. In the figure below is the snippet of the mouseover and mouseout effect implemented by JavaScript.

```
this.$el.on('mouseover.starrr', 'a', (function(_this) {
  return function(e) {
    return _this.syncRating(_this.getStars().index(e.currentTarget) + 1);
  };
})(this));
this.$el.on('mouseout.starrr', (function(_this) {
  return function() {
    return _this.syncRating();
  };
})(this));
```

Figure 13JavaScript for review rating

3.3 Validation of Email in Contact Form

In a contact, the user is required to insert his username and email address followed by the message he wishes to send. However, the user may make a mistake in typing the right format of the email address. Therefore, a validation of email is inserted in such a way that the user gets an error message if the format is wrong. For example, if '@' is missing, the email address is considered wrong therefore an error message is displayed.

```
var x=document.contact.email.value;
var atposition=x.indexOf("@");
var dotposition=x.lastIndexOf(".");
if (atposition<1 || dotposition<atposition+2 || dotposition+2>=x.length){
    alert("Please enter a valid e-mail address");
    return false;
```

Figure 14JavaScript validation for email address

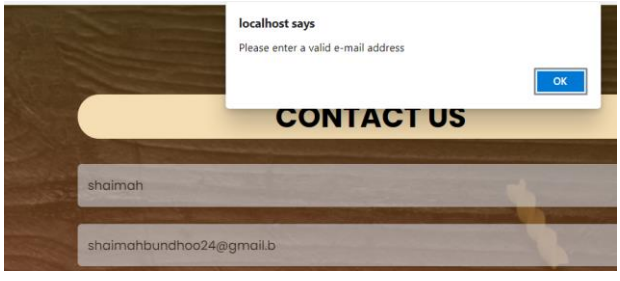


Figure 15Validation result

3.4 Validation for No Empty Fields in Payment Form

In order to prevent the user from submitting the form with empty fields, we inserted a JavaScript validation as follows:

```
<script>
    function validateForm() {
    var a = document.forms["payment"]["name"].value;
    var b = document.forms["payment"]["phone"].value;
    var c = document.forms["payment"]["address"].value;
    var d = document.forms["payment"]["cardname"].value;
    var e = document.forms["payment"]["cardno"].value;
    var f = document.forms["payment"]["exdate"].value;
    var g = document.forms["payment"]["ccv"].value;

    if (a == "") || (b = "") || (c = "") || (d = "") || (e = "") || (f = "") || (g = "") {
    alert("Fields shoul be filled");
    return false;
    }
}
</script>
```

Figure 16JavaScript validation for no empty fields

3.5 Slider in Homepage

```
<script type="text/javascript">
  var slides = document.querySelectorAll('.slide'); //fetch the variables
  var btns = document.querySelectorAll('.btn');
  let currentSlide = 1; //initialise it at the first slide

  // manual navigation by pressing button
  var manualNav = function(manual){
    slides.forEach((slide) => {
      slide.classList.remove('active');

      btns.forEach((btn) => {
        btn.classList.remove('active');
      });
    });

    slides[manual].classList.add('active');
    btns[manual].classList.add('active');
  }

  btns.forEach((btn, i) => {
    btn.addEventListener("click", () => {
      manualNav(i);
      currentSlide = i; //changes to the next slide
    });
  });
```

The slide in the homepage is done using a recursive function. After each button click, the function is called again with the next slide number. Hence, the slides are changed.

The button has an EventListener which triggers after a click.

Figure 17JavaScript slider in homepage

3.6 Responsiveness of Navigation

```
<script>
    var navLinks = document.getElementById("navLinks");
    function showMenu(){
        navLinks.style.right = "0";
    }

    function hideMenu(){
        navLinks.style.right = "-200px";
    }
</script>
```

This is used to hide the navigation bar while the screen size is relatively small. After clicking the dropdown icon, the function showMenu/hideMenu is called, which then changes the alignment by -200 px. Hence the full menu is shown in figure 18.

Figure 18JavaScript responsiveness for navigation

4.0 Forms

4.1 Login and Registration	The login and registration are 2 forms that work in parallel. That is, a new user creates an account using the registration form and the credentials are saved in a database. Then the same table in the same database is used for the user to login again with the same credentials.
4.2 Change password form	After the user logs in, he can change his password and the password is then updated in the database of the same table used by login and registration.
4.3 Contact Form	With the contact form, the user can write a message and submit it which SamShai will then receive by mail and can reply by mail itself.
4.4 Admin Form	An admin form, which is accessible after logging in, is created for the admin to add products into the database.
4.5 Payment Form	After adding to cart, to proceed with payment, a payment is created to which the details are then stored in a database.
4.6 Review Form	A review form is created with a 5-star rating where a user can add his review which is first stored in the database and then displayed from the database.

4.7 Form Format

All forms are all in same format as shown below:

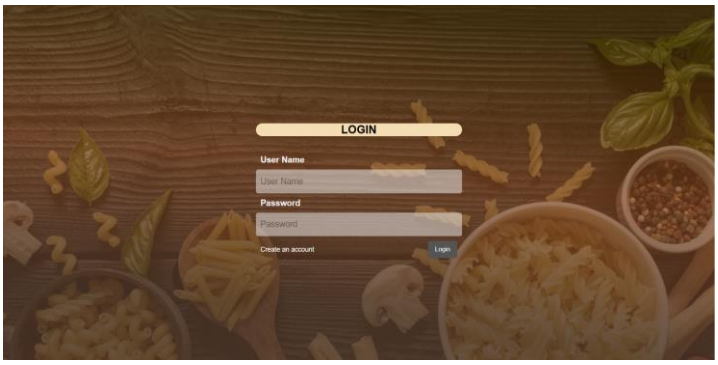


Figure 19Form format

5.0 PHP as backend services

5.1 Proper Handling of User Registration/Login

For the login, registration and change password, we used PHP to implement the proper handling and form action to connect the form with its PHP backend as shown in the figure below:

```
<form name="form" action="signup-check.php" method="post">
```

Figure 20Form action

Moreover, we implemented error handling in the form where an error message is displayed. The error message is displayed on top of the form on this location:

```
<?php if (isset($_GET['error'])) { ?>
    <p class="error"><?php echo $_GET['error']; ?></p>
<?php } ?>

<?php if (isset($_GET['success'])) { ?>
    <p class="success"><?php echo $_GET['success']; ?></p>
<?php } ?>
```

The error message is displayed using echo in this space. For an error message to be displayed, an error handling must be implemented for some situations like:

Figure 21Error message using php

1. If the fields for username, name, password and/or confirmation password remain empty an error message is displayed to inform the user that the fields are required.
2. Even if a user inserts a password and it does not match with the confirmation, an error message is displayed to indicate the password does not match.

```
36    else if(empty($name)){
37        header("Location: signup.php?error=Name is required&$user_data");
38        exit();
39    }
40
41    else if($pass !== $re_pass){
42        header("Location: signup.php?error=The confirmation password  does not match&$user_data");
43        exit();
44    }
```

Figure 22Validation for matching password

Figure 23Login Form with validation result

According to the user registration, name, username, password and confirm password. The input of variable “name” and “username” is done as shown in the figure on the right where the system is connected to a database and when a user creates an account with an existing username or name an error message is displayed as shown in the figure on the right:

```
<label>Name</label>
<?php if (isset($_GET['name'])) { ?>
    <input type="text"
           name="name"
           placeholder="Name"
           value="<?php echo $_GET['name']; ?>"><br>
<?php }else{ ?>
    <input type="text"
           name="name"
           placeholder="Name"><br>
<?php }?>
```

Figure 24validation for existing username

5.2 isset()

```
if(isset($_POST['name']) && isset($_POST['phone'])
&& isset($_POST['address']) && isset($_POST['cardname'])
&& isset($_POST['cardno']) && isset($_POST['exdate']) && isset($_POST['ccv']) && isset($_POST['total'])) {
```

Figure 25Use of isset

The isset() function is used to check whether a variable is set or not. Following its else statement, it will output alert messages so that the user knows if he missed a text field.

5.3 Use of sessions

Sessions are used to hold temporary values and forward them to other forms. For example, the session created in *cart.php* is used again in the *payment.php*. A session is started in the *index.php.*

The array *$product* stores the product details received from by clicking the Add to cart. Only the ID and product quantity bought are received through a POST method. The remaining product details are fetched by a SQL query performed with the same product ID.

```php
//product details assigned to an array
while($row = $result->fetch_assoc()) {
    $products = array(
        'id'            =>  $product_id,
        'name'          => $row['name'],
        'quantity'      =>  $quantity,
        'stock'         => $row['stock'],
        'price'         =>  $row['price'],
        'img'           =>  $row['path']
    );
}
```

Figure 26Store product details in array

The session *shopping_cart* stores the count (which serves as an index) and the array product (see figure 27). If the cart is created for the first time, it initialises it at 0.

Else, each session will hold the count and the product array. This facilitates the add to cart process.

```php
//check if session already exists
if(isset($_SESSION["shopping_cart"])){

    //get the index of the array
    $productArrayID = array_column($_SESSION["shopping_cart"],"id");

    //check if the index already is in the array, then does not add the item again
    if(!in_array($product_id, $productArrayID))
    {
        //adds item for the first time
        $count = count($_SESSION["shopping_cart"]);
        $_SESSION["shopping_cart"][$count] = $products;
    }
    else{
        echo '<script>alert("Item Already Added")</script>';
    }
}
else{
    //session created for the first time
    $_SESSION["shopping_cart"][0] = $products;
}
```

Figure 27Insert into session

```php
if(isset($_GET["action"])){
    if($_GET["action"] == "delete"){
        foreach($_SESSION["shopping_cart"] as $keys => $values){
            if($values["id"] == $_GET["id"])
            {
                unset($_SESSION["shopping_cart"][$keys]);
                echo '<script>alert("Item Removed")</script>';
                echo '<script>window.location="index.php?page=cart"</script>';
                // session_start();
            }
        }
    }
}
```

Unset is used to remove a session alongside its content. It used to remove a product from the cart.

Figure 28Use of unsert(session)

```php
<?php $total = 0;
foreach($_SESSION["shopping_cart"] as $keys => $values)
{
 ?>
    <tr>
        <td><?php echo $values['name']; ?></td>
        <td>Rs. <?php echo $values['price']; ?></td>
        <td><?php echo $values['quantity']; ?></td>
        <?php $finalprice = $values['quantity'] * $values['price'] ?>
        <td>Rs. <?php echo $finalprice; ?></td>
        <td><a href="cart.php?action=delete&id=<?php echo $values["id"]; ?>
    </tr>
```

Here, the same session is called again, and is looped. *$value contains* each individual item. *$values*[‘price’] will output the price.

Figure 29Calling shopping_cart session

After *cart.php*, *payment.php* is used. A session is started and the shopping_cart is called again to update the database. Finally, session_destroy() is used to eliminate the current session, and prepare a new one for the payment.

Figure 30Use of session_destroy()

5.4 Product viewing from Database

```php
<?php
    $sql = "SELECT * FROM products WHERE cat = 'Starter' ORDER BY id ASC";
    $result = $conn->query($sql);
    if(mysqli_num_rows($result) > 0)
    {
    while($row = $result->fetch_assoc()) {
?>
```

The query will select all the products where the category is starter. The while loop is used to select all the products.

Figure 31Retrieve starter category of products

```php
<div class="product-content">
    <div class="product-img">
        <?php $imageURL = 'uploads/'.$row["path"];?>
        <img src="<?php echo $imageURL; ?>" alt="product image">
    </div>
</div>

<div class="product-info">
    <a href = "index.php?page=productDetails&id=<?=$row['id']?>" class="product-name"><?php echo  $row["name"] ?></a>
    <p class="product-price"> Rs.<?php echo  $row["price"] ?></p>

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

Figure 32Displaying product details

After the while loop is open, it will repeat the divs and all the elements inside according to the number of values received from the query. *$row*[“path”] will return the image path from the database, and then it is concatenated with “uploads/” to get the full path. Similarly, all the product's details are retrieved in this way. This prevents writing multiple divs and instead loop only one according to the number of items.

5.5 Insert, Update, Retrieve from Database

5.5.1 Insert

```php
if (isset($_POST['upload'])) {

    $filename = $_FILES["uploadfile"]["name"];

    $pid = $_POST['pid'];
    $pname = $_POST['pname'];
    $pprice = $_POST['pprice'];
    $pstock = $_POST['pstock'];
    $pcat = $_POST['pcat'];
    $pinfo = $_POST['pinfo'];


    $db = mysqli_connect("localhost", "root", "", "samshai");

        $sql = "INSERT INTO products VALUES ('$pid', '$pname', '$pprice', '$pstock', '$filename', '$pcat', '$pinfo')";
        // Execute query
        mysqli_query($db, $sql);
        header("Location: productInput.html");
}
```

The admin form is used to insert new products in the database. After the data are received through a POST method, they are stored in variables. The INSERT statement passes the variables and the *mysqli_query* performs the query.
Insert statements have also been used in the sign-up form (to insert new customers) and the payment form (to insert new orders)

Figure 33Insert product details

5.5.2 Update

```php
$sql = "SELECT password
        FROM users WHERE
        id='$id' AND password='$op'";
$result = mysqli_query($conn, $sql);
if(mysqli_num_rows($result) === 1){

    $sql_2 = "UPDATE users
              SET password='$np'
              WHERE id='$id'";
    mysqli_query($conn, $sql_2);
    header("Location: change-password.php?success=Your password has been changed successfully");
    exit();
```

Figure 34Updating password

The change password part is done by updating the old password with the new one. The SQL will update the password where the ID is equal to the current account logged in.

Update statements are also used to update the stock after an item has been bought. (see figure 34)

5.5.3 Retrieve

```php
<?php
  if (isset($_GET['id'])){
    $pid = $_GET['id'];
  }
  $sql = "SELECT * FROM products WHERE id = '$pid'";
  $result = $conn->query($sql);

  if ($result->num_rows > 0) {
  }
  while($row = $result->fetch_assoc()) {
?>
```

Figure 35Retrieving products details

This is inside the *productDetails.php*. After the ID is received through a GET in the products.php, a SQL query uses it to get the product's details.

Then the results are looped similarly to how the products are loaded in the *products.php*

```php
<div class = "product-content">
  <h3 class = "product-title"><?php echo  $row["name"] ?></h3>
  <div class = "product-price">
    <p class = "new-price">Price: Rs.<span><?php echo  $row["price"] ?></span></p>
  </div>

  <div class = "product-detail">
    <h2>About this item: </h2>
    <p><?php echo  $row["info"] ?></p>
    <ul>
      <li>Category: <span><?php echo  $row["cat"] ?></span></li>
      <li>Quality: <span>Freshly Cooked/Baked</span></li>
    </ul>
  </div>
```

Figure 36Displaying product details

The product's details such as the name, price and its ingredients are then retrieved from the database.

5.6 Successful connection to the database server

includes > db_connection.php > ...

```php
<?php
function OpenCon()
 {
 $dbhost = "localhost";
 $dbuser = "root";
 $dbpass = "";
 $db = "samshai";
 $conn = new mysqli($dbhost, $dbuser, $dbpass,$db) or die("Connect failed: %s\n". $conn -> error);
 return $conn;
}

function CloseCon($conn){
 $conn -> close(); }
?>
```

Figure 37Connection to database

To connect to the MYSQL database, the XAMPP server needs to be running. *$conn* is a mySQL procedure which takes the host, username, password (in our case there is none) and the database name. This php file is called by using "include 'db_connection.php'" and then the function OpenCon() is called.

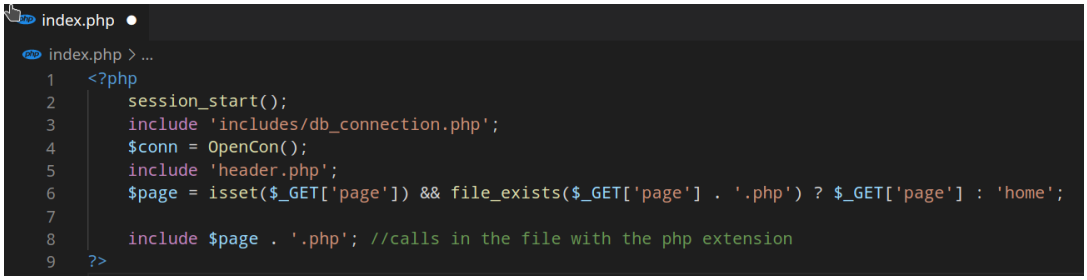


Figure 38Basic routing of index.php

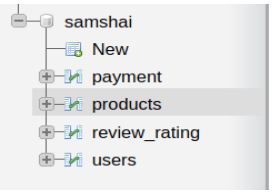
After a session is started, the database connection and its function are called. The header files contain the nav bar.

The $page serves as a routing to facilitate going through other pages while still using the database. By default, it is set to open the home page.

For example, the URL of the product page will be *index.php?page=products*. If the user clicks on a product to view its details, the URL will then be *index.php?page=productDetails&id=4*.

By using index.php as a "parent" URL, there is no need to have the header and database connection in each file, it is automatically in every file loaded via index.php.

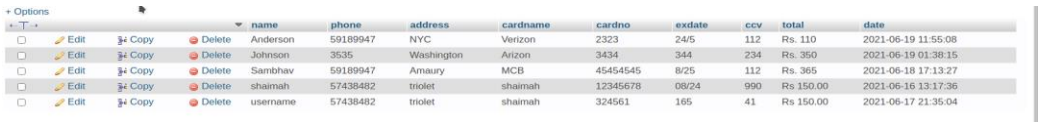
6.0 MySQL Database



The XAMPP package contains the MySQL database which is accessed through a web browser. In our database, the tables on the left were used. The database connection which has been explained previously, is used to connect XAMPP with Visual Studio Code.

Figure 39All tables from the Database

- **Payment table**



	name	phone	address	cardname	cardno	exdate	ccv	total	date
Edit Copy Delete	Anderson	59189947	NYC	Verizon	2323	24/5	112	Rs. 110	2021-06-19 11:55:08
Edit Copy Delete	Johnson	3535	Washington	Arizon	3434	344	234	Rs. 350	2021-06-19 01:38:15
Edit Copy Delete	Sambhav	59189947	Amaury	MCB	45454545	8/25	112	Rs. 365	2021-06-18 17:13:27
Edit Copy Delete	shaimah	57438482	triolet	shaimah	12345678	08/24	990	Rs 150.00	2021-06-16 13:17:36
Edit Copy Delete	username	57438482	triolet	shaimah	324561	165	41	Rs 150.00	2021-06-17 21:35:04

Figure 40Payment Table

The payment table contains all the relevant details of the customer, card details and the total. The date column uses a timestamp data-type, and it is automatically updated.

- **Product Table**

id	name	price	stock	path	cat	info
1	Chicken Spring Rolls	65	46	chicken spring rolls.png	Starter	Contains chicken, cabbage, carrots, oyster sauce, ...
2	Garlic Bread	55	34	garlic bread.png	Starter	Main ingredients consists of olive oil, parsley, g...
3	Brochette Poulet	80	43	brochettes-de-poulet-aux-fines-herbes-et-citron.jp...	Starter	Consists of chicken, carrots, green bell peppers, ...
4	Brochette Crevette	85	46	crevettes-jumbo-sauce-satay.jpeg	Starter	Main ingredients are: Shrimps, fillet fish, green ...
5	Crispy Prawns	95	36	crispy prowns.jpg	Starter	Main ingrediants are: prawns, garlic, breadcrumbs ...
6	Crispy Chicken	95	49	Crispy fried chicken.jpg	Starter	Main ingrediants are: chicken, garlic, breadcrumbs...
7	FETTUCCINE ALFREDO	365	50	Fettuccine Alfredo.jpeg	Main Course	Main ingredients: fettucine pasta, butter, chicken...
8	GARLIC BUTTER PASTA	375	50	Garlic-Butter-Pasta-with-Garlic-Chicken.jpg	Main Course	Main ingredients consists of chicken breast, butte...
9	LOBSTER PASTA	350	50	Lobster-Pasta.jpg	Main Course	Main ingredients consists of lobster, spaghetti, b...
10	VEGAN MUSHROOM PASTA	270	50	Super green pasta.jpg	Main Course	Specially added to the menu for our vegan customer...
11	BRUSCHETTA PASTA	275	50	bruschetta-pasta.jpg	Main Course	Main ingredients consists of penne pasta, cherry t...

Figure 41Product Table

The product table contains the product details. The path of the image is stored and then later called to display the images.

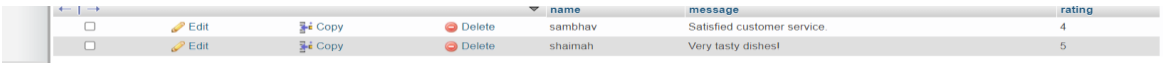
- **Users Table**

	id	user_name	password	name
Edit Copy Delete	5	Shaimah	1234	shaimah
Edit Copy Delete	6	sambhav03	12345	sambhav
Edit Copy Delete	7	admin1	admin	admin
Edit Copy Delete	8	Elliot3	h4ck	Elliot Anderson

Figure 42User Table

The users table contains the name, username, password and the id of all the accounts created.

- **Rating_Review Table**



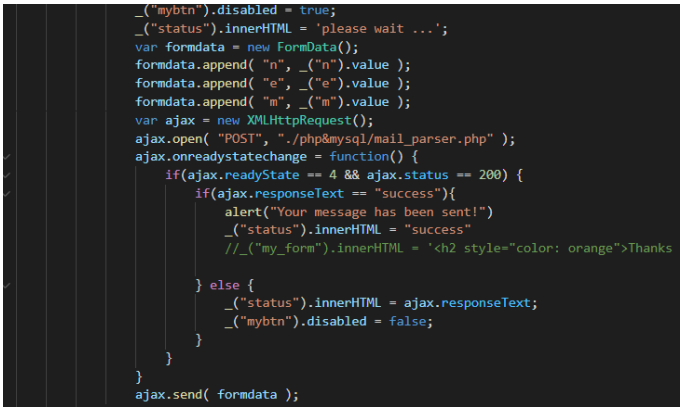
	name	message	rating
Edit Copy Delete	sambhav	Satisfied customer service.	4
Edit Copy Delete	shaimah	Very tasty dishes!	5

Figure 43Ratings Review Table

The review submission is stored in the database as shown in the figure above.

7.0 AJAX

7.1 Use of PHP AJAX in Contact Form



```
_("mybtn").disabled = true;
_("status").innerHTML = 'please wait ...';
var formdata = new FormData();
formdata.append( "n", _("n").value );
formdata.append( "e", _("e").value );
formdata.append( "m", _("m").value );
var ajax = new XMLHttpRequest();
ajax.open( "POST", "./php&mysql/mail_parser.php" );
ajax.onreadystatechange = function() {
    if(ajax.readyState == 4 && ajax.status == 200) {
        if(ajax.responseText == "success"){
            alert("Your message has been sent!")
            _("status").innerHTML = "success"
            //_("my_form").innerHTML = '<h2 style="color: orange">Thanks

        } else {
            _("status").innerHTML = ajax.responseText;
            _("mybtn").disabled = false;
        }
    }
}
ajax.send( formdata );
```

According to the code, when the button "submit form" with id= "my-btn" is clicked, a text "please wait" appears in the span created with id= "status". Them the values of name, email and message is appended followed by the creation of an XMLHttpRequest object, thus creating a function to be executed when the server response is ready, sends the request off to a PHP file (mail_parser.php) on the server. Then the message "success" appears when the form is successfully submitted. This gives the user an indication when his form is submitted.

Figure 44Contact form using AJAX

7.2 AJAX in Review Rating Page

In order to let the user know that his review is submitted a message appears "Review Submitted". For this message to appear ajax has been as shown in figure xxx.

```
$("#submit").click(function(){
    var name = $('#name').val();
    var message = $('#message').val();
    $.ajax({
        url: "rating.php",
        type: 'post',
        data: {v1 : name, v2 : message, v3 : rate},
        success: function (status) {
            if(status == 1){
                $('.msg').html('<b style="color: blue"> Review Submitted !</b>');
            }else{
                $('.msg').html('<b>Review not submitted!</b>');
            }
        }
    });
```

When the "Submit" button is clicked, the name and email value is taken which is transferred to the *rating.php.* The third variable is the rate which is the value from the stars. And when the values are successfully added to the database, the message "Review submitted" is displayed.

Figure 45Ratings using AJAX

7.3 AJAX in About Us

On clicking the button "Show Surprise", a function (loadDoc()) is called in which data is requested from the web server and displays it. That is, the text form *surprise.txt* is displayed. This has been implemented using AJAX as shown below.

```
<script>
    function loadDoc() {
      var xhttp = new XMLHttpRequest();
      xhttp.onreadystatechange = function() {
        if (this.readyState == 4 && this.status == 200) {
          document.getElementById("surprise").innerHTML =
          this.responseText;
        }
      };
      xhttp.open("GET", "surprise.txt", true);
      xhttp.send();
    }
</script>
```

Figure 46Show Surprise using AJAX