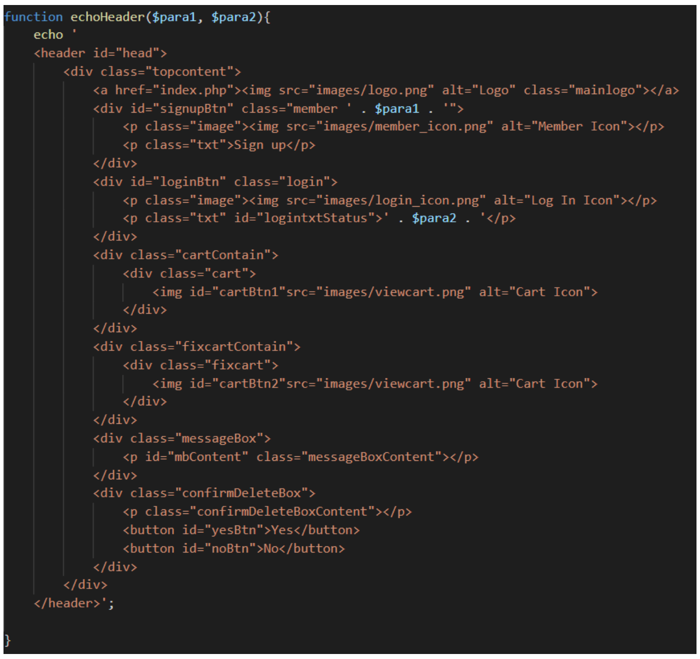
[COS10011] Assignment 3 Report	
Team Name: Ceromates	Date: [2020-06-17]
Title: Assignment 3: Business Website for Ceronics	

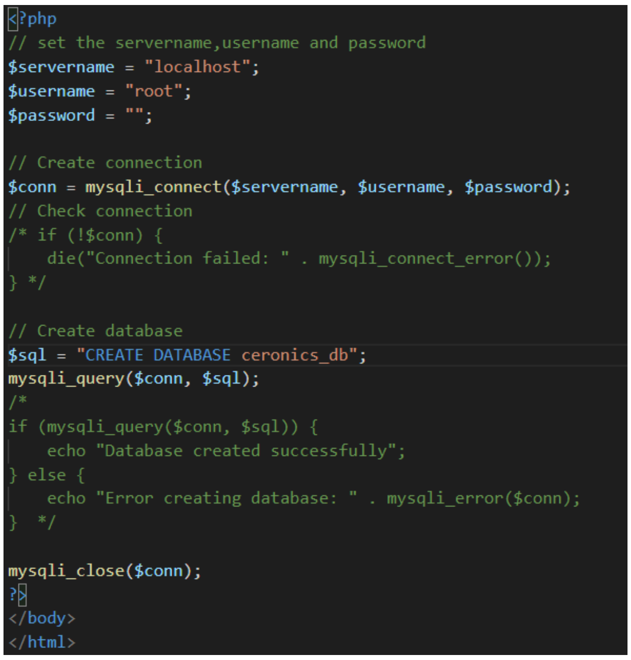
Use PHP to reuse common elements



```
function echoHeader($para1, $para2){
    echo '
    <header id="head">
        <div class="topcontent">
            <a href="index.php"><img src="images/logo.png" alt="Logo" class="mainlogo"></a>
            <div id="signupBtn" class="member ' . $para1 . '">
                <p class="image"><img src="images/member_icon.png" alt="Member Icon"></p>
                <p class="txt">Sign up</p>
            </div>
            <div id="loginBtn" class="login">
                <p class="image"><img src="images/login_icon.png" alt="Log In Icon"></p>
                <p class="txt" id="logintxtStatus">' . $para2 . '</p>
            </div>
            <div class="cartContain">
                <div class="cart">
                    <img id="cartBtn1"src="images/viewcart.png" alt="Cart Icon">
                </div>
            </div>
            <div class="fixcartContain">
                <div class="fixcart">
                    <img id="cartBtn2"src="images/viewcart.png" alt="Cart Icon">
                </div>
            </div>
            <div class="messageBox">
                <p id="mbContent" class="messageBoxContent"></p>
            </div>
            <div class="confirmDeleteBox">
                <p class="confirmDeleteBoxContent"></p>
                <button id="yesBtn">Yes</button>
                <button id="noBtn">No</button>
            </div>
        </div>
    </header>';

}
```

All the common elements from the Web site are put into a separate files such as Header, Navigation, Footer, Login Form, Sign Up Form, Feedback Form, Calculator Panel, and Cart Panel. This can let every page contain common elements as all pages are using the same PHP file. So, unity of the website can be maintained. This helps editing for the website much easier as all the reuse elements are using the same PHP file. All the common elements are regrouped in a file name "include". The original file before the implementation of the PHP are from JavaScript.

Create database

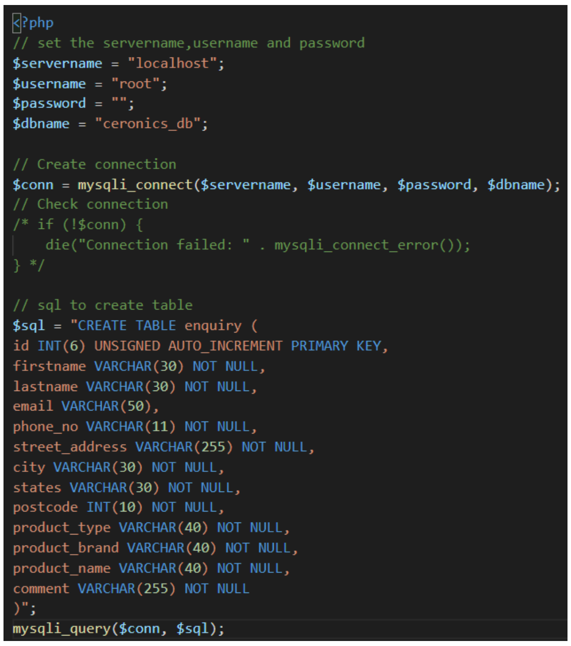


```
<?php
// set the servername,username and password
$servername = "localhost";
$username = "root";
$password = "";

// Create connection
$conn = mysqli_connect($servername, $username, $password);
// Check connection
/* if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
} */

// Create database
$sql = "CREATE DATABASE ceronics_db";
mysqli_query($conn, $sql);
/*
if (mysqli_query($conn, $sql)) {
    echo "Database created successfully";
} else {
    echo "Error creating database: " . mysqli_error($conn);
}  */

mysqli_close($conn);
?>
</body>
</html>
```

Create_Database.php is a php server-side webpage used to create a database to store information regarding our business. Ceronics like enquiry form information and member details. Database offers a huge capacity to handle and store a huge amount of data which is especially useful for online business webpages. Similarly, we create a connection to the server name after we set up the server name, username and password. When the connection to the server is valid, we set up our database by using the command “CREATE DATABASE ceronics_db”. The mysqli_query() function is used to performs a query against a database.

The database is now created with our business name called “ceronics_db”. The message of checking the connection of localhost and the creation database is being removed/commented as to prevent the user from seeing the messages shown on the webpages. This php page is only linked on the home page (index.php) as database only need to be created once.

Create Tables



```
<?php
// set the servername,username and password
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "ceronics_db";

// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
/* if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
} */

// sql to create table
$sql = "CREATE TABLE enquiry (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(30) NOT NULL,
lastname VARCHAR(30) NOT NULL,
email VARCHAR(50),
phone_no VARCHAR(11) NOT NULL,
street_address VARCHAR(255) NOT NULL,
city VARCHAR(30) NOT NULL,
states VARCHAR(30) NOT NULL,
postcode INT(10) NOT NULL,
product_type VARCHAR(40) NOT NULL,
product_brand VARCHAR(40) NOT NULL,
product_name VARCHAR(40) NOT NULL,
comment VARCHAR(255) NOT NULL
)";
mysqli_query($conn, $sql);
```

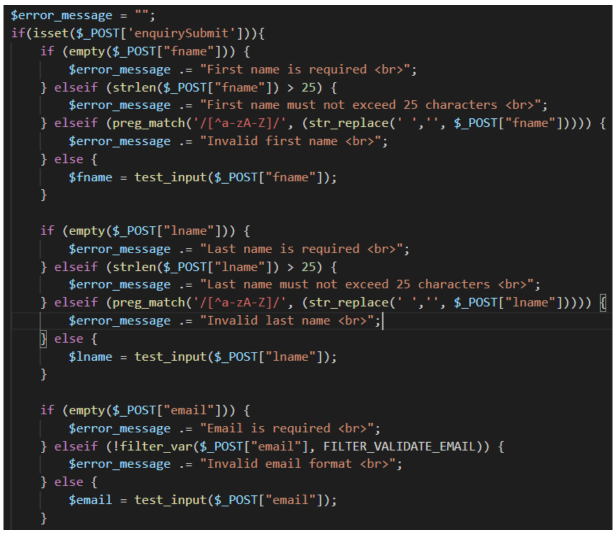
After the database is created, we need to organize all the data in their respective tables follow by their data types and sequence. First of all, we need to create table in the database to store the data in a proper manner. Here, we are creating a table named "enquiry" to store all the enquiry form details submitted by customers using the MySQL command "CREATE TABLE enquiry ()". From the MySQL command, in the bracket, we can define the table columns in the enquiry table by entering the "column name" followed by DATATYPES (number of characters). For instance,

"CREATE TABLE enquiry (
id INT (6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR (30) NOT NULL,
lastname VARCHAR (30) NOT NULL)"

One of the most important steps is that we define our first column of the enquiry table as user id which is "id" and the UNSIGNED allows only non-negative numbers to be used to record the user id. AUTO_INCREMENT helps the database table to generate a sequence whenever any new record is added into the table. The user id is our primary key for the enquiry table in the database which defines the unique identifier for every single record. All tables that needed to be created are coded in the 'create_tables.php' page. Similar to the create_database.php, this php page will only be linked once at the home page.

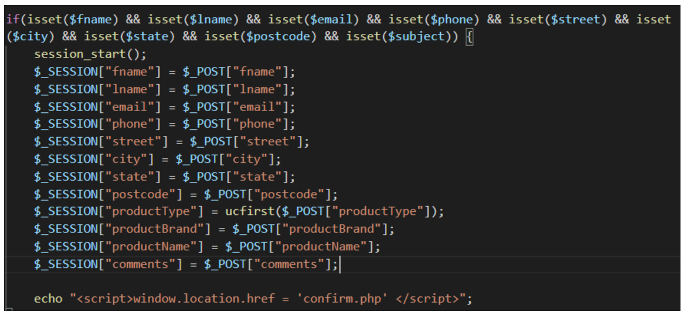
Enquiry Form Validation

Php side validation has also been implemented. The previous use of JavaScript implementation has been replaced completely with php validation in both Enquiry Form (enquiry.php) and Sign Up Form (login_signup_tab.php). This allows more convenient editing as the php code is directly written internally in the page. A server-side validation is more secured as user at the client side will not be able to see and edit the php codes. This also helps with maintenance and editing for future uses as the php is inside the page itself. Other than that, php also makes the webpage more flexible as it allows parts of the page to change without having to reload the entire page for example the error messages that pop up when a field is incorrectly filled in. The picture below shows some part of the validation's code. It is quite similar to JavaScript, just that the syntax used are different. For instance, empty() to check the input is empty or not, strlen() to check the length of string.



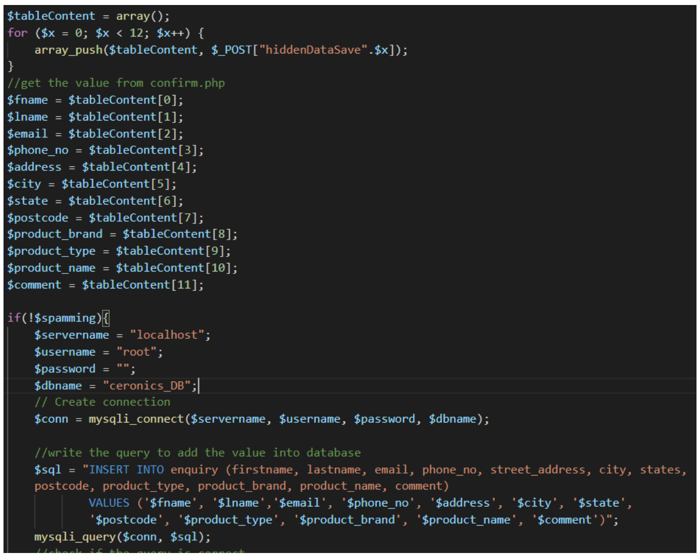
```php
$error_message = "";
if(isset($_POST['enquirySubmit'])){
    if (empty($_POST["fname"])) {
        $error_message .= "First name is required <br>";
    } elseif (strlen($_POST["fname"]) > 25) {
        $error_message .= "First name must not exceed 25 characters <br>";
    } elseif (preg_match('/[^a-zA-Z]/', (str_replace(' ','', $_POST["fname"])))) {
        $error_message .= "Invalid first name <br>";
    } else {
        $fname = test_input($_POST["fname"]);
    }

    if (empty($_POST["lname"])) {
        $error_message .= "Last name is required <br>";
    } elseif (strlen($_POST["lname"]) > 25) {
        $error_message .= "Last name must not exceed 25 characters <br>";
    } elseif (preg_match('/[^a-zA-Z]/', (str_replace(' ','', $_POST["lname"])))) {
        $error_message .= "Invalid last name <br>";
    } else {
        $lname = test_input($_POST["lname"]);
    }

    if (empty($_POST["email"])) {
        $error_message .= "Email is required <br>";
    } elseif (!filter_var($_POST["email"], FILTER_VALIDATE_EMAIL)) {
        $error_message .= "Invalid email format <br>";
    } else {
        $email = test_input($_POST["email"]);
    }
```

Meanwhile, the data is being stored into session storage before direct the user to confirm.php page. The code used here is $_SESSION["sessionName"]. The data will then be retrieved from confirm.php after they clicked the submit button.



```php
if(isset($fname) && isset($lname) && isset($email) && isset($phone) && isset($street) && isset
($city) && isset($state) && isset($postcode) && isset($subject)) {
    session_start();
    $_SESSION["fname"] = $_POST["fname"];
    $_SESSION["lname"] = $_POST["lname"];
    $_SESSION["email"] = $_POST["email"];
    $_SESSION["phone"] = $_POST["phone"];
    $_SESSION["street"] = $_POST["street"];
    $_SESSION["city"] = $_POST["city"];
    $_SESSION["state"] = $_POST["state"];
    $_SESSION["postcode"] = $_POST["postcode"];
    $_SESSION["productType"] = ucfirst($_POST["productType"]);
    $_SESSION["productBrand"] = $_POST["productBrand"];
    $_SESSION["productName"] = $_POST["productName"];
    $_SESSION["comments"] = $_POST["comments"];

    echo "<script>window.location.href = 'confirm.php' </script>";
```

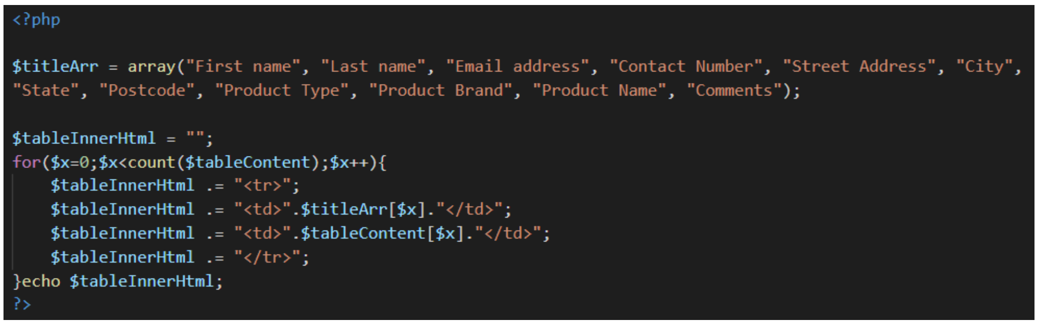
Submitting an enquiry



```php
$tableContent = array();
for ($x = 0; $x < 12; $x++) {
    array_push($tableContent, $_POST["hiddenDataSave".$x]);
}
//get the value from confirm.php
$fname = $tableContent[0];
$lname = $tableContent[1];
$email = $tableContent[2];
$phone_no = $tableContent[3];
$address = $tableContent[4];
$city = $tableContent[5];
$state = $tableContent[6];
$postcode = $tableContent[7];
$product_brand = $tableContent[8];
$product_type = $tableContent[9];
$product_name = $tableContent[10];
$comment = $tableContent[11];

if(!$spamming){
    $servername = "localhost";
    $username = "root";
    $password = "";
    $dbname = "ceronics_DB";
    // Create connection
    $conn = mysqli_connect($servername, $username, $password, $dbname);

    //write the query to add the value into database
    $sql = "INSERT INTO enquiry (firstname, lastname, email, phone_no, street_address, city, states,
    postcode, product_type, product_brand, product_name, comment)
            VALUES ('$fname', '$lname','$email', '$phone_no', '$address', '$city', '$state',
            '$postcode', '$product_type', '$product_brand', '$product_name', '$comment')";
    mysqli_query($conn, $sql);
```

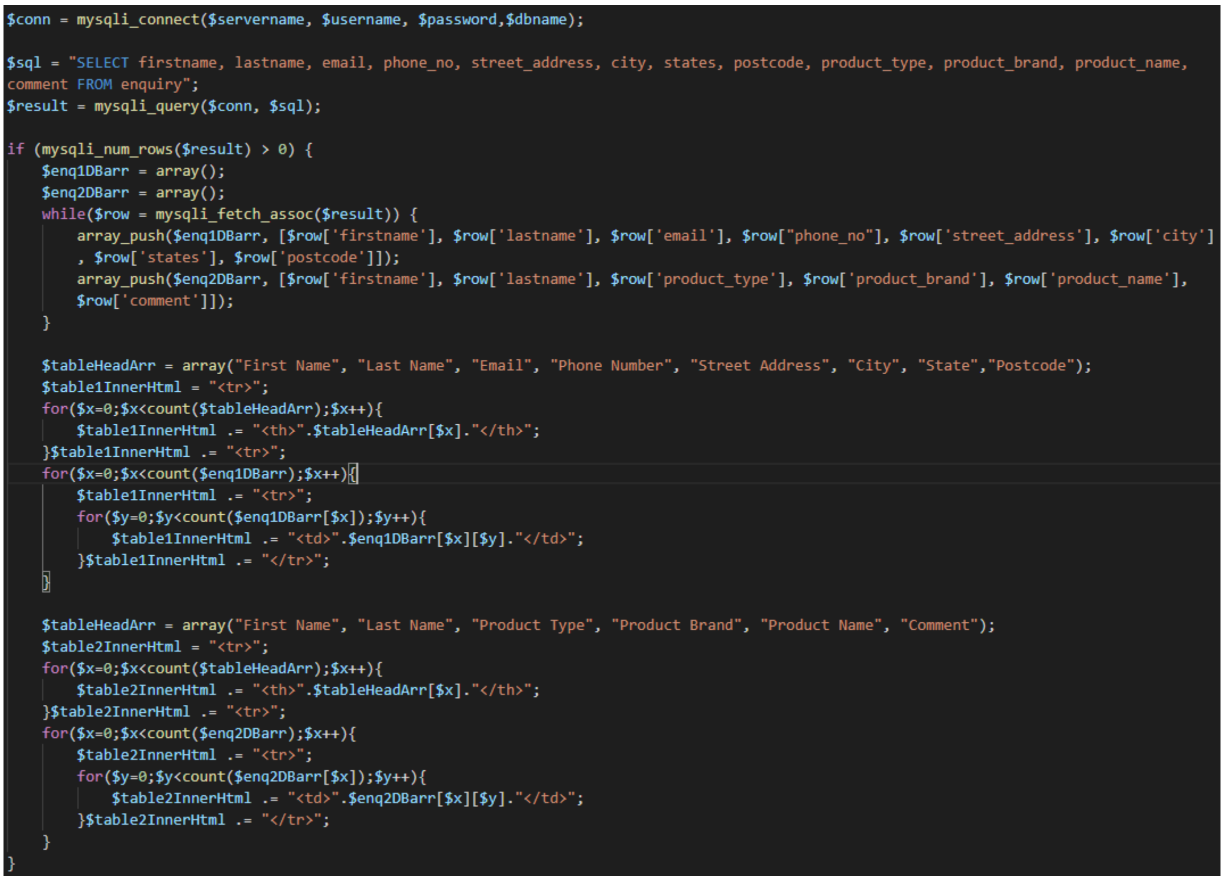
Enquiry_process.php is a server-side webpage generated in PHP form. It basically gets all the post data from confirm.php using superglobal "$_POST" and insert all the data into the "enquiry" table from our own customized database called ceronics_DB. To store the data into the database, we first create a connection to our server name called "localhost", once the connection is established, we insert data into the database tables using MySQL query. For eg, "INSERT INTO enquiry (firstname, lastname) VALUES ('Adrian', 'Sim')". The command is executed automatically once the user from the confirm.php clicks on the "Submit" button to submit their enquiry form.

After that, all the data is stored into the database table accurately according to their own data types to form an enquiry form record. For instance, the First name of our customer is stored into the firstname column while the Last name of our customer is stored into the lastname column within the same row of its respective User ID. All the enquiry form information will be organized accurately in the database table called enquiry. The team had used array to handle all the data when echo the php codes to display it in a table format on the web page.



```php
<?php

$titleArr = array("First name", "Last name", "Email address", "Contact Number", "Street Address", "City",
"State", "Postcode", "Product Type", "Product Brand", "Product Name", "Comments");

$tableInnerHtml = "";
for($x=0;$x<count($tableContent);$x++){
    $tableInnerHtml .= "<tr>";
    $tableInnerHtml .= "<td>".$titleArr[$x]."</td>";
    $tableInnerHtml .= "<td>".$tableContent[$x]."</td>";
    $tableInnerHtml .= "</tr>";
}echo $tableInnerHtml;
?>
```

View enquiries



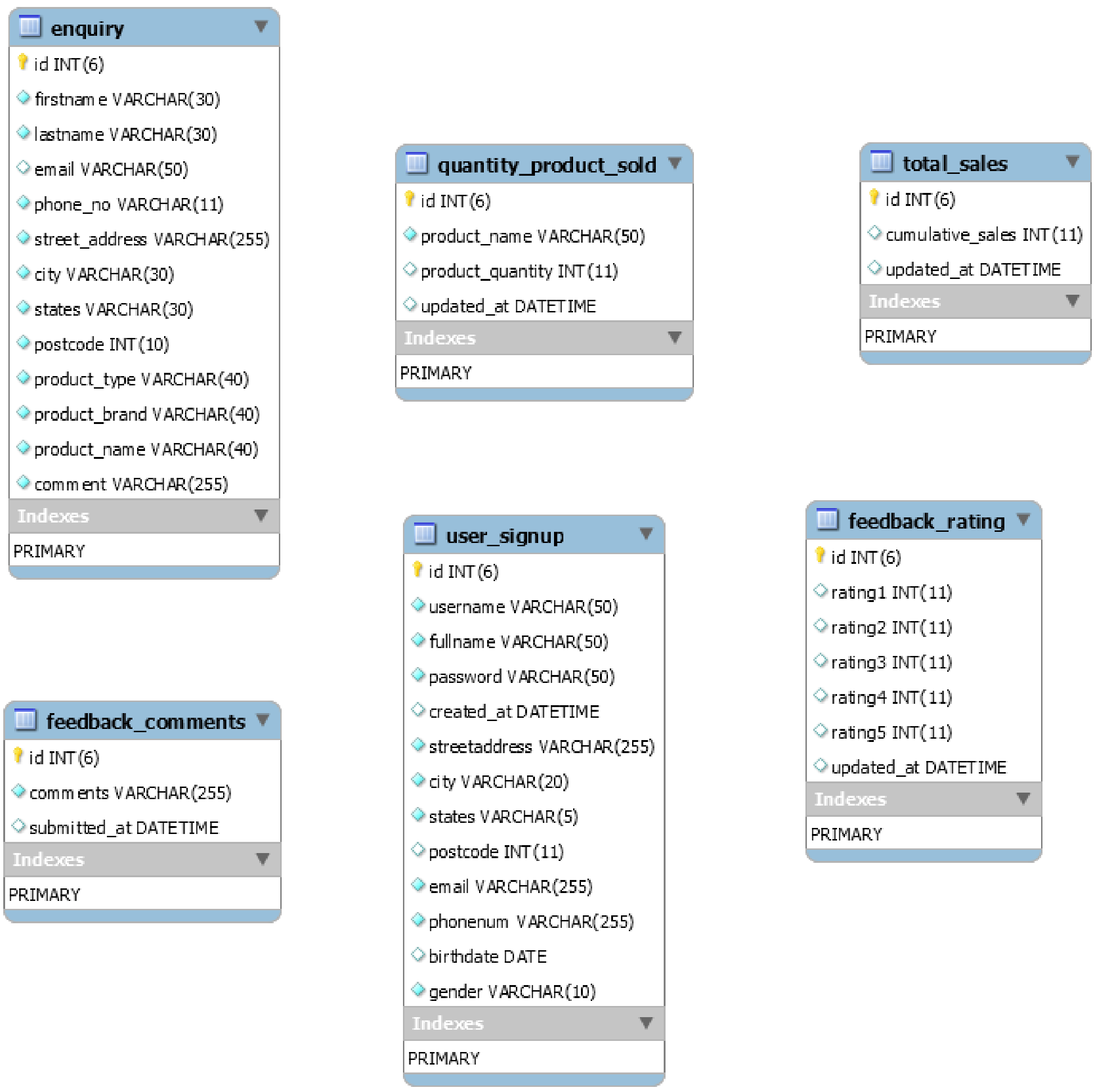
```php
$conn = mysqli_connect($servername, $username, $password,$dbname);

$sql = "SELECT firstname, lastname, email, phone_no, street_address, city, states, postcode, product_type, product_brand, product_name,
comment FROM enquiry";
$result = mysqli_query($conn, $sql);

if (mysqli_num_rows($result) > 0) {
    $enq1DBarr = array();
    $enq2DBarr = array();
    while($row = mysqli_fetch_assoc($result)) {
        array_push($enq1DBarr, [$row['firstname'], $row['lastname'], $row['email'], $row["phone_no"], $row['street_address'], $row['city']
        , $row['states'], $row['postcode']]);
        array_push($enq2DBarr, [$row['firstname'], $row['lastname'], $row['product_type'], $row['product_brand'], $row['product_name'],
        $row['comment']]);
    }

    $tableHeadArr = array("First Name", "Last Name", "Email", "Phone Number", "Street Address", "City", "State","Postcode");
    $table1InnerHtml = "<tr>";
    for($x=0;$x<count($tableHeadArr);$x++){
        $table1InnerHtml .= "<th>".$tableHeadArr[$x]."</th>";
    }$table1InnerHtml .= "<tr>";
    for($x=0;$x<count($enq1DBarr);$x++){
        $table1InnerHtml .= "<tr>";
        for($y=0;$y<count($enq1DBarr[$x]);$y++){
            $table1InnerHtml .= "<td>".$enq1DBarr[$x][$y]."</td>";
        }$table1InnerHtml .= "</tr>";
    }

    $tableHeadArr = array("First Name", "Last Name", "Product Type", "Product Brand", "Product Name", "Comment");
    $table2InnerHtml = "<tr>";
    for($x=0;$x<count($tableHeadArr);$x++){
        $table2InnerHtml .= "<th>".$tableHeadArr[$x]."</th>";
    }$table2InnerHtml .= "<tr>";
    for($x=0;$x<count($enq2DBarr);$x++){
        $table2InnerHtml .= "<tr>";
        for($y=0;$y<count($enq2DBarr[$x]);$y++){
            $table2InnerHtml .= "<td>".$enq2DBarr[$x][$y]."</td>";
        }$table2InnerHtml .= "</tr>";
    }
}
```

View_enquiries.php is a private webpage which is designed to display all the enquiry form information which are submitted to the ceronics_DB database table called “enquiry”. This is mainly a private website which cannot be accessed by any customers but only the business owner and allow the business owner to view and analyze all the enquiry forms submitted by customers. Therefore, the view_enquiries.php will retrieve all the information of each single enquiry form from the “enquiry” table using MySQL query. Similarly, we first create a connection to our server name called “localhost” by command “my_sqli_query(servername, username, password, databasename);”.

After that, we select all the enquiry form information from the “enquiry” database table using query “SELECT (firstname, lastname, email, …..) FROM enquiry;”. Once information is retrieved from the database table, the information is displayed to the business owner in table form. The row of the tables is generated based on the number of records from the database using the command “while ($row = mysqli_fetch_assoc($result)){…..}”.

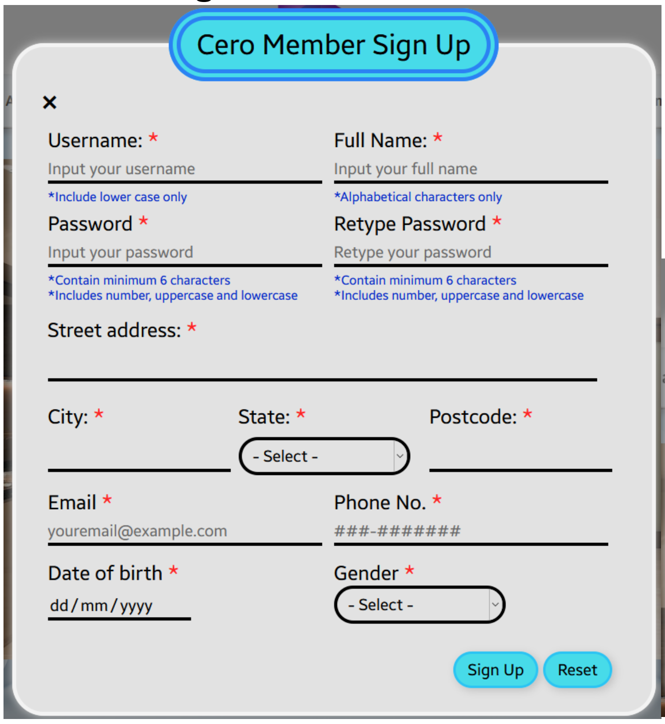
The mysqli_fetch_assoc($result) will return the field records in the current row of the results set into an associative array and moves the result pointer to the next row. Therefore, the enquiry form table will be displayed with all the information available from the database enquiry table using while loop. Lastly, the server connection is terminated once the table is generated completedly using the function mysqli_close().

MySQL database schema



Enhancements of Web Page

1. User Management Module


Cero Member Sign Up

×

Username: *
Input your username
*Include lower case only

Full Name: *
Input your full name
*Alphabetical characters only

Password *
Input your password
*Contain minimum 6 characters
*Includes number, uppercase and lowercase

Retype Password *
Retype your password
*Contain minimum 6 characters
*Includes number, uppercase and lowercase

Street address: *

City: *

State: *
- Select -

Postcode: *

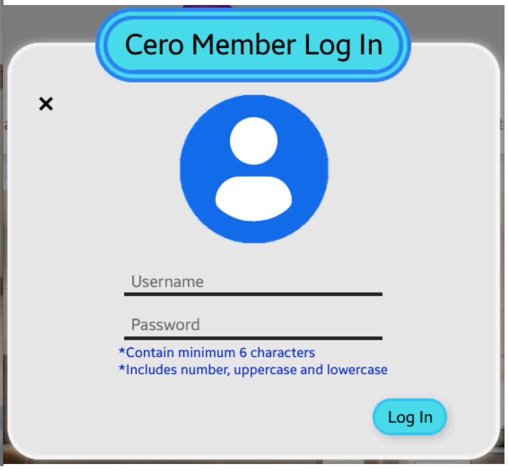
Email *
youremail@example.com

Phone No. *
###-#######

Date of birth *
dd/mm/yyyy

Gender *
- Select -

Sign Up Reset


Cero Member Log In

×

Username

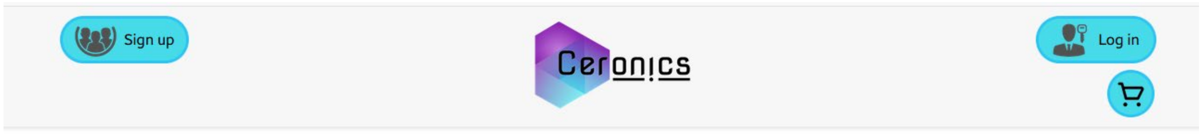
Password
*Contain minimum 6 characters
*Includes number, uppercase and lowercase

Log In

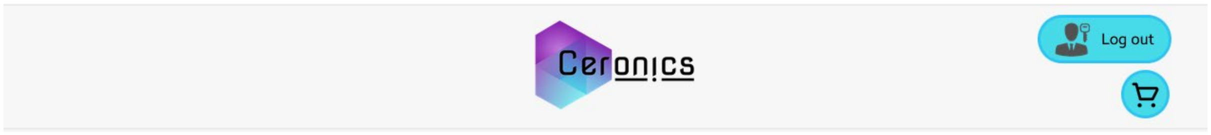
The sign up and login form are included in all pages. When user click the sign up or login buttons from the header, these forms will pop out. The sign up form is also validated using Php codes to preserve the integrity of the server data. Validation like checking the password and the re-type password field are similar or not are done from Php codes also. After the user submitted the from, the system will check whether the username the user keyed in is used by others or not. If the username is used by others, the system will output a message that ask the user to input the username again as the username is used by others. If everything had pass through the server side validation, the data will then being saved into the database's table named user_signup.

After the user sign up using their respective username, they will be able to login with their username and password now. After they login, the header will change to a button where they can logout.

Header before login

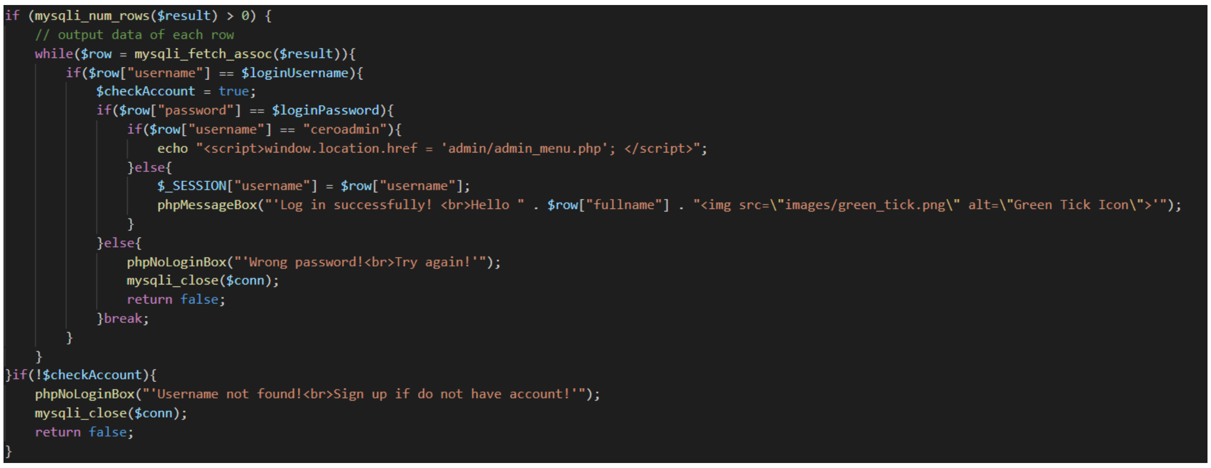


Header after login



After that, if the admin wants to check the user's information, the admin had to input the username ceroadmin and password Ceroadminpw12 in the login panel (login form) to get into the admin's menu.

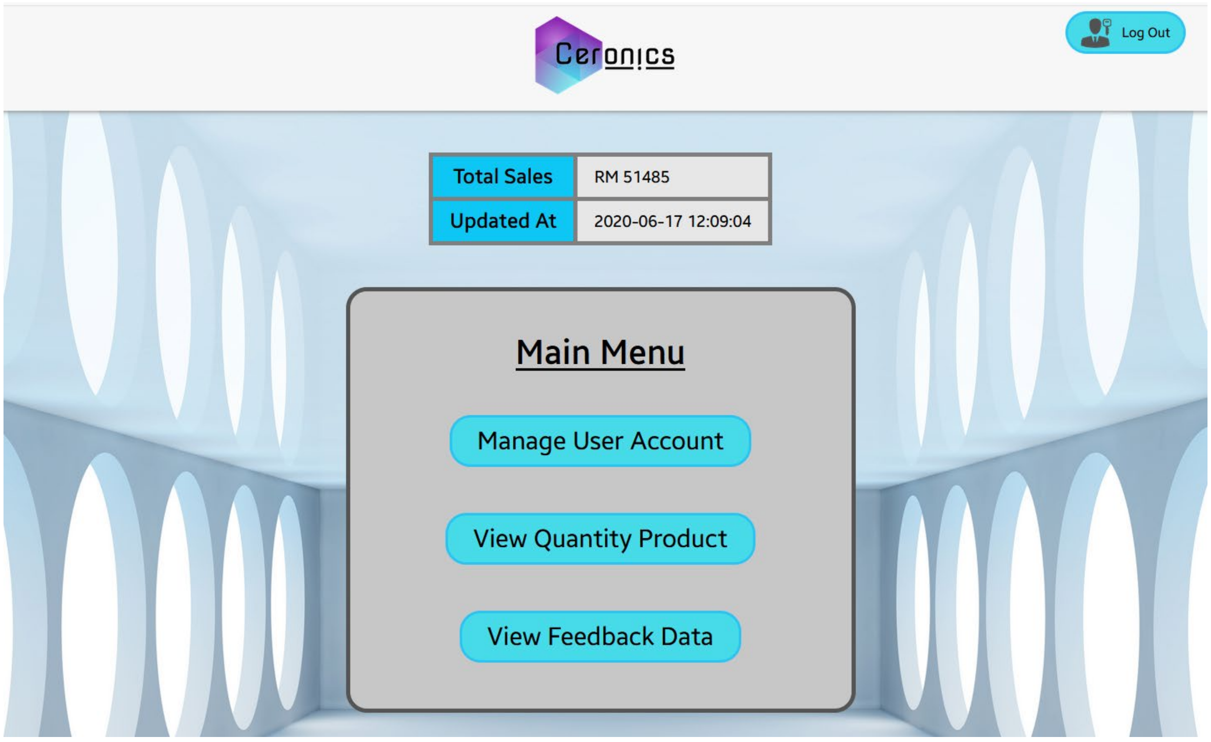
Admin username	ceroadmin
Admin password	Ceroadminpw12



```php
if (mysqli_num_rows($result) > 0) {
    // output data of each row
    while($row = mysqli_fetch_assoc($result)){
        if($row["username"] == $loginUsername){
            $checkAccount = true;
            if($row["password"] == $loginPassword){
                if($row["username"] == "ceroadmin"){
                    echo "<script>window.location.href = 'admin/admin_menu.php'; </script>";
                }else{
                    $_SESSION["username"] = $row["username"];
                    phpMessageBox("'Log in successfully! <br>Hello " . $row["fullname"] . "<img src=\"images/green_tick.png\" alt=\"Green Tick Icon\">'");
                }
            }else{
                phpNoLoginBox("'Wrong password!<br>Try again!'");
                mysqli_close($conn);
                return false;
            }break;
        }
    }
}if(!$checkAccount){
    phpNoLoginBox("'Username not found!<br>Sign up if do not have account!'");
    mysqli_close($conn);
    return false;
}
```

The codes above shows the direct of admin to admin's interface. The system will only direct to admin's interface if an admin's username and password in entered correctly.

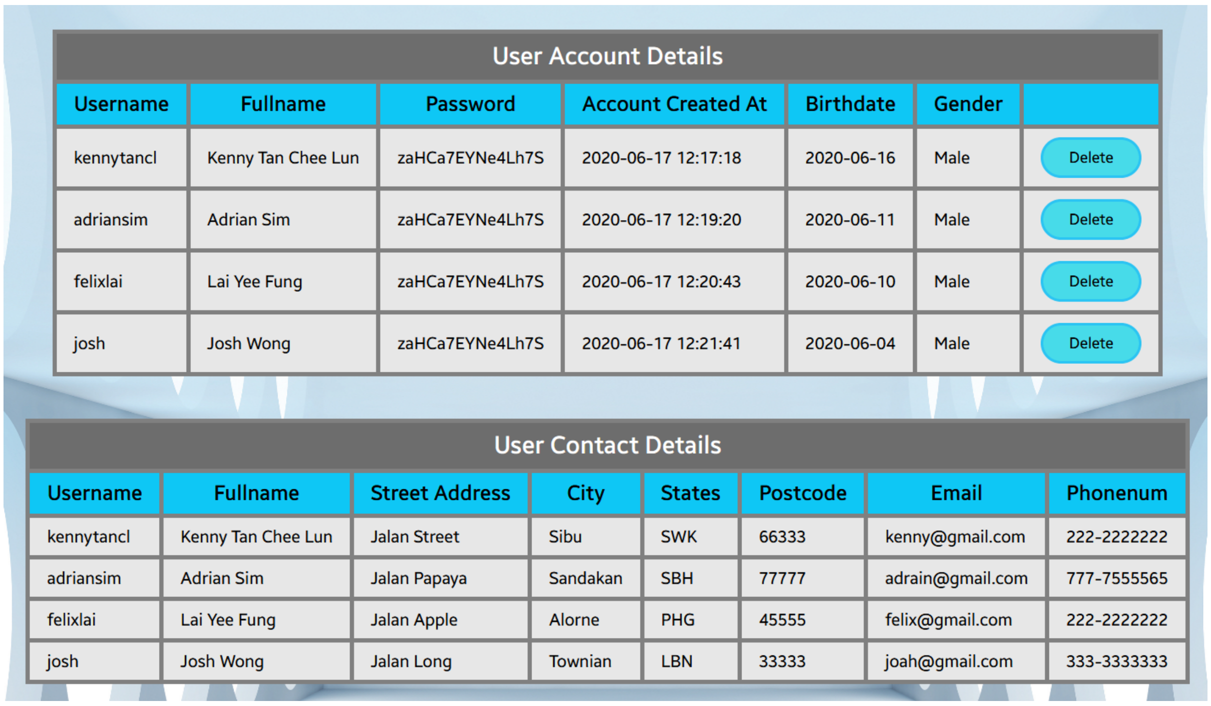
Admin Menu interface



Total Sales	RM 51485
Updated At	2020-06-17 12:09:04

The image above shows the main menu of admin. The admin can click into the manage user account to view all the user’s account. All details are as shown.

Manage User Account Interface



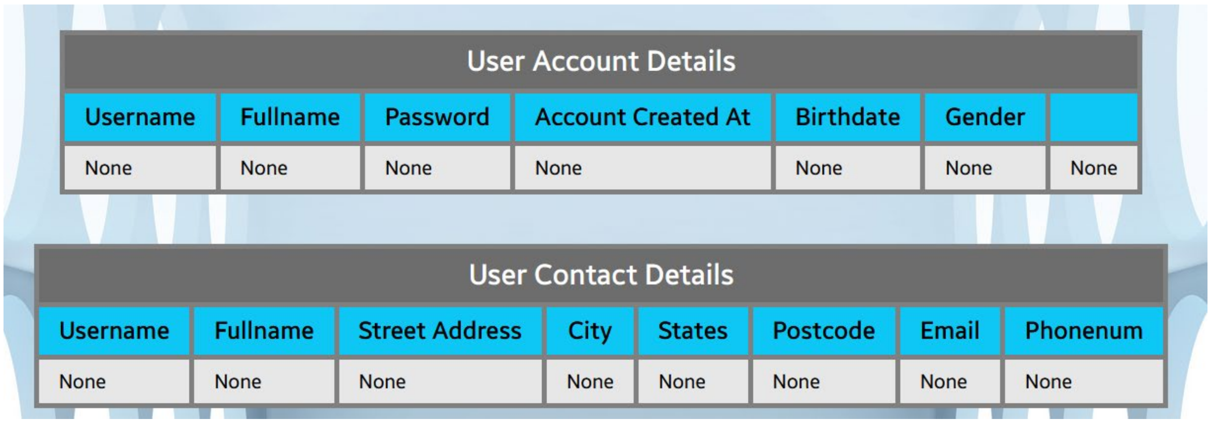
User Account Details

Username	Fullname	Password	Account Created At	Birthdate	Gender	
kennytancl	Kenny Tan Chee Lun	zaHCa7EYNe4Lh7S	2020-06-17 12:17:18	2020-06-16	Male	Delete
adriansim	Adrian Sim	zaHCa7EYNe4Lh7S	2020-06-17 12:19:20	2020-06-11	Male	Delete
felixlai	Lai Yee Fung	zaHCa7EYNe4Lh7S	2020-06-17 12:20:43	2020-06-10	Male	Delete
josh	Josh Wong	zaHCa7EYNe4Lh7S	2020-06-17 12:21:41	2020-06-04	Male	Delete

User Contact Details

Username	Fullname	Street Address	City	States	Postcode	Email	Phonenum
kennytancl	Kenny Tan Chee Lun	Jalan Street	Sibu	SWK	66333	kenny@gmail.com	222-2222222
adriansim	Adrian Sim	Jalan Papaya	Sandakan	SBH	77777	adrain@gmail.com	777-7555565
felixlai	Lai Yee Fung	Jalan Apple	Alorne	PHG	45555	felix@gmail.com	222-2222222
josh	Josh Wong	Jalan Long	Townian	LBN	33333	joah@gmail.com	333-3333333

The team had used the looping method to generate the table row and column. The admin can even delete the account of user by clicking the delete button allocated on the respective row. All the admin's php pages are stored in a folder named admin. After the admin click the log out button, the system will direct to the home page.



User Account Details

Username	Fullname	Password	Account Created At	Birthdate	Gender	
None	None	None	None	None	None	None

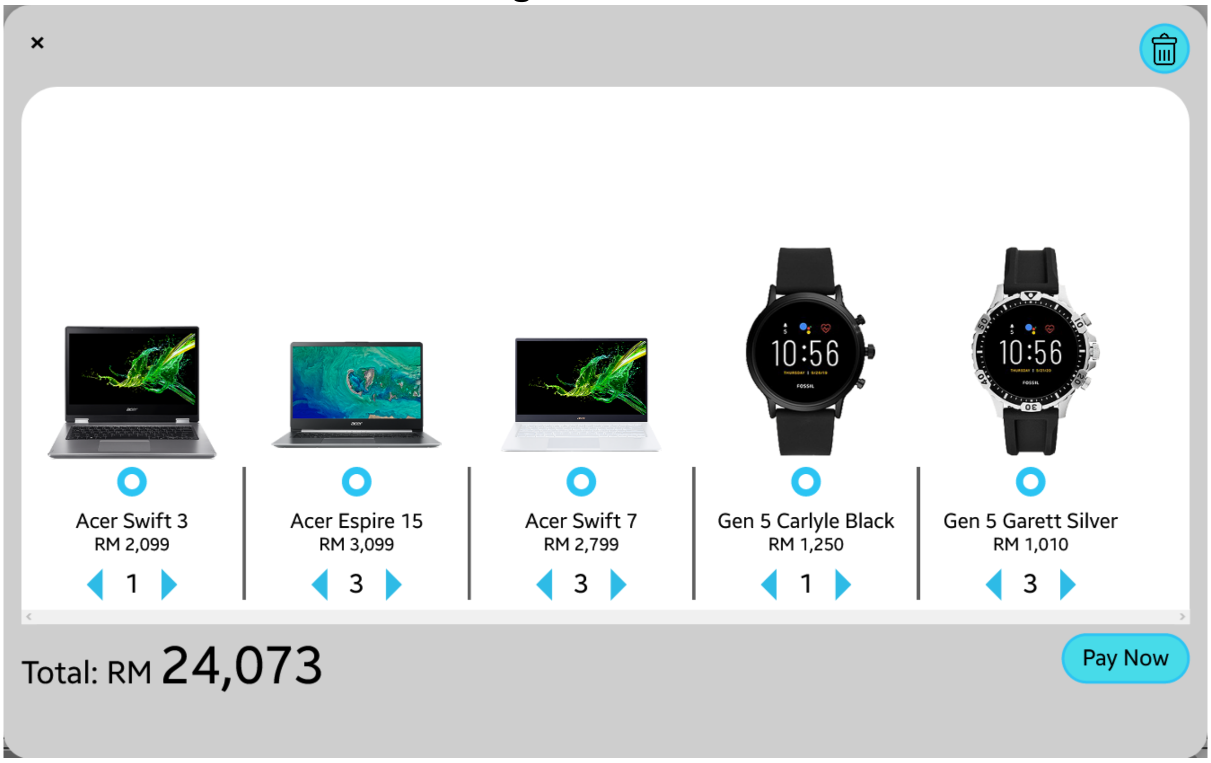
User Contact Details

Username	Fullname	Street Address	City	States	Postcode	Email	Phonenum
None	None	None	None	None	None	None	None

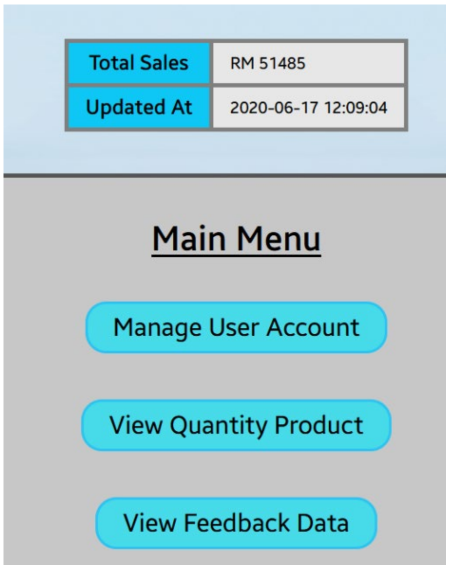
If there is no user in the database, the system will display the table as shown as above.

Some part of the user management module is referenced and paraphrased form https://www.tutorialrepublic.com/php-tutorial/php-mysql-login-system.php.

2. Product and Sales Tracking Module

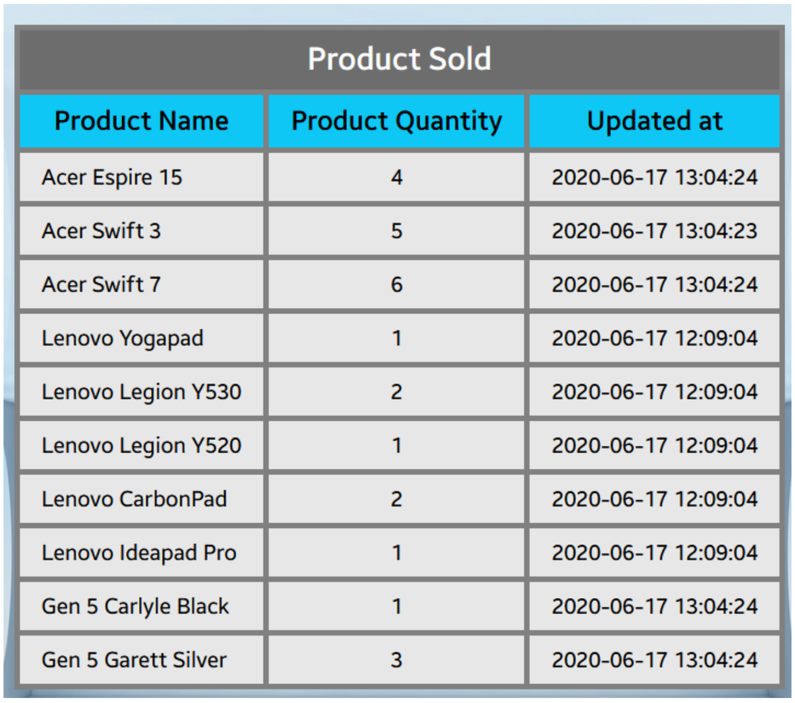


After the products are added to the cart, the user can now press the pay now button. After the cutomer click the pay now button, the system will save the total bills into the database's table named total_sales. Besides, the system will also save the product name and product quantity to database. The code used here is the <input type="hidden">. The pay now button is actually a form's submit button. The values are DOM by JavaScript before submitting the form. The product name and product quantity is stored to a table named quantity_product_sold in the database.



Total Sales	RM 51485
Updated At	2020-06-17 12:09:04

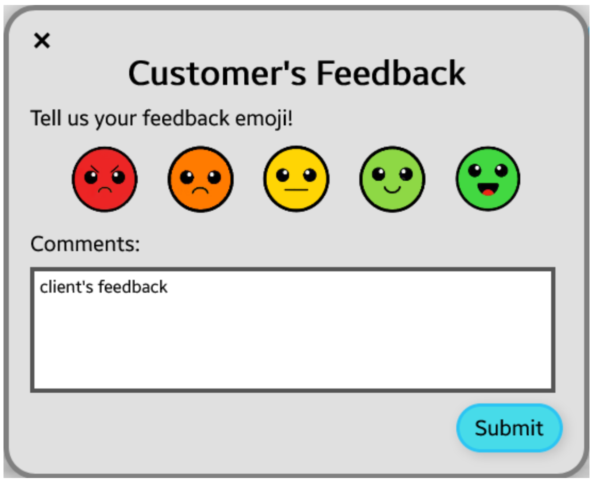
Refer to the figure on the left, the table above the main menu shows the total sales from the customer. When the customer click the pay now button in the cart, the sales will be added in cumulatively and recorded in this table. To view the quantity of product, the admin had to press the "View Quantity Product" button.



Product Sold		
Product Name	Product Quantity	Updated at
Acer Espire 15	4	2020-06-17 13:04:24
Acer Swift 3	5	2020-06-17 13:04:23
Acer Swift 7	6	2020-06-17 13:04:24
Lenovo Yogapad	1	2020-06-17 12:09:04
Lenovo Legion Y530	2	2020-06-17 12:09:04
Lenovo Legion Y520	1	2020-06-17 12:09:04
Lenovo CarbonPad	2	2020-06-17 12:09:04
Lenovo Ideapad Pro	1	2020-06-17 12:09:04
Gen 5 Carlyle Black	1	2020-06-17 13:04:24
Gen 5 Garett Silver	3	2020-06-17 13:04:24

The figure above show the interface on the table about the data retrieved from the database. There is also a column called "Updated At" to show the date and time the field is being updated.

3. Feedback Tracking Module



×

Customer's Feedback

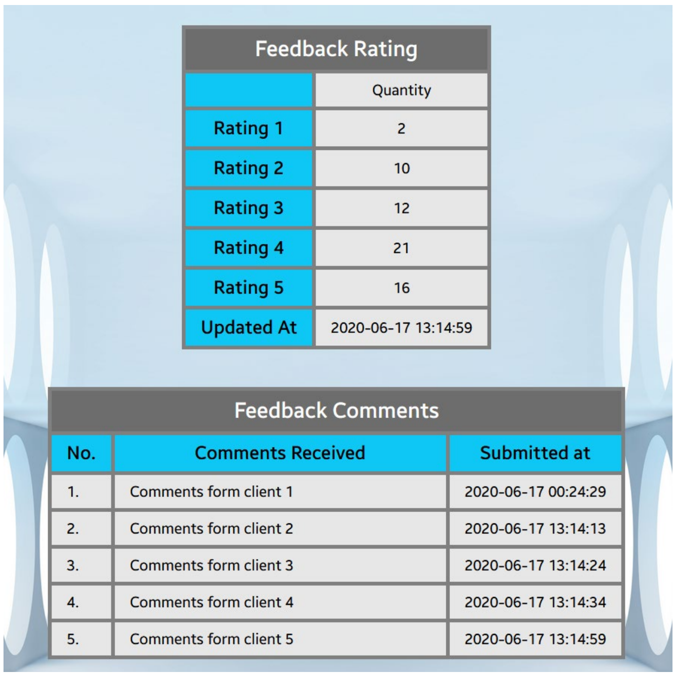
Tell us your feedback emoji!

🙁 🙁 😐 🙂 😀

Comments:

client's feedback

After the client submit the feedback rating emoji and give comments(optional), the system will then add the quantity of that particular rating (rating from 1 to 5, red emoji to green emoji) and save it to the tables feedback_rating and feedback_comments in the database.



Feedback Rating	
	Quantity
Rating 1	2
Rating 2	10
Rating 3	12
Rating 4	21
Rating 5	16
Updated At	2020-06-17 13:14:59

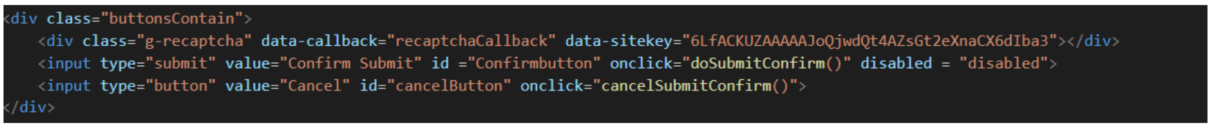
Feedback Comments

No.	Comments Received	Submitted at
1.	Comments form client 1	2020-06-17 00:24:29
2.	Comments form client 2	2020-06-17 13:14:13
3.	Comments form client 3	2020-06-17 13:14:24
4.	Comments form client 4	2020-06-17 13:14:34
5.	Comments form client 5	2020-06-17 13:14:59

If the admin clicks the “View Feedback Data” button from the admin’s main menu, the system will direct to this interface. The feedback rating table shows the quantity of user rate the company. The feedback comments tables shows the comments from user.

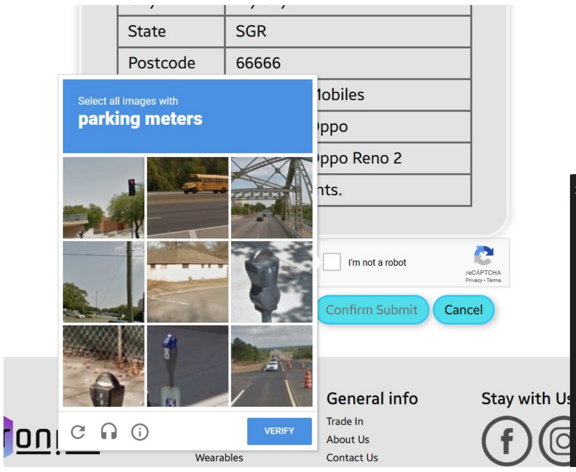
4. Anti-Spam Feature: Re-captcha

For the anti-spam feature, the team implemented one of the most common and widely used feature called “Re-captcha” which is developed by Google. This feature is used to prevent malicious attack or to avoid any spam by hackers to enable the online business model to run smoothly and safely.

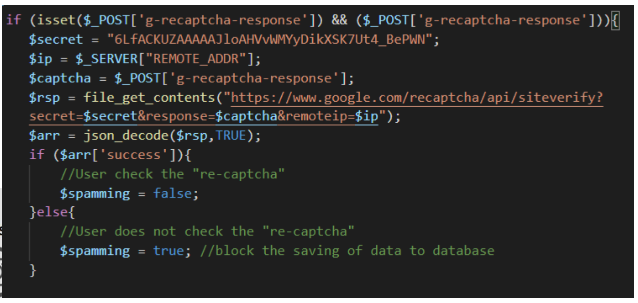


```
<div class="buttonsContain">
    <div class="g-recaptcha" data-callback="recaptchaCallback" data-sitekey="6LfACKUZAAAAAJoQjwdQt4AZsGt2eXnaCX6dIba3"></div>
    <input type="submit" value="Confirm Submit" id ="Confirmbutton" onclick="doSubmitConfirm()" disabled = "disabled">
    <input type="button" value="Cancel" id="cancelButton" onclick="cancelSubmitConfirm()">
</div>
```

First and foremost the re-captcha feature is implemented in the confirm.php page when user is prompted to confirm their enquiry form information prior submit to the database enquiry table. Before the submission is executed, user must check the “re-captcha” feature just before they submit it. To check it, the user had to select some images related to the field generated. The “Re-captcha” has a time-limit to remain valid, once the time-limit is exceeded, it will be expired and the user will have to re-check the “re-captcha” in order to submit the form.



State	SGR
Postcode	66666



```
if (isset($_POST['g-recaptcha-response']) && ($_POST['g-recaptcha-response'])){
    $secret = "6LfACKUZAAAAAJloAHVvWMYyDikXSK7Ut4_BePWN";
    $ip = $_SERVER["REMOTE_ADDR"];
    $captcha = $_POST['g-recaptcha-response'];
    $rsp = file_get_contents("https://www.google.com/recaptcha/api/siteverify?
    secret=$secret&response=$captcha&remoteip=$ip");
    $arr = json_decode($rsp,TRUE);
    if ($arr['success']){
        //User check the "re-captcha"
        $spamming = false;
    }else{
        //User does not check the "re-captcha"
        $spamming = true; //block the saving of data to database
    }
```

When the “re-captcha” is successfully checked and the confirm button is clicked, it will validate and check for the validity of the “g-captcha-response” in the enquiry_process.php. The validity of the “g-captcha-response” is checked by using “isset($_POST[“g-captcha-response”]) and ($_POST[“g-captcha-response”])” to see whether the captcha box is entered correctly or not. If the user does not check the “re-captcha” no data will be saved to the database. This can avoid user to send many similar data and waste the storage of the database.

This information is referenced from https://developers.google.com/recaptcha/docs/verify and https://developers.google.com/recaptcha/docs/display.

Member Contribution

Team Member 1	Kenny Tan Chee Lun (Group Leader)
Contribution (%)	25%
List of Contribution	Assign task to team members
	Completed user management module that has login and logout function
	Create admin’s main menu interface and additional php form and php pages
	Create product sold quantity and feedback tracking module
	Save and retrieve the data to and form the database’s recpective tables.
	Involved in doing the assignment report
	Submit the assignment to the submission link
Team Member 2	Josh Wong Howe Zheng
Contribution (%)	25%
List of Contribution	Use PHP to reuse common elements.
	Move all Header, Navigation, Footer, Login Form, Sign Up Form, Feedback Form, Calculator Panel, and Cart Panel to external php file.
	Link all the external file at respective location in all pages.
	Involved in doing the assignment report.
Team Member 3	Lai Yee Fung
Contribution (%)	25%
List of Contribution	Validate enquiry form using php code.
	Validate sign up form using php code.
	Ensure all data save to database is in proper manner.
	Ensure all required fields are not empty before submitting form.
	Involved in doing the assignment report.
Team Member 4	Adrian Sim Huan Tze
Contribution (%)	25%
List of Contribution	Create database and tables named enquiry.
	Make sure that all submitted enquiry details can be stored.
	Created enquiry_process.php for user to see the data submitted.
	Created the view_enquiries.php page for admin to see the data submitted
	Implement enhancement anti-spam feature for enhancement.
	Involved in doing the assignment report.