

Introduction of your project. Gives an overview of the project, rationale behind the creation of this website and the requirements.

The company is named as Re-Den. The founders of this company would be Chin Zi Ming. The company was headquartered in Kuala Lumpur, Malaysia. It is a newly established company since 2020. In 2020, the founder noticed that there is a denim trend, and he came across a street product where the seller customized the bags by using different pieces of clothes to become a pragmatic and fashionable handbag. Hence, the process of coming out with the idea of Re-Den was serendipitous. The idea of coming out with this name is that he would like to create awareness on protecting and conserving our environment. The 'Re' means recycle and recreate and the 'Den' is the denim which the items he would like to convert into a fashionable handbag through the 'Re' process. The main customer segment would be female as handbag can be described as one of their necessities. The founder planned to build this brand to be a well-known and continuous trend in 5 years' time and the slogan for this company would be 'Ride on the Fashion Green Wave' where it emphasizes the awareness on protecting the environment. Thus, the brand, Re-Den could contribute small part of the efforts for the conservation of environment.

Basically, the Re-Den company is operated via online. On the e-commerce website, Re-Den will be selling handbag mainly for the female. The selling of the handbag is unique. This is because the company required the customers to send their unwanted denim jeans to them in order for the further process and recreation to become a new fashionable denim handbag. The customers could choose on the design that they are interested in. Hence, the charges would be varied according to their preferences. After the end product is done, our company will deliver it to the customers' shipping address without any surcharge. The delivering fee is fully funded by the company and the 2.5% of the profit earned by each order will be taken to do the charity.

Screenshots of user interface on how to implement the PHP functions listed on the Web Features table & Implementation details of the website and justification of the approaches (excerpts of PHP codes for the PHP functions)

Registration function

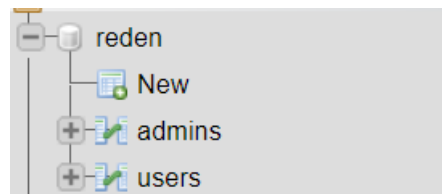


Figure a

```
Create TABLE users(  
  userId int(11) PRIMARY KEY AUTO_INCREMENT NOT NULL,  
  userName varchar(128) NOT NULL,  
  userEmail varchar(128) NOT NULL,  
  usersUid varchar(128) NOT NULL,  
  usersPwd varchar(128) NOT NULL,  
  billingName varchar(128) DEFAULT NULL,  
  usersPhone varchar(128) DEFAULT NULL,  
  usersAddress varchar(128) DEFAULT NULL,  
  usersCity varchar(128) DEFAULT NULL,  
  usersState varchar(128) DEFAULT NULL,  
  usersZip varchar(128) DEFAULT NULL,  
  userNameCard varchar(128) DEFAULT NULL,  
  usersCreditCardNumber varchar(128) DEFAULT NULL,  
  usersCreditCardExpM varchar(128) DEFAULT NULL,  
  usersCreditCardExpY varchar(128) DEFAULT NULL,  
  usersCreditCardExpCVV varchar(128) DEFAULT NULL  
);
```

Figure b

As the figure a shown above, I have created the reden database with users table to store the user data and information. I use the sql code shown in figure b to create the users table. In order to ensure that each row is unique, I created a column named userId that would serve as our table's primary key. To ensure that it can increase incrementally, I use auto increment. When a new user registers, their userId will always be the previous customer's number plus one. The userId, user email, username, and user password were then set to be not null. This is to ensure that users have access to this data when they are added to the database. I left the user's shipping and billing information null by default so that even if they don't enter any information, they may still register an account. We're doing this to make it simple for users to register. It is preferable to make it simple for users to create accounts when they want to register on our website. As a result, we let users join up without providing their billing and shipping information initially. Only when customers want to buy our products do they need to input more personal information.

Figure a

```

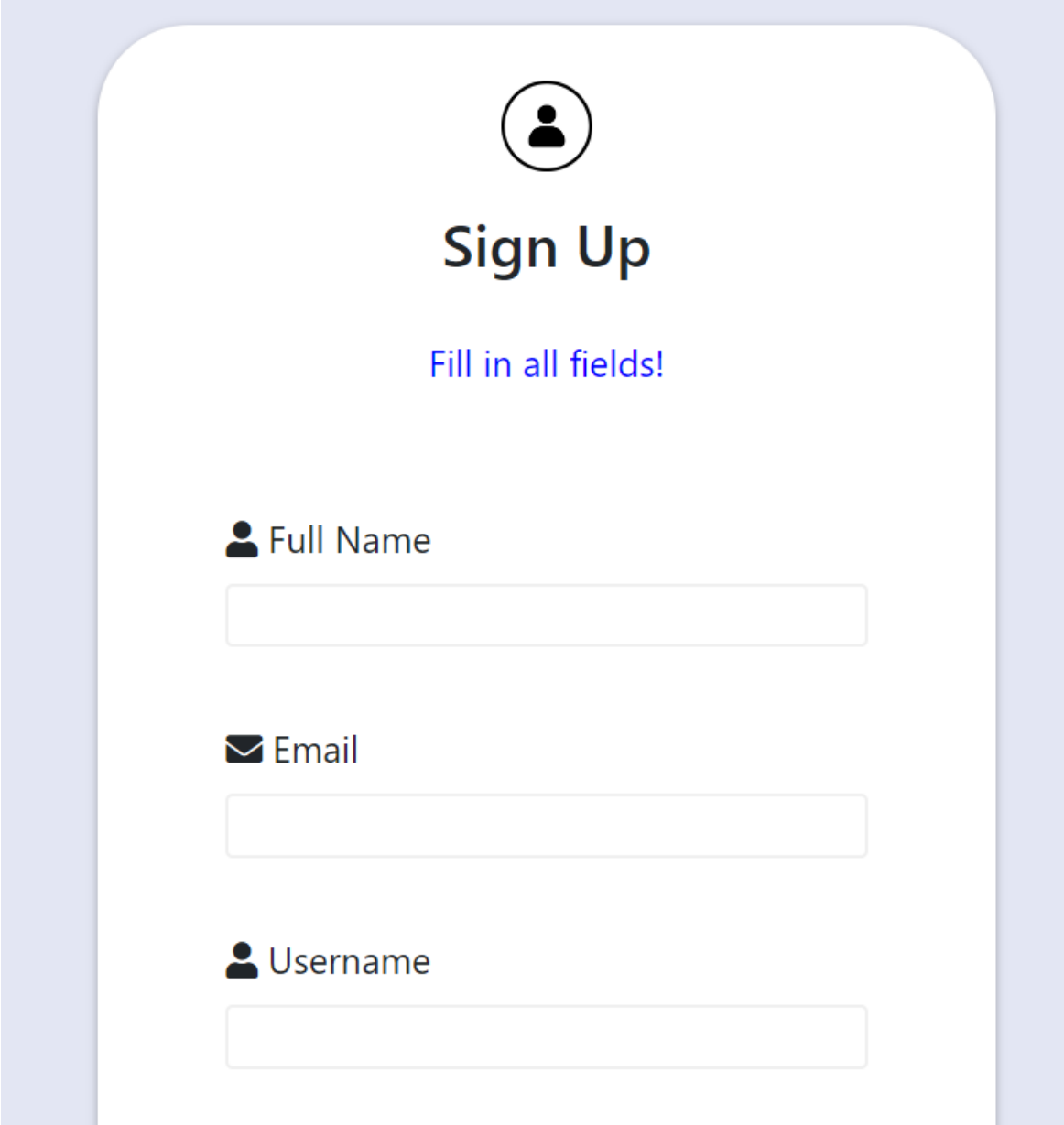
</div>
<form class="form" action="signup.inc.php" method="post">
<div class="field">
<label><i class="fa fa-user"></i> Full Name</label>
|   <input type="text" name="name" placeholder="">
</div>
<div class="field">
<label><i class="fa fa-envelope"></i> Email</label>
|   <input type="text" name="email" placeholder="">
</div>
<div class="field">
<label><i class="fa fa-user"></i> Username</label>
|   <input type="text" name="uid" placeholder="">
</div>
<div class="field">
<label><i class="fa fa-key" aria-hidden="true"></i> Password</label>
|   <input type="password" id="pwd" name="pwd" placeholder="">
<div id="psw_re">
|   <h3>Password must contain the following:</h3>
|   <p id="letter" class="invalid">At least one <b>lowercase</b> letter</p>
|   <p id="capital" class="invalid">At least one <b>capital (uppercase)</b> letter</p>
|   <p id="number" class="invalid">At least one <b>number</b></p>
|   <p id="length" class="invalid">At least <b>8 characters</b></p>
</div>
</div>


```

Figure b

As the figure a shown, this is the user interface for the user to sign up. We require the user to sign up first so that they can create an account on our website, and we are able to store their


data and information in our database. As shown in the figure b, I have integrated the signup.inc.php file in the form action which also act as a function file. Next, I'm using the post method as I want to get the input from the user securely when they are filling up the personal details at the signup page rather than using get method to handle the input from user.






Sign Up

Fill in all fields!

 Full Name

 Email


 Username

Figure a

```

if(isset($_POST["submit"])){

    $name = $_POST["name"];
    $email = $_POST["email"];
    $username = $_POST["uid"];
    $pwd = $_POST["pwd"];
    $pwdRepeat= $_POST["pwdrepeat"];

    require_once 'dbh.inc.php';
    require_once 'functions.inc.php';

    if(emptyInputSignup($name, $email, $username, $pwd, $pwdRepeat) !== false){
        header("location: sign_up.php?error=emptyinput");
        exit();
    }
}

```

Figure b

```

function emptyInputSignup($name, $email, $username, $pwd, $pwdRepeat){
    $result;
    if(empty($name)||empty($email)||empty($username)||empty($pwd)||empty($pwdRepeat)){
        $result = true;
    }
    else{
        $result = false;
    }
    return $result;
}

```

Figure c

```

if(isset($_GET["error"])){
    if($_GET["error"] == "emptyinput"){
        echo"<p>Fill in all fields!</p>";
    }
}

```

Figure d

As shown in figure a above, there is a notice which shows that user need to fill in all fields when they straight away leave the form blank and click on the sign-up button directly. As shown in figure b, I use the post method to get the input from user. Then, I use the emptyInputSignup to validate if the user has entered all the fields. Let's say the user did not fill up all the fields, he or she will prompt back to the sign_up page with an error that is same as the emptyinput. I also use the exit() function if the header statement does not execute. As shown in the figure c, the emptyInputSignup function in in the functions.inc.php file thus I use the require_once of the functions.inc.php file in the signup.inc.php file as shown in figure b to do the validation. As shown in figure c, the function emptyInputSignup contains five arguments which are name, email, username, password, and password repeat. They allow user input to be passed to the function. Then, a result variable is created. We use the the if statement to check whether the arguments are empty or not. Hence, we can validate whether the user enters all the fields or not when the system returns either true or false results. We will utilise the get function as shown in figure d to obtain the error if the user does not complete all the fields. We will display 'fill in all fields' at the top of the sign-up form as

shown in figure a above if the error equals emptyinput. In essence, this is how the validation function operates.

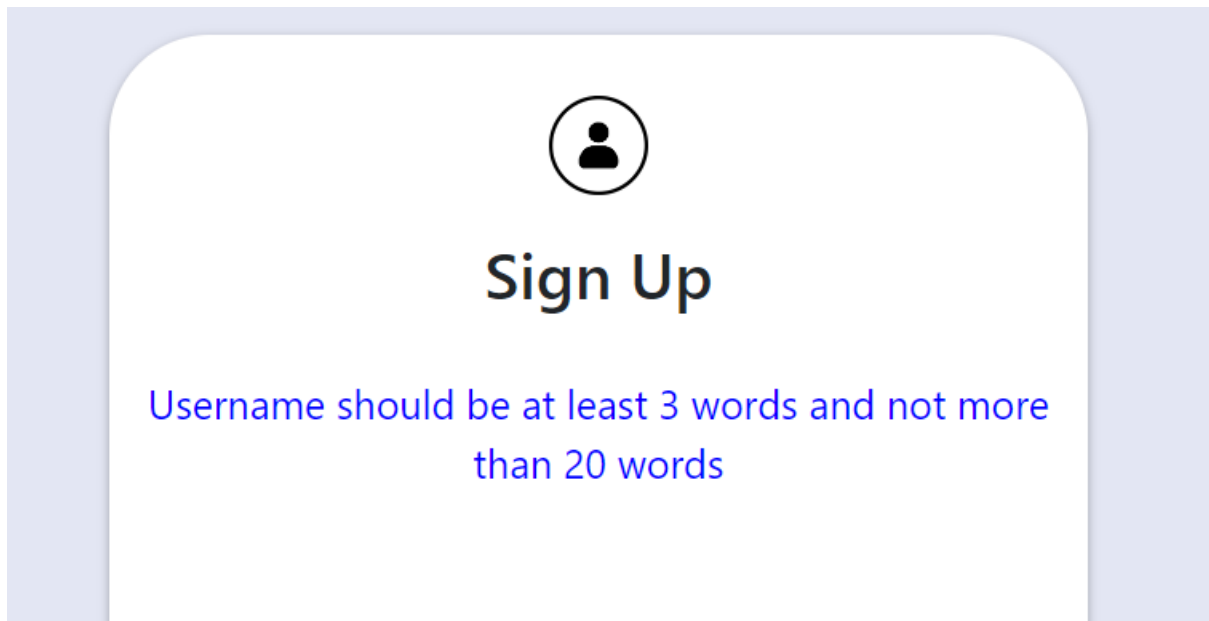


Figure a

```
if(invalidUid($username) !== false){  
    header("location: sign_up.php?error=invaliduid");  
    exit();  
}
```

Figure b

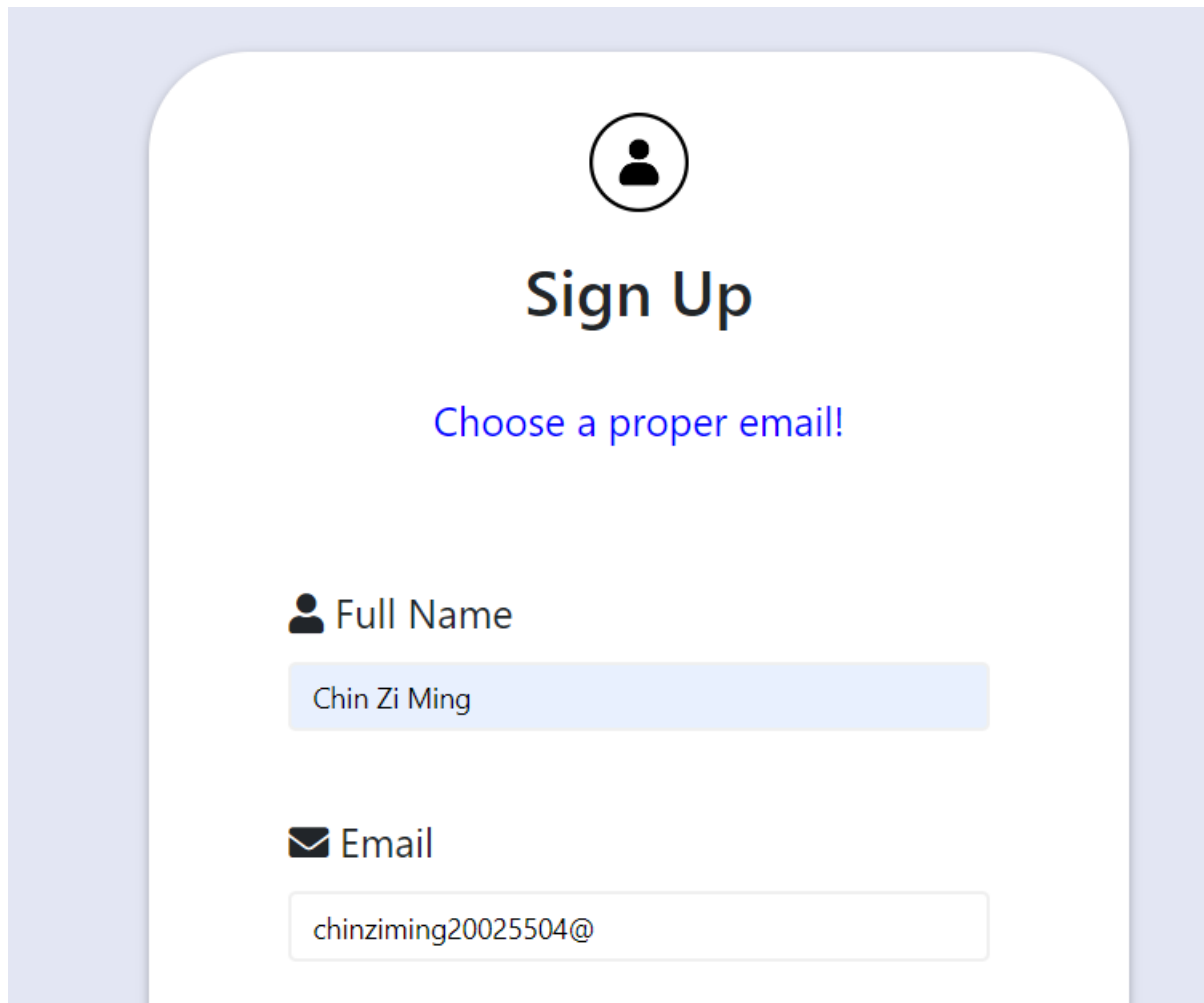
```
function invalidUid($username){  
    $result;  
    if(!preg_match("/^[a-zA-Z0-9]{3,20}$/", $username)){  
        $result = true;  
    }  
    else{  
        $result = false;  
    }  
    return $result;  
}
```


Figure c

```
else if($_GET["error"] == "invaliduid"){  
    echo"<p>Username should be at least 3 words and not more than 20 words</p>";  
}
```

Figure d


As the figure a shown, this is a username validation using php. If the user does not enter a proper username length, the page will prompt the user the message of 'Username should be at least 3 words and not more than 20 words' as shown in figure a. As shown in figure b, I use the post method to get the input from user. Then, I use the invalidUid to validate if the user has entered correct username format. Let's say the user did not enter a proper username, he or she will prompt back to the sign_up page with an error that is same as the invaliduid. I also use the exit() function if the header statement does not execute. As shown in figure c, it has an argument which allow the input username from user. It allows user's input to be passed to the function. Then, a result variable is created. We use the if statement and preg_match function to check whether the email is valid or not. The user is allowed to enter username with formats of lowercase, capital letter, number and also username between 3 to 20 letters. Hence, we can validate whether the user enters correct format of username when the system returns either true or false results. We will utilise the get function as shown in figure d to obtain the error if the user does not enter correct format of username. We will display 'Username should be at least 3 words and not more than 20 words' at the top of the sign-up form as shown in figure a above if the error equals invaliduid.






Sign Up

Choose a proper email!

 Full Name

Chin Zi Ming

 Email

chinziming20025504@

Figure a

```
if(invalidEmail($email) !== false){  
    header("location: sign_up.php?error=invalidemail");  
    exit();  
}
```

Figure b

```
function invalidEmail($email){  
    $result;  
    if(!filter_var($email, FILTER_VALIDATE_EMAIL)){  
        $result = true;  
    }  
    else{  
        $result = false;  
    }  
    return $result;  
}
```

Figure c


```

else if($_GET["error"] == "invalidemail"){
    echo"<p>Choose a proper email!</p>";
}

```

Figure d

As the figure a shown, this is an email validation using php. If the user does not enter a proper email like figure a shown, the page will prompt the user to choose a proper email. As shown in figure b, I use the post method to get the input from user. Then, I use the invalidEmail to validate if the user has entered correct email format. Let's say the user did not enter a proper email, he or she will prompt back to the sign_up page with an error that is same as the invalid email. I also use the exit() function if the header statement does not execute. As shown in figure c, it has an argument which allow the input email from user. It allows user's input to be passed to the function. Then, a result variable is created. We use the if statement and filter_var to check whether the email is valid or not. Hence, we can validate whether the user enters correct format of email when the system returns either true or false results. We will utilise the get function as shown in figure d to obtain the error if the user does not enter correct format of email. We will display 'Choose a proper email!' at the top of the sign-up form as shown in figure a above if the error equals invalid email.

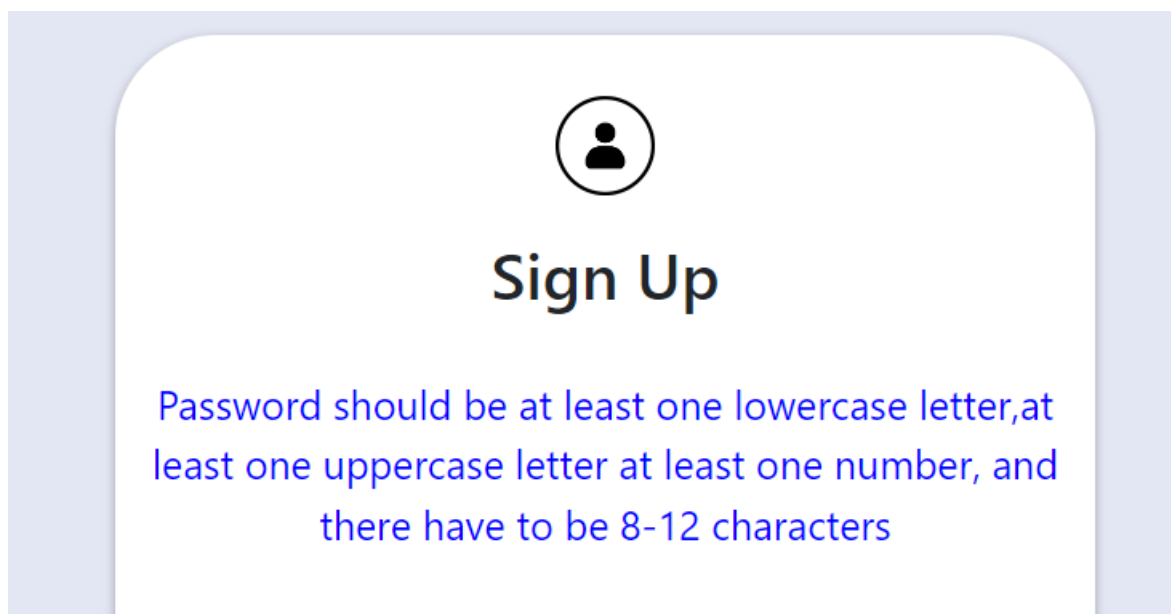


Figure a

```

if(invalidPwd($pwd) != false){
    header("location: sign_up.php?error=invalidpwd");
    exit();
}

```

Figure b

```

function invalidPwd($pwd){
    $result;
    if (strlen($pwd) <= '8') {
        $result = true;
    }
    elseif(!preg_match("#[0-9]+#", $pwd)) {
        $result = true;
    }
    elseif(!preg_match("#[A-Z]+#", $pwd)) {
        $result = true;
    }
    elseif(!preg_match("#[a-z]+#", $pwd)) {
        $result = true;
    }
    else{
        $result = false;
    }
    return $result;
}

```

Figure c

```

else if($GET["error"] == "invalidpwd"){
    echo"<p>Password should be at least one lowercase letter,at least one uppercase letter at least one number, and there have to be 8-12 characters</p>";
}

```

Figure d

As the figure a shown, this is a password validation using php. If the user does not enter a proper password format, the page will prompt the user a message 'Password should be at least one lowercase letter,at least one uppercase letter at least one number, and there have to be 8-12 characters'. As shown in figure b, I use the post method to get the input password from user. Then, I use the invalidPwd to validate if the user has entered correct password format. Let's say the user did not enter a proper password, he or she will prompt back to the sign_up page with an error that is same as the invalidpwd. I also use the exit() function if the header statement does not execute. As shown in figure c, it has an argument which allow the input password from user. It allows user's input to be passed to the function. Then, a result variable is created. We use the if statement, preg_match and strlen function to check whether the password is valid or not. Strlen function is used to check the length of the passwords while the preg_match function is used to check whether the password entered contain number, lowercase, and capital letter. Hence, we can validate whether the user enters correct format of password when the system returns either true or false results. We will utilise the get function as shown in figure d to obtain the error if the user does not enter correct format of password. We will display 'Password should be at least one lowercase letter, at least one uppercase letter at least one number, and there have to be 8-12 characters' at the top of the sign-up form as shown in figure a above if the error equals invalidpwd.

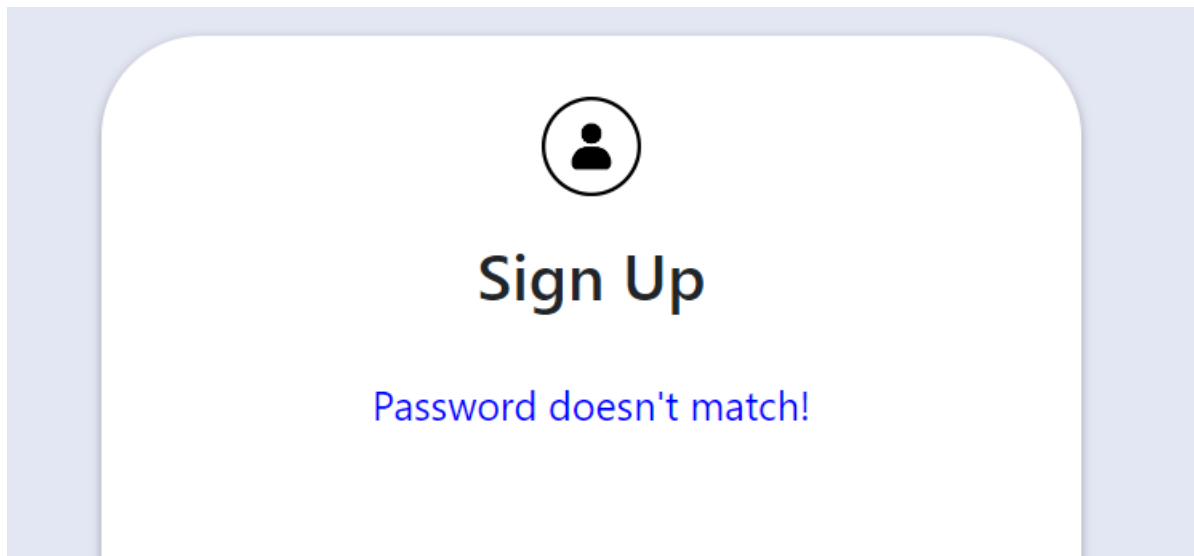


Figure a

```
if(pwdMatch($pwd, $pwdRepeat) !== false){  
    header("location: sign_up.php?error=passworddontmatch");  
    exit();  
}
```

Figure b

```
function pwdMatch($pwd, $pwdRepeat){  
    $result;  
    if($pwd != $pwdRepeat){  
        $result = true;  
    }  
    else{  
        $result = false;  
    }  
    return $result;  
}
```

Figure c

```
else if($_GET["error"] == "passworddontmatch"){  
    echo"<p>Password doesn't match!</p>";  
}
```

Figure d

As the figure a shown, this is a password validation that validate user to key in the same password while signing up using php. If the user does not enter a same password correctly, the page will prompt the user a message 'Password doesn't match'. As shown in figure b, I

use the post method to get the input password from user. Then, I use the pwdMatch to validate if the user has entered same password or not. Let's say the user did not enter a same password, he or she will prompt back to the sign_up page with an error that is same as the passworddontmatch. I also use the exit() function if the header statement does not execute. As shown in figure c, it has two arguments which allow the input password and repeated password from user. It allows user's input to be passed to the function. Then, a result variable is created. We use the if statement to check whether the password and repeated password is the same or not. Hence, we can validate whether the user enters the same password twice when the system returns either true or false results. We will utilise the get function as shown in figure d to obtain the error if the user does not enter same password twice. We will display 'Password doesn't match' at the top of the sign-up form as shown in figure a above if the error equals passworddontmatch.

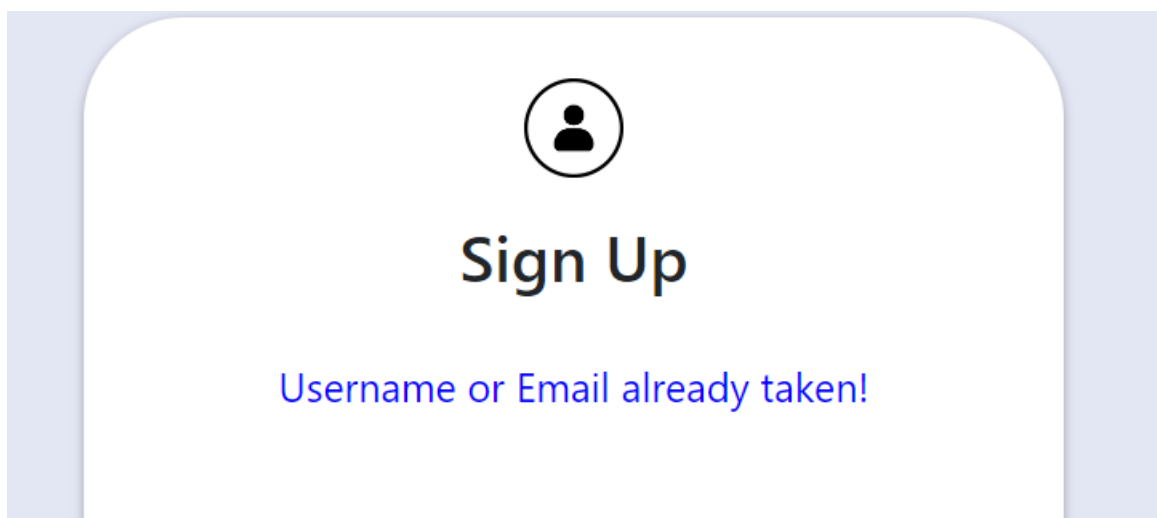


Figure a

```
dbh.inc.php
1  <?php
2
3  $serverName ="localhost";
4  $dbUserName ="root";
5  $dbPassword ="";
6  $dbName ="reden";
7
8  $conn = mysqli_connect($serverName, $dbUserName, $dbPassword, $dbName);
9
10 if(!$conn){
11     die("Connection failed: ".mysqli_connect_error());
12 }
13 ?>
```

Figure b

```

if(uidExists($conn, $username, $email) !== false){
    header("location: sign_up.php?error=usnametaken");
    exit();
}

```

Figure c

```

function uidExists($conn, $username, $email){
    $sql = "SELECT * FROM users WHERE usersUid = ? OR usersEmail = ?";
    $stmt = mysqli_stmt_init($conn);
    if(!mysqli_stmt_prepare($stmt,$sql)){
        header("location: sign_up.php?error=stmtfailed");
        exit();
    }

    mysqli_stmt_bind_param($stmt, "ss", $username, $email);
    mysqli_stmt_execute($stmt);

    $resultData = mysqli_stmt_get_result($stmt);

    if($row = mysqli_fetch_assoc($resultData)){
        return $row;
    }
    else{
        $result = false;
        return $result;
    }

    mysqli_stmt_close($stmt);
}

```

Figure d

```

else if($_GET["error"] == "usnametaken"){
    echo"<p>Username or Email already taken!</p>";
}

```

Figure e

As the figure a shown, we must determine whether the user has provided a username or email that already exists in the database in order to prevent duplication of user information in the customer's database. This means that if a user inputs a username or email that already exists in the database, they will receive a message to that consequence. Before checking the data and information from the database, we need to do the database connection as shown in figure b. We use the if statement to check whether the connection is successful, if not we use the die

function to echo the connection failed messages. As shown in figure c, I use the `uidExists` to validate if the username and email had already existed in the database after the database connection is successful. Let's say the database return that already existed the username and email, he or she will prompt back to the `sign_up` page with an error equal to `usnametaken`. I also use the `exit()` function if the header statement does not execute. As shown in figure d, it has three arguments which allow the input username and email from user. It allows user's input to be passed to the function. Then, `$conn` variable is used for the connection of database. We will create a SQL statement that selects all data from the users table in the users database where the username and user email match the values given by the user. To strengthen its security, we employ the prepare statement. The new prepare statement is initialised when we create it. To ensure that the code runs into the database without any user input, and then we add user input to execute them independently. This will stop any code from being inserted into our database. The user will be sent to the sign-up page with an error code equal to `stmtfailed` if the prepare statement fails. The `exit` function is then used to ensure that the code below the `this if` expression is not run. If the prepare statement is successful, we will pass the user's data using the `mysqli_stmt_bind_param` function. The result data will then be stored in a variable called `$resultData` after our prepare statement has been executed. Hence, we can validate whether the username and email existed in database by using the `if` statement and functions above when the system returns either true or false results. We will utilise the `get` function as shown in figure e to obtain the error if the username and email entered already existed in the database. We will display 'Username or Email already taken' at the top of the sign-up form as shown in figure a above if the error equals `usnametaken`.

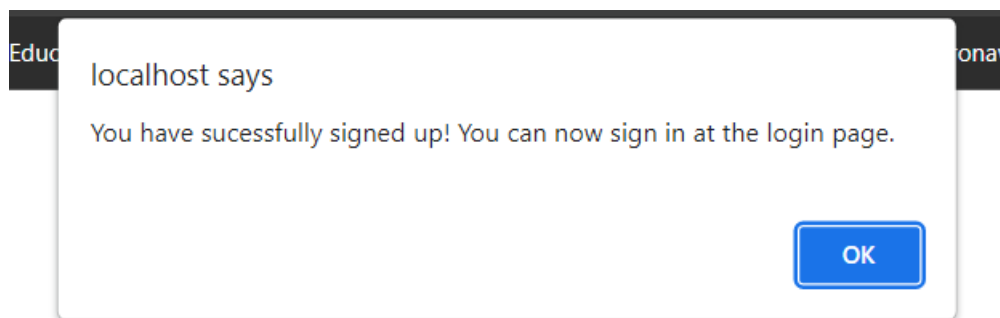


Figure a

```
createUser($conn, $name, $email, $username, $pwd);
```

Figure b

```
function createUser($conn, $name, $email, $username, $pwd){
    $sql = "INSERT INTO users(usersName, usersEmail, usersUid, usersPwd) VALUES(?,?,?,?);";
    $stmt = mysqli_stmt_init($conn);
    if(!mysqli_stmt_prepare($stmt,$sql)){
        header("location: sign_up.php?error=stmtfailed");
        exit();
    }

    $hashedPwd = password_hash($pwd, PASSWORD_DEFAULT);

    mysqli_stmt_bind_param($stmt, "ssss", $name, $email, $username, $hashedPwd);
    mysqli_stmt_execute($stmt);
    mysqli_stmt_close($stmt);
    header("location: sign_up.php?error=none");
    exit();
}
```

Figure c

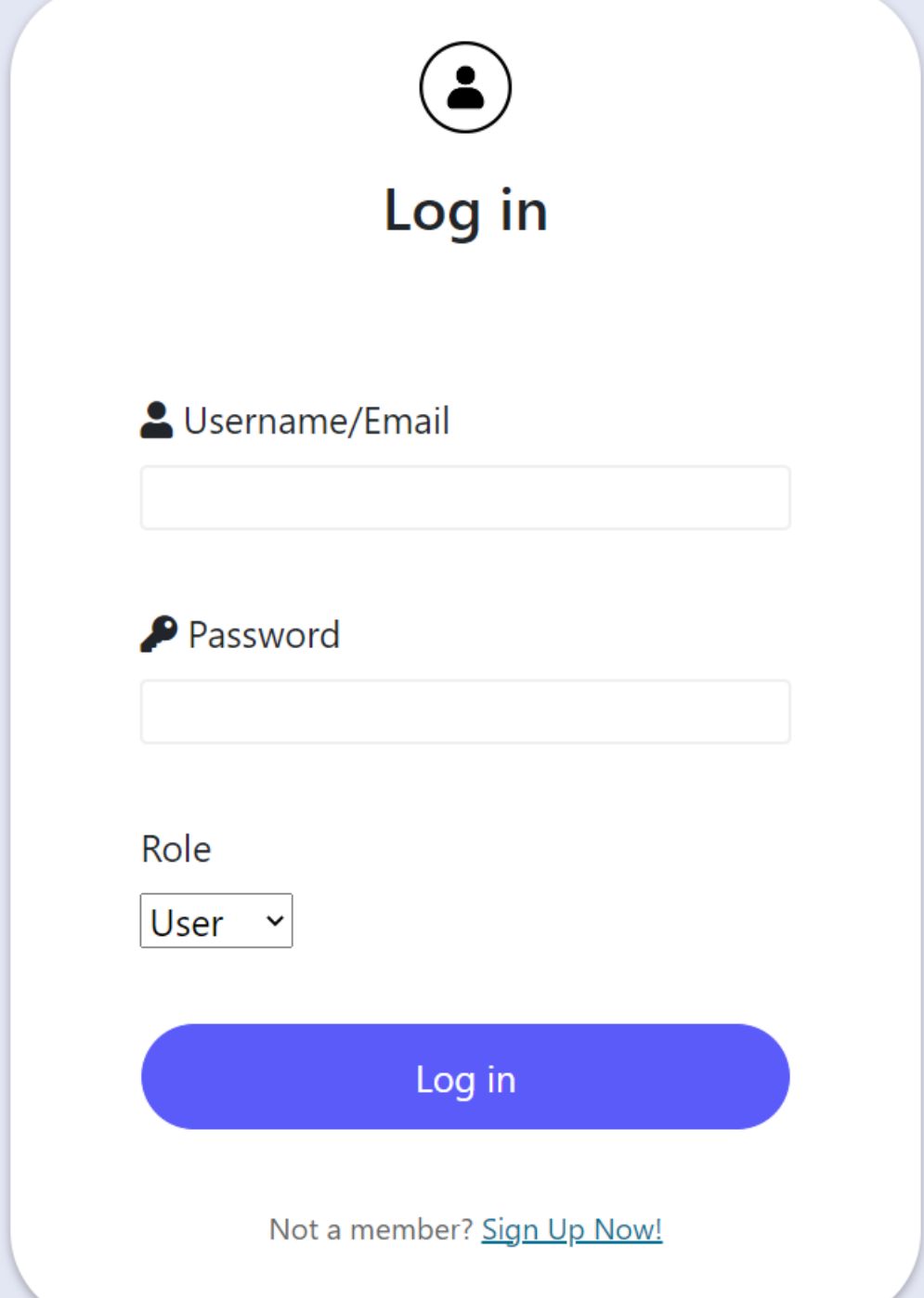
```
else if($_GET["error"] == "none"){
    echo "<script type='text/JavaScript'>alert('You have sucessfully signed up! You can now sign in at the login page.');


```

Figure d


As shown in figure a and b, when the user fills up the details correctly, I use the createUser to create a data for the user in the database. As shown in figure c, the customer information is added to our customer database using the createUser function. It has five arguments. The other is is used to retrieve the data the user entered in the form while the \$conn is used to establish a connection to the database. In order to insert customer input into the users table that we have created in our database, we create a sql code. To strengthen its security, we employ the prepare statement. The new prepare statement is initialised when we create it. To ensure that the code runs into the database without any user input, and then we add user input to execute them independently. This will stop any code from being inserted into our database. The user will be sent to the sign-up page with an error code equal to stmtfailed if the prepare statement fails. The exit function is then used to ensure that the code below the this if expression is not run. Furthermore, I use the password_hash function to hide the user's password and to prevent the actual password to be display in the database. Therefore, we may run the statement that adds user information to our database. The statement is then closed, and the user is directed to the sign-up page with a none error message. Next in figure d, I will use the get method to get the error and let's say the error is same as the none, thus I will use javascript alert to prompt the user that they have successfully sign up and prompt them to login using the sign up details they have inputed.


Login function includes Admin login and User login

A login form with a white background and rounded corners, set against a light blue background. At the top center is a circular icon containing a black silhouette of a person. Below this is the text "Log in" in a large, bold, black font. The form contains three input fields: a text field for "Username/Email" with a person icon, a text field for "Password" with a key icon, and a dropdown menu for "Role" currently showing "User". Below these fields is a large, rounded blue button with the text "Log in" in white. At the bottom, there is a link that says "Not a member? [Sign Up Now!](#)".



Log in

 Username/Email

 Password

Role

User ▾

Log in

Not a member? [Sign Up Now!](#)

Figure a


```


</div>
<form class = "form" action="login.inc.php" method="post">
  <div class="field">
    <label><i class="fa fa-user"></i> Username/Email</label>
    <input type="text" name="uid" placeholder=""><br>
  </div>
  <div class="field">
    <label><i class="fa fa-key" aria-hidden="true"></i> Password</label>
    <input type="password" name="pwd" placeholder=""><br>
  </div>
  <div class="field">
    <label>Role</label>
    <select name="role">
      <option>User</option>
      <option>Admin</option>
    </select>
  </div>
  <div class="field">
    <button type="submit" name="submit">Log in</button>
  </div>
  <div class = "signup_link">
    Not a member? <a href = "sign_up.php">Sign Up Now!</a>
  </div>
</form>

</div>

```


Figure b


As the figure a shown, this is the user interface for the user to log in. We require the user to sign up first so that they can log in on our website. As shown in the figure b, I have integrated the login.inc.php file in the form action which also act as a function file. Next, I'm using the post method as I want to get the input from the user securely when they are filling up the personal details at the log in page rather than using get method to handle the input from user. As a result, we may retrieve the user's input and determine whether the user has an account or not.



Log in

Fill in all fields!

 Username/Email

 Password

Role

User ▾

Log in

Figure a

```
if($role == "User"){  
    loginUser($conn, $username, $pwd);  
}else if($role == "Admin"){  
    loginAdmin($conn, $username, $pwd);  
}
```

Figure b

```
function loginUser($conn, $username, $pwd){
    $uidExists = uidExists($conn, $username, $username);

    if($uidExists === false){
        header("location: sign_in.php?error=wronglogin");
        exit();
    }

    $pwdHashed = $uidExists["usersPwd"];
    $checkPwd = password_verify($pwd, $pwdHashed);

    if($checkPwd === false){
        header("location: sign_in.php?error=wronglogin");
        exit();
    }
    else if($checkPwd === true){
        session_start();
        $_SESSION["userid"] = $uidExists["usersId"];
        $_SESSION["useruid"] = $uidExists["usersUid"];
        echo '<script type="text/JavaScript">alert("You have successfully login! You will now be redirected back to the home page"); window.location.href="index.php"</script>';
        exit();
    }
}
```

Figure c

```
else if($_GET["error"] == "wronglogin"){
    echo"<p>Incorrect login information</p>";
}
```

Figure d

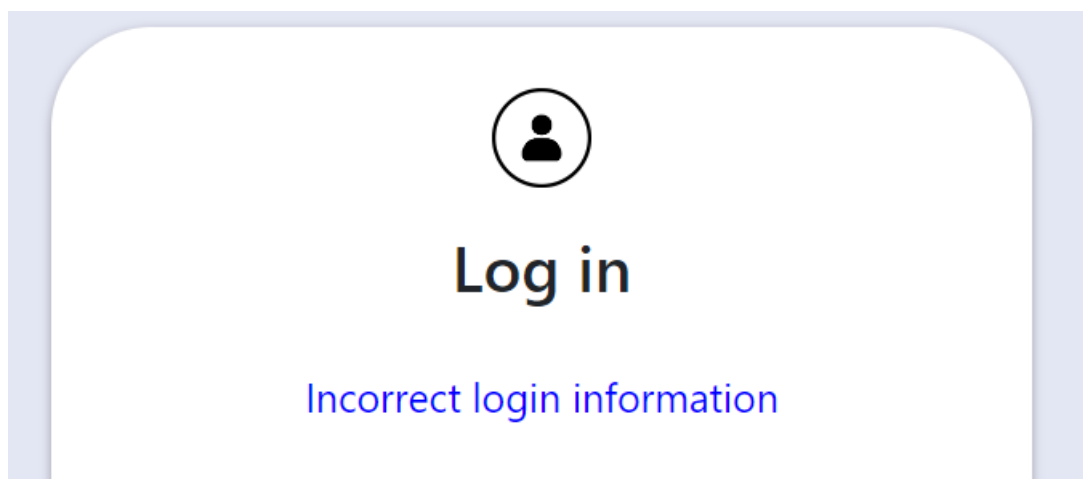


Figure e

As shown in figure a, the log in validations is similar to the sign-up validations. But in the log in page, to determine if the user chose to log in as a user or as an administrator, we will utilise an if statement. If they decide to log in as user, we will do it by using a user-defined function named loginUser. If they decide to log in as the administrator, we will do it by calling a user-defined function named loginAdmin. In loginUser function, it has three arguments which are \$conn, \$username and \$password. The \$conn is used for database connection and for the rest are used to retrieve the username and password. We use the uidExists function to check whether the inputs enter already existed in the database. If the inputs don't exist in the database, the user will be prompted back to the sign in page where an error message wronglogin will be given and stop the code underneath it from running by using the exit method. The variable \$pwdHashed will be used to obtain the user password from the database if the user enters an existing username or email address. We can access the

passwords that are in the same row as the username or email that the user entered, because we use associative arrays in the `uidExists` method and associative arrays employ names for each index inside the array. The built-in PHP function `password_verify` will then be used to determine whether the password entered by the user matches the password stored in the database or not. If the inputs are not match, the user will be prompted back to the sign in page where an error message `wronglogin` will be given and stop the code underneath it from running by using the `exit` method. If the information is match, we start a session and generate two session variables: one with the username and one with the user id. These are the only two database values that can never be duplicated and also unique. The user will then be directed to the home page after we utilise the `javascript alert` keyword to confirm their login. As shown in the figure d, let's say the error is same as the `wronglogin`, the users will be prompted the message of 'Incorrect login information' as shown in figure e.

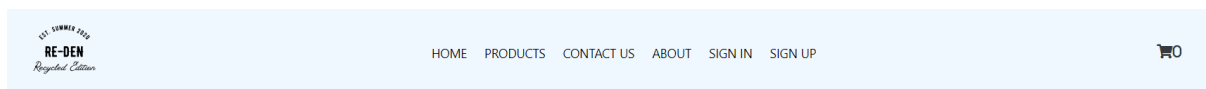


Figure a

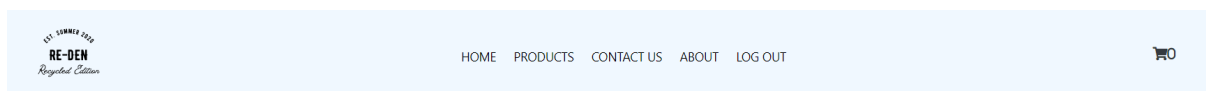


Figure b

```
<?php
if(isset($_SESSION["userid"])){
    echo"<li><a href='logout.inc.php'>LOG OUT</a></li>";
}
else{
    echo"<li><a href='sign_in.php'>SIGN IN</a></li>";
    echo"<li><a href='sign_up.php'>SIGN UP</a></li>";
}
?>
```

Figure c

When the user login successfully, the navigation bar will change as shown in figure a to figure b. We use the `if` statement to check whether the `$_SESSION["userid"]` is set or not. If it is set, it indicates that the user has previously logged in since, when a user logs in, a session variable containing their username is created. As a result, we will show the user the log out navigator rather than the sign in and sign-up navigators.

Session timeout

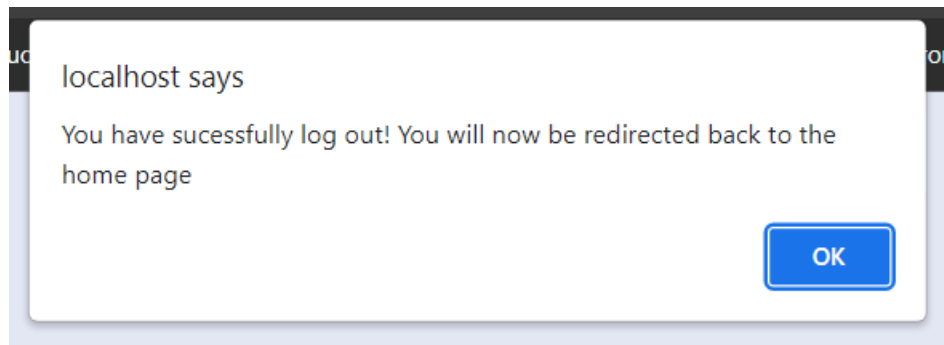


Figure a

```
<?php
session_start();
unset($_SESSION['userid']);
$_SESSION = array();
session_unset();
setcookie(session_name(), '', time() - 2592000, '/');
session_destroy();
echo '<script type="text/JavaScript">alert("You have sucessfully log out! You will now be redirected back to the home page"); window.location.href='
?>
```

Figure b

As shown in figure a, it shows that user have successfully log out. But how does it work? The logout.inc.php which acts as a functional file will start a session and then unset the session variables that were defined earlier as long as the user sign in as shown in figure b. Therefore, we utilize the session_unset() function to unset all the session variables that were registered. The cookie is then cleared by setting the value to "" and the expiration date to any exact time. Finally, we terminate the session using the session destroy function. Once the user has successfully logged out, we utilise the javascript alert keyword to notify them and direct them back to the home page.



Figure a

```
if (isset($_SESSION['userid']))
{
    $userid = htmlspecialchars($_SESSION['userid']);

    if (isset($_SESSION['LAST_ACTIVITY']) && (time() - $_SESSION['LAST_ACTIVITY'] > 900)) {
        destroy_session_and_data();

        echo '<script type="text/JavaScript">alert("session time out");</script>';
    }
    else
        $_SESSION['LAST_ACTIVITY'] = time(); // update last activity time stamp
}
```

Figure b

```
function destroy_session_and_data()
{
    //session_start();
    //$_SESSION = array();

    unset($_SESSION['userid']);
    $_SESSION = array();
    session_unset();
    setcookie(session_name(), '', time() - 2592000, '/');
    session_destroy();
}
```

Figure c

The scenario will happen in figure a when the user is not active after 15 minutes as set in figure b. We use the if statement to check whether the `$_SESSION["userid"]` is set or not. If it is set, it indicates that the user has previously logged in since, when a user logs in, a session variable containing their username is created. The session variable will be kept in a variable called `$userid`. The if statement will then be used to determine whether or not the user's most last activity occurred more than 15 minutes ago. It is 900 seconds for 15 minutes since we must utilise seconds in this situation. The user will be logged out using the user-defined function call `destroy session and data` if their last activity has been for more than 15 minutes as shown in figure b. The user will be informed that their session has timed out using the JavaScript alert keyword as shown in figure a. Furthermore as shown in figure c, the function `destroy_session_and_data()` works the same as log out function.

Admin function: Edit and Delete Customer and Product details

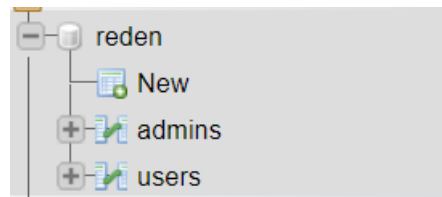


Figure a


```
Create TABLE admins(  
adminsId int(11) PRIMARY KEY AUTO_INCREMENT NOT NULL,  
adminsName varchar(128) NOT NULL,  
adminsEmail varchar(128) NOT NULL,  
adminsUid varchar(128) NOT NULL,  
adminsPhone varchar(128) NOT NULL,  
adminsPwd varchar(128) NOT NULL  
);  
  
INSERT INTO admins (adminsName, adminsEmail,adminsUid, adminsPhone, adminsPwd)VALUES('CHAI JING HANG', 'jinghangchai95@gmail.com','CJH', '+601022197
```

Figure b

As shown in figure a, we create a database for the admin, and we use the sql code as shown in figure b to create the database table. It actually works the same as user table that have been created earlier.



Log in

 Username/Email



 Password

Role

Log in

Not a member? [Sign Up Now!](#)

Figure a

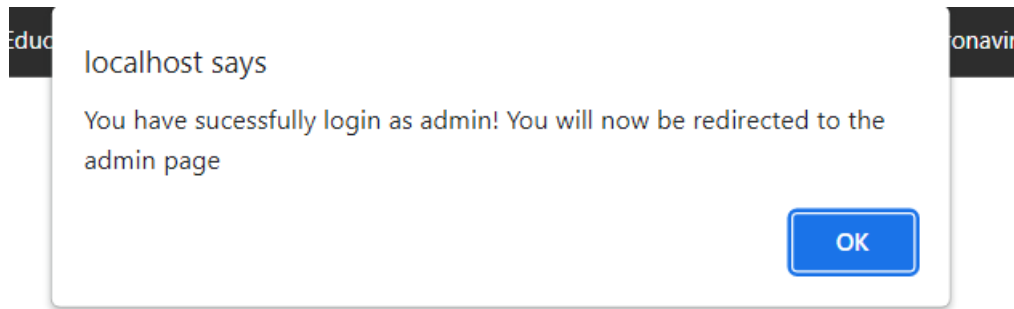


Figure b

```
if($role == "User"){
    loginUser($conn, $username, $pwd);
}else if($role == "Admin"){
    loginAdmin($conn, $username, $pwd);
}
```

Figure c

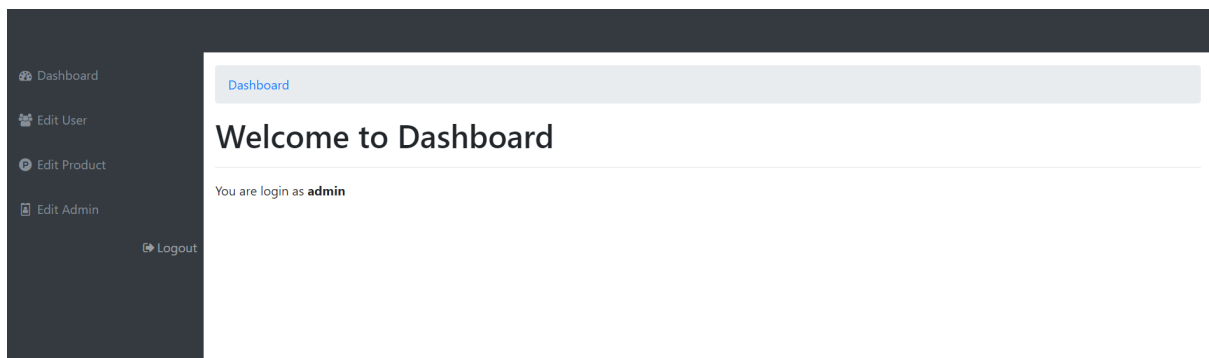


Figure d

Customer Edit and Update

Create New Client

Total Records -

Customer ID	Customer Name	Customer Email	Customer Username	
1	CZ	chinziming20025504@gmail.com	CZ2	<button>Update</button> <button>Delete</button>
2	ZC21	chinziming2002@gmail.com	aaaaa	<button>Update</button> <button>Delete</button>
3	CHIN ZI MING	zimingchin.vivahomes@gmail.com	Max	<button>Update</button> <button>Delete</button>

Figure e

In order to be an admin, the user need to choose the role of admin and then fill up the fields needed as shown in figure a. For this admin login, it is used for the admin to edit product and edit user as shown in figure d. As you can see in figure e, the user data is being displayed

where the data were retrieved from the database as shown in the table. The admin can update and delete the customer information and the admin can create a client if needed.

```
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "reden";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "SELECT * FROM users";
$result = $conn->query($sql);

if (!$result) {
    die("Invalid query: " . $conn->error);
}
```

Figure f

```
while($row = $result->fetch_assoc()) {
    echo "<tr>
        <td>" . $row["usersId"] . "</td>
        <td>" . $row["usersName"] . "</td>
        <td>" . $row["usersEmail"] . "</td>
        <td>" . $row["usersUid"] . "</td>
        <td>
            <a class='btn btn-primary btn-sm' href='edit_user.php?id=$row[usersId]''>Update</a>
            <a class='btn btn-danger btn-sm' href='delete_user.php?id=$row[usersId]''>Delete</a>
        </td>
    </tr>";
}
```

Figure g

In order to create table shown in figure e, we need to create the database connection. We use the variable \$conn to connect the database with mysqli function. We use if statement to test if there is a connection error, if yes, a die function will be executed showing that connection fail. Then, we use sql command to retrieve data from the database user table. Again, I use the if statement to test if there is any error and if yes, a die function will be used where it will show the message of invalid query. In the figure g above shown, the user information that were retrieved earlier will go into the while loop to print all the user details in the table. The while loop function to read each row in the database and then store it as array inside the variable \$row as associative array is used to name each index in the array. Thus, we can enter

the column name to get the data from each row. As the figure g shown, the user will be directed to the edit user page by clicking the update button, while they will be directed to the delete user page by clicking the delete user button. Each row's user id will be stored in one of them using an id variable. It is crucial because we only want to update and delete the user's data for that specific row when we update and delete. The edit and update pages' purposes for the user and the admin are essentially the same. Just the validation sets them apart from one another.

Edit User

Update Client Details

Name	<input type="text" value="Chin Zi Ming"/>
Billing Name	<input type="text" value="Chin Zi Ming"/>
Phone	<input type="text" value="0162350735"/>
Address	<input type="text" value="11, Jalan Remia 3, Taman Kota Jaya"/>
City	<input type="text" value="Kota tinggi"/>
State	<input type="text" value="Johor"/>
Zip	<input type="text" value="81900"/>
<div><input type="button" value="Update"/> <input type="button" value="Cancel"/></div>	

Figure a

```
if( $_SERVER['REQUEST_METHOD'] == 'GET'){

    if(! isset($_GET["id"])){
        header("location: customer_edit.php");
        exit;
    }

    $id = $_GET["id"];

    $sql = "SELECT * FROM users WHERE usersId=$id";
    $result = $conn->query($sql);
    $row = $result->fetch_assoc();

    if(!$row){
        header("location: customer_edit.php");
        exit;
    }

    $name = $row["userName"];
    $fname = $row["billingName"];
    $phone = $row["usersPhone"];
    $address = $row["usersAddress"];
    $city = $row["usersCity"];
    $state = $row["usersState"];
    $zip = $row["usersZip"];
```

Figure b

```
<label class="col-sm-3 col-form-label">Name</label>
<div class="col-sm-6">
|   <input type="text" class="form-control" name="name" value="<?php echo $name; ?>">
</div>
```

Figure c

```
$sql = "UPDATE users " .
"SET userName = '$name', billingName = '$fname', usersPhone = '$phone', usersAddress = '$address', usersCity = '$city', usersState = '$state', use
"WHERE usersId = $id";
```

Figure d

As you can see in the figure a above, a form will already have the client data that was taken from the database. As a result, it is simpler for the administrators to see what needs to change. In addition, since the username and email are the only variables that can be modified and are needed to uniquely identify each customer, we do not permit admin to modify them. In order to preserve the privacy of our customers, we also do not permit admin to modify their passwords or billing information. As shown in the figure b and based on the selected row, we must extract the specific client information. In order for us to identify the customer id of each row, keep in mind that we have established an id variable that is used to record the user's id for each row at the link in the update button and the delete button. The user's ID, which we store in the id variable at the link in the update button, will first be obtained using the get method. We must stop the admin from modifying the data if we do not receive the id, therefore we will send them to the previous page and then use the exit method to stop the code below from running. If the ID is appropriately obtained, all the data in the table that are in the same row as that user id are selected. The data that we have just retrieved is then stored using an associative array. We can obtain the data for each column by entering the column name inside the bracket because we are using an associative array, which uses names for each index inside the array as shown in figure b. The data that we have pulled from the database is printed out at each field in the form using the php echo keyword as shown in figure c. As the figure d shown, the PHP code is identical for creating a new customer as well as updating the client's data. The only distinction between updating an existing customer and generating a new one is that we use update in our sql code to replace the database's existing data with the new data rather than insert as shown in figure d.

Delete User

```
<?php
if(isset($_GET["id"])){
    $id = $_GET["id"];

    $servername = "localhost";
    $Username = "root";
    $password = "";
    $database = "reden";

    $conn = new mysqli($servername, $Username, $password, $database);

    $sql = "DELETE FROM users WHERE usersId=$id";
    $conn->query($sql);
}
header("location: customer_edit.php");
exit;
?>
```

As the figure a shown. Similar to the update page, depending on the row that is chosen, we must retrieve the specific client information. In order to identify the customer id for each row, we utilise the same id variable that is used to keep the user id for each row at the link in the delete button. We use the sql statement to delete all the user data that is located in the same row as that user after we get the user id for that row.

Product Page & Shopping Cart

```
CREATE TABLE IF NOT EXISTS `redenproduct` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(200) NOT NULL,
  `description` text NOT NULL,
  `price` decimal(7,2) NOT NULL,
  `rrp` decimal(7,2) NOT NULL DEFAULT '0.00',
  `quantity` int(11) NOT NULL,
  `img` text NOT NULL,
  `dateadded` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=utf8;

INSERT INTO `redenproduct` (`id`, `name`, `description`, `price`, `rrp`, `quantity`, `img`, `dateadded`) VALUES
(1, 'Birdy Denim Handbag', '<p>Customized denim handbag with birds, leaves and fruits pictures.</p>\r\n<h3>Functionality</h3>\r\n<ul>\r\n<li>For casual use.</li>\r\n<li>A',
(2, 'Office Denim Handbag', '<p>Customized denim handbag with office working style.</p>\r\n<h3>Functionality</h3>\r\n<ul>\r\n<li>For office use.</li>\r\n<li>Alternative f',
(3, 'Ancient Denim Handbag', '<p>Customized denim handbag with oldschoool design.</p>\r\n<h3>Functionality</h3>\r\n<ul>\r\n<li>For classical use.</li>\r\n<li>Alternative f',
(4, 'Jewel Denim Handbag', '<p>Customized denim handbag with oem Jewelleries.</p>\r\n<h3>Functionality</h3>\r\n<ul>\r\n<li>For ootd use.</li>\r\n<li>Alternative for sling',
(5, 'Luxurious Denim Handbag', '<p>Customized denim handbag luxurious and advanced feels.</p>\r\n<h3>Functionality</h3>\r\n<ul>\r\n<li>For occasional use.</li>\r\n<li>Alt',
(6, 'Feather Denim Handbag', '<p>Customized denim handbag with handcrafted feathers embroidery.</p>\r\n<h3>Functionality</h3>\r\n<ul>\r\n<li>For work out use.</li>\r\n<li>
```

Figure a



















		id	name	description	price	rrp	quantity	img	dateadded
<input type="checkbox"/>	 Edit	 Copy	 Delete	1 Birdy Denim Handbag	<p>Customized denim handbag with birds, leaves and...	168.80	188.80	10 re-den_001.jpg	2020-03-13 17:55:22
<input type="checkbox"/>	 Edit	 Copy	 Delete	2 Office Denim Handbag	<p>Customized denim handbag with office working st...	188.80	200.00	34 re-den_002.jpg	2020-03-13 18:52:49
<input type="checkbox"/>	 Edit	 Copy	 Delete	3 Ancient Denim Handbag	<p>Customized denim handbag with oldschoool design....	155.50	180.00	23 re-den_003.jpg	2020-03-13 18:47:56
<input type="checkbox"/>	 Edit	 Copy	 Delete	4 Jewel Denim Handbag	<p>Customized denim handbag with oem Jewelleries.<...	228.80	250.00	7 re-den_004.jpg	2020-03-13 17:42:04
<input type="checkbox"/>	 Edit	 Copy	 Delete	5 Luxurious Denim Handbag	<p>Customized denim handbag luxurious and advanced...	299.99	399.99	15 re-den_005.jpg	2020-03-13 17:33:23
<input type="checkbox"/>	 Edit	 Copy	 Delete	6 Feather Denim Handbag	<p>Customized denim handbag with handcrafted feath...	129.99	150.00	21 re-den_006.jpg	2020-03-13 17:21:08

Figure b

I use this table to store all our redenproducts, along with the id, name, description, price, rrp price, quantity, image, and the dateadded columns as shown in figure a. After I inserted the SQL scripts into the database, it should look like the figure b above.

```

dbcontroller.php
1  <?php
2  function pdo_connect_mysql() {
3      // Update the details below with your MySQL details
4      $hostname = 'localhost';
5      $username = 'root';
6      $password = '';
7      $dbname = 'redentableproduct';
8      try {
9          return new PDO('mysql:host=' . $hostname . ';dbname=' . $dbname . ';charset=utf8', $username, $password);
10     } catch (PDOException $exception) {
11         // If there is an error with the connection, stop the script and display the error.
12         exit('Failed to connect to database!');
13     }
14 }
15
16 ?>

```

Figure a

As shown in figure a, I have created a dbcontroller.php file that I will use it for the cart which is the database connection function.

```

index.php
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8">
5      <title>REDEN</title>
6      <link href="style3.0.css" rel="stylesheet" type="text/css">
7      <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.1/css/all.css">
8  </head>
9  <body class= 'body'>
10 <?php
11 session_start();
12 error_reporting(E_ERROR | E_WARNING | E_PARSE);
13 include 'dbcontroller.php';
14 $pdo = pdo_connect_mysql();
15
16 $page = isset($_GET['page']) && file_exists($_GET['page'] . '.php') ? $_GET['page'] : 'home';
17 include $page . '.php';
18
19 ?>
20
21 </body>
22 </html>

```

Figure b

As shown in figure b, I have created an index.php file that will be the main page which can access to related pages. I use the session_start() function that enable me to store the products that will be added to the cart. The script above will be connected to the MySQL using the database connection function that I have created in the dbcontroller.php file. I have also created a basic routing to check whether the GET request variable \$_GET['page'] exists or not. Let's say if the page does not exist, the default page will be our home page and if it exists, that will direct the user to the specific page requested.

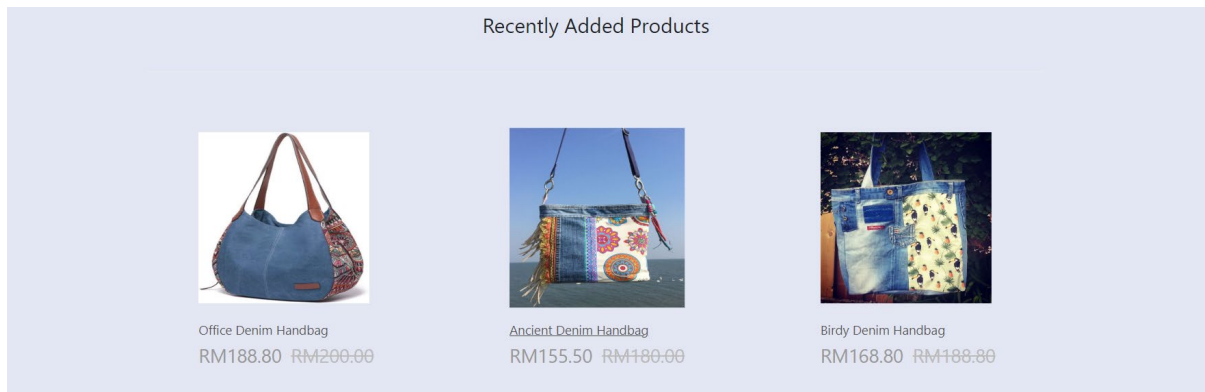


Figure a

```
home.php
1  <?php
2  $stmt = $pdo->prepare('SELECT * FROM redenproduct ORDER BY dateadded DESC LIMIT 3');
3  $stmt->execute();
4  $recently_added_products = $stmt->fetchAll(PDO::FETCH_ASSOC);
5  ?>
6
```

Figure b

As the figure a above shown, these are the 3 recently added products that will be display in our home page. I'm using the php codes shown in figure b that will execute the SQL query to retrieve the three most recently added products from the database. The query will be order by dateadded and the amount will be limited by 3. Then, the results will be stored om the \$recently_added_products variable that act as an associated array.

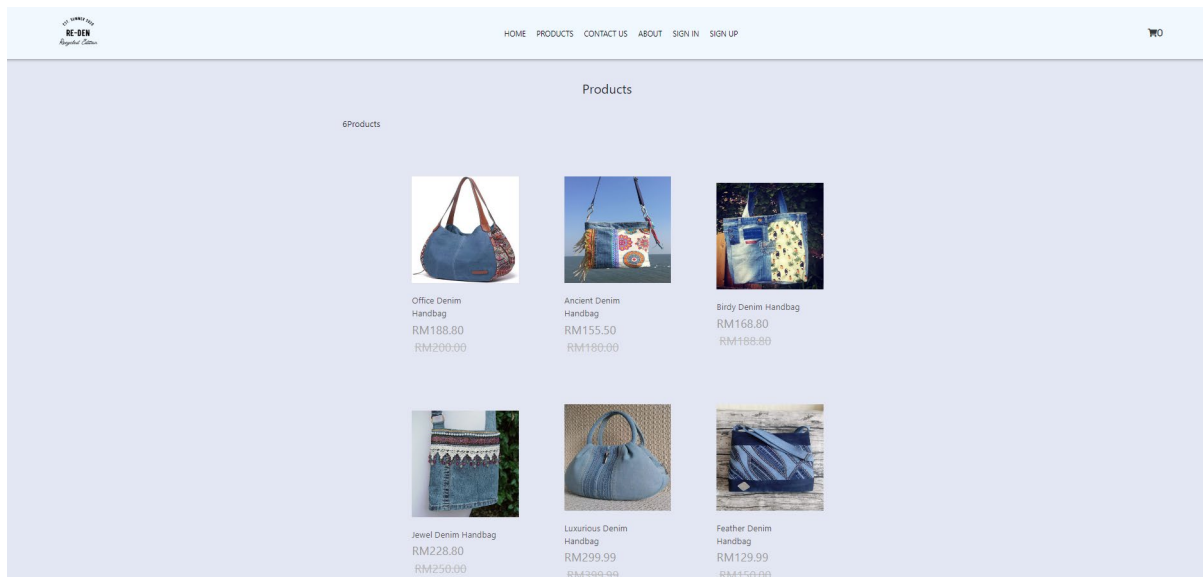


Figure a

```

products.php
1
2 <?php
3
4 include_once "dbcontroller.php";
5 $pdo = pdo_connect_mysql();
6 $num_products_on_each_page = 6;
7
8 $current_page = isset($_GET['p']) && is_numeric($_GET['p']) ? (int)$_GET['p'] : 1;
9
10 $stmt = $pdo->prepare('SELECT * FROM redenproduct ORDER BY dateadded DESC LIMIT ?,?');
11
12 $stmt->bindValue(1, ($current_page - 1) * $num_products_on_each_page, PDO::PARAM_INT);
13 $stmt->bindValue(2, $num_products_on_each_page, PDO::PARAM_INT);
14 $stmt->execute();
15
16 $products = $stmt->fetchAll(PDO::FETCH_ASSOC);
17
18 $total_products = $pdo->query('SELECT * FROM redenproduct')->rowCount();
19 ?>

```

Figure b

I have created the products page for the user to browse through the products as shown in the figure a. I have set a limitation of displaying specific products on this products page and also add pagination that will allow the users to navigate between pages. As shown in the figure b, I'm using \$num_products_on_each_page variable to limit the numbers of product display. I will utilise a GET request to find out which page the user is now on. This will display in the URL like index.php?page=products&p=1 etc., and in our PHP script, I can retrieve the parameter p using the \$_GET['p'] variable. The code will run a query to obtain the restricted products from the database, supposing the request is legitimate. Furthermore, I will use the code \$total_products = \$pdo->query('SELECT * FROM redenproduct')->rowCount(); shown in figure to show that the total number of products display in the top left products.php page.

```

<div class="buttons">
  <?php if ($current_page > 1): ?>
    <a href="index.php?page=products&p=<?=$current_page-1?>">Prev</a>
  <?php endif; ?>
  <?php if ($total_products > ($current_page * $num_products_on_each_page) - $num_products_on_each_page + count($products)): ?>
    <a href="index.php?page=products&p=<?=$current_page+1?>">Next</a>
  <?php endif; ?>
</div>

```

Figure a

As shown in figure a, only when there are more products overall than the variable \$num products on each page, therefore the user will see the Next and Prev buttons.

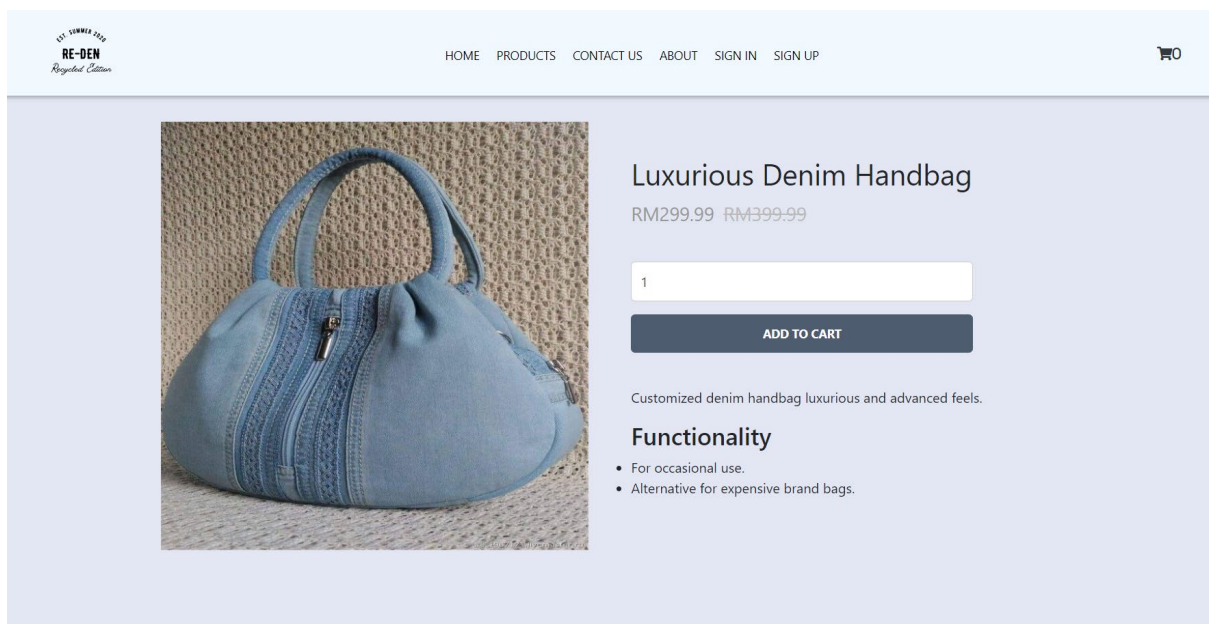


Figure a

```

<?php
if (isset($_GET['id'])) {
    $stmt = $pdo->prepare('SELECT * FROM redenproduct WHERE id = ?');
    $stmt->execute([$_GET['id']]);
    $product = $stmt->fetch(PDO::FETCH_ASSOC);
    if (!$product) {
        exit('Product does not exist!');
    }
} else {
    exit('Product does not exist!');
}
?>

```

Figure b

I have created the product page shown in figure a above. The GET request ID variable, which identifies a specific product, will be used to display all of its details on the product page. users have access to the description, image, and price. With a click of a button, the users can

adjust the quantity and add the item to their cart. The GET request's requested id variable will be verified by the code shown in figure b above. The code will proceed to obtain the product from our database's redenproduct table if the option is provided. The code will print a straightforward error and the exit() function will stop the script from running further and display the error if the product doesn't exist in the database.

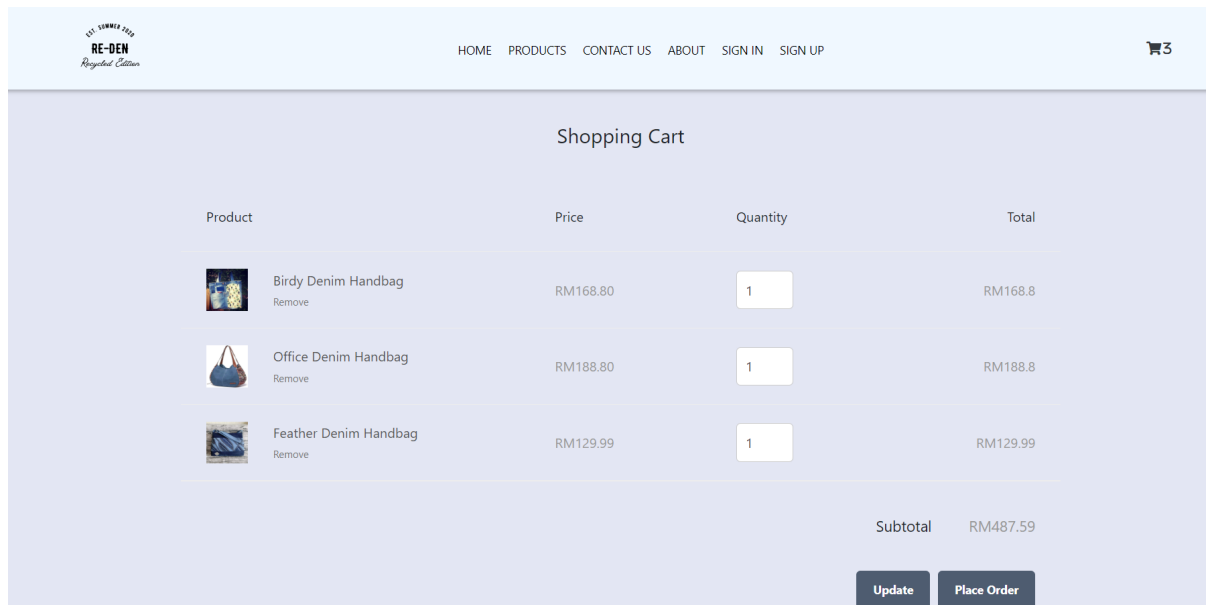


Figure a

```

1  <?php
2
3  if (isset($_POST['product_id'], $_POST['quantity']) && is_numeric($_POST['product_id']) && is_numeric($_POST['quantity'])) {
4      $product_id = (int)$_POST['product_id'];
5      $quantity = (int)$_POST['quantity'];
6      $stmt = $pdo->prepare('SELECT * FROM redenproduct WHERE id = ?');
7      $stmt->execute([$POST['product_id']]);
8      $product = $stmt->fetch(PDO::FETCH_ASSOC);
9      if ($product && $quantity > 0) {
10         if (isset($_SESSION['cart']) && is_array($_SESSION['cart'])) {
11             if (array_key_exists($product_id, $_SESSION['cart'])) {
12                 $_SESSION['cart'][$product_id] += $quantity;
13             } else {
14                 $_SESSION['cart'][$product_id] = $quantity;
15             }
16         } else {
17             $_SESSION['cart'] = array($product_id => $quantity);
18         }
19     }
20     header('location: index.php?page=cart');
21     exit;
22 }
23

```

Figure b

I have created a cart.php file as shown in the figure a above, the user can view a list of the items they have put to their shopping cart on the cart page. They will be able to adjust the quantities and remove products. The PHP session variables are used in the code as shown in figure b above. For instance, when a user switches to another page, I use PHP sessions to remember the products in the shopping basket so that they remain there until the session ends. The code above will verify whether a product was added to the shopping cart. You will notice that I made an HTML form if you return to the product.php file. I am looking for certain form

values, and if the product is found, I can verify it by choosing it from the database's `redenproduct` table. We can add several items to the shopping cart by using the related array of products in the session variable `cart`. The product ID will serve as the array key and the quantity as the array value. All we need to do to update a product if it already exists in the shopping cart is change the amount.

```
if (isset($_GET['remove']) && is_numeric($_GET['remove']) && isset($_SESSION['cart']) && isset($_SESSION['cart'][$_GET['remove']])) {  
    unset($_SESSION['cart'][$_GET['remove']]);  
}
```

Figure a

As shown in figure a, the customer will be able to delete a product from the cart on the cart page. When the button is clicked, we may perform a GET request to determine which item to remove from the shopping cart. For instance, if we have an item with the ID 1, the following URL will delete it: http://localhost/assignment_web/index.php?page=cart&remove=1.

```
if (isset($_POST['update']) && isset($_SESSION['cart'])) {  
    foreach ($_POST as $k => $v) {  
        if (strpos($k, 'quantity') !== false && is_numeric($v)) {  
            $id = str_replace('quantity-', '', $k);  
            $quantity = (int)$v;  
            if (is_numeric($id) && isset($_SESSION['cart'][$id]) && $quantity > 0) {  
                $_SESSION['cart'][$id] = $quantity;  
            }  
        }  
    }  
    header('location: index.php?page=cart');  
    exit;  
}  
  
if (isset($_POST['placeorder']) && isset($_SESSION['cart']) && !empty($_SESSION['cart'])) {  
    header('Location: billing_address.php');  
    exit;  
}
```

Figure b

As shown in figure b, the items in the shopping cart will iterate and the amounts will be updated by the code above. The quantity changes can be made by the user on the cart page. The code will use the name of the Update button to determine when to use a POST request to update the quantities.

```

if (isset($_POST['placeorder']) && isset($_SESSION['cart']) && !empty($_SESSION['cart'])) {
    header('Location: billing_address.php');
    exit;
}

```

Figure a

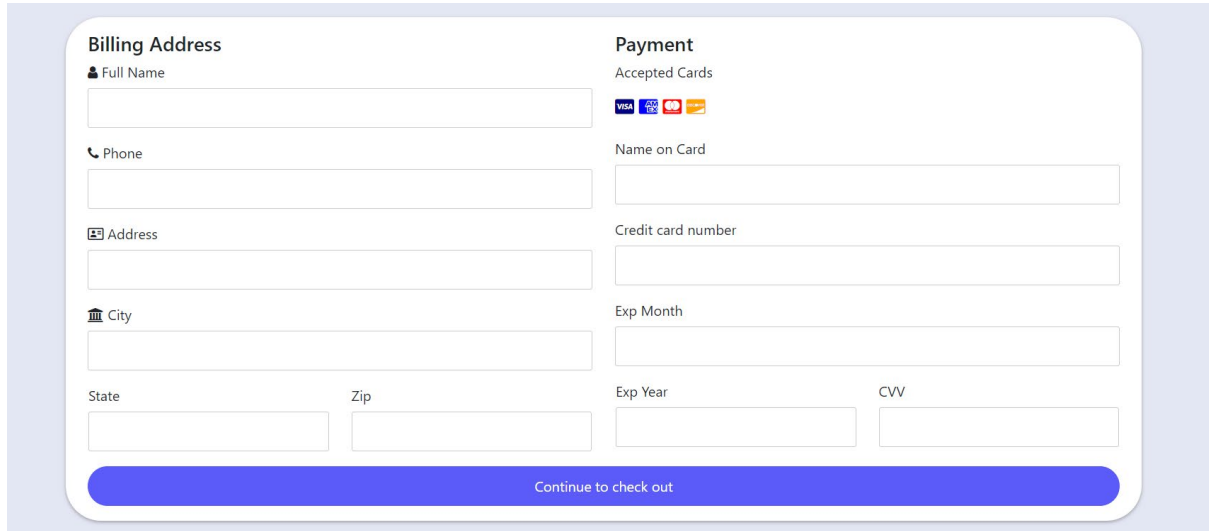


Figure a shows a form with two main sections: **Billing Address** and **Payment**. The **Billing Address** section includes fields for Full Name, Phone, Address, City, State, and Zip. The **Payment** section includes a header for Accepted Cards (Visa, MasterCard, American Express), a field for Name on Card, a field for Credit card number, a field for Exp Month, and a field for Exp Year. A CVV field is also present. A blue button at the bottom says "Continue to check out".

Figure b

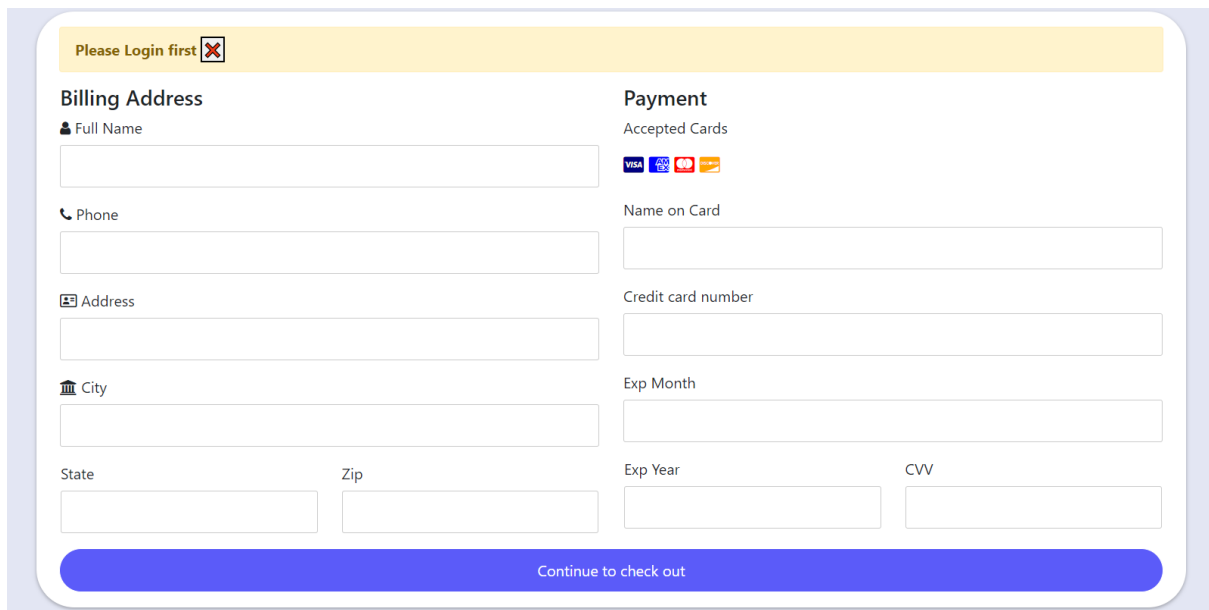


Figure b shows the same form as Figure a, but with an additional yellow banner at the top that says "Please Login first" with a red 'X' icon. The form fields and the "Continue to check out" button are identical to Figure a.

Figure c

```

if (isset($_SESSION['userid']))
{
    $userid = htmlspecialchars($_SESSION['userid']);
}
else{
    $errorMessage = "Please Login first";
    break;
}

```

Figure d

As the code shown in figure a above, this will redirect the user to the billing address page shown in figure b if they click the Place Order button. To keep their shipping and payment details in the customer database, we must first make sure they have a login. If not, it will look like the figure c above. As shown in figure d, in order to determine if they have logged in already or not, we will utilise the if statement. The session variable `$_SESSION["userid"]` is checked to see if it has been set using an if statement. If it is set, it indicates that the user has previously logged in since, when a user logs in, a session variable containing their username is created. If not, a message asking the user to log in first and use the break keyword to stop the code below from running will be displayed.

Billing Address		Payment	
<p>Full Name</p> <input type="text" value="Chin Zi Ming"/>		<p>Accepted Cards</p>	
<p>Phone</p> <input type="text" value="0162350735"/>		<p>Name on Card</p> <input type="text" value="1234567891011111"/>	
<p>Address</p> <input type="text" value="11, Jalan Remia 3, Taman Kota Jaya"/>		<p>Credit card number</p> <input type="text" value="1234567891011111"/>	
<p>City</p> <input type="text" value="Kota tinggi"/>		<p>Exp Month</p> <input type="text" value="12"/>	
<p>State</p> <input type="text" value="Johor"/>	<p>Zip</p> <input type="text" value="81900"/>	<p>Exp Year</p> <input type="text" value="26"/>	<p>CVV</p> <input type="text" value="999"/>
<p>Continue to check out</p>			

Figure a

```

if (isset($_SESSION['userid']))
{
    $userid = htmlspecialchars($_SESSION['userid']);
    $sql = "SELECT * FROM users WHERE usersUid='$userid'";
    $result = $conn->query($sql);
    $row = $result->fetch_assoc();
}

```

Figure b

As shown in figure a, the user details already been there if they have login and enter the billing details before. Hence, their information will immediately be retrieved from the client database and shown in the form. As shown in figure b, to identify the user, we are utilising the session variable `$_SESSION["userid"]`. The username of the person who has logged in will be stored in the session variable. In order to choose the customer information that belongs to the customer who has logged in, we can utilise this username as a condition.

```
$products_in_cart = isset($_SESSION['cart']) ? $_SESSION['cart'] : array();
$products = array();
$subtotal = 0.00;
if ($products_in_cart) {
    $array_to_question_marks = implode(',', array_fill(0, count($products_in_cart), '?'));
    $stmt = $pdo->prepare('SELECT * FROM redenproduct WHERE id IN (' . $array_to_question_marks . ')');
    $stmt->execute(array_keys($products_in_cart));
    $products = $stmt->fetchAll(PDO::FETCH_ASSOC);
    foreach ($products as $product) {
        $subtotal += (float)$product['price'] * (int)$products_in_cart[$product['id']];
    }
}
```

Figure a

As shown in figure a, the code allows to add items to your cart and choose from the database. Since we previously didn't store this data in our session variable, if there are any items in the shopping cart, get them from our redenproduct table along with the following column names: product name, description, image, and price. Additionally, we determine the subtotal by iterating through the products and dividing the cost by the number of quantities.



Figure a

In order to get the quantity to display beside the cart icon as shown in figure a, I'm using `$num_items_in_cart = isset($_SESSION['cart']) ? count($_SESSION['cart']) : 0;` and `<?php echo $num_items_in_cart?>` to display the numbers of item currently in the cart

Test cases and test results

Test Case ID	TC-A-001
Objective	Test the sign-up page with error empty input
Steps to test	1. Enter full name 2. Enter email 3. Leave the username blank 4. Enter password 5. Enter repeat password 6. Click signup button
Data inputs	Full name: Chin Zi Ming Email: chinziming20025504@gmail.com Password: Aba12345@ Repeat Password: Aba12345@
Expected Results	An alert message "Fill in all the fields" will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-A-002
Objective	Test the sign-up page with error invalid uid
Steps to test	1. Enter full name 2. Enter email 3. Enter username less than 3 characters 4. Enter password 5. Enter repeat password 6. Click signup button
Data inputs	Full name: Chin Zi Ming Email: chinziming20025504@gmail.com Username: zc Password: Aba12345@ Repeat Password: Aba12345@
Expected Results	An alert message "Username should be at least 3 words and not more than 20 words" will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-A-003
Objective	Test the sign-up page with error invalid email
Steps to test	<ol style="list-style-type: none"> 1. Enter full name 2. Enter email format of XXX@ 3. Enter username 4. Enter password 5. Enter repeat password 6. Click signup button
Data inputs	Full name: Chin Zi Ming Email: chinziming20025504@ Username: zc1 Password: Aba12345@ Repeat Password: Aba12345@
Expected Results	An alert message “Choose a proper email!” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-A-004
Objective	Test the sign-up page with error invalid password
Steps to test	<ol style="list-style-type: none"> 1. Enter full name 2. Enter email 3. Enter username 4. Enter password format of 1234 5. Enter repeat password 6. Click signup button
Data inputs	Full name: Chin Zi Ming Email: chinziming20025504@gmail.com Username: zc1 Password: 1234 Repeat Password: 1234
Expected Results	An alert message “Password should be at least one lowercase letter,at least one uppercase letter at least one number, and there have to be 8-12 characters” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-A-005
Objective	Test the sign-up page with error password don't match
Steps to test	<ol style="list-style-type: none"> 1. Enter full name 2. Enter email 3. Enter username 4. Enter password 5. Enter repeat password that is different from original password 6. Click signup button
Data inputs	Full name: Chin Zi Ming Email: chinziming20025504@ Username: zc1 Password: Aba12345@ Repeat Password: Aba12345#
Expected Results	An alert message "Password doesn't match!" will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-A-006
Objective	Test the sign-up page with error username taken
Steps to test	<ol style="list-style-type: none"> 1. Enter full name 2. Enter email that have been signed up 3. Enter username that have been signed up 4. Enter password 5. Enter repeat password 6. Click signup button
Data inputs	Full name: Chin Zi Ming Email: chinziming20025504@gmail.com Username: zc1 Password: Aba12345@ Repeat Password: Aba12345#
Expected Results	An alert message "Username or Email already taken!" will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-B-001
Objective	Test the sign-in page with error empty input
Steps to test	1. Enter username/email 2. Leave the password blank 3. Click sign in button
Data inputs	Username: zc1 Password:
Expected Results	An alert message “Fill in all fields!” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-B-002
Objective	Test the sign-in page with error wrong login
Steps to test	1. Enter username/email without sign up 2. Enter password without sign up 3. Click sign in button
Data inputs	Username: zc1 Password: AbaAba1234
Expected Results	An alert message “Incorrect login information” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-C-001
Objective	Test the billing address page with error user did not sign in account when placing order and checking out
Steps to test	1. No need to sign in account 2. Add product into the cart and place order 3. Filling up all the billing details 4. Click on the “Continue to check out” button
Data inputs	Full name: Chin Zi Ming Phone:0162350735 Address: 11, Jalan Remia 3, Taman Kota Jaya City: Kota tinggi State: Johor Zip: 81900 Name on Card: Chin Zi Ming Credit card number: 1111111111111111 Exp Month: 12 Exp Year: 24 CVV:999
Expected Results	An alert message “Please Login first” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-C-002
Objective	Test the billing address page with error empty input
Steps to test	<ol style="list-style-type: none"> 1. Sign in account 2. Add product into the cart and place order 3. Filling up all the billing details except for the name 4. Click on the “Continue to check out” button
Data inputs	Full name: Phone:0162350735 Address: 11, Jalan Remia 3, Taman Kota Jaya City: Kota tinggi State: Johor Zip: 81900 Name on Card: Chin Zi Ming Credit card number: 1111111111111111 Exp Month: 12 Exp Year: 24 CVV:999
Expected Results	An alert message “All the fields are required” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-C-003
Objective	Test the billing address page with error phone format
Steps to test	<ol style="list-style-type: none"> 1. Sign in account 2. Add product into the cart and place order 3. Filling up all the billing details and key in the phone number that is more than 11 digits 4. Click on the “Continue to check out” button
Data inputs	Full name: Chin Zi Ming Phone: 0162350735125 Address: 11, Jalan Remia 3, Taman Kota Jaya City: Kota tinggi State: Johor Zip: 81900 Name on Card: Chin Zi Ming Credit card number: 1111111111111111 Exp Month: 12 Exp Year: 24 CVV:999
Expected Results	An alert message “Malaysia Phone Number should be made up of 10 or 11 digits” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-C-004
Objective	Test the billing address page with error non-existing state in Malaysia
Steps to test	<ol style="list-style-type: none"> 1. Sign in account 2. Add product into the cart and place order 3. Filling up all the billing details and key in non-existing state in Malaysia 4. Click on the “Continue to check out” button
Data inputs	Full name: Chin Zi Ming Phone: 0162350735 Address: 11, Jalan Remia 3, Taman Kota Jaya City: Kota tinggi State: Thailand Zip: 81900 Name on Card: Chin Zi Ming Credit card number: 1111111111111111 Exp Month: 12 Exp Year: 24 CVV:999
Expected Results	An alert message “Not a valid state” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-C-005
Objective	Test the billing address page with error zip format
Steps to test	<ol style="list-style-type: none"> 1. Sign in account 2. Add product into the cart and place order 3. Filling up all the billing details and key in zip format that is less than 5 digits 4. Click on the “Continue to check out” button
Data inputs	Full name: Chin Zi Ming Phone: 0162350735 Address: 11, Jalan Remia 3, Taman Kota Jaya City: Kota tinggi State: Johor Zip: 8190 Name on Card: Chin Zi Ming Credit card number: 1111111111111111 Exp Month: 12 Exp Year: 24 CVV:999
Expected Results	An alert message “Zip code should be 5 digits” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-C-006
Objective	Test the billing address page with error card number format
Steps to test	<ol style="list-style-type: none"> 1. Sign in account 2. Add product into the cart and place order 3. Filling up all the billing details and key in card number format that is less than 16 digits 4. Click on the “Continue to check out” button
Data inputs	Full name: Chin Zi Ming Phone: 0162350735 Address: 11, Jalan Remia 3, Taman Kota Jaya City: Kota tinggi State: Johor Zip: 81900 Name on Card: Chin Zi Ming Credit card number: 1111 Exp Month: 12 Exp Year: 24 CVV:999
Expected Results	An alert message “Card number should be 16 digits” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-C-007
Objective	Test the billing address page with error card expire month format
Steps to test	<ol style="list-style-type: none"> 1. Sign in account 2. Add product into the cart and place order 3. Filling up all the billing details and key in card expire month format that is not within January to December digits 4. Click on the “Continue to check out” button
Data inputs	Full name: Chin Zi Ming Phone: 0162350735 Address: 11, Jalan Remia 3, Taman Kota Jaya City: Kota tinggi State: Johor Zip: 81900 Name on Card: Chin Zi Ming Credit card number: 1111111111111111 Exp Month: 13 Exp Year: 24 CVV:999
Expected Results	An alert message “Not a valid month” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-C-008
Objective	Test the billing address page with error card expire year format
Steps to test	<ol style="list-style-type: none"> 1. Sign in account 2. Add product into the cart and place order 3. Filling up all the billing details and key in card expire year format that is not double digits 4. Click on the “Continue to check out” button
Data inputs	Full name: Chin Zi Ming Phone: 0162350735 Address: 11, Jalan Remia 3, Taman Kota Jaya City: Kota tinggi State: Johor Zip: 81900 Name on Card: Chin Zi Ming Credit card number: 1111111111111111 Exp Month: 12 Exp Year: 2412 CVV:999
Expected Results	An alert message “Year should only be 2 digits. For example: 2024 = 24” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-C-009
Objective	Test the billing address page with error CVV format
Steps to test	<ol style="list-style-type: none"> 1. Sign in account 2. Add product into the cart and place order 3. Filling up all the billing details and key in CVV format that is less than 3 digits 4. Click on the “Continue to check out” button
Data inputs	Full name: Chin Zi Ming Phone: 0162350735 Address: 11, Jalan Remia 3, Taman Kota Jaya City: Kota tinggi State: Johor Zip: 81900 Name on Card: Chin Zi Ming Credit card number: 1111111111111111 Exp Month: 12 Exp Year: 24 CVV:99
Expected Results	An alert message “CVV should be 3 digits” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-D-001
Objective	Test the create admin page with error empty input
Steps to test	<ol style="list-style-type: none"> 1. Enter name 2. Enter email 3. Leave the username blank 4. Enter Phone number 5. Enter password 6. Click submit button
Data inputs	Name: Chin Zi Ming Email: chinziming20025504@gmail.com Username: Phone:0162350735 Password: Aba12345@
Expected Results	An alert message “All the fields are required” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-D-002
Objective	Test the create admin page with error username format
Steps to test	<ol style="list-style-type: none"> 1. Enter name 2. Enter email 3. Enter the username less than 3 characters 4. Enter Phone number 5. Enter password 6. Click submit button
Data inputs	Name: Chin Zi Ming Email: chinziming20025504@gmail.com Username: ZC Phone:0162350735 Password: Aba12345@
Expected Results	An alert message “Username should be at least 3 words and not more than 20 words” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-D-003
Objective	Test the create admin page with error email format
Steps to test	<ol style="list-style-type: none"> 1. Enter name 2. Enter email with format chinzing20025504@ 3. Enter the username 4. Enter Phone number 5. Enter password 6. Click submit button
Data inputs	Name: Chin Zi Ming Email: chinzing20025504@ Username: ZC1 Phone:0162350735 Password: Aba12345@
Expected Results	An alert message “Choose a proper email!” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-D-004
Objective	Test the create admin page with error password format
Steps to test	<ol style="list-style-type: none"> 1. Enter name 2. Enter email 3. Enter the username 4. Enter Phone number 5. Enter password with the format 1234 6. Click submit button
Data inputs	Name: Chin Zi Ming Email: chinzing20025504@gmail.com Username: ZC1 Phone:0162350735 Password: 1234
Expected Results	An alert message “Password should be at least one lowercase letter, at least one uppercase letter at least one number, and there have to be 8-12 characters” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-D-005
Objective	Test the create admin page with error username taken
Steps to test	<ol style="list-style-type: none"> 1. Enter name 2. Enter email that have been signed up 3. Enter username that have been signed up 4. Enter Phone number 5. Enter password 6. Click submit button
Data inputs	Name: Chin Zi Ming Email: chinziming20025504@gmail.com Username: ZC1 Phone:0162350735 Password: Aba12345@
Expected Results	An alert message "Username or Email already taken!" will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-D-006
Objective	Test the create admin page with error phone format
Steps to test	<ol style="list-style-type: none"> 1. Enter name 2. Enter email 3. Enter the username 4. Enter Phone number with digits more 11 5. Enter password 6. Click submit button
Data inputs	Name: Chin Zi Ming Email: chinziming20025504@gmail.com Username: ZC1 Phone:0162350735125 Password: Aba12345@
Expected Results	An alert message "Malaysia Phone Number should be made up of 10 or 11 digits" will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-D-007
Objective	Test the update admin page with error empty input
Steps to test	<ol style="list-style-type: none"> 1. Leave the name blank 2. Enter Phone number 3. Click submit button
Data inputs	Username: Phone:0162350735
Expected Results	An alert message "All the fields are required" will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-D-008
Objective	Test the update admin page with error phone format
Steps to test	1. Enter name 2. Enter Phone number with digits more 11 3. Click submit button
Data inputs	Username: ZC1 Phone:0162350735125
Expected Results	An alert message “Malaysia Phone Number should be made up of 10 or 11 digits” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-E-001
Objective	Test the create user page with error empty input
Steps to test	1. Enter name 2. Enter email 3. Leave the username blank 4. Enter password 5. Click submit button
Data inputs	Name: Chin Min Liang Email: chinmingliang@gmail.com Username: Phone:0167056968 Password: Aba12345#
Expected Results	An alert message “All the fields are required” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-E-002
Objective	Test the create user page with error username format
Steps to test	1. Enter name 2. Enter email 3. Enter the username less than 3 characters 4. Enter password 5. Click submit button
Data inputs	Name: Chin Min Liang Email: chinmingliang@gmail.com Username: CL Phone:0167056968 Password: Aba12345#
Expected Results	An alert message “Username should be at least 3 words and not more than 20 words” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-E-003
Objective	Test the create user page with error email format
Steps to test	<ol style="list-style-type: none"> 1. Enter name 2. Enter email with format chinminliang@ 3. Enter the username 4. Enter password 5. Click submit button
Data inputs	Name: Chin Min Liang Email: chinminliang@ Username: CL1 Phone:0167056968 Password: Aba12345#
Expected Results	An alert message “Choose a proper email!” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-E-004
Objective	Test the create user page with error password format
Steps to test	<ol style="list-style-type: none"> 1. Enter name 2. Enter email 3. Enter the username 4. Enter password with the format 1234 5. Click submit button
Data inputs	Name: Chin Zi Ming Email: chinminliang@gmail.com Username: CL1 Phone:0167056968 Password: 1234
Expected Results	An alert message “Password should be at least one lowercase letter, at least one uppercase letter at least one number, and there have to be 8-12 characters” will be displayed
Actual Results	Meet the expected results
Test Results	Pass


Test Case ID	TC-E-005
Objective	Test the create user page with error username taken
Steps to test	<ol style="list-style-type: none"> 1. Enter name 2. Enter email that have been signed up 3. Enter username that have been signed up 4. Enter password 5. Click submit button
Data inputs	Name: Chin Min Liang Email: chinminliang@gmail.com Username: CL1 Phone:0167056968 Password: Aba12345#
Expected Results	An alert message “Username or Email already taken!” will be displayed
Actual Results	Meet the expected results
Test Results	Pass


Test Case ID	TC-E-006
Objective	Test the update user page with error empty input
Steps to test	<ol style="list-style-type: none"> 1. Enter name 2. Enter billing name 3. Leave phone number blank 4. Enter address 5. Enter city 6. Enter state 7. Enter zip
Data inputs	Name: Chin Zi Ming Billing name: Chin Zi Ming Phone: Address: 11, Jalan Remia 3, Taman Kota Jaya City: Kota tinggi State: Johor Zip: 81900
Expected Results	An alert message “All the fields are required” will be displayed
Actual Results	Meet the expected results
Test Results	Pass


Test Case ID	TC-E-007
Objective	Test the update user page with error phone format
Steps to test	<ol style="list-style-type: none"> 1. Enter name 2. Enter billing name 3. Enter phone number that is more than 11 digits 4. Enter address 5. Enter city 6. Enter state 7. Enter zip
Data inputs	Name: Chin Zi Ming Billing name: Chin Zi Ming Phone:0162350735125 Address: 11, Jalan Remia 3, Taman Kota Jaya City: Kota tinggi State: Johor Zip: 81900
Expected Results	An alert message “Malaysia Phone Number should be made up of 10 or 11 digits” will be displayed
Actual Results	Meet the expected results
Test Results	Pass


Test Case ID	TC-E-008
Objective	Test the update user page with error non-existing state in Malaysia
Steps to test	<ol style="list-style-type: none"> 1. Enter name 2. Enter billing name 3. Enter phone number 4. Enter address 5. Enter city 6. Enter non-existing state in Malaysia 7. Enter zip
Data inputs	Name: Chin Zi Ming Billing name: Chin Zi Ming Phone:0162350735 Address: 11, Jalan Remia 3, Taman Kota Jaya City: Kota tinggi State: Thailand Zip: 81900
Expected Results	An alert message “Not a valid state” will be displayed
Actual Results	Meet the expected results
Test Results	Pass


Test Case ID	TC-E-009
Objective	Test the update user page with error zip format
Steps to test	<ol style="list-style-type: none"> 1. Enter name 2. Enter billing name 3. Enter phone number 4. Enter address 5. Enter city 6. Enter state 7. Enter zip than is less than 5 digits
Data inputs	Name: Chin Zi Ming Billing name: Chin Zi Ming Phone:0162350735 Address: 11, Jalan Remia 3, Taman Kota Jaya City: Kota tinggi State: Johor Zip: 8190
Expected Results	An alert message “Zip code should be 5 digits” will be displayed
Actual Results	Meet the expected results
Test Results	Pass


Test Case ID	TC-F-001
Objective	Test the create product page with error empty input
Steps to test	<ol style="list-style-type: none"> 1. Leave the product name blank 2. Enter description 3. Enter price 4. Enter RRP 5. Enter Quantity 6. Upload image 7. Click submit button
Data inputs	Name: Description: classical design Price: 50 RRP: 70 Quantity: 50 Upload image: 
Expected Results	An alert message “All the fields are required” will be displayed
Actual Results	Meet the expected results
Test Results	Pass


Test Case ID	TC-F-002
Objective	Test the create product page with error price format
Steps to test	<ol style="list-style-type: none"> 1. Enter product name 2. Enter description 3. Enter price with alphabets input 4. Enter RRP 5. Enter Quantity 6. Upload image 7. Click submit button
Data inputs	Name: Classic denim handbag Description: classical design Price: ab RRP: 70 Quantity: 50 Upload image: 
Expected Results	An alert message “Price must be a number” will be displayed
Actual Results	Meet the expected results
Test Results	Pass


Test Case ID	TC-F-003
Objective	Test the create product page with error RRP format
Steps to test	1. Enter product name 2. Enter description 3. Enter price 4. Enter RRP with alphabets input 5. Enter Quantity 6. Upload image 7. Click submit button
Data inputs	Name: Classic denim handbag Description: classical design Price: 50 RRP: ab Quantity: 50 Upload image: 
Expected Results	An alert message “RRP must be a number” will be displayed
Actual Results	Meet the expected results
Test Results	Pass


Test Case ID	TC-F-004
Objective	Test the create product page with error quantity format
Steps to test	1. Enter product name 2. Enter description 3. Enter price 4. Enter RRP 5. Enter Quantity with alphabets input 6. Upload image 7. Click submit button
Data inputs	Name: Classic denim handbag Description: classical design Price: 50 RRP: 70 Quantity: ab Upload image: 
Expected Results	An alert message “Quantity must be a number” will be displayed
Actual Results	Meet the expected results
Test Results	Pass


Test Case ID	TC-F-005
Objective	Test the create product page with error product name already existed
Steps to test	1. Enter product name that the website already existed 2. Enter description 3. Enter price 4. Enter RRP 5. Enter Quantity with alphabets input 6. Upload image 7. Click submit button
Data inputs	Name: Birdy Denim Handbag Description: classical design Price: 50 RRP: 70 Quantity: 50 Upload image: 
Expected Results	An alert message “Product Name already existed!” will be displayed
Actual Results	Meet the expected results
Test Results	Pass


Test Case ID	TC-F-006
Objective	Test the create product page with error uploading image that is too big
Steps to test	1. Enter product name 2. Enter description 3. Enter price 4. Enter RRP 5. Enter Quantity 6. Upload image >10000000 7. Click submit button
Data inputs	Name: Classic denim handbag Description: classical design Price: 50 RRP: 70 Quantity: 50 Upload image: 
Expected Results	An alert message “Your files is too big” will be displayed
Actual Results	Meet the expected results
Test Results	Pass


Test Case ID	TC-F-007
Objective	Test the create product page with error uploading image that is destroy
Steps to test	<ol style="list-style-type: none"> 1. Enter product name 2. Enter description 3. Enter price 4. Enter RRP 5. Enter Quantity 6. Upload image that is destroy 7. Click submit button
Data inputs	Name: Classic denim handbag Description: classical design Price: 50 RRP: 70 Quantity: 50 Upload image: 
Expected Results	An alert message “There was an error uploading your file!” will be displayed
Actual Results	Meet the expected results
Test Results	Pass


Test Case ID	TC-F-008
Objective	Test the create product page with error uploading image that did not meet the format
Steps to test	<ol style="list-style-type: none"> 1. Enter product name 2. Enter description 3. Enter price 4. Enter RRP 5. Enter Quantity 6. Upload image other than JPG, JPEG and PNG format 7. Click submit button
Data inputs	Name: Classic denim handbag Description: classical design Price: 50 RRP: 70 Quantity: 50 Upload image: 
Expected Results	An alert message “You cannot upload files of this type!” will be displayed
Actual Results	Meet the expected results
Test Results	Pass


Test Case ID	TC-F-009
Objective	Test the update product page with error empty input
Steps to test	<ol style="list-style-type: none"> 1. Leave the product name blank 2. Enter description 3. Enter price 4. Enter RRP 5. Enter Quantity 6. Upload image 7. Click submit button
Data inputs	Name: Description: classical design Price: 50 RRP: 70 Quantity: 50 Upload image: 
Expected Results	An alert message “All the fields are required” will be displayed
Actual Results	Meet the expected results
Test Results	Pass


Test Case ID	TC-F-010
Objective	Test the update product page with error price format
Steps to test	<ol style="list-style-type: none"> 1. Enter product name 2. Enter description 3. Enter price with alphabets input 4. Enter RRP 5. Enter Quantity 6. Upload image 7. Click submit button
Data inputs	Name: Classic denim handbag Description: classical design Price: ab RRP: 70 Quantity: 50 Upload image: 
Expected Results	An alert message “Price must be a number” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-F-011
Objective	Test the update product page with error RRP format
Steps to test	<ol style="list-style-type: none"> 1. Enter product name 2. Enter description 3. Enter price 4. Enter RRP with alphabets input 5. Enter Quantity 6. Upload image 7. Click submit button
Data inputs	Name: Classic denim handbag Description: classical design Price: 50 RRP: ab Quantity: 50 Upload image: 
Expected Results	An alert message “RRP must be a number” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-F-012
Objective	Test the update product page with error quantity format
Steps to test	<ol style="list-style-type: none"> 1. Enter product name 2. Enter description 3. Enter price 4. Enter RRP 5. Enter Quantity with alphabets input 6. Upload image 7. Click submit button
Data inputs	Name: Classic denim handbag Description: classical design Price: 50 RRP: 70 Quantity: ab Upload image: 
Expected Results	An alert message “Quantity must be a number” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-F-013
Objective	Test the update product page with error uploading image that is too big
Steps to test	<ol style="list-style-type: none"> 1. Enter product name 2. Enter description 3. Enter price 4. Enter RRP 5. Enter Quantity 6. Upload image >10000000 7. Click submit button
Data inputs	Name: Classic denim handbag Description: classical design Price: 50 RRP: 70 Quantity: 50 Upload image: 
Expected Results	An alert message “Your files is too big” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-F-014
Objective	Test the update product page with error uploading image that is destroy
Steps to test	<ol style="list-style-type: none"> 1. Enter product name 2. Enter description 3. Enter price 4. Enter RRP 5. Enter Quantity 6. Upload image that is destroy 7. Click submit button
Data inputs	Name: Classic denim handbag Description: classical design Price: 50 RRP: 70 Quantity: 50 Upload image: 
Expected Results	An alert message “There was an error uploading your file!” will be displayed
Actual Results	Meet the expected results
Test Results	Pass

Test Case ID	TC-F-015
Objective	Test the update product page with error uploading image that did not meet the format
Steps to test	<ol style="list-style-type: none"> 1. Enter product name 2. Enter description 3. Enter price 4. Enter RRP 5. Enter Quantity 6. Upload image other than JPG, JPEG and PNG format 7. Click submit button
Data inputs	Name: Classic denim handbag Description: classical design Price: 50 RRP: 70 Quantity: 50 Upload image: 
Expected Results	An alert message “You cannot upload files of this type!” will be displayed
Actual Results	Meet the expected results
Test Results	Pass