



School of Electrical Engineering and
Computer Science

COS5019-B Enterprise Pro Acceptance Testing Document

Project: My Supermarket Shop

Team 22

Table of Contents

Introduction	4
Code Inspection	5
Project Structure	5
Baseline	5
index.php	5
Algorithmic Correctness.....	5
Layout.....	6
Links to other pages	7
login.php	7
Algorithmic Correctness.....	7
Layout.....	8
Links to other pages	9
register.php.....	10
Algorithmic Correctness.....	10
Layout.....	11
Links to other pages	12
basket.php	12
Algorithmic Correctness.....	12
Layout.....	13
Links to other pages	13
profile.php (orders).....	14
Algorithmic Correctness.....	14
Layout.....	14
Links to other pages	15
order.php	16
Algorithmic Correctness.....	16
Layout.....	16
Links to other pages	17
admin.php	17
Algorithmic Correctness.....	17
Layout.....	17
Links to other pages	17
Acceptance Testing	18
Login.....	18
Admin	18

Logout	18
Register	18
View Products	19
View Supermarkets	19
Add To Basket	19
Order from Basket.....	19
Empty Basket	19

Introduction

The following document consists mostly of Code Inspection. Due to the nature of the project, most of the testing is done through Code Inspection as there is no object-oriented code. Unit Testing could have applied to this project as most of the files were the static webpages that encompassed the PHP functions inside to call dynamic functionalities. In the end, the Acceptance Testing document will be provided to further test all the functional requirements.

Code Inspection

The following section will address the Code Inspection done on each webpage the project has present to discuss code writing standards and webpage layout and structure consistency. Every webpage will be brought and examined thoroughly and evaluated for technical review. The “Algorithm Correctness” will explain briefly the logic behind some implementations, the “Layout” will focus on the visual look of the page and the “Links to Other Pages” will focus if all the links present are correctly set.

Project Structure

The project consists of 7 web pages that use HTML and PHP to display dynamic functionality. Each page is dedicated to a specific functionality while some might extend a bit further and implement more complex functionalities within the same page using PHP or JavaScript.

Baseline

There are some common attributes that all pages of the project share. These are:

- Navigation Bar (with different links and buttons but following the same structure).
- Style and Theme (Blue Background / Yellow Navigation Bar with Supermarket Icon next to the Title).
- Links to another website. Depending on the state of the application (ex. If a user is logged in, admin is logged in) some functionalities and links might be available or not. Ex. Logout is only present when a user is logged in.

index.php

Algorithmic Correctness

The following snippet is responsible for getting all the supermarkets and showing them to the UI. For each supermarket of the product, display name of the supermarket and price they sell it for. The code creates unique buttons that when pressed will get the quantity and add it to the cookie “basket”.

```
// for every supermarket that sells the product, create a card showing supermarket name, price and quantity
foreach ($supermarkets as $supermarket) {
    // the reference to the supermarket information
    $supermarket_reference = $database->getReference('/Products/'.$product.'/'.$supermarket.'/');
    // the name of the supermarket
    $supermarket_name = $supermarket_reference->getChild('/supermarketName/')->getSnapshot()->getValue();
    // the price of the product in that supermarket
    $supermarket_price = $supermarket_reference->getChild('/priceOfUnit/')->getSnapshot()->getValue();
    // for every third item, make a new row
    if($i==0 || $i==3 || $i==6){
        echo '<div class="row justify-content-center">';
    }
    echo //display a card, with supermarket name, price and a functionality to set quantity and add to basket
    '<div class="col text-center">';
    echo '
    <div class="card" style="width: 18em;margin: auto; margin-bottom: 0.5em;">
        <div class="card-header">
            '.$supermarket_name.'
        </div>
        <ul class="list-group list-group-flush text-center">
            <li class="list-group-item">Price : <span style="display: inline;" id = "'.$supermarket.'-price">'.$supermarket_price.'</span></li>
            <li class="list-group-item">Quantity: <input class="w-25" type="number" value="0" id="'.$supermarket.'-quantity" name="quantity" min="1" max="
            <li class="list-group-item"><button id="'.$supermarket.'-button" onclick="addToBasket('.$product.', '.$supermarket.'" type="butt
        </ul>
    </div>';
    //for every third supermarket close the row div
    if($i==3 || $i==6 || $i==9){
        echo '</div>';
    }
    $i = $i + 1;
}
echo '</div>';
echo '<hr>';
echo '<a class="btn btn-primary" href="/mysupermarketshopt22/index.php">New Search</a>';
==>
```

Figure 1 Algorithm responsible for loading all the supermarkets that have the product.

Layout

The layout of the index page has been consistent as the main card style is followed where the user needs to pick the products and afterwards select their desired supermarket.

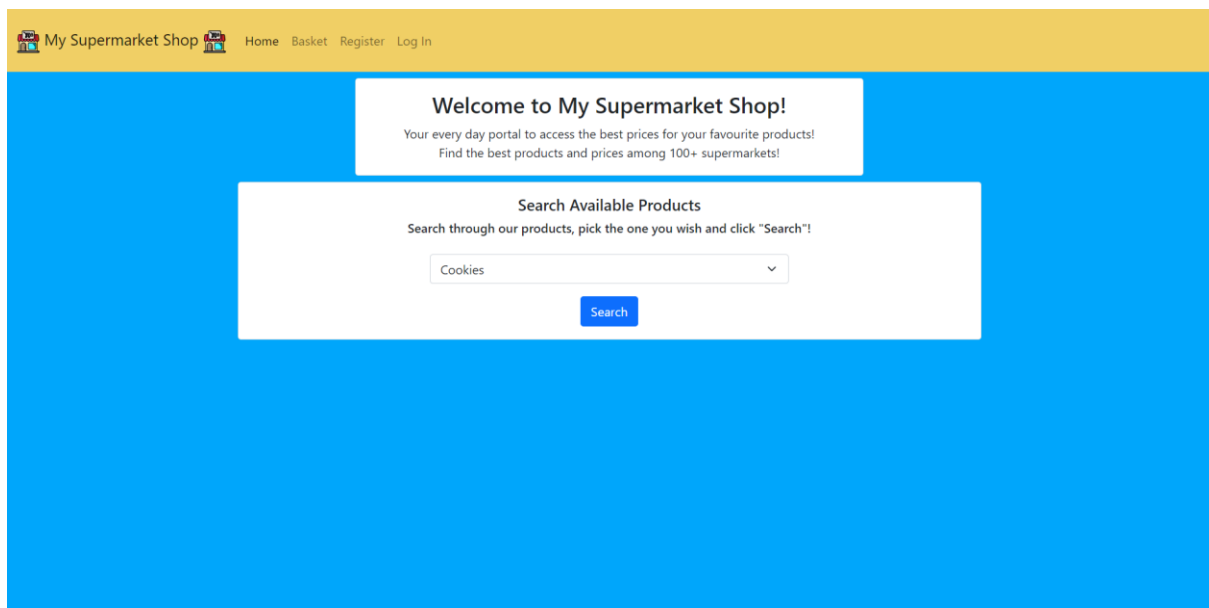


Figure 2 Main Page

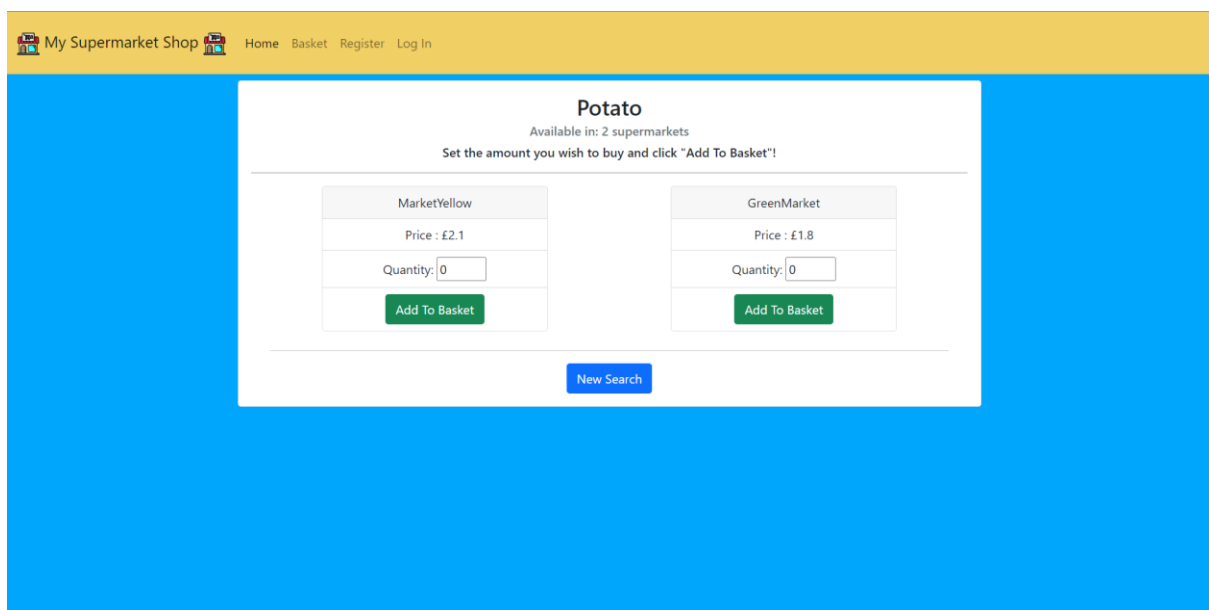


Figure 3 Potato Picked - Supermarkets available

Links to other pages

Snippet of redirecting to the pages.

```
<div class="collapse navbar-collapse" id="navbarSupportedContent">
  <ul class="navbar-nav me-auto mb-2 mb-lg-0">
    <li class="nav-item">
      <a class="nav-link active" aria-current="page" href="/mysupermarketshopt22/index.php">Home</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" aria-current="page" href="/mysupermarketshopt22/basket.php">Basket</a>
    </li>
    <?php
      // check from the cookies if there is a user logged in
      // if yes then show profile option (with their name) and logout option
      if (isset($_COOKIE["logged_in"])) {
        if ($_COOKIE["logged_in"] == true) {
          echo '<li class="nav-item">
            <a class="nav-link" href="profile.php">Orders ( ' . $_COOKIE['logged_in_username'] . ') </a>
          </li>
          <li class="nav-item">
            <button type="button" class="btn btn-link" style="color: red;text-decoration: none !important;"
          </li>
          ' ;
        }
      }
      // if user not logged in, then show the register and login options
      else {
        echo '<li class="nav-item">
          <a class="nav-link" href="/mysupermarketshopt22/register.php">Register</a>
        </li>
        <li class="nav-item dropdown">
          <a class="nav-link" href="/mysupermarketshopt22/login.php">Log In</a>
        </li>';
      }
    <?>
  </ul>
</div>
```

Figure 4 Links to other pages from Login

login.php

Algorithmic Correctness

The following snippet is responsible for checking if the account is an admin account by checking in the Firebase Database for the “username” and “password” attribute of the “Admin” key.

```
// check if it is admin account
$adminUsername = $database->getReference('/Admin/username')->getSnapshot()->getValue();
$adminPassword = $database->getReference('/Admin/password')->getSnapshot()->getValue();

if($username === $adminUsername && $password === $adminPassword){
  setcookie("logged_in", true, time() + (86400 * 30), "/");
  setcookie("logged_in_username", $username, time() + (86400 * 30), "/");
  echo '<div class="container">
    <div class="row">
      <div class="col-sm">
        <br>
      </div>
    </div>
    <div class="row">
      <div class="col-sm">
        <div class="alert alert-success" role="alert">
          <span class="spinner-border spinner-border-sm" role="status" aria-hidden="true"></span>Welcome ' . strval($username) . '! Please wait...</div>
        </div>
      </div>
    </div>';
  echo '<script type="text/javascript">location.href = "/mysupermarketshopt22/admin.php";</script>';
}
```

Figure 5 Check if Admin account

The following snippet is responsible for going through all the users. Since the usernames are part of the database structure and they are nodes themselves, checking if the node Users/username node exists is a viable solution. If the password field matches as well, then the log in is correct.

```

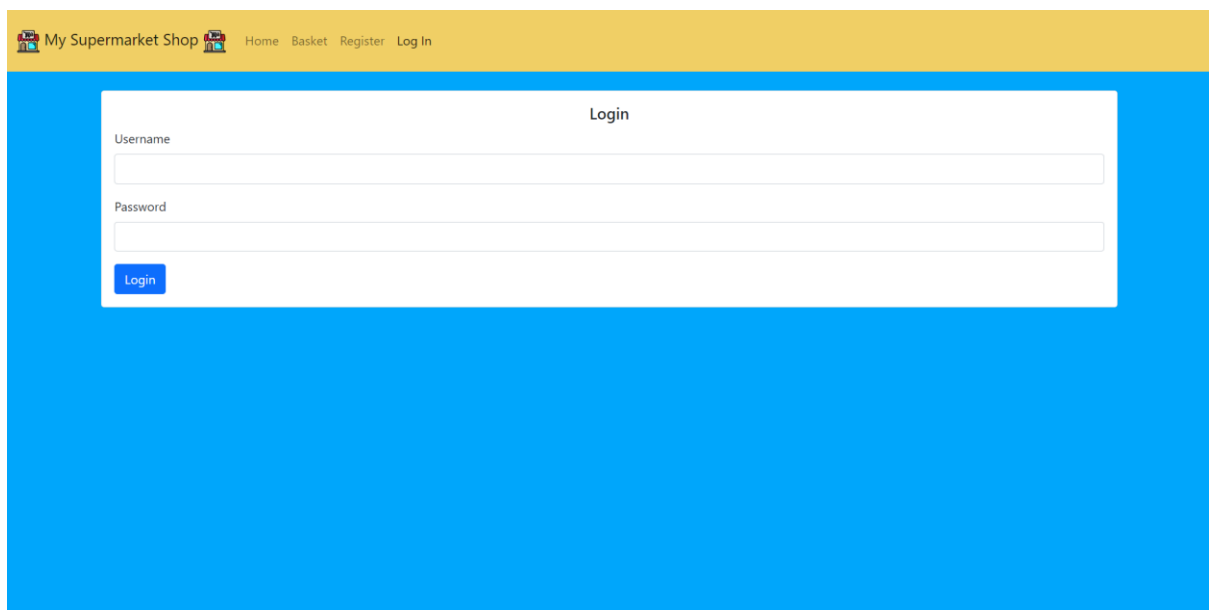
// get the user keys from the database
$user_keys = $database->getReference('/Users/')->getChildKeys();
$found = false;
// for each user key
foreach ($user_keys as $user_key) {
    // find the reference and if the username exists under that key
    $usernameFromDatabase = $database->getReference('/Users/' . strval($user_key) . '/' . strval($username) . '/')->getSnapshot();
    if ($usernameFromDatabase->exists()) {
        // if the password from that username also matches
        if (strval($usernameFromDatabase->getChild('/password/'))->getValue() == strval($password)) {
            $found = true;
            // create cookies to hold the logged in user
            setcookie("logged_in", true, time() + (86400 * 30), "/");
            setcookie("logged_in_username", $username, time() + (86400 * 30), "/");
            echo '<div class="container">
                <div class="row">
                    <div class="col-sm">
                        <br>
                    </div>
                </div>
                <div class="row">
                    <div class="col-sm">
                        <div class="alert alert-success" role="alert">
                            <span class="spinner-border spinner-border-sm" role="status" aria-hidden="true"></span>Welcome back '. strval($username).'
                        </div>
                    </div>
                </div>
            </div>';
            echo '<script type="text/javascript">location.href = "/mysupermarketshopt22/profile.php";</script>';
        }
    }
}

```

Figure 6 Check User credentials

Layout

The layout of the login page is consistent with the fields being present and the login button being available. An alert is shown in case the user inputted invalid values.



The screenshot shows the 'My Supermarket Shop' login page. The header is yellow and contains the shop name, a home icon, and links for Home, Basket, Register, and Log In. The main body is blue. A white login form is centered, titled 'Login'. It contains two input fields: 'Username' and 'Password'. Below the password field is a blue 'Login' button.

Figure 7 Login Page Default

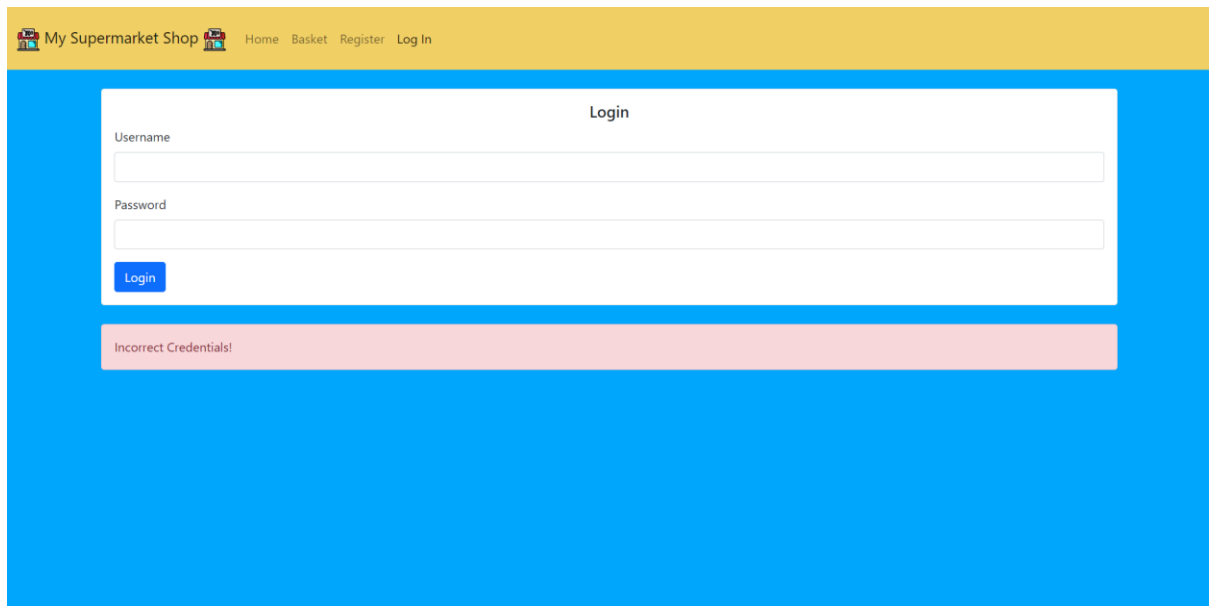


Figure 8 Login page - Incorrect Credentials

Links to other pages

Snippet of redirecting to the pages.

```
<body style="background-color: #00A6FB;"></body>
<nav class="navbar navbar-expand-lg navbar-light" style="background-color: #F0CF65;">
  <div class="container-fluid">
    <a class="navbar-brand" href="#">
      
      My Supermarket Shop
    </a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls=
    <span class="navbar-toggler-icon"></span>
  </button>
  <div class="collapse navbar-collapse" id="navbarSupportedContent">
    <ul class="navbar-nav me-auto mb-2 mb-lg-0">
      <li class="nav-item">
        <a class="nav-link" aria-current="page" href="/mysupermarketshopt22/index.php">Home</a>
      </li>
      <li class="nav-item">
        <a class="nav-link" aria-current="page" href="/mysupermarketshopt22/basket.php">Basket</a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="/mysupermarketshopt22/register.php">Register</a>
      </li>
      <li class="nav-item dropdown">
        <a class="nav-link active" href="/mysupermarketshopt22/login.php">Log In</a>
      </li>
    </ul>
  </div>
</div>
</nav>
```

Figure 9 Redirecting links from Login.

register.php

Algorithmic Correctness

The following snippet explains the possibility of the user registering with a username that already exists. Since usernames are nodes under the /Users/ tree, checking if the inputted username is a valid node will confirm if the node exists.

```
// check if all fields have been provided
if (!ifNull($_POST['username']) && !ifNull($_POST['email']) && !ifNull($_POST['password']) && !ifNull($_POST['dateofbirth'])) {

    // create variables for all the fields inputted
    $username = $_POST['username'];
    $email = $_POST['email'];
    $password = $_POST['password'];
    $dateofbirth = $_POST['dateofbirth'];

    // get all user keys which correspond to users to check if username exists
    $user_keys = $database->getReference('/Users/')->getChildKeys();
    foreach ($user_keys as $user_key) {
        // check if the key has a username inside as the one provided
        $usernameFromDatabase = $database->getReference('/Users/'.strval($user_key).'/'.$username.'/')->getSnapshot();
        // if yes, it means a user exists!
        if($usernameFromDatabase->exists()){
            echo '<div class="container">
                <div class="row">
                    <div class="col-sm">
                        <br>
                    </div>
                </div>
                <div class="row">
                    <div class="col-sm">
                        <div class="alert alert-danger" role="alert">Username already exists!</div>
                    </div>
                </div>
            </div>';
            exit;
        }
    }
}
```

Figure 10 Check if username already exists.

The following snippet is responsible for filling a data load that will be sent to the firebase database to create a new node containing the created user. The cookie is then created since the user is logged in immediately.

```
// if not, then get the reference to the users
$reference = $database->getReference('/Users/');
$snapshot = $reference->getSnapshot();

// data to push to the tree holding all the users
$postData = [
    strval($username) => [
        'password' => $password,
        'email' => $email,
        'dateofbirth' => date($dateofbirth),
    ],
];

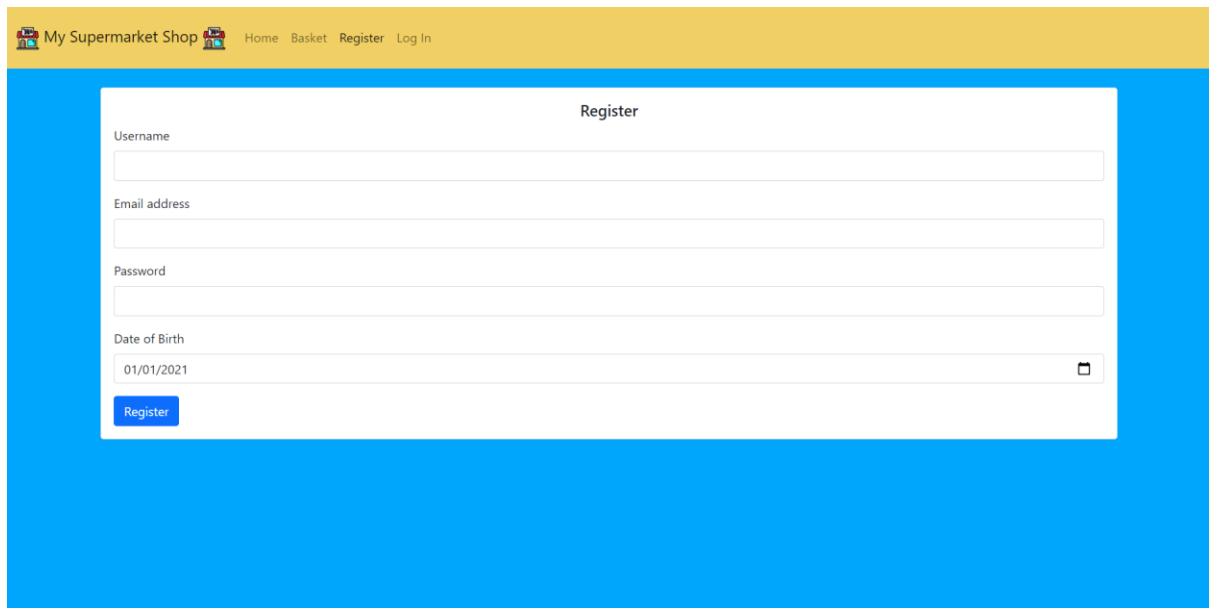
//push to the tree holding all the users, this will also create a unique key
$database->getReference('/Users')->push($postData);

// set cookie to show logged in user
setcookie("logged_in", true, time() + (86400 * 30), "/");
setcookie("logged_in_username", $username, time() + (86400 * 30), "/");
```

Figure 11 Create new user, send to database.

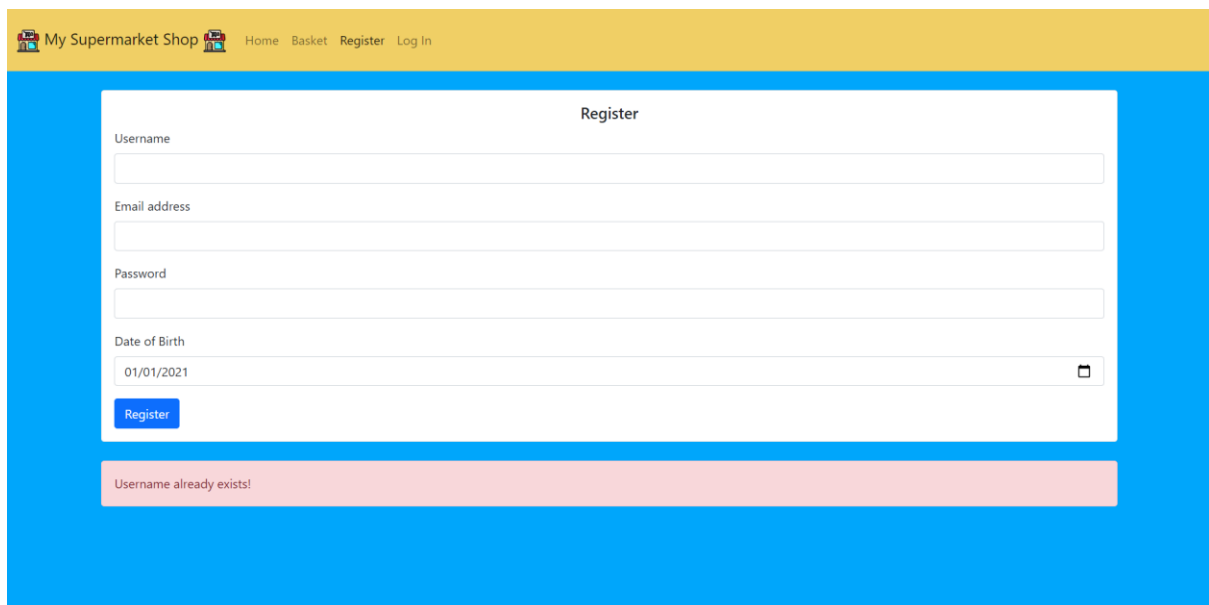
Layout

The following displays show that the layout is consisted across different pages states such as when alerting the user that the username already exists.



The screenshot shows the 'Register' page of 'My Supermarket Shop'. The page has a yellow header with the shop name and navigation links: Home, Basket, Register, and Log In. The main content area has a blue background. A white form titled 'Register' is centered. It contains four input fields: 'Username', 'Email address', 'Password', and 'Date of Birth'. The 'Date of Birth' field is a date picker showing '01/01/2021'. A blue 'Register' button is at the bottom of the form.

Figure 12 Register – Default



This screenshot shows the same 'Register' form as Figure 12, but with an error message. Below the form, a pink message box displays the text 'Username already exists!'. The form fields and the 'Register' button remain visible above the message.

Figure 13 Register - Username Already Exists

Links to other pages

Snippet of redirecting to the pages.

```
<nav class="navbar navbar-expand-lg navbar-light" style="background-color: #F0CF65;">
  <div class="container-fluid">
    <a class="navbar-brand" href="#">
      
      My Supermarket Shop
      </a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" data-bs-keyboard="true" data-bs-ripple="true" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarSupportedContent">
      <ul class="navbar-nav me-auto mb-2 mb-lg-0">
        <li class="nav-item">
          <a class="nav-link" aria-current="page" href="/mysupermarketshopt22/index.php">Home</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" aria-current="page" href="/mysupermarketshopt22/basket.php">Basket</a>
        </li>
        <li class="nav-item">
          <a class="nav-link active" href="/mysupermarketshopt22/register.php">Register</a>
        </li>
        <li class="nav-item dropdown">
          <a class="nav-link" href="/mysupermarketshopt22/login.php">Log In</a>
        </li>
      </ul>
    </div>
  </div>
</nav>
```

Figure 14 Redirect Links

basket.php

Algorithmic Correctness

The following snippet is responsible for getting the basket cookie and reading it to create a table which will display the list of items and quantity of each items that has been selected for purchase.

```
// since all orders are concatenated and separated by a comma (,)
// create an array to hold all the orders
$orders = explode("!", strval($basket));
echo '<table class="table" id="tableOrders">
  <thead>
    <tr>
      <th scope="col">#</th>
      <th scope="col">Product</th>
      <th scope="col">Supermarket</th>
      <th scope="col">Quantity</th>
      <th scope="col">Price</th>
      <th scope="col">Total</th>
    </tr>
  </thead>
  <tbody>';

$i = 0;
// for each order in the orders array
foreach ($orders as $order) {
  // order format = product-supermarket-quantity-price!
  // split the string to get each value
  $words = explode("-", strval($order));
  // if the word exists
  if (isset($words) && !empty($words) && isset($words[1])) {
    echo '<tr>
      <th scope="row">' . $i . '</th>
      <td>' . $words[0] . '</td>
      <td>' . $words[1] . '</td>
      <td>' . $words[3] . '</td>
      <td>' . (float)$words[2] . '</td>
      <td>' . $words[2] * (float)$words[3][0] . '</td>
    </tr>';
    $i = $i + 1;
  }
}
echo '</tbody></table>';
?>

<!-- Button to erase all from the basket -->
<button type="button" id="emptyBasket" onclick="emptyBasket()" class="btn btn-danger">Empty Basket</button>
<button type="button" id="orderBtn" onclick="order()" class="btn btn-success">Order</button>
```

Layout

The layout of the basket page is consistent as the table created from the “basket” cookie correctly displays all the items selected, the quantity preferred and the total sum cost.

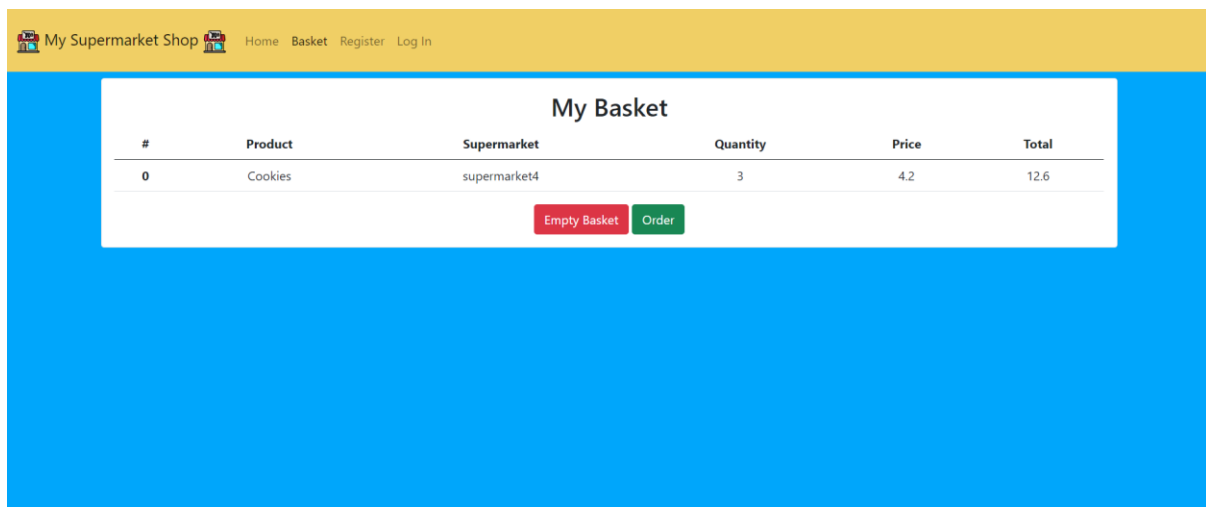


Figure 15 Basket will items

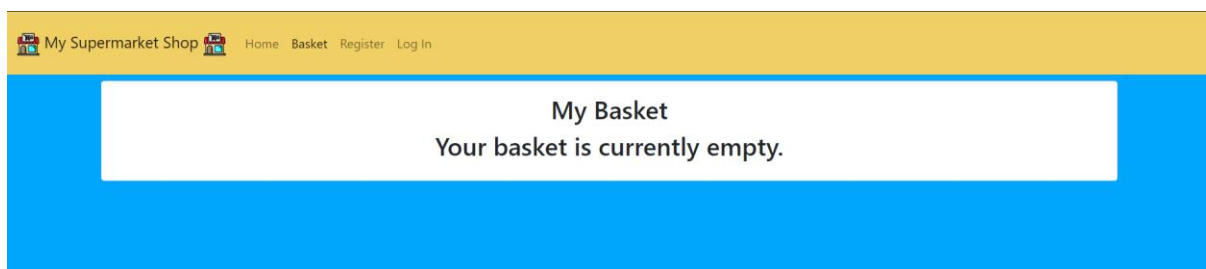


Figure 16 Empty Basket

Links to other pages

Snippet of redirecting to the pages.

```
<div class="collapse navbar-collapse" id="navbarSupportedContent">
  <ul class="navbar-nav me-auto mb-2 mb-lg-0">
    <li class="nav-item">
      <a class="nav-link active" aria-current="page" href="/mysupermarketshopt22/index.php">Home</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" aria-current="page" href="/mysupermarketshopt22/basket.php">Basket</a>
    </li>
  </ul>
  <?php
    // check from the cookies if there is a user logged in
    // if yes then show profile option (with their name) and logout option
    if (isset($_COOKIE["logged_in"])) {
      if ($_COOKIE["logged_in"] == true) {
        echo '<li class="nav-item">
          <a class="nav-link" href="profile.php">Orders ( ' . $_COOKIE["logged_in_username"] . ') </a>
        </li>
        <li class="nav-item">
          <button type="button" class="btn btn-link" style="color: red;text-decoration: none !important;" onclick="logout()">Log out</button>
        </li>
      ';
      }
    }
    // if user not logged in, then show the register and login options
    else {
      echo '<li class="nav-item">
        <a class="nav-link" href="/mysupermarketshopt22/register.php">Register</a>
      </li>
      <li class="nav-item dropdown">
        <a class="nav-link" href="/mysupermarketshopt22/login.php">Log In</a>
      </li>';
    }
  <?>
```

Figure 17 Redirect Links

profile.php (orders)

Algorithmic Correctness

The following snippet is like the basket functionality but now the orders are drawn from the database and the formatted order (same as cookie format) is displayed to the user in a table format. For each order in the orders where the order has the username logged in, display it.

```
foreach ($orders as $order) {
    $order_reference = $database->getReference('/Orders/'.$order);

    $user_from_order = $order_reference->getChild('/user/')->getSnapshot()->getValue();

    if($user_from_order==$COOKIE['logged_in_username']){
        $found_an_order = true;

        $items_from_order = $order_reference->getChild('/items/')->getSnapshot()->getValue();

        $items = explode("!", strval($items_from_order));
        echo '<div class="row mt-2 mb-2">
            <div class="col-sm text-center">
                <div class="card">
                    <div class="card-body text-center">';
        echo '<h4>Order ID: '.$order.'</h4>';
        echo '<table class="table" id="tableOrders">
            <thead>
                <tr>
                    <th scope="col">#</th>
                    <th scope="col">Product</th>
                    <th scope="col">Supermarket</th>
                    <th scope="col">Quantity</th>
                    <th scope="col">Price</th>
                    <th scope="col">Total</th>
                </tr>
            </thead>
            <tbody>';

        $i = 0;
        // for each order in the orders array
        foreach ($items as $item) {
            // order format = product-supermarket-quantity-price!
            // split the string to get each value
            $words = explode("-", strval($item));
            // if the word exists
            if(isset($words)&&!empty($words)&&isset($words[1])){
                echo '<tr>
                    <th scope="row">' . $i . '</th>
                    <td>'.$words[0].</td>
                    <td>'.$words[1].</td>
                    <td>'.$words[3].</td>
                    <td>'.(float)$words[2].</td>
                    <td>'. $words[2] * (float) $words[3][0].</td>
                </tr>';

                $i = $i + 1;
            }
        }
    }
}
```

Figure 18 Get Orders for user from database

Layout

The layout is consisted when orders are present as well as when there are no orders pending, correctly displaying the ordered items, quantities and total, as well as showing the order id.

My Supermarket Shop

Home Basket Orders (testuser10) Log out

Order ID: -MaGL0BckjkRCbA4c-va					
#	Product	Supermarket	Quantity	Price	Total
0	Apple	supermarket5	3	1.1	3.3

Figure 19 Ordes

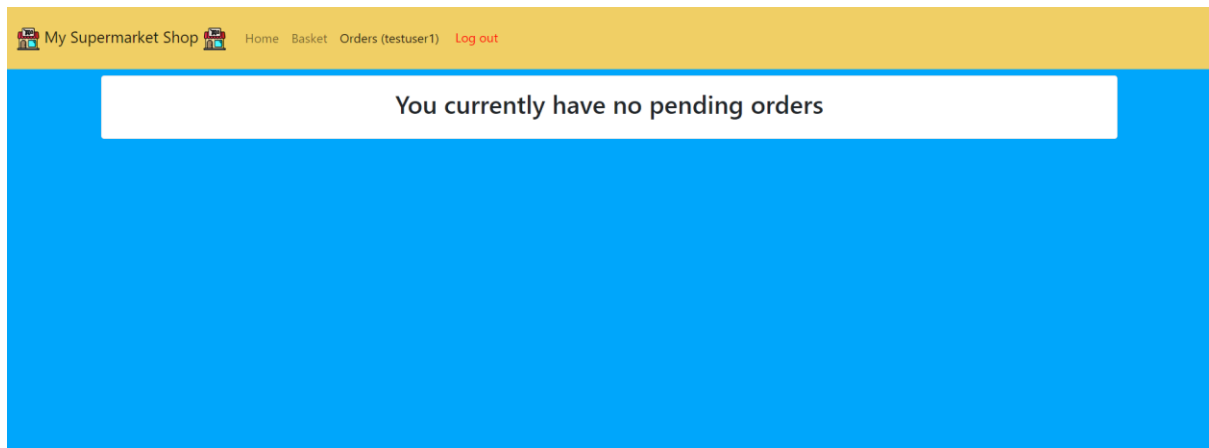


Figure 20 No Orders present for user.

Links to other pages

Snippet of redirecting to the pages.

```

<div class="collapse navbar-collapse" id="navbarSupportedContent">
  <ul class="navbar-nav me-auto mb-2 mb-lg-0">
    <li class="nav-item">
      <a class="nav-link" aria-current="page" href="/mysupermarketshopt22/index.php">Home</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" aria-current="page" href="/mysupermarketshopt22/basket.php">Basket</a>
    </li>
    <?php
    if (isset($_COOKIE["logged_in"])) {
      if ($_COOKIE["logged_in"] == true) {
        echo '<li class="nav-item">
          <a class="nav-link active" href="/mysupermarketshopt22/profile.php">Orders ('.$_COOKIE['logged_in_username'].') <
        </li>
        <li class="nav-item">
          <button type="button" class="btn btn-link" style="color: red;text-decoration: none !important;" onclick="logout()
        </li>
        ';
      }
    } else {
      echo '<li class="nav-item">
        <a class="nav-link" href="/mysupermarketshopt22/register.php">Register</a>
      </li>
      <li class="nav-item dropdown">
        <a class="nav-link" href="/mysupermarketshopt22/login.php">Log In</a>
      </li>';
      echo '<script type="text/javascript">location.href = "/mysupermarketshopt22/login.php";</script>';
    }
  </ul>

```

Figure 21 Links

order.php

Algorithmic Correctness

The following snippet is responsible for registering the order and alerting the user of the correct order submission.

```
if ($_SERVER['REQUEST_METHOD'] == 'POST') {  
    // check if all fields have been provided  
    if (!ifNull($_POST['address1']) && !ifNull($_POST['address2']) && !ifNull($_POST['postcode']))  
        // create variables for all the fields inputted  
        $address1 = $_POST['address1'];  
        $address2 = $_POST['address2'];  
        $postcode = $_POST['postcode'];  
        $phone = $_POST['phone'];  
  
        // if not, then ge the reference to the users  
        $reference = $database->getReference('/Users/');  
        $snapshot = $reference->getSnapshot();  
  
        // data to push to the tree holding all the users  
        $postData = [  
            'address1' => $address1,  
            'address2' => $address2,  
            'postcode' => $postcode,  
            'phone' => $phone,  
            'user' => $_COOKIE['logged_in_username'],  
            'items' => $_COOKIE['basket']  
        ];  
  
        //push to the tree holding all the users, this will also create a unique key  
        $database->getReference('/Orders')->push($postData);  
  
        // clear basket  
        setcookie("basket", "", time() - 3600);  
  
        echo '<h2>Congratulations, your order was successful.</h2>';  
        echo '<h3>Order Details</h3>';  
        echo '<p>Address 1: ' . $address1 . '</p>';  
        echo '<p>Address 2: ' . $address2 . '</p>';  
        echo '<p>Postcode: ' . $postcode . '</p>';  
        echo '<p>Phone: ' . $phone . '</p>';  
    } else { // if data sent is corrupted  
        echo '<h2>Something went wrong, please try again.</h2>';  
    }  
}
```

Figure 22 Order Submission.

Layout

The layout is consistent when the order has been submitted displaying properly all the details provided for the order.

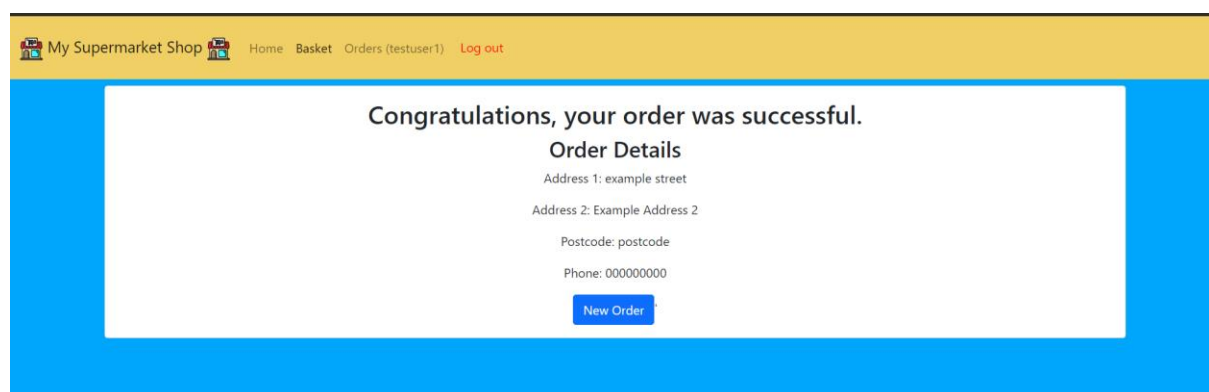


Figure 23 Order Submission Confirm

Links to other pages
Snippet of redirecting to the pages.

```
<div class="Collapse navbar-collapse" id="navbarSupportedContent">
<ul class="navbar-nav me-auto mb-2 mb-lg-0">
<li class="nav-item">
<a class="nav-link" aria-current="page" href="/mysupermarketshopt22/index.php">Home</a>
</li>
<li class="nav-item active">
<a class="nav-link active" aria-current="page" href="/mysupermarketshopt22/basket.php">Basket</a>
</li>
<?php
// check from the cookies if there is a user logged in
// if yes then show profile option (with their name) and logout option
if (isset($_COOKIE["logged_in"])) {
    if ($_COOKIE["logged_in"] == true) {
        echo '<li class="nav-item">
            <a class="nav-link" href="profile.php">Orders ( ' . $_COOKIE["logged_in_username"] . ' ) </a>
        </li>
        <li class="nav-item">
            <button type="button" class="btn btn-link" style="color: red;text-decoration: none !important;" onclick="logout()">Log out</button>
        </li>
    ';
    }
}
// if user not logged in, then show the register and login options
else {
    echo '<li class="nav-item">
        <a class="nav-link" href="/mysupermarketshopt22/register.php">Register</a>
    </li>
    <li class="nav-item dropdown">
        <a class="nav-link" href="/mysupermarketshopt22/login.php">Log In</a>
    </li>';
}
?>
</div>
```

Figure 24 Redirects

admin.php
Algorithmic Correctness
Layout

The layout is consistent where the admin can see all the order from all the users and the details for each order as well as provide the button to mark the order as completed.

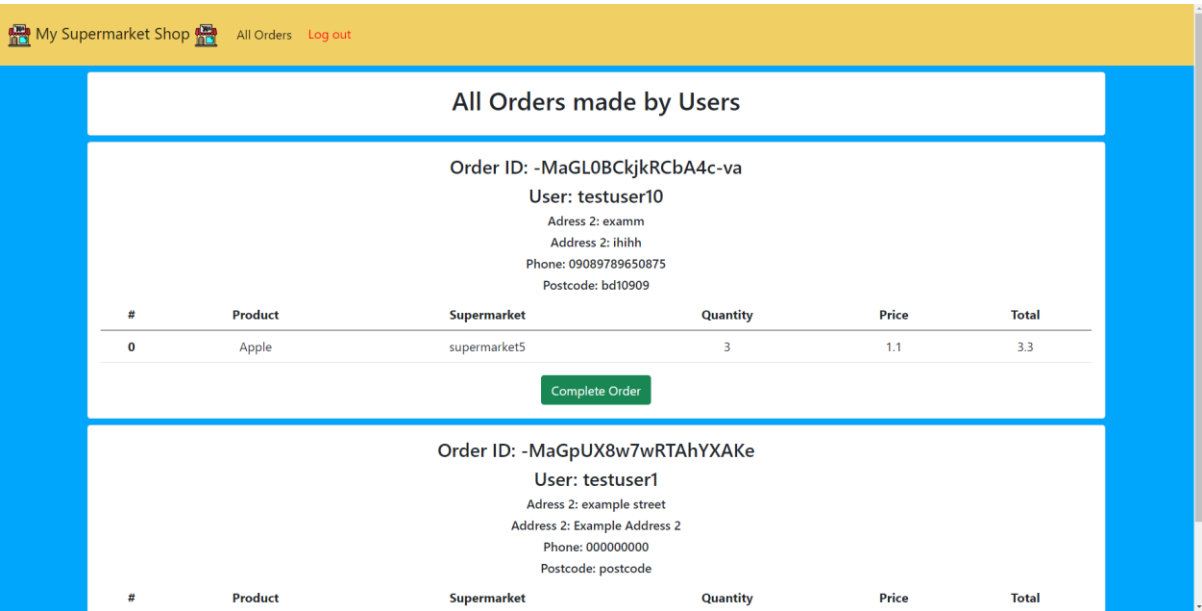


Figure 25 Admin View Orders

Links to other pages
Snippet of redirecting to the pages.

```

<div class="collapse navbar-collapse" id="navbarSupportedContent">
<ul class="navbar-nav me-auto mb-2 mb-lg-0">
<li class="nav-item">
<a class="nav-link" aria-current="page" href="/mysupermarketshopt22/index.php">Home</a>
</li>
<li class="nav-item active">
<a class="nav-link active" aria-current="page" href="/mysupermarketshopt22/basket.php">Basket</a>
</li>
<?php
// check from the cookies if there is a user logged in
// if yes then show profile option (with their name) and logout option
if (isset($_COOKIE["logged_in"])) {
    if ($_COOKIE["logged_in"] == true) {
        echo '<li class="nav-item">
<a class="nav-link" href="profile.php">Orders (' . $_COOKIE['logged_in_username'] . ') </a>
</li>
<li class="nav-item">
<button type="button" class="btn btn-link" style="color: red;text-decoration: none !important;" onclick="lc
</li>
';
    }
}
// if user not logged in, then show the register and login options
else {
    echo '<li class="nav-item">
<a class="nav-link" href="/mysupermarketshopt22/register.php">Register</a>
</li>
<li class="nav-item dropdown">
<a class="nav-link" href="/mysupermarketshopt22/login.php">Log In</a>
</li>';
}
?>

```

Figure 26 Redirects

Acceptance Testing

In this section all the functional requirements from the story cards will be tested and examined.

Login

Login			
Input	Expected Result	Current Output	Comments
"testuser1","password"	Login Successful	Login Successful	Passed
"ihihhi","password"	Wrong Credentials	Wrong Credentials	Passed
"testuser1","ojajj"	Wrong Credentials	Wrong Credentials	Passed
"ihihhi","ninini"	Wrong Credentials	Wrong Credentials	Passed

Admin

Admin Login			
Input	Expected Result	Current Output	Comments
"admin","password"	Login Successful	Login Successful	Passed
"huhuh","password"	Wrong Credentials	Wrong Credentials	Passed
"admin","jiiijijj"	Wrong Credentials	Wrong Credentials	Passed
"ihihhih","Nininin"	Wrong Credentials	Wrong Credentials	Passed

Logout

Logout			
Input	Expected Result	Current Output	Comments
User is logged in and presses Logout Button	Logout	Logout	Passed

Register

Register			
Input	Expected Result	Current Output	Comments
"testuser2","password", "mail@mail.com",	Register Successful	Register Successful	Passed

"979899898"			
"testuser1","password", "mail@mail.com", "979899898"	Username already exists!	Username already exists!	Passed
"testuser1","", "", ""	Missing fields	Missing fields	Passed

View Products

View Products			
Input	Expected Result	Current Output	Comments
User sees list of products in Main Page	Correct list of all products	Correct list of all products	Passed

View Supermarkets

View Supermarkets			
Input	Expected Result	Current Output	Comments
User sees list of supermarkets that offer the product in Main Page	Correct list of all products	Correct list of all products	Passed
User has not selected a product beforehand	Please select a product	Please select a product	Passed

Add To Basket

Add To Basket			
Input	Expected Result	Current Output	Comments
User sets the desired quantity and pressed "Added to Basket"	Item and Quantity added to basket	Item and Quantity added to basket	Passed
User sets desired quantity to 0.	No Items added to the basket	No Items added to the basket	Passed

Order from Basket

Order from Basket			
Input	Expected Result	Current Output	Comments
User clicks "order" and fills all fields in personal information.	Order Successful	Order Successful	Passed
User clicks "order" and does not fill all fields in personal information.	Please fill all fields	Please fill all fields	Passed
User has no items in basket.	Ordering is not available	Ordering is not available	Passed

Empty Basket

Empty Basket			
Input	Expected Result	Current Output	Comments

User has items in the Basket and presses respective button.	Basket emptied	Basket emptied	Passed
User does not have items in the Basket and presses respective button.	Basket not emptied	Basket not emptied	Passed