## Existing User:

Account 1: wl17865

Password 1: 17865

Account 2: jz17501

Password 2: 17501

## 1. HTML

In the beginning, we thought it might be interesting that we can make a e-commerce website. So we decided to make a website which sells a wide variety of laptops. We learned that HTML tags contain several key parts including tags (and their attributes), character-based data types, character references, and entity references. Basically, the structure should be like this:

```
<!DOCTYPE html xmlns="http://www.w3.org/1999/xhtml" lang="en-GB" xml:lang="en-GB" >
<html>
  <head>
    <title>title</title>
  </head>
  <body>
    <p>Hello world!</p>
  </body>
</html>
```

- We used xmlns: xhtml=http://www.w3.org/1999/xhtml to declare and reference namespaces so name conflicts between elements or attributes with the same name can be resolved.

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml" lang="en-GB" xml:lang="en-GB">
```

- Metadata is information about data. The <meta> tag provides metadata about HTML documents. Metadata does not appear on the page, but it is readable by the machine.

```
<head>

  <title>Machiko by Wilson</title>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <link rel="stylesheet" href="assets/css/main.css" />
  <link rel="stylesheet" href="assets/css/default.css" />
  <link rel="stylesheet" href="assets/css/component.css" />
  <script src="assets/js/modernizr.custom.js"></script>
  <script type="text/javascript" src="assets/js/jquery.js"></script>
  <script src="assets/js/index.js"></script>


</head>
```

- We made some independent html pages in the app folder, such as header/footer/nav, which is convenient to manage duplicate information in many pages. While users only have to insert a script in each page then each page will automatically load those html pages from app folder.

```
</head>

<body>

  <!-- Header -->
  <div id="header"></div>

  <!-- Nav -->
  <div id="nav"></div>
```

```
  <!-- Footer -->
  <div id="footer"></div>

  <!-- Scripts -->
  <script src="assets/js/jquery.min.js"></script>
  <script src="assets/js/jquery.scrollex.min.js"></script>
  <script src="assets/js/skel.min.js"></script>
  <script src="assets/js/util.js"></script>
  <script src="assets/js/main.js"></script>
  <script src="assets/js/toucheffects.js"></script>
  <script>
    $(function () {
      $('#header').load('app/header.html');
      $('#nav').load('app/nav.html');
      $('#footer').load('app/footer.html');
    })
  </script>
```

- It is quite interesting that we have learned the importance of using HTML's tag like div, section and fieldset to make the web page to be more clean.

```
<section id="two" class="wrapper style2">
  <div class="inner">
    <fieldset>
      <legend><h3>Product Specification</h3></legend>
      <div class="container1">
        <div class="product">
          <img id = "picture" height="470px" width="auto" src="" />
          <div class="product-info">
            <p>Product: <b><span id="title"> </span></b></p><br></br>
            <p>Price: <b><span id="price"> </span> &#163;</b></p><br></br>
            <div class="clear"></div>
            <p></p>
            <p><span>Quantity: </span>
              <input id="squan" type="number" name="points" min="0" max="10" value="0"/>
            </p>
            <div class="cart">
              <input type="submit" id= "BuyI" value="Buy" class="button special"/>
            </div>
          </div>
          <div class="clear"></div>
        </div>
        <br></br>

        <!-- Product Table -->
        <div id="spec"></div>

      </div>
    </fieldset>
  </div>
</section>
```

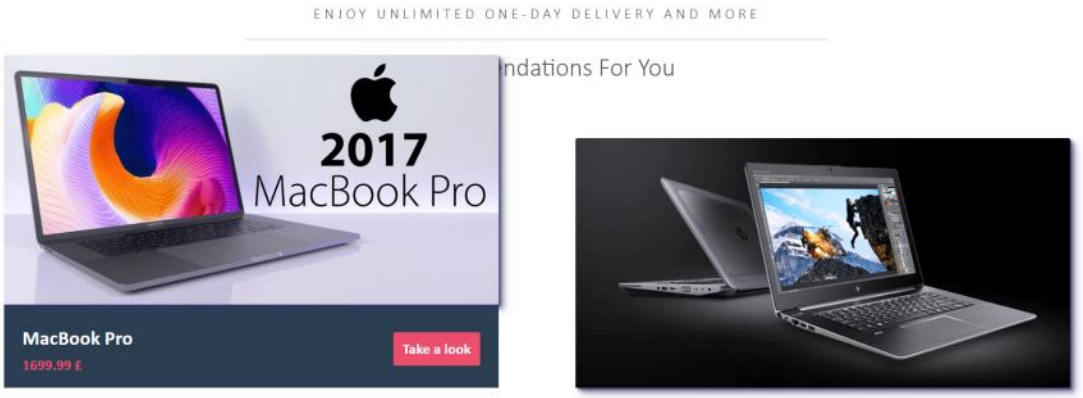

- HTML paragraphs are defined by  tags. For example Enjoy Unlimited One-Day Delivery and More The browser automatically adds a blank line before and after the paragraph.

ENJOY UNLIMITED ONE-DAY DELIVERY AND MORE

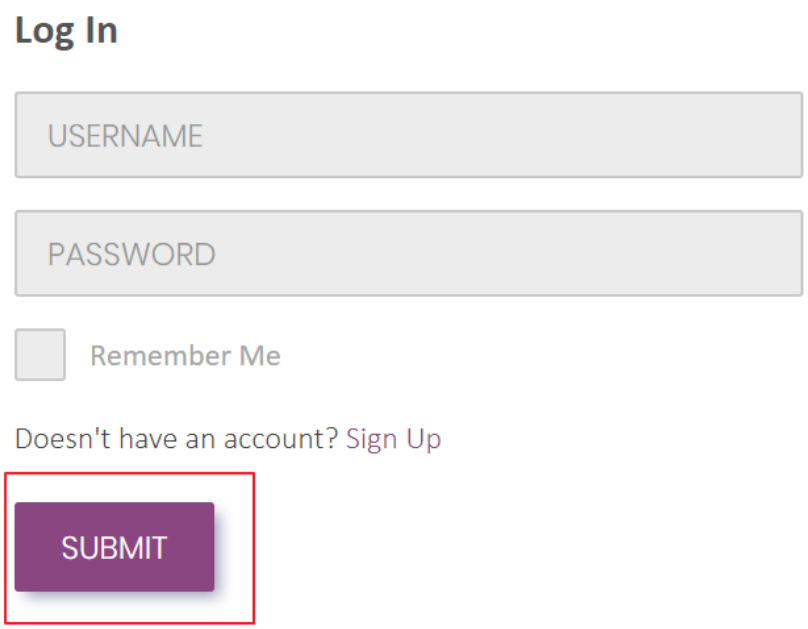
Recommendations For You

- HTML links are defined by the <a> tag.

```
<li>
  <figure>
    <div class="image fit">
      <a><img id = "product_image_1" height="270" width="535" src="images/products/macpro01.png" alt="" /></a>
      <figcaption>
        <h3 id="product_name_1">MacBook Pro</h3>
        <span id="product_price_3"><b>1699.99 &#163;</b></span>
        <a href="product.html" id = "look0" class = "look">Take a look</a>
      </figcaption>
    </div>
  </figure>
</li>
```




- The <input> tag specifies the input fields for the user to enter data. Depending on the type attribute, there are several forms of input fields. The input fields can be text fields, check boxes, password fields, radio buttons, buttons, and the like.



```
<ul class="actions">
  <input type="submit" id="submit" value="Submit" class="button special"/>
</ul>
```

- Using the table method in the HTML, the information on the WWW can be made clear and more readable. In the HTML Table, you can fill in each of the following types of information in each cell like Text (Text), Images, Inputs (Forms), List of items (Lists) and Information (Anchors).

**Specification**

| Event | Description |
|---|---|
| Brand: | Apple |
| Memory: | 4GB DDR3 RAM |
| Storage: | 256GB |
| Processor: | Intel Core i5 |
| Screen Size: | 13.3 inches |
| Graphics: | Intel HD Graphics 4000 |

```
<table>
  <thead>
    <tr>
      <th>Event</th>
      <th>Description</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td style="width:200px">Brand: </td>
      <td><span id="brand"> </span></td>
    </tr>
    <tr>
      <td>Memory: </td>
      <td><span id="memory"> </span></td>
    </tr>
    <tr>
      <td>Storage: </td>
      <td><span id="storage"> </span></td>
    </tr>
    <tr>
      <td>Processor: </td>
      <td><span id="processor"> </span></td>
    </tr>
    <tr>
      <td>Screen Size: </td>
      <td><span id="screen"> </span></td>
    </tr>
    <tr>
      <td>Graphics: </td>
      <td><span id="graphics"> </span></td>
    </tr>
  </tbody>
</table>
```

## 2. CSS

CSS Cascading Style Sheets Cascading Style Sheets. CSS is mainly used to set the text content (fonts, size, alignment, etc.) in the HTML page, appearance (width and height, border style, margins, etc.) of the image, layout appearance of the layout, etc.

- The property is the style attribute you want to set. Each property has a value. The property and value are separated by a colon. The CSS declaration always ends with a semicolon (;) and the declaration group is enclosed in braces ({}).

```css
@charset "UTF-8";
@import url(font-awesome.min.css);
@import url("https://fonts.googleapis.com/css?family=Poppins:300,400,500,600,700");

/* Reset */

 html, body, div, span, applet, object, iframe, h1, h2, h3, h4, h5, h6, p, blockquote, pre, a, abbr, acronym, address, big
 , cite, code, del, dfn, em, img, ins, kbd, q, s, samp, small, strike, strong, sub, sup, tt, var, b, u, i, center, dl, dt,
  dd, ol, ul, li, fieldset, form, label, legend, table, caption, tbody, tfoot, thead, tr, th, td, article, aside, canvas,
 details, embed, figure, figcaption, footer, header, hgroup, menu, nav, output, ruby, section, summary, time, mark, audio,
  video {
   margin: 0;
   padding: 0;
   border: 0;
   font-size: 100%;
   font: inherit;
   vertical-align: baseline;
 }

 article, aside, details, figcaption, figure, footer, header, hgroup, menu, nav, section {
   display: block;
 }

 body {
   line-height: 1;
 }

 ol, ul {
   list-style: none;
 }
   content: '';
   content: none;
 }
```

- ID selector: If you want to set CSS styles in HTML elements, you need to set the "id" and "class" selectors in the element. The id selector can specify a specific style for HTML elements marked with a specific id. The HTML element sets the id selector with the id attribute, and the id selector in CSS is defined with "#".

```css
h1, h2, h3, h4, h5, h6 {
  font-weight: 300;
  line-height: 1.5;
  margin: 0 0 1rem 0;
}

  h1 a, h2 a, h3 a, h4 a, h5 a, h6 a {
    color: inherit;
    text-decoration: none;
  }
```

```html
<section id="four">
  <div class="inner">
    <ul class="grid cs-style-2">
      <header class="align-center">
        <p class="special">Enjoy Unlimited One-Day Delivery and More</p>
        <h2>Recommendations For You</h2>
      </header>
```


- Class selector: The class selector is used to describe the style of a set of elements. The class selector is different from the id selector. The class can be used in multiple elements. It is represented in HTML as a class attribute. In CSS, the class selector is shown with a dot ".".

```css
.image.right {
  float: right;
  margin: 0 0 1rem 1.5rem;
  top: 0.25rem;
}

.image.fit {
  display: block;
  margin: 0 0 2rem 0;
  width: 100%;
}

 .image.fit img {
    width: 100%;
  }

.image.main {
  display: block;
  margin: 0 0 3rem 0;
  width: 100%;
}
```

```html
<figure>
  <div class="image fit">
    <a><img id = "product_image_1" height="270" width="535" src="images/products/macpro01.png" alt=""
    /></a>
    <figcaption>
      <h3 id="product_name_1">MacBook Pro</h3>
      <span id="product_price_3"><b>1699.99 &#163;</b></span>
      <a href="product.html" id = "look0" class = "look">Take a look</a>
    </figcaption>
  </div>
</figure>
```


```css
.image.right {
  float: right;
  margin: 0 0 1rem 1.5rem;
  top: 0.25rem;
}

.image.fit {
  display: block;
  margin: 0 0 2rem 0;
  width: 100%;
}

.image.fit img {
  width: 50%;
}

.image.main {
  display: block;
  margin: 0 0 3rem 0;
  width: 100%;
}
```



- External style sheet: Applicable to situations where styles need to be applied to many pages, each page is linked to a style sheet using a <link> tag to get css files from \assets\css. The <link> tag is at the (document's) head:

```
<head>

  <title>Machiko by Wilson</title>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <link rel="stylesheet" href="assets/css/main.css" />
  <link rel="stylesheet" href="assets/css/default.css" />
  <link rel="stylesheet" href="assets/css/component.css" />
  <script src="assets/js/modernizr.custom.js"></script>
  <script type="text/javascript" src="assets/js/jquery.js"></script>
  <script src="assets/js/index.js"></script>


</head>
```

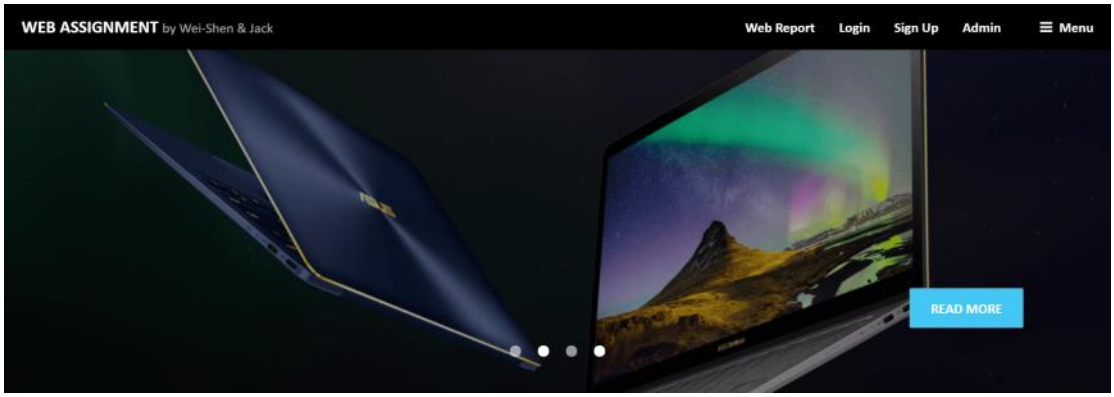
- Due to the need of some web pages, more in-depth use of CSS's "Pseudo Element". It is not an element of a real web page, but its behavior and performance are the same as those of a real web page element. It can also be manipulated using CSS.

- ::before, ::after are probably the most commonly used pseudo-elements, both exist with the attribute display:inline-block, ::before adds content "before" the original element, ::after is the original element "after" adds content, and the pseudo element also "inherits" the original element's attributes

```css
.banner > article {
  -moz-transition: opacity 1.5s ease, visibility 1.5s;
  -webkit-transition: opacity 1.5s ease, visibility 1.5s;
  -ms-transition: opacity 1.5s ease, visibility 1.5s;
  transition: opacity 1.5s ease, visibility 1.5s;
  background-attachment: fixed;
  background-position: center;
  background-repeat: no-repeat;
  background-size: cover;
  height: 100%;
  left: 0;
  opacity: 0;
  position: absolute;
  text-align: center;
  top: 0;
  visibility: hidden;
  width: 100%;
  z-index: 0;
  box-shadow:4px 4px 12px -2px rgba(20%,20%,40%,0.5);
}

.banner > article:before {
  content: '';
  display: inline-block;
  height: 100%;
  vertical-align: middle;
}

.banner > article:after {
  content: '';
  display: block;
  width: 100%;
  height: 100%;
  position: absolute;
  top: 0;
  left: 0;
  background: #000;
  opacity: 0.35;
}
```




## 3. Javascript

JavaScript programming language allows developers to build complex things on web pages. As long as the webpage displays not only static information, but also timely content updates, interactive maps, 2D/3D animation, rolling video jukeboxes, and the like, you can assume that this web page is built using JavaScript.

Literally, we found a module from Github, which provided a great feature to decorate our webpage. We not only used html and CSS to adjust the animation, but also insert a JavaScript to move the object when touch it.

```
/** Used Only For Touch Devices **/
( function( window ) {

  // for touch devices: add class cs-hover to the figures when touching the items
  if( Modernizr.touch ) {

    // classie.js https://github.com/desandro/classie/blob/master/classie.js
    // class helper functions from bonzo https://github.com/ded/bonzo

    function classReg( className ) {
      return new RegExp("(^|\\s+)" + className + "(\\s+|$)");
    }

    // classList support for class management
    // altho to be fair, the api sucks because it won't accept multiple classes at once
    var hasClass, addClass, removeClass;

    if ( 'classList' in document.documentElement ) {
      hasClass = function( elem, c ) {
        return elem.classList.contains( c );
      };
      addClass = function( elem, c ) {
        elem.classList.add( c );
      };
      removeClass = function( elem, c ) {
        elem.classList.remove( c );
      };
    }
    else {
      hasClass = function( elem, c ) {
        return classReg( c ).test( elem.className );
      };
      addClass = function( elem, c ) {
        if ( !hasClass( elem, c ) ) {
          elem.className = elem.className + ' ' + c;
        }
```


For now, we can skillfully use scripts to gather information from input web pages and send the information to the (POST) server. Or get (GET) data from the server and alter the parameters in the web page.

```
$("#submit").on("click", function() {
  $.ajax({
    type: "post",
    url: "/login",
    data: {
      username: $("#user").val(),
      password: $("#pass").val()
    },
    dataType: "json",
    success: function(res, status, xhr) {
      if (res.code != 200) {
        alert(res.msg);
        return;
      }
      window.location.href = "userinfo.html";
    },
    error: function(err) {
      console.log("ajax error");
      console.log("ajax error" + JSON.stringify(err));
    }
  });
});
```

```
$(function(){
    $.ajax({
        type:"GET",
        url:"/detail",
        data:{},
        dataType:"json",
        success:function(res, status, xhr){
            if(res.code != 200){
                alert(res.msg);
                window.location.href = "login.html";
                return;
            }
            $("#user").html(res.user);
            $("#name").html(res.name);
            $("#birt").html(res.birt);
            $("#addr").html(res.addr);
            $("#tele").html(res.tele);
        },
        error:function(err){
            console.log("ajax error"+JSON.stringify(err));
        }
    });
});
```

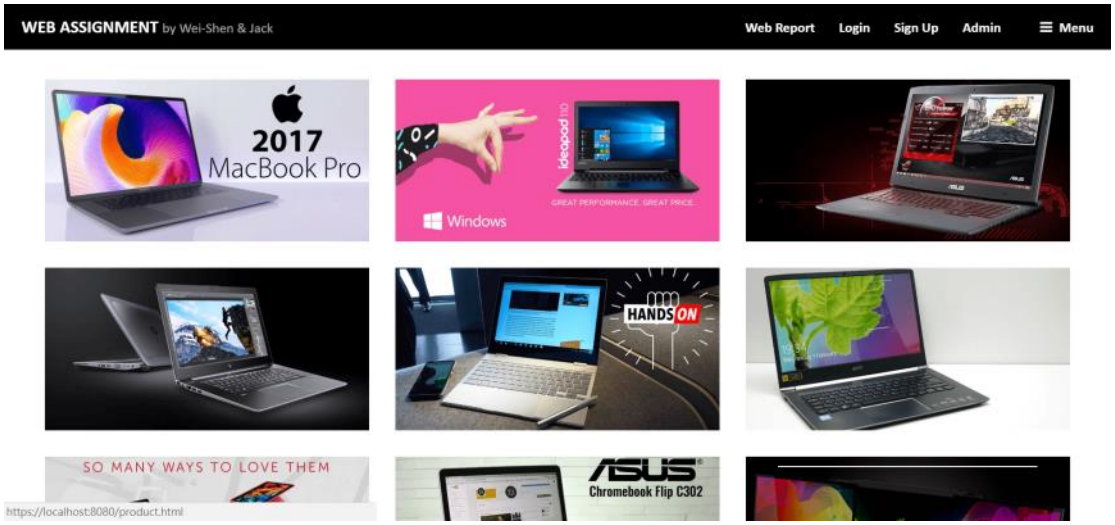
We packaged the html, such as storing mutual information in a single html, such as header/footer/nav, which is convenient to manage duplicate information in many pages. So we can update all the pages immediately if we make any changes with our header, footer and nav.

In About_us, we set a time interval to achieve a systematic replacement of the picture and play back.  And set the click event so that the image can be replaced after clicking continuously.

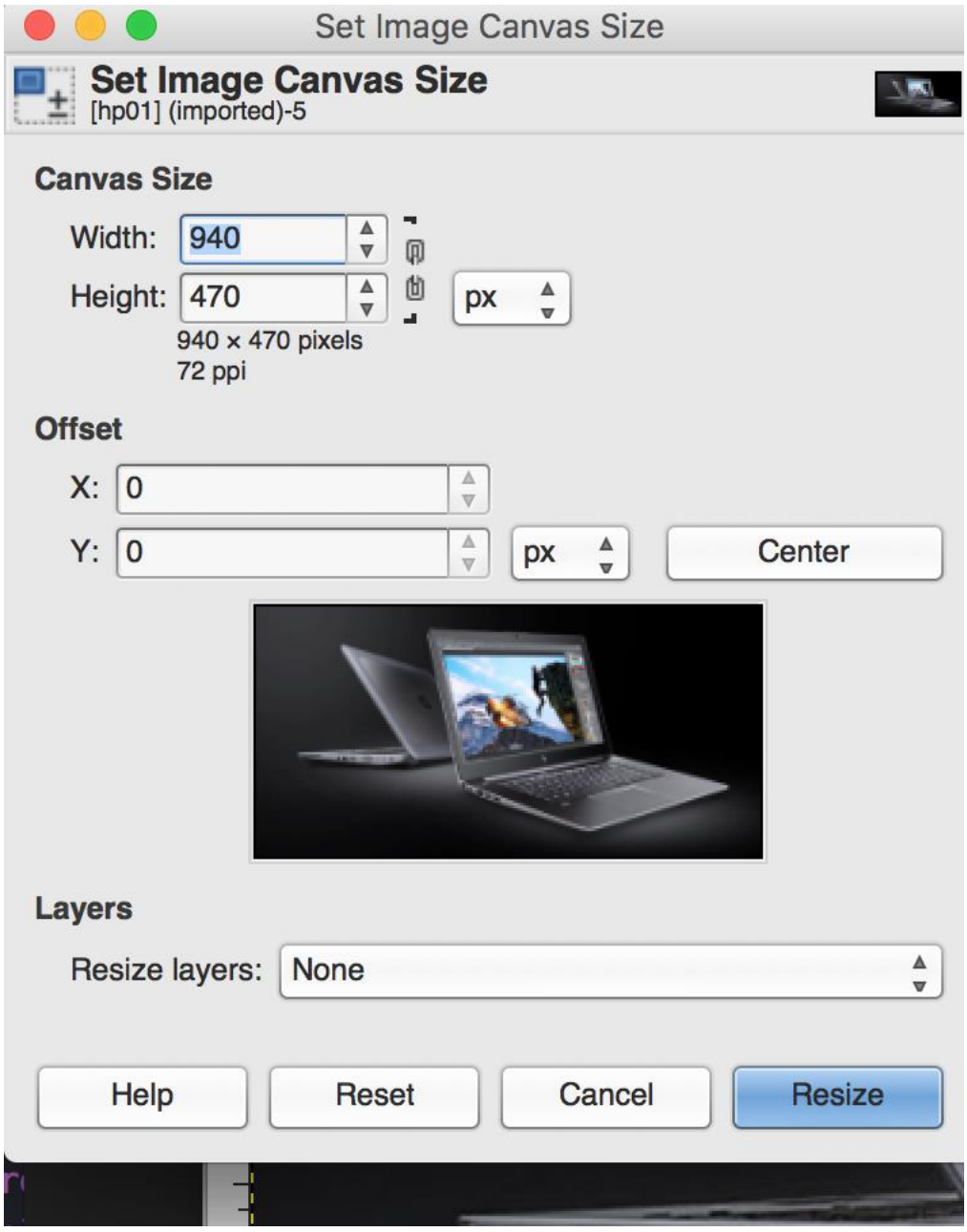
```
setInterval(changeImg,timeInterval);
var i = 0;
function changeImg()
{
  $("#img1").fadeOut(300, function() {
     $("#img1").attr("src", imageWei[i]);
     if(i == 2){i = 0;}
     else{
       i ++;
     }
     $("#img1").fadeIn(300);
   })
}
```

## 4. PNG

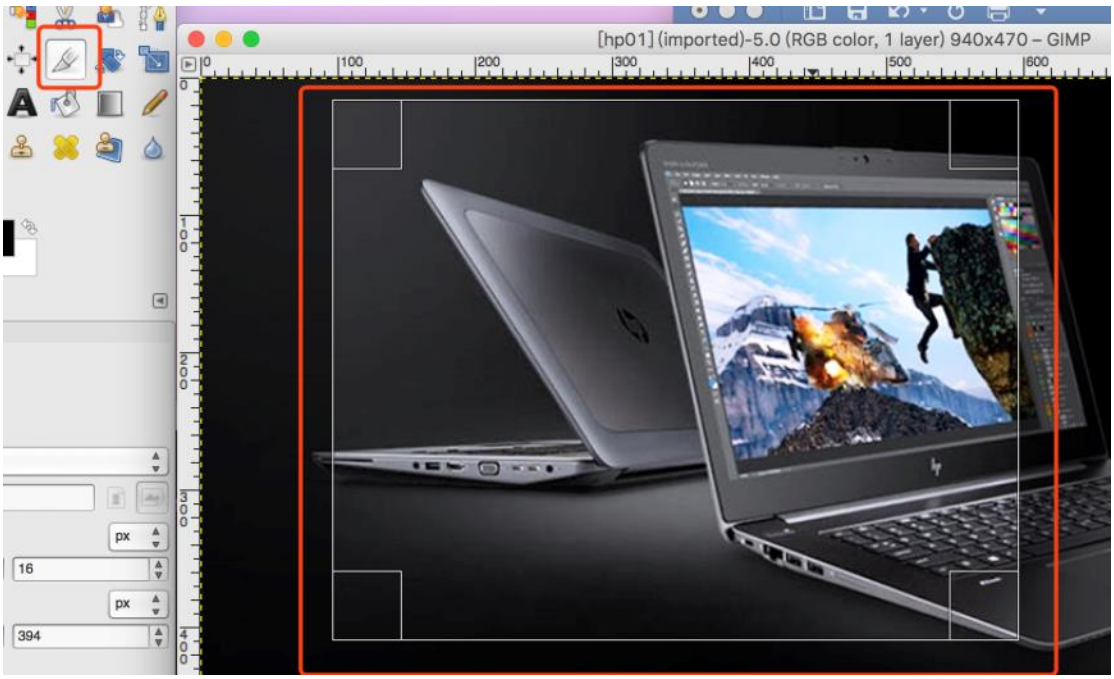
Basically, most of the pictures were found on the internet, then we used GIMP to crop each picture and adjust their size and resolution.

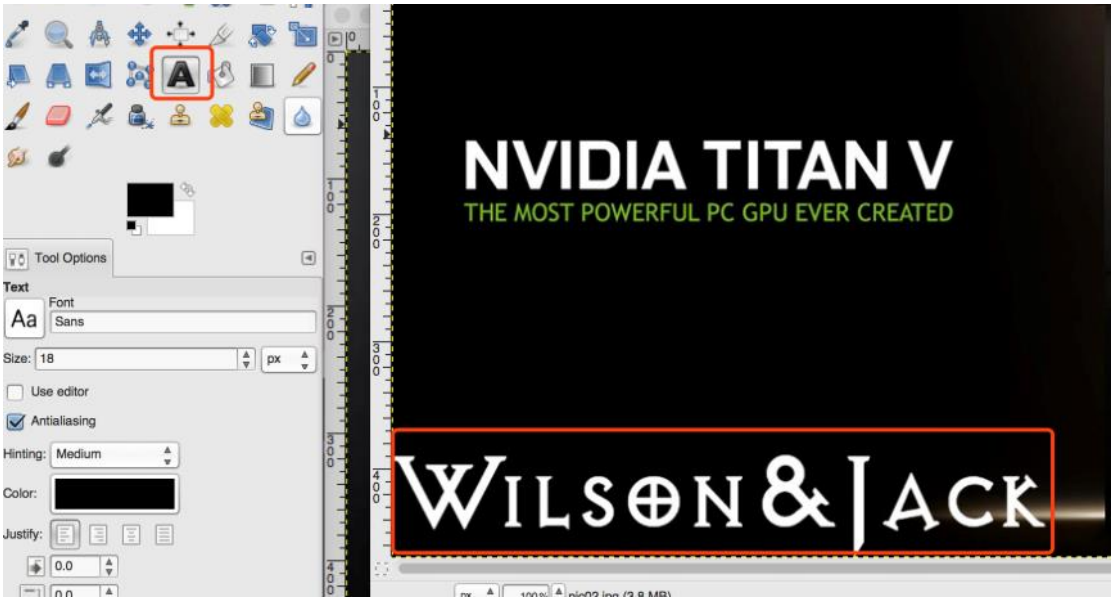

Set Image Canvas Size



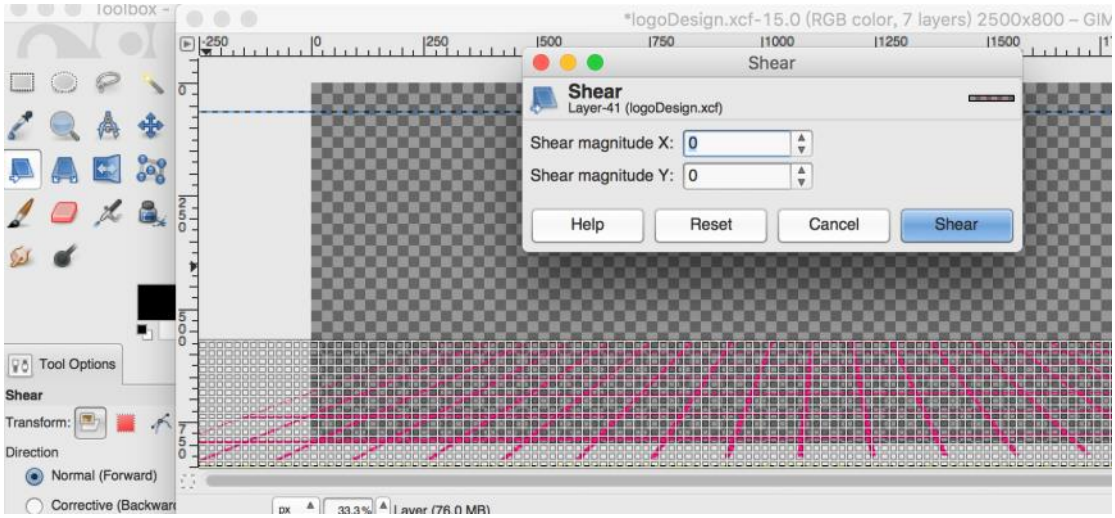

Crop tool: Remove edge of one image




Text Tool




Shear Tool: shear the layer, path or selection

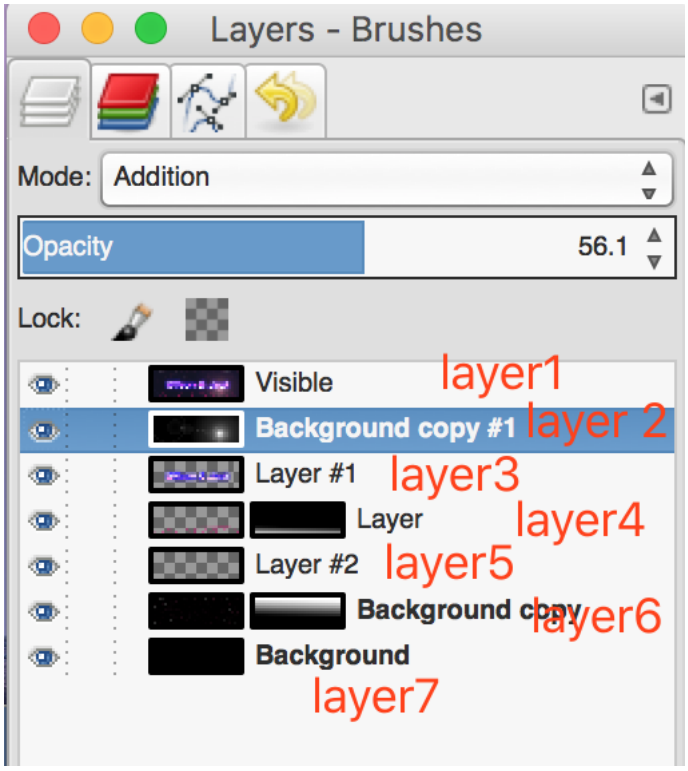



We used GIMP to design the picture at the top of the About.html page.


- Bucket Fill Tool to fill an area with color or pattern fill (FG color fill and Pattern fill)

- Layer – Brushes: We used this tool to implement multiple combinations of elements in this background. Like there are 7 layers in this background image.

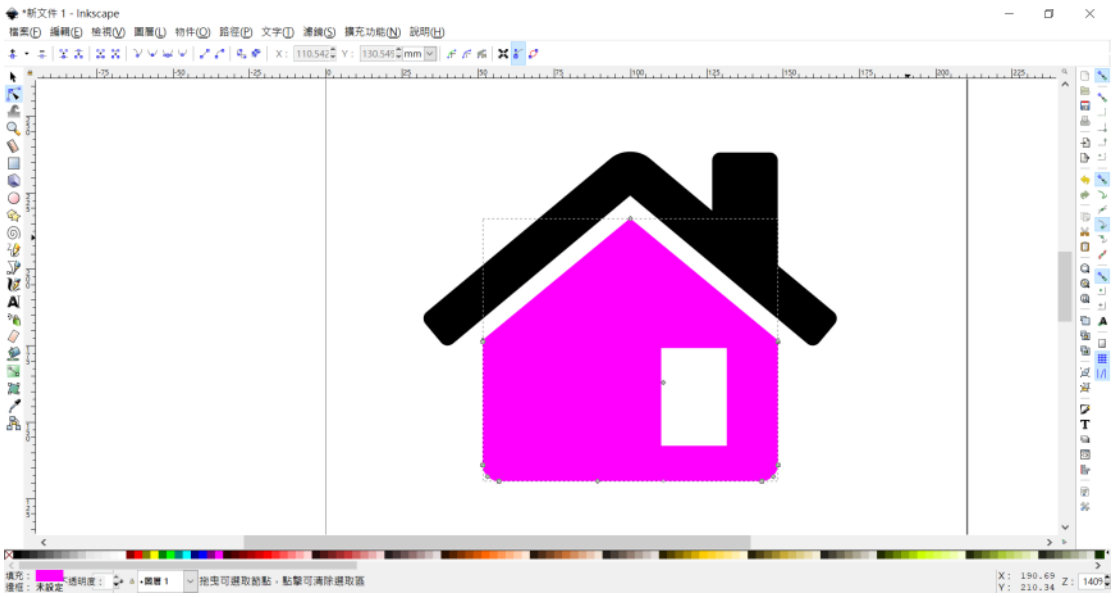
- layer 7 is the black background, layer 6 is “stars” using Gaussian Blur, layer 5 and layer 4 are for the red grid, layer 3 is the Wilson & Jack text, layer 2 is the round light using Lighting Effects, layer1 is the complete image which has been set to visible.

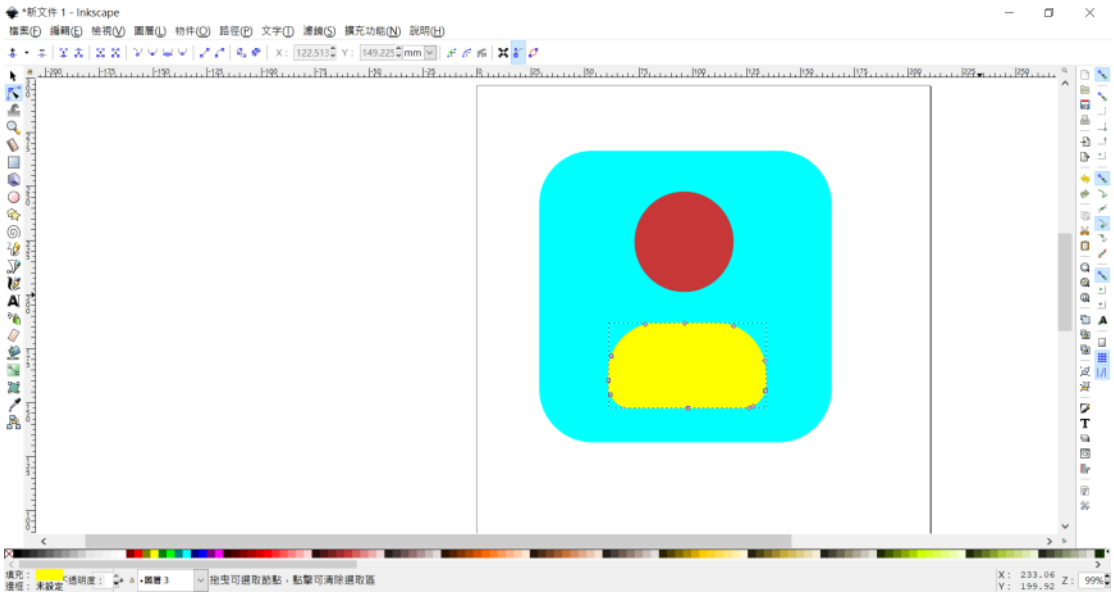


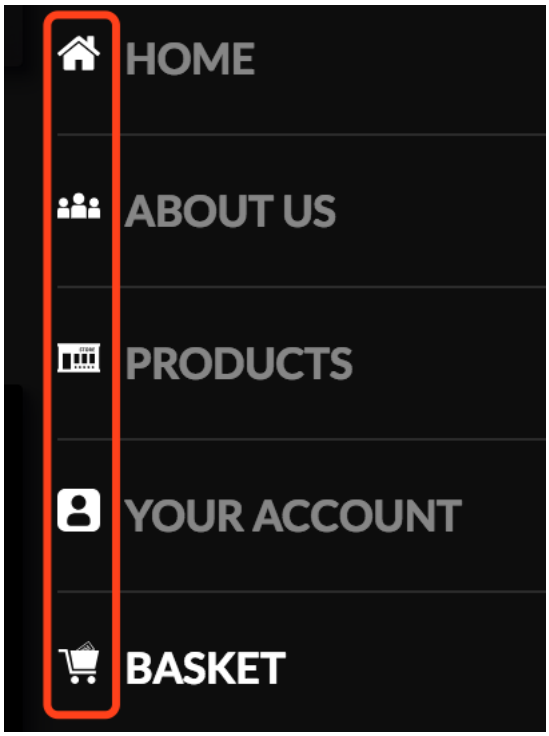

**5. SVG**

This might be our first time to implement the vector design, while Inkscape is a professional level vector drawing software! It can pull out the Bezier curve, create a variety of three-dimensional objects, and draw and paint on vector image files. There are also rich text design tools.

- Rich vector drawing tools, layer editing, vector objects, text design, color fill
- Supporting the transfer of the image file format is also very rich, basically meet the vector map editing and output requirements.
- Supports many advanced SVG features (such as: mark, copy, alpha blending, etc.). In addition to basic points, lines, faces, circles, rectangles, curve. Users can also do advanced shape drawing techniques such as editing nodes and path operations. In the color performance can also be presented gradient and transparent and other special color effects


*新文件 1 - Inkscape
檔案(F) 編輯(E) 檢視(V) 圖層(L) 物件(O) 路徑(P) 文字(T) 濾鏡(S) 擴充功能(N) 說明(H)
填充：
描框：未設定
透明度：
圖層 1
拖曳可選取節點，點擊可清除選取器
X：190.69
Y：210.34
Z：1409


*新文件 1 - Inkscape
檔案(F) 編輯(E) 檢視(V) 圖層(L) 物件(O) 路徑(P) 文字(T) 濾鏡(S) 擴充功能(N) 說明(H)
填充：
描框：未設定
透明度：
圖層 3
拖曳可選取節點，點擊可清除選取器
X：233.06
Y：199.92
Z：99%




## 6. Server:

We used express as a server framework and use the 8080 as a method to increase security. On the one hand, this server will treat unreasonable URLs as 404 pages, which we used Gimp to make a 404 picture as 404 pages to inform users. On the other hand, if users try to visit the admin page, page will be directed to the login interface.

- Content negotiation

```
//content negotiation
app.use(function(req, res, next){
  var otype = "text/html";
  var ntype = "application/xhtml+xml";
  if(path.extname(req.url) == ".html"){
    var header = req.headers.accept;
    var accepts  = header.split(",");
    var type = accepts.indexOf(ntype)>= 0 ? ntype: otype;
    res.header("Content-Type", type);
  }
  next();
});
```

- To avoid uppercase and lowercase issues of URLs, we convert all the URLs into lowercase.

- We made the server's https certificate (certificates and authentication files stored in /encrypt) and used it in the server. Next, the user's login information is saved in the server through express-session. Finally, use the helmet framework to increase the security of your web pages.

- If you wish to view your accounts and shopping cart, users are required to log in advance, otherwise it returns to the corresponding page.

- For the wrong server information, the web page will send the corresponding error code to the user and prompt the analogous information.

## 7. Database:

We utilized SQLite3 to record our data. The file of the database is located in "web/database". There are three files (setup.js, db.db_connection.js and web.db). In addition, there is a setup.js to initialize the database in that it allows users to insert some example data.

Basically, we designed three tables which have been currently used in our website. Table 1 is “user” with 6 attributes to record user's basic information. Table 2 is “product” with 10 attributes to record product's information. Table 3 is “basket” with 3 attributes to record each purchase data.

The db_connection.js is to set up the database and deal with the error using callback, while there are three different methods which have been used to deal with the data. Finally, Function db_connection is tantamount to searching for one row. The function db_connectionAll is to find all rows of one query and the function db_connectionRun is to run update/insert/delete queries.

## 8. Dynamic pages

There are three dynamic pages (userinfo.html, product.html, basket.html). Data from these pages requesting from the server are from the web.db database. We definitely designed these pages by ourselves. We used the function of ajax to request data and deliver it from the server.

The userinfo.html and product.html only request one set of data, but the basket.html requests several rows of data. The basket.html is the most interesting and difficult part of this project.

The detailed information in basket.html can be explained as following steps:

- Firstly, it will check the state of login, while it is not until one user has logged in then it will jump to the login.html.

- Secondly, if the user has not purchased any products, this page will show the message “There is no product in your basket.” When the user bought one product, the user’s information and the product’s information will be shown on this page. The SQL statement of this query is that:

```
var sql = util.format("SELECT * FROM (select SUM(quantity) AS sum, product_id" +
"AS pid, user_id AS uid from basket where user_id = '%s' group by product_id ) "+
"AS a JOIN product p ON a.pid = p.id JOIN user u ON u.username = a.uid",
req.session.username);
```

The result of this query may include many rows as one user can buy numerous products, so the whole data (an array) will be sent to the client side. In basket.js (client side script), the following loop will be used to change the content of the basket.html.

```
for (var i = 0; i < size; i++) {
  $('#noorder').fadeOut(5);
  $('#d' + i).fadeIn(5);
  var productname = "#productname" + i;
  var quan = "#quan" + i;
  var price = "#price" + i;
  var amount = "#amount" + i;
  $(productname).html(data[i].title);
  $(quan).html(data[i].sum);
  $(price).html(data[i].price);
  var total = data[i].sum * data[i].price;
  $(amount).html(total);
  sum += total;
}
sum = sum.toFixed(2);
$('#total').html(sum);
```

**9. Depth:**

The depth of exploration of this site includes but is not limited to the following:

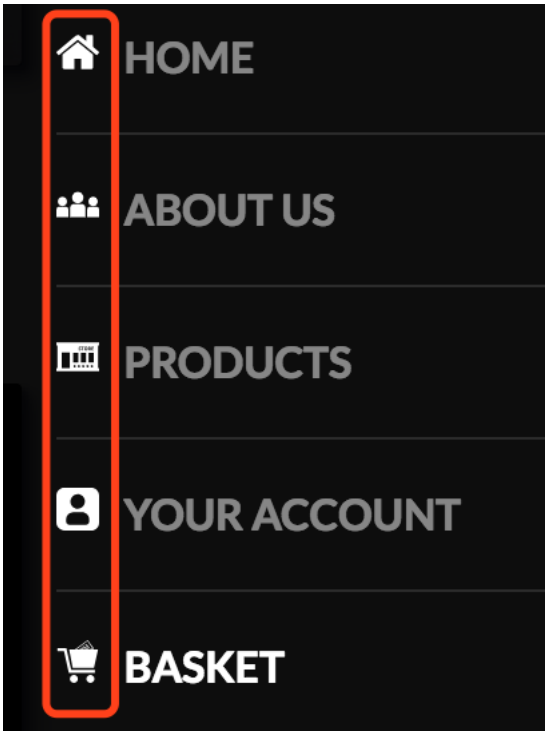
- The use of HTML tags, text content, hyperlinks, import of various scripts, etc. This website also shows reasonable uses of DIV blocks to make each HTML to be more concise and easy managed.

- We learned how to adjust the properties of each tag (border, background, color, alignment, etc.) by controlling the CSS files; similarly, we have also explored the use of some CSS libraries like bootstrap.css and style.css, tried to incorporate them into this website in order to make the display beautiful.

- In terms of JavaScript, we skillfully use a variety of functions to enable the elements to have animation and transformations. We have mastered the method of server exchange using AJAX. For now, we can use it flexibly

regardless of whether it accepts server data or transfers data to the server. In addition, we can manipulate various attributes of HTML. For instance, detecting a single attribute value and responding to a click action. All can be processed in this website. To a lesser extent, we can utilize excellent frameworks to increase the aesthetics of animation.

- In terms of PNG, we not only explored basic operations, but also enhanced the ability to use layers, and make use of the open source tool like GIMP to create our own website Logo. For instance, we designed multiple tag symbols for our website navigator.




- We made HTTPS encrypted information processing through self-made certificates, and store the key file in the private domain.

- This website makes reasonable use of the session to record user login information, and also uses app.use (helmet()) to increase the security of the site. When users access a non-existent page, use res.status.sendFile() to output a 404 picture.

- In addressing databases, this web server can handle SQL statements. And the results (single or multiple) are transmitted to the client by using Json type data.

- This site stores the database file (Type: SQLite3), initialization database script, and database connection script in a separate folder in the restricted domain. And it has been built three tables which can store user, product, and shopping cart information. This website utilizes SQLite3 query statements to achieve access to the database and return specific or multiple pieces of information. However, if users make any wrong operations to the database, the website will use callback to protect database information automatically.

- Last but not least we explored a definite dynamic web page and a dynamic web page containing multiple similar records and successfully applied it to this site. Like the basket.html automatically adjusts the spacing and content of web pages based on how many records are recorded, and successfully integrates the data which is queried into web pages.