



# Project Web Programming

By

<b>Miss Kanyavee</b>	<b>Likitwattanakij</b>	<b>6488038</b>
<b>Miss Arada</b>	<b>Raksapatcharawong</b>	<b>6488066</b>
<b>Mr. Tanawat</b>	<b>Raweebachwarathon</b>	<b>6488067</b>
<b>Mr. Ittikorn</b>	<b>Suksai</b>	<b>6488088</b>
<b>Mr. Sarttra</b>	<b>Prasongtichol</b>	<b>6488124</b>

A Report Submitted in Partial Fulfillment of

the Requirements for

**ITCS212\_Web Programming**

**Faculty of Information and Communication Technology**

**Mahidol University**

**2022**

## Project of Web Programming

### 1. Project Overview

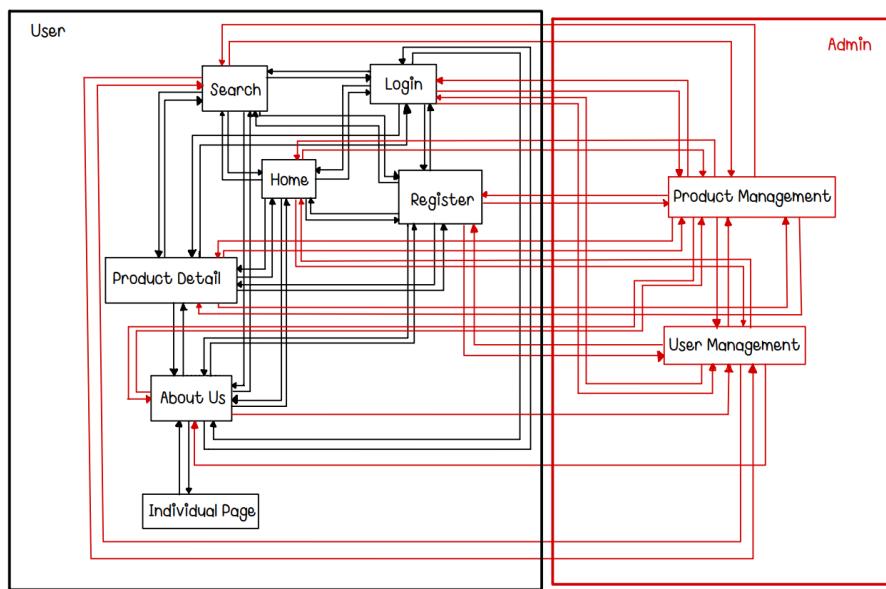
- Phase 1 (HTML,CSS)

For this phase, it will be the design of website pages in each section using html to design various sections within the web page. Whether it's a search box navigation bar or input field. Which will be put in various pages such as home page, Admin page or Product management page. Then, CSS code will be used to decorate and design the website to be more beautiful to be ready for Phase 2.

- Phase 2 (JavaScript [Node.js], Web Services, and DB)

Phase 2 will be the back end which for this phase will connect data with SQL or Database and API to make web service, which will use Javascript (Node.js) to help in connecting data in different parts by uses index.js as the center for all parts of the web service.

### 2. Navigation Diagram of your web application.



### 3. Details of your web application and code.

- All pages have the same header navigation bar and footer navigation bar template but there are changes in variations of color in individual.

Fundamental navigation bar code for all html files

style.css: this is css class source code on header and footer nav bar on IGD Website

```
*{
    font-family:'Dangrek';
}
@keyframes fadeIn {
    0% { opacity: 0; }
    100% { opacity: 1; }
}
.fade-in-image {
    animation: fadeIn 2s;
}
.fade-in-text {
    animation: fadeIn 1.5s;
}
header, footer{
    background-color: #1FC454;
    padding: 5px;
    text-align: center;
}
header{
    background: #rgb(167,255,153);
    background: linear-gradient(90deg, #rgba(167,255,153,1) 0%, #rgba(95,224,107,1) 0%, #rgba(0,168,73,1) 100%);
    border-bottom: 3.5px solid #009834;
    height: 200px;
    display: block;
    position: relative;
    overflow: hidden;
    z-index: 1;
}

footer {
    background: #rgb(167,255,153);
    background: linear-gradient(90deg, #rgba(167,255,153,1) 0%, #rgba(95,224,107,1) 0%, #rgba(0,168,73,1) 100%);
    display: block;
    position: absolute;
    bottom: 0;
    left: 0;
    right: 0;
    margin: 0;
    min-height: 7%;
    width: 100%;
}
```

```
nav ul li {
    display: inline-block;
}
.head{
    background: #fcfc2e;
    border:4px solid #009834;
    border-bottom:none;
    text-align: center;
    height: 70px;
    width: 60px;
    margin: 0 auto;
    padding-bottom: 30px;
    position: relative;
    top: 0;
    left: 0;
    z-index: 2;
}
nav ul li a.headnav{
    text-decoration: none;
    padding: 0px 30px;
    color: #009834 ;
    border-left: 2px solid #009834 ;
    border-right: 2px solid #009834 ;
}
nav ul li a.headnav:hover{
    color:#18508 ;
}
nav ul li a.footnav{
    text-decoration: none;
    padding: 30px 20px;
    color: black;
}
```

<individual about us> .css : there are numerous variations of navigation bar for this, the example below here.

```
*{
    font-family:'Bad Script';
}
header{
    background: #rgb(150,230,124);
    background: linear-gradient(90deg, #rgba(150,230,124,1) 0%, #rgba(0,175,111,1) 100%);
    border-bottom: 3px solid #white;
    padding: 15px;
    text-align: center;
    height: 180px;
    display: block;
    position: relative;
    overflow: hidden;
    z-index: 1;
}
body{
    background:#whitesmoke;
    min-height: 100vh;
    position: relative;
    flex-direction: column;
    margin: 0;
    padding-bottom: 100px;
}
```

```
nav ul li {
    display: inline-block;
    margin-right: auto;
    margin-left: auto;
}
nav ul li :hover.headnav{
    color: #A06300;
}
nav ul li :hover.footnav{
    color: #A06300;
    text-decoration: underline;
}
.head{
    background: #F7C535;
    border: 3px solid #white;
    text-align: center;
    padding-bottom: 30px;
    width: 80px;
    margin: 0 auto;
    position: relative;
    top: 0;
    left: 0;
    z-index: 2;
}

nav ul li a.headnav{
    text-decoration: none;
    color: #white;
    padding:0px 40px;
    border-right: 2px solid #white;
    border-left: 2px solid #white;
    margin: 0px auto;
}
nav ul li a.footnav{
    text-align: center;
    text-decoration: none;
    font-weight: bold;
    color: #white;
    padding:10px 30px;
    margin: 0px auto;
}
```

- This css use to determines the design for each page on the website.

```

# 6488038_aboutus_style.css
▷ 6488038_aboutus.html
# 6488066_aboutus_style.css
▷ 6488066_aboutus.html
# 6488067_aboutus_style.css
▷ 6488067_aboutus.html
# 6488088_aboutus_style.css
▷ 6488088_aboutus.html
# 6488124_aboutus_style.css
▷ 6488124_aboutus.html
▷ AboutUs.html
▷ Account.html
▷ BoiledRiceWithGrouper.html
▷ ClearSoupWithOmlete.html
▷ GrilledSeaBass_withMisoSauce....
▷ Home.html
▷ Login.html
▷ Product.html
▷ ProductManage.html
(i) readme.txt
▷ Register.html
JS search.js
▷ SpicyChickenSoupWithMushro...
▷ SpicyMixedVegetableSoup.html
# style.css

```

> 6488038\_aboutus\_pic  
 > 6488066\_aboutus\_pic  
 > 6488067\_aboutus\_pic  
 > 6488088\_aboutus\_pic  
 > 6488124\_aboutus\_pic  
 > Phase1\_pic

## - Home



There are three components that provide in the home page.

1. Title part which uses to show the name of the website in graphic representation.

```
<div class="foodheader">
  <div class="titlehome">
    
    <h1 class="foodtopic" style="color: #ffc92e;">IGD's Website</h1>
  </div>
</div>
```

```
.foodheader {
  background: #008356;
  background-image: url(Phase1_pic/pngimg.com\ -\ pasta_PNG15.png);
  background-repeat: no-repeat;
  background-position: 1200px 250px;
  background-size: contain;
  height: 420px;
  overflow: hidden;
}
```

2. Recommend menus, each menu can be examined by clicking on its picture as well as more information link reference.

```
<article class="container1">
  <div class="box2">
    <a href="ClearSoupWithOmelette.html"></a>
    <h3 class="recommend">Clear soup with Omelette</h3>
    <div class="description">
      <a class="more" href="ClearSoupWithOmelette.html">more information</a>
    </div>
  </div>
  <div class="box2">
    <a href="GrilledSeaBassWithMisoSauce.html"></a>
    <h3 class="recommend">Grilled Sea Bass with Miso Sauce</h3>
    <div class="description">
      <a class="more" href="GrilledSeaBassWithMisoSauce.html">more information</a>
    </div>
  </div>
  <div class="box2">
    <a href="SpicyChickenSoupWithMushroom.html"></a>
    <h3 class="recommend">Spicy chicken soup with mushrooms</h3>
    <div class="description">
      <a class="more" style="top:15px;" href="SpicyChickenSoupWithMushroom.html">more information</a>
    </div>
  </div>
  <div class="box2">
    <a href="SpicyMixedVegetableSoup.html"></a>
    <div class="description">
      <h3 class="recommend">Spicy mixed vegetable soup</h3>
      <a class="more" style="padding-top: 55px;" href="SpicyMixedVegetableSoup.html">more information</a>
    </div>
  </div>
  <div class="box2">
    <a href="BoiledRiceWithGrouper.html"></a>
    <div class="description">
      <h3 class="recommend">Boiled Rice with Grouper</h3>
      <a class="more" style="padding-top: 55px;" href="BoiledRiceWithGrouper.html">more information</a>
    </div>
  </div>
</article>
```

### 3. Categories can be divided into 6 types.

```
<article class="container1">
  <div class="box">
    <a href=""></a>
    <div class="description">
      <h3>Low calories cuisine</h3>
    </div>
  </div>
  <div class="box">
    <a href=""></a>
    <div class="description">
      <h3>Fruit and Vegetables</h3>
    </div>
  </div>
  <div class="box">
    <a href=""></a>
    <div class="description">
      <h3>High protein cuisine</h3>
    </div>
  </div>
</article>
```

```
<div class="fade-in-image">
  <article class="container1">
    <div class="box">
      <a href=""></a>
      <div class="description">
        <h3>Beverages</h3>
      </div>
    </div>
    <div class="box">
      <a href=""></a>
      <h3>Dessert</h3>
    </div>
    <div class="box">
      <a href=""></a>
      <h3>Appertizer</h3>
    </div>
  </article>
</div>
```



Low calories cuisine



Fruit and Vegetables



High protein cuisine



Beverages



Dessert



Appertizer

## - Product

Search which includes 3 criteria consist of Type Nutrition Other factors.



```
<form id="search-form">
  <section class="searchpart" id="search">
    <h1 class="foodelection1" style="font-size: 40px; color: #ffc92e;">
      Explore more food!
    </h1>
    <div class="searchbar">
      <input class="input1" type="search" id="search-input" placeholder="Search" />
      <button type="submit" onclick="searchFood()">Search</button>
    </div>
    <div class="custom-select">
      <select id="type">
        <option class="dropdown" value="0">Type</option>
        <option class="dropdown" value="main dish">Main Dish</option>
        <option class="dropdown" value="appetizer">Appetizer</option>
        <option class="dropdown" value="dessert">Dessert</option>
        <option class="dropdown" value="beverage">Beverage</option>
      </select>
      <select id="nutrition">
        <option class="dropdown" value="0">Nutrition</option>
        <option class="dropdown" value="protein">Protein</option>
        <option class="dropdown" value="fat">Fat</option>
        <option class="dropdown" value="fruit and vegetables">Fruit and vegetables</option>
        <option class="dropdown" value="starch">Starch</option>
        <option class="dropdown" value="dairy">Dairy</option>
      </select>
      <select id="factor">
        <option class="dropdown" value="0">Other Factor</option>
        <option class="dropdown" value="seafood allergic">Seafood allergic</option>
        <option class="dropdown" value="low calories">Low Calories</option>
        <option class="dropdown" value="heart disease">Heart disease</option>
        <option class="dropdown" value="diabetes">Diabetes</option>
        <option class="dropdown" value="obesity">Obesity</option>
      </select>
    </div>
  </section>
</form>
```

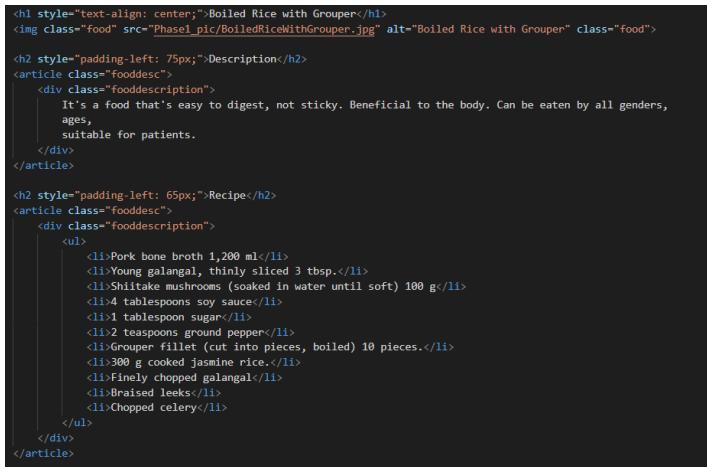
The product will be indicated here on div id= "search-output"

```
<div id="search-output"></div>
```



## - Product detail

Product Details in each page. The components

```
<h1 style="text-align: center;">Boiled Rice with Grouper</h1>

<h2 style="padding-left: 75px;">Description</h2>
<article class="fooddesc">
  <div class="fooddescription">
    It's a food that's easy to digest, not sticky. Beneficial to the body. Can be eaten by all genders, ages, suitable for patients.
  </div>
</article>

<h2 style="padding-left: 65px;">Recipe</h2>
<article class="fooddesc">
  <div class="fooddescription">
    <ul>
      <li>Pork bone broth 1,200 ml</li>
      <li>Young galangal, thinly sliced 3 tbsp.</li>
      <li>Shiitake mushrooms (soaked in water until soft) 100 g</li>
      <li>4 tablespoons soy sauce</li>
      <li>1 tablespoon sugar</li>
      <li>2 teaspoons ground pepper</li>
      <li>Grouper fillet (cut into pieces, boiled) 10 pieces.</li>
      <li>300 g cooked jasmine rice.</li>
      <li>Finely chopped galangal</li>
      <li>Braised leeks</li>
      <li>Chopped celery</li>
    </ul>
  </div>
</article>
```

```
<h2 style="padding-left: 75px;">Method</h2>
<article class="fooddesc">
  <div class="fooddescription">
    <ol>
      <li>Boil pork bone broth with young galangal until boiling, add shiitake mushrooms, season with light soy sauce, sugar and ground pepper, reduce heat and simmer until boiling broth. Set aside.</li>
      <li>Heat the water in another pot until it boils. Add jasmine rice to cook in boiling water until soft, scoop into a bowl, and place blanched grouper. Put the pork bone broth, sprinkle finely pounded galangal, fried garlic, and celery. Eat while hot with soybean sauce.</li>
    </ol>
  </div>
</article>
```

## Boiled Rice with Grouper



### Description

It's a food that's easy to digest, not sticky. Beneficial to the body. Can be eaten by all genders, ages, suitable for patients.

### Recipe

- Pork bone broth 1,200 ml
- Young galangal, thinly sliced 3 tbsp.
- Shiitake mushrooms (soaked in water until soft) 100 g
- 4 tablespoons soy sauce
- 1 tablespoon sugar
- 2 teaspoons ground pepper
- Grouper fillet (cut into pieces, boiled) 10 pieces.
- 300 g cooked jasmine rice.
- Finely chopped galangal
- Braised leeks
- Chopped celery

### Method

1. Boil pork bone broth with young galangal until boiling, add shiitake mushrooms, season with light soy sauce, sugar and ground pepper, reduce heat and simmer until boiling broth. Set aside.
2. Heat the water in another pot until it boils. Add jasmine rice to cook in boiling water until soft, scoop into a bowl, and place blanched grouper. Put the pork bone broth, sprinkle finely pounded galangal, fried garlic, and celery. Eat while hot with soybean sauce.

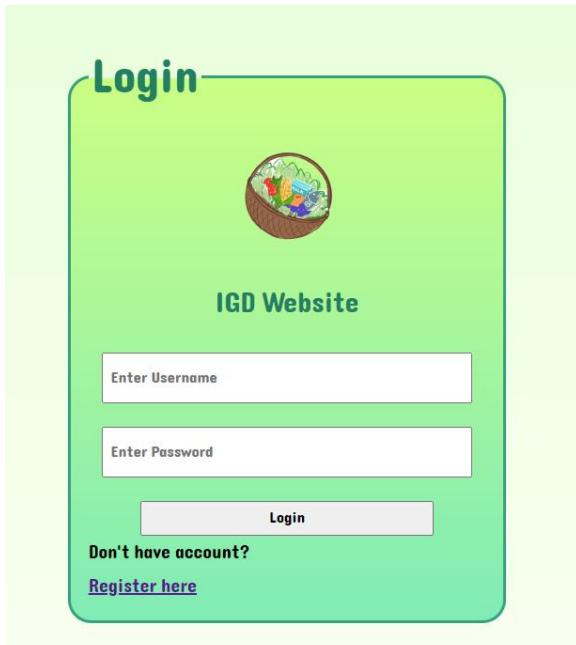
## - Search

Search section already include in the product page. In fact, this part call function SearchFood() from search.js

```
<form id="search-form">
  <section class="searchpart" id="search">
    <h1 class="foodselection1" style="font-size: 40px; color: #ffc92e;">
      | Explore more food!
    </h1>
    <div class="searchbar">
      <input class="input1" type="search" id="search-input" placeholder="Search" />
      <button type="submit" onclick="searchFood()">Search</button>
    </div>
    <div class="custom-select">
      <select id="type">
        <option class="dropdown" value="0">Type</option>
        <option class="dropdown" value="main dish">Main Dish</option>
        <option class="dropdown" value="appetizer">Appetizer</option>
        <option class="dropdown" value="dessert">Dessert</option>
        <option class="dropdown" value="beverage">Beverage</option>
      </select>
      <select id="nutrition">
        <option class="dropdown" value="0">Nutrition</option>
        <option class="dropdown" value="protein">Protein</option>
        <option class="dropdown" value="fat">Fat</option>
        <option class="dropdown" value="fruit and vegetables">Fruit and vegetables</option>
        <option class="dropdown" value="starch">Starch</option>
        <option class="dropdown" value="dairy">Dairy</option>
      </select>
      <select id="factor">
        <option class="dropdown" value="0">Other Factor</option>
        <option class="dropdown" value="seafood allergic">Seafood allergic</option>
        <option class="dropdown" value="low calories">Low Calories</option>
        <option class="dropdown" value="heart disease">Heart disease</option>
        <option class="dropdown" value="diabetes">Diabetes</option>
        <option class="dropdown" value="obesity">Obesity</option>
      </select>
    </div>
  </section>
</form>
```

## - Login

There is a form for user to input username and password to access to website.



```
<form action="/login" method="post">
  <fieldset class="fieldset1">
    <legend style="color: #228160">Login</legend>
    
    <h2>IGD Website</h2>
    <input class="input1" type="text" placeholder="Enter Username" name="username" required>
    <input class="input1" type="password" placeholder="Enter Password" name="password" required>
    <button type="login">Login</button>
    Don't have account?
    <a href="Register.html">Register here</a>
  </fieldset>
</form>
```

## - Register

**Register**



**IGD's Member**

**Write down your information here**

First Name:  Last Name:  Nickname:

Email:  Tel:

Username:  Password:

```
<form action="/register" method="post">
<fieldset class="fieldset2">
<legend>Register</legend>
<div class="titleereg">

<h1 style="color: #fff92e;">IGD's MemberWrite down your information here

```

## - User Account management

**Account Management**



**IGD's Administrator**

Search

ID	Username	Password	Role	Modification
6488038	Kanyaveeza	2468102	Web developer	<a href="#">edit</a>
6488066	AradaAbsent	6600660066	Web designer	<a href="#">edit</a>
6488067	TanawatNamTaaKabao	666999666	Web developer	<a href="#">edit</a>
6488088	IttikornSukPaiLoeu	8800880088	Web developer	<a href="#">edit</a>
6488124	SoratraYipPree	2002Euib!	Web developer	<a href="#">edit</a>
6666666	KirinAha	11100111	Mascot	<a href="#">edit</a>

```
form>
<div class="titleacc">

<h1 style="color: #32B259;">IGD's AdministratorUpdate</button>
</div>
<table>
<thead>
<tr>
<th>ID</th>
<th>Username</th>
<th>Password</th>
<th>Role</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>6488038</td>
<td>Kanyaveeza</td>
<td>2468102</td>
<td>Web developer</td>
<td>edit</td>
</tr>
<tr>
<td>6488066</td>
<td>AradaAbsent</td>
<td>6600660066</td>
<td>Web designer</td>
<td>edit</td>
</tr>
<tr>
<td>6488067</td>
<td>TanawatNamTaaKabao</td>
<td>666999666</td>
<td>Web developer</td>
<td>edit</td>
</tr>
<tr>
<td>6488088</td>
<td>IttikornSukPaiLoeu</td>
<td>8800880088</td>
<td>Web developer</td>
<td>edit</td>
</tr>
<tr>
<td>6488124</td>
<td>SoratraYipPree</td>
<td>2002Euib!</td>
<td>Web developer</td>
<td>edit</td>
</tr>
<tr>
<td>6666666</td>
<td>KirinAha</td>
<td>11100111</td>
<td>Mascot</td>
<td>edit</td>
</tr>


```

## - Product Management

**Product Management**



**IGD's Administrator**

Search  Search

Add  Delete  Clear  Update

ID	Name	Type	Nutrition	Factor	Modification
001	Grilled sea bass with miso sauce	Main dish	Protein	None	<a href="#">edit</a>
002	Clear soup with omlete	Appertizer	Starch	Low calories	<a href="#">edit</a>
003	Spicy mixed vegetable soup	Main dish	Fruit and vegetable	Low calories	<a href="#">edit</a>
004	Boiled rice with grouper	Appertizer	Protein	Low calories	<a href="#">edit</a>
005	Spicy chicken soup with mushroom	Appertizer	Protein	Low calories	<a href="#">edit</a>

```

<form action="" method="post">
<fieldset class="fieldset3" style="background-color:#FAFFED;">
<legend>Product Management</legend>
<br>
<div class="titleprod">

<h1 class="Foodtopic" style="color: #328259;">"IGD's Administrator"
</div>
<div class="searchbar">
<input class="input1" style="height: 30px;" type="search" id="search" placeholder="Search">
<button style="border: radius 20px; width: 100px;" type="submit" id="search"><input type="button" value="Search" /><a href="#" style="text-decoration: none; color: black;">Add</a></button>
<button class="manage two" type="clear"><a href="#" style="text-decoration: none; color: black;">Delete</a></button>
<button class="manage two" type="reset"><a href="#" style="text-decoration: none; color: black;">Clear</a></button>
<button id="Fetch-data-button" class="manage two" type="button" style="text-decoration: none; color: black;">Fetch Data</button>
<button class="manage two" type="submit"><a href="#" style="text-decoration: none; color: black;">Update</a></button>
</div>
<table>
<tr>
<th>ID</th>
<th>Name</th>
<th>Type</th>
<th>Nutrition</th>
<th>Factor</th>
<th>Modification</th>
</tr>
<tbody>
<tr>
<td>001</td>
<td>Grilled sea bass with miso sauce</td>
<td>Main dish</td>
<td>Protein</td>
<td>None</td>
<td><a href="#">edit</a></td>
</tr>
</tbody>
</table>

```

## - About Us

```

<article class="containerabout">
<div class="boxabout">

<div class="description">
<h2 style="margin: 1px; padding-bottom: 0px;">Kanyavee Likitwattanakij</h2>
<div class="memberdesc">
<h4 style="font-style: italic;">Position: Web Developer,Web Designer</h4>
<a class="aboutnav" href="/6488038aboutus">
<h4 style="font-style: italic;">About this member</h4>
</a>
</div>
</div>
</div>
<div class="boxabout">

<div class="description">
<h2 style="margin: 1px; padding-bottom: 0px;">Arada Raksapatcharawong</h2>
<div class="memberdesc">
<h4 style="font-style: italic;">Position:Web Designer</h4>
<a class="aboutnav" href="/6488066aboutus">
<h4 style="font-style: italic;">About this member</h4>
</a>
</div>
</div>
</div>
<div class="boxabout">

<div class="description">
<h2 style="margin: 1px; padding-bottom: 0px; font-size: 22px;">Tanawat Raweebacharathon</h2>
<div class="memberdesc">
<h4 style="font-style: italic;">Position:Web Developer</h4>
<a class="aboutnav" href="/6488067aboutus">
<h4 style="font-style: italic;">About this member</h4>
</a>
</div>
</div>
</div>
</article>

```

```

<div class="fade-in-image">
  <article class="containerabout">
    <div class="boxabout">
      
      <div class="description">
        <h2 style="margin: 1px;padding-bottom: 0px;">Ittikorn Suksai</h2>
        <div class="memberdesc">
          <h4 style="font-style: italic;">Position:Web Developer</h4>
          <a class="aboutnav" href="/6488088aboutus">
            <h4 style="font-style: italic;">About this member</h4>
          </a>
        </div>
      </div>
    </div>
    <div class="boxabout">
      
      <div class="description">
        <h2 style="margin: 1px;padding-bottom: 0px;">Sarttra Prasongtichol</h2>
        <div class="memberdesc">
          <h4 style="font-style: italic;">Position:Web Developer,Web Designer</h4>
          <a class="aboutnav" href="/6488124aboutus">
            <h4 style="font-style: italic;">About this member</h4>
          </a>
        </div>
      </div>
    </div>
  </article>

```

**-Member-**



**Kanyavee Likitwattanakij**

Position: Web Developer, Web Designer

[About this member](#)



**Arada Raksapatcharawong**

Position: Web Designer

[About this member](#)



**Tanawat Rawepachwarathon**

Position: Web Developer

[About this member](#)



**Ittikorn Suksai**

Position: Web Developer

[About this member](#)



**Sarttra Prasongtichol**

Position: Web Developer, Web Designer

[About this member](#)

## - Individual About Us

Everyone has different template and css of their individual page.

```
<body>
<header>
    <h2 style="color: #a1cbe8;"> Blue's Website </h2>
    <nav class="head" id="navbar">
        <ul>
            <li><a href="/" class="midfont headnav">Home</a></li>
            <li><a href="/product" class="midfont headnav">Product</a></li>
            <li><a href="/search" class="midfont headnav">Search</a></li>
            <li><a href="/login" class="midfont headnav">Login</a></li>
            <li><a href="/register" class="midfont headnav">Register</a></li>
        </ul>
    </nav>
</header>
<p style="font-size: 36px; text-align: center;">◊Welcome to my page◊</p>

<h2 style="color: #a1cbe8;">Description</h2>
<p>Hello, My name is Blue , I am Sophomore student at Faculty of Information and Communication Technology. I take a responsibility on Web Designer and Web Developer.</p>
<p style="font-style: italic;">◊This is my Page about me feel free to joy◊</p>
<section>
    <h2>Santtra Prasongthiol</h2>
    <h2>Position: Web Designer,Web Developer</h2>
    <br>
    <ul>
        <li>Address:87/180 Burasiri Village Ratchapruk ,Pak Kret,Nonthaburi,11120</li>
        <li>santtra.pra@student.mahidol.ac.th</li>
    </ul>
</section>
<div class="section1">
    <h2>Education Background</h2>
    <ul>
        <li>Faculty of Information and Communication Technology,Mahidol University, Thailand
            <br>
            Bachelor of Science(Information Technology); GPA 2.73/4.00; August 2021 - Present
        </li>
        <li>Saint Gabriel's College, Thailand(Primary 1 to Secondary 6)
            <br>
            Major: Math-Science; GPA 3.32/4.00; May 2018-February 2020
        </li>
    </ul>
</div>
</div>
```

```
<body>
<header>
    <h2 style="color: #a1cbe8;"> Blue's Website </h2>
    <nav class="head" id="navbar">
        <ul>
            <li><a href="/" class="midfont headnav">Home</a></li>
            <li><a href="/product" class="midfont headnav">Product</a></li>
            <li><a href="/search" class="midfont headnav">Search</a></li>
            <li><a href="/login" class="midfont headnav">Login</a></li>
            <li><a href="/register" class="midfont headnav">Register</a></li>
        </ul>
    </nav>
</header>
<p style="font-size: 36px; text-align: center;">◊Welcome to my page◊</p>

<h2 style="color: #a1cbe8;">Description</h2>
<p>Hello, My name is Blue , I am Sophomore student at Faculty of Information and Communication Technology. I take a responsibility on Web Designer and Web Developer.</p>
<p style="font-style: italic;">◊This is my Page about me feel free to joy◊</p>
<section>
    <h2>Santtra Prasongthiol</h2>
    <h2>Position: Web Designer,Web Developer</h2>
    <br>
    <ul>
        <li>Address:87/180 Burasiri Village Ratchapruk ,Pak Kret,Nonthaburi,11120</li>
        <li>santtra.pra@student.mahidol.ac.th</li>
    </ul>
</section>
<div class="section1">
    <h2>Education Background</h2>
    <ul>
        <li>Faculty of Information and Communication Technology,Mahidol University, Thailand
            <br>
            Bachelor of Science(Information Technology); GPA 2.73/4.00; August 2021 - Present
        </li>
        <li>Saint Gabriel's College, Thailand(Primary 1 to Secondary 6)
            <br>
            Major: Math-Science; GPA 3.32/4.00; May 2018-February 2020
        </li>
    </ul>
</div>
</div>
```

| Home | Product | Search | Login | Register |

**AR's Website**

**-Welcome to my page-**



**Arada Raksapatcharawong**  
**Position: Web Designer**  
Address:15/222 Naithiwon Kancharaphisek Bangkok Bangkok 10160  
Email:arada\_rak@student.mahidol.ac.th

| Home | Product | Search | Login | Register |

**Blue's Website**

**◊Welcome to my page◊**



**Description**

Hello, My name is Blue , I am Sophomore student at Faculty of Information and Communication Technology. I take a responsibility on Web Designer and Web Developer.

*This is my Page about me feel free to joy~*

#### 4. Details of your web services and code.

- **Front-end**

Front-end server will be run on localhost:3000. The javascript file for front-end server consists of:

- o Index.js

```
const express = require('express');
const path = require('path');

const app = express();
const port = 3000;
const session = require('express-session');

// Initialize the session middleware
app.use(session({
  secret: 'my-secret-key',
  resave: false,
  saveUninitialized: false
}));

// Serve static files from the 'public' folder
app.use(express.static(path.join(__dirname, '.', '.', 'Phase 1 Web')));
app.use(express.urlencoded({ extended: false }));

app.get('/register', (req, res) => {
  console.log('Request at /register')
  res.sendFile(path.join(__dirname, '.', '.', 'Phase 1 Web', 'Register.html'));
});

app.get('/aboutus', (req, res) => {
  console.log('Request at /aboutus')
  res.sendFile(path.join(__dirname, '.', '.', 'Phase 1 Web', 'AboutUs.html'));
});

app.get('/accountmanage', (req, res) => {
  console.log('Request at /accountmanage')
  res.sendFile(path.join(__dirname, '.', '.', 'Phase 1 Web', 'Account.html'));
});

app.get('/productmanage', (req, res) => {
  console.log('Request at /productmanage')
  res.sendFile(path.join(__dirname, '.', '.', 'Phase 1 Web', 'ProductManage.html'));
});

app.get('/6488038aboutus', (req, res) => {
  console.log('Request at /6488038aboutus')
  res.sendFile(path.join(__dirname, '.', '.', 'Phase 1 Web', '6488038_aboutus.html'));
});

app.get('/6488066aboutus', (req, res) => {
  console.log('Request at /6488066aboutus')
  res.sendFile(path.join(__dirname, '.', '.', 'Phase 1 Web', '6488066_aboutus.html'));
});

app.get('/6488067aboutus', (req, res) => {
  console.log('Request at /6488067aboutus')
  res.sendFile(path.join(__dirname, '.', '.', 'Phase 1 Web', '6488067_aboutus.html'));
});

app.get('/6488088aboutus', (req, res) => {
  console.log('Request at /6488088aboutus')
  res.sendFile(path.join(__dirname, '.', '.', 'Phase 1 Web', '6488088_aboutus.html'));
});

app.get('/6488124aboutus', (req, res) => {
  console.log('Request at /6488124aboutus')
  res.sendFile(path.join(__dirname, '.', '.', 'Phase 1 Web', '6488124_aboutus.html'));
});

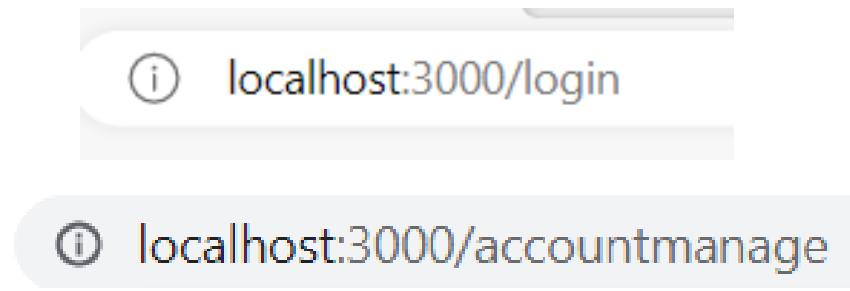
app.get('/', (req, res) => {
  console.log('Request at /')
  res.sendFile(path.join(__dirname, '.', '.', 'Phase 1 Web', 'Home.html'));
});

app.get('/product', (req, res) => {
  console.log('Request at /product')
  res.sendFile(path.join(__dirname, '.', '.', 'Phase 1 Web', 'Product.html'));
});

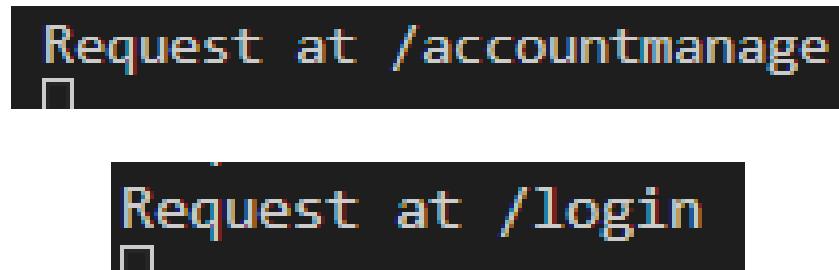
app.get('/search', (req, res) => {
  console.log('Request at /search')
  res.redirect('/Product.html#search');
});

app.get('/login', (req, res) => {
  console.log('Request at /login')
  res.sendFile(path.join(__dirname, '.', '.', 'Phase 1 Web', 'Login.html'));
});
app.post('/login', (req, res) => {
  res.redirect('/login');
});
```

If there are request to the page from the user, the address bar will show link like this.



Also, with the terminal since there are console.log command to indicate the status of web service.



## - Back-end

Front-end server will be run on localhost:3030. The javascript file for front-end server consists of:

- index.js

index.js played as a significant javascript file as it the center server that connect all the javascript file together when start node.js.

```
1  require('dotenv').config();
2  const express = require('express');
3  const path = require('path');
4  const db = require('./database');
5  const navbar = require('../navbar');
6  const auth = require('../auth');
7  const adminservice = require('../admin');
8  const productservice = require('../prod');
9  const app = express()
10 // Import the express-session module
11 const session = require('express-session');
12
13 // Initialize the session middleware
14 app.use(session({
15   secret: 'my-secret-key',
16   resave: false,
17   saveUninitialized: false
18 }));
19
20 // Serve static files from the 'public' folder
21 app.use(express.static(path.join(__dirname, '.', '..', 'Phase 1 Web')));
22 app.use(express.urlencoded({ extended: false }));
23 app.use(express.json());
24
25 //MAKE ID FOR NAV BAR
26 app.use('/', navbar);
27
28 app.use(auth);
29 app.use(adminservice);
30 app.use(productservice);
31 app.listen(process.env.PORT, () => {
32   console.log(`Server is listening on port ${process.env.PORT}`);
33 });
34
```

- database.js

this database.js use to set the connection to sec2\_gr2\_database.sql to display the information from table to the system

sec2\_gr2\_database.sql table:

```
create table if not exists AdminInformation(
    AdminID      char(7) primary key,
    AdminFname   varchar(50) not null,
    AdminLname   varchar(50) not null,
    AdminInName  varchar(25) not null,
    AdminEmail   varchar(25) not null,
    AdminPhone   varchar(12) not null
);

create table if not exists AdminLogin(
    ID_Admin     char(7),
    AdminUser    varchar(50) primary key,
    AdminPass    varchar(50) not null,
    AdminRole    varchar(25),
    LoginLog     timestamp,
    constraint FK_AdminInLog   foreign key(ID_Admin) references AdminInformation(AdminID));

create table if not exists ProductInfomation(
    ProdID      char(3) primary key,
    ProdName    varchar(50) not null,
    ProdType    varchar(25),
    ProdNeutri  varchar(25),
    ProdFactor  varchar(25),
    ProdDesc    text,
    ProdRecipe  Text,
```

this is the code in database.js.

```
require('dotenv').config()
const sql = require('mysql2')
const db = sql.createPool({
  host: process.env.MYSQL_HOST,
  user: process.env.MYSQL_USERNAME,
  password: process.env.MYSQL_PASSWORD,
  database: process.env.MYSQL_DATABASE
});

//Connect to the DB
db.getConnection((err, connection) => {
  if (err) {
    console.error('Error getting connection from pool:', err);
    return;
  }
  connection.release();
});

//DB on connect
if (db) {
  db.on('connection', (connection) => {
    console.log(`Connection established on connection ${connection.config.database}`);
  });
} else {
  console.log('Error');
}
module.exports = db;
```

- .env

This environment variable (.env) use to determines mysql access condition include port ,username , password, and schema.

```
const db = sql.createPool({
  host: process.env.MYSQL_HOST,
  user: process.env.MYSQL_USERNAME,
  password: process.env.MYSQL_PASSWORD,
  database: process.env.MYSQL_DATABASE
});
```

```
1 # don't forget to add the schema privileges to your username in sec2_gr2_database schema
2 PORT=3030
3 MYSQL_HOST=localhost
4 MYSQL_USERNAME= tinycollegeUser # use your own username
5 MYSQL_PASSWORD= 2002Eulb! #use your own password
6 MYSQL_DATABASE= sec2_gr2_database
```

- navbar.js

the same as mentioned in the index.js front-end section but it will change from app.get to route.get since this is the secondary javascript file so it need to export route by module.export = route

```
const express = require('express');
const path = require('path');
const route = express.Router();

route.get('/', (req, res) => {
  console.log('Request at /')
  res.sendFile(path.join(__dirname, '..', '..', 'Phase 1 Web', 'Home.html'));
});
```

```
module.exports = route;
```

Then we import this javascript file to index.js and call app.use function

```
const navbar = require('./navbar');

//MAKE ID FOR NAV BAR
app.use('/', navbar);
```

- auth.js

this part is the authentication with getting information from user and administrators to access the web services. If the data is in the database, the login will be succeed and sent user to homepage, but if not the server will sent user back to login form again

```
const express = require('express');
const db = require('./database');

const router = express.Router();

router.post('/login', (req, res) => {
  const { username, password } = req.body;
  db.query(
    'SELECT * FROM AdminLogin WHERE AdminUser = ? AND AdminPass = ?',
    [username, password],
    (error, results) => {
      if (error) {
        res.json({ success: false, error: error.message });
      } else if (results.length > 0) {
        // login successful, create session and return success response
        req.session.authenticated = true;
        res.redirect('/');
      } else {
        // login failed, return error response
        res.redirect('/login');
      }
    });
});
```

- If login is true,

**Login**



IGD Website

[Don't have account?](#)

[Register here](#)



- If login is false,

**Login**



IGD Website

[Don't have account?](#)

[Register here](#)

**Login**



IGD Website

[Don't have account?](#)

[Register here](#)

For the register part, the website provides the form for user to input their data and submit it to mysql database to store the information in the system, after that they can use login function.

## Register



**IGD's Member**

*Write down your information here*

First Name:  Last Name:  Nickname:

Email:  Tel:

Username:  Password:

## Login



**IGD Website**

**Don't have account?**

[\*\*Register here\*\*](#)

- o **admin.js**

admin.js is the function for administrator to modify their information in the system consist of:

no criteria search: type localhost:3030/reg in address bar to display all admin data in the system.

```
router.get('/reg', async (req, res) => {
  try {
    const [UserData] = await db.promise().execute(
      `SELECT * FROM AdminInformation`,
    );
    const [UserData2] = await db.promise().execute(
      `SELECT * FROM AdminLogin`,
    );
    const data = {
      adminInformation: UserData,
      adminLogin: UserData2
    };
    console.log(data); // Log the merged data to the console
    return res.status(200).json({ error: false, data: data, message: "User successfully retrieved" });
  } catch (error) {
    console.error(error);
    return res.status(500).json({ error: true, message: 'Error retrieving user data' });
  }
});
```

```
{"error":false,"data":[{"adminInformation":{"AdminID":6488038,"AdminName":"Kanyavee","AdminName2":"Kanyavee","AdminEmail":"Babe","AdminPhone":"066-468-2969"}, {"AdminID":6488066,"AdminName":"Arada","AdminName2":"Raksapatcharawong","AdminEmail":"Arada@gmail.com","AdminPhone":"090-000-0000"}, {"AdminID":6488067,"AdminName":"Tanawat","AdminName2":"Raweebacharathon","AdminEmail":"Pluem","AdminPhone":"092-625-2924"}, {"AdminID":6488088,"AdminName":"Ittikorn","AdminName2":"Suksa1","AdminEmail":"Tanawat@gmail.com","AdminPhone":"098-584-5035"}, {"AdminID":6488124,"AdminName":"Sarttra","AdminName2":"Parsongtichol","AdminEmail":"Blue","AdminPhone":"088-673-5514"}, {"AdminID":6488125,"AdminName":"Kirin","AdminName2":"Elou","AdminEmail":"Kirin240@gmail.com","AdminPhone":"089-555-1111"}, {"AdminID":6488146,"AdminName":"Panang","AdminName2":"Prasongsintha","AdminEmail":"Sho","AdminPhone":"0618632002"}, {"ID_Admin":6488066,"AdminUser":"AradaAbsent","AdminPass":"6600650066","AdminRole":"Web designer","AdminEmail": "GpXAGYv", "AdminUser": "Inosuke", "AdminPass": "420698", "AdminRole": "Web designer", "LoginLog": "2023-04-25T05:43:16.000Z"}, {"ID_Admin":6488088,"AdminUser": "IttikornSuKaiLaeu", "AdminPass": "8800880088", "AdminRole": "Web developer", "LoginLog": null}, {"ID_Admin":6488038,"AdminUser": "SarttraIpFree", "AdminPass": "2002Eub1", "AdminRole": "Web developer", "LoginLog": null}, {"ID_Admin":6488124,"AdminUser": "SarttraYipFree", "AdminPass": "2002Eub1", "AdminRole": "Web developer", "LoginLog": null}, {"ID_Admin":6488125,"AdminUser": "KirinNaNa", "AdminPass": "1100111", "AdminRole": "Mascot", "LoginLog": null}, {"ID_Admin":6488067,"AdminUser": "TanawatNaTaakBao", "AdminPass": "666999666", "AdminRole": "Web developer", "LoginLog": null}], "message": "User successfully retrieved")
```

Criteria search: Search by ID type localhost:3030/reg /:id in address bar to display the admin data that match input ID in the system.

```
router.get('/reg/:id', async (req, res) => {
  try {
    const { id } = req.params; // Get the admin ID from the request parameters
    const [UserData] = await db.promise().execute(
      `SELECT * FROM AdminInformation WHERE AdminID = ?`, [id] // Use the admin ID in the SQL query
    );
    const [UserData2] = await db.promise().execute(
      `SELECT * FROM AdminLogin WHERE ID_Admin = ?`, [id] // Use the admin ID in the SQL query
    );
    if (UserData && UserData2) {
      const data = {
        adminInformation: UserData[0], // The result of the SQL query is an array, so we need to access the first element
        adminLogin: UserData2[0]
      };
      console.log(data); // Log the merged data to the console
      return res.status(200).json({ error: false, data: data, message: "User successfully retrieved" });
    } else {
      return res.status(404).json({ error: true, message: "Admin not found" });
    }
  } catch (error) {
    console.error(error);
    return res.status(500).json({ error: true, message: 'Error retrieving admin data' });
  }
});
```

```
{"error":false,"data":{"adminInformation":{"AdminID":6488067,"AdminName":"Tanawat","AdminName2":"Raweebacharathon","AdminEmail":"Pluem","AdminPhone":"092-625-2924"}, "adminLogin": {"ID_Admin":6488067,"AdminUser": "TanawatNaTaakBao", "AdminPass": "666999666", "AdminRole": "Web developer", "LoginLog": null}}, "message": "User successfully retrieved")
```

Criteria search: Search by adminrole: type localhost:3030/reg /adminrole/:adminrole in address bar to display the admin data that match input admin role in the system.

```
router.get('/reg/adminrole/:AdminRole', async (req, res) => {
  try {
    const { AdminRole } = req.params;
    const [result] = await db.promise().execute(
      `SELECT AdminInformation.AdminID, AdminInformation.AdminName, AdminInformation.AdminLname, AdminInformation.AdminName, AdminInformation.AdminEmail, AdminInformation
      FROM AdminInformation
      JOIN AdminLogin ON AdminInformation.AdminID = AdminLogin.ID_Admin
      WHERE AdminLogin.AdminRole = ?`,
      [AdminRole]
    );
    const adminInformation = result.map((row) => ({
      AdminID: row.AdminID,
      AdminName: row.AdminName,
      AdminLname: row.AdminLname,
      AdminName: row.AdminName,
      AdminEmail: row.AdminEmail,
      AdminPhone: row.AdminPhone,
    }));
    const adminLogin = result.map((row) => ({
      ID_Admin: row.AdminID,
      AdminUser: row.AdminUser,
      AdminPass: row.AdminPass,
      AdminRole: row.AdminRole,
      LoginLog: row.LoginLog,
    }));
    return res.status(200).json({
      error: false,
      data: {
        adminInformation,
        adminLogin,
      },
      message: "Admin data successfully retrieved",
    });
  } catch (error) {
    console.error(error);
    return res.status(500).json({ error: true, message: 'Error retrieving admin data' });
  }
});
```

```
{"error":false,"data":[{"adminInformation":[{"AdminID":"6488008","AdminName":"Tittikorn","AdminLname":"Suksai","AdminRole":"Web","AdminEmail":"Tittikorn@gmail.com","AdminPhone":"098-584-5035"}, {"AdminID":"6488038","AdminName":"Kanyavee","AdminLname":"Likitwattanakij","AdminRole":"Web developer","AdminEmail":"Kanyavee@gmail.com","AdminPhone":"+66-468-2969"}, {"AdminID":"6488104","AdminName":"Shaman","AdminLname":"Koapan","AdminEmail":"shamking@gmail.com","AdminPhone":"+66-468-468-2969"}, {"AdminID":"6488124","AdminName":"Moscow","AdminLname":"Kaopan","AdminEmail":"shamking@gmail.com","AdminPhone":"+66-468-468-2969"}, {"AdminID":"6488124","AdminName":"Parsonsgetchol","AdminEmail":"Blue","AdminPhone":"088-673-5591"}, {"AdminID":"6488067","AdminName":"Tanawat","AdminLname":"Raweepaccharathon","AdminEmail":"Plum","AdminEmail":"tanawat@gmail.com","AdminPhone":"+092-625-2924"}],"adminLogin":[{"ID_Admin":6488008,"AdminUser":"TittikornSupPailaeu","AdminPass":"88000800088","AdminRole":"Web developer","LoginLog":null}, {"ID_Admin":6488038,"AdminUser":"Kanyavee","AdminPass":"2002eulb!","AdminRole":"Web developer","LoginLog":null}, {"ID_Admin":6488124,"AdminUser":"SartrraYipPreed","AdminPass":"2002eulb!","AdminRole":"Web developer","LoginLog":null}, {"ID_Admin":6488067,"AdminUser":"TanauhtNamTaakBao","AdminPass":"666999666","AdminRole":"Web developer","LoginLog":null}], "message": "Admin data successfully retrieved"}
```

Criteria search: Search by adminuser: type localhost:3030/reg /adminuser/: adminuser in address bar to display the admin data that match input admin role in the system.

```
router.get('/reg/adminuser/:adminuser', async (req, res) => {
  try {
    const { adminuser } = req.params;
    const [result] = await db.promise().execute(
      `SELECT AdminInformation.AdminID, AdminInformation.AdminName, AdminInformation.AdminLname, AdminInformation.AdminEmail, AdminInformation
      FROM AdminInformation
      JOIN AdminLogin ON AdminInformation.AdminID = AdminLogin.ID_Admin
      WHERE AdminLogin.AdminUser = ?`,
      [adminuser]
    );
    const adminInformation = result.map((row) => ({
      AdminID: row.AdminID,
      AdminName: row.AdminName,
      AdminLname: row.AdminLname,
      AdminEmail: row.AdminEmail,
      AdminPhone: row.AdminPhone,
    }));
    const adminLogin = result.map((row) => ({
      ID_Admin: row.AdminID,
      AdminUser: row.AdminUser,
      AdminPass: row.AdminPass,
      AdminRole: row.AdminRole,
      LoginLog: row.LoginLog,
    }));
    return res.status(200).json({
      error: false,
      data: {
        adminInformation,
        adminLogin,
      },
      message: "Admin data successfully retrieved",
    });
  } catch (error) {
    console.error(error);
    return res.status(500).json({ error: true, message: 'Error retrieving admin data' });
  }
});
```

```
{"error":false,"data":{"adminInformation":[{"AdminID":"6488038","AdminName":"Kanyavee","AdminLname":"Likitwattanakij","AdminRole":"Web developer","AdminEmail":"Kanyavee@gmail.com","AdminPhone":"+66-468-2969"}],"adminLogin":[{"ID_Admin":6488038,"AdminUser":"Kanyavee","AdminPass":"2468102","AdminRole":"Web developer","LoginLog":null}], "message": "Admin data successfully retrieved"}
```

Insert data: Insert data can proceed by call POST method on localhost:3030/reg in postman application to insert the admin data which required attributes that matched the table in mysql to the system.

```
router.post('/reg', async (req, res) => {
  try {
    // Destructure the required fields from the request body
    const { AdminID, AdminFname, AdminLname, AdminNname, AdminEmail, AdminPhone, AdminUser, AdminPass, AdminRole } = req.body;

    // Insert a new row into the AdminInformation table
    const [AdminInformation] = await db.promise().execute(
      `INSERT INTO AdminInformation (AdminID,AdminFname, AdminLname, AdminNname, AdminEmail, AdminPhone) VALUES (?, ?, ?, ?, ?, ?)`,
      [AdminID, AdminFname, AdminLname, AdminNname, AdminEmail, AdminPhone]
    );

    // Insert a new row into the AdminLogin table, using the AdminInformation ID as a foreign key
    const [AdminLogin] = await db.promise().execute(
      `INSERT INTO AdminLogin (ID_Admin, AdminUser, AdminPass, AdminRole) VALUES (?, ?, ?, ?)`,
      [AdminID, AdminUser, AdminPass, AdminRole]
    );

    // Return a success response to the client
    return res.status(200).json({ error: false, message: 'Admin registration successful' });
  } catch (error) {
    console.error(error);
    return res.status(500).json({ error: true, message: 'Error registering admin' });
  }
});
```

```
// router.post /reg : this will add a new administrator data to the system
//Testing: Insert a new administrator
//method: POST
//URL: localhost:3030/reg
//body: raw JSON
//Test case here:
//1st Test case:
//{
//  "AdminID": "328569",
//  "AdminFname": "James",
//  "AdminLname": "Sunderland",
//  "AdminNname": "SH2",
//  "AdminEmail": "silenthill@hotmail.com",
//  "AdminPhone": "085-669-8545",
//  "AdminUser": "JameZAP",
//  "AdminPass": "nopassword",
//  "AdminRole": "Mascot"
//}
```

update data: Insert data can proceed by call PUT method on localhost:3030/reg/:id in postman application to insert the admin data which required attributes that matched the table as well as matched the id parameter in MySQL to the system.

```
router.put('/reg/:id', async (req, res) => {
  const { id } = req.params; // Get the admin ID from the request parameters

  try {
    // Destructure the required fields from the request body
    const { AdminFname, AdminLname, AdminNname, AdminEmail, AdminPhone, AdminUser, AdminPass, AdminRole } = req.body;

    // Update the row in the AdminInformation table
    await db.promise().query(
      `UPDATE AdminInformation SET AdminFname = ?, AdminLname = ?, AdminNname = ?, AdminEmail = ?, AdminPhone = ? WHERE AdminID = ?`,
      [AdminFname, AdminLname, AdminNname, AdminEmail, AdminPhone, id]
    );

    // Update the row in the AdminLogin table
    await db.promise().query(
      `UPDATE AdminLogin SET AdminUser = ?, AdminPass = ?, AdminRole = ? WHERE ID_Admin = ?`,
      [AdminUser, AdminPass, AdminRole, id]
    );

    // Return a success response to the client
    return res.status(200).json({ error: false, message: 'Admin information updated successfully' });
  } catch (error) {
    console.error(error);
    return res.status(500).json({ error: true, message: 'Error updating admin information' });
  }
});
```

```
//2nd Test Case:
//{
//  "AdminFname": "Chayanisa",
//  "AdminLname": "Chukiatphatnakij",
//  "AdminNname": "Mind",
//  "AdminEmail": "melodymind@gmail.com",
//  "AdminPhone": "061-033-8945",
//  "AdminUser": "Conductress",
//  "AdminPass": "ilovegirl",
//  "AdminRole": "Receptionist"
//}
```

Delete data: Delete data can proceed by call DELETE method on localhost:3030/reg/:id in postman application to delete the admin data which matched the id parameter in mysql on the system.

```
// router.delete /reg/:id : this will delete an administrator data that matched required ID in the system
//Testing: delete an administrator data in required ID
//method: DELETE
//URL: localhost:3030/reg/6488067, localhost:3030/reg/6488038 , or localhost:3030/reg/6666666
router.delete('/reg/:id', (req, res) => {
  const { id } = req.params; // Get the admin ID from the request parameters

  // Delete the admin login data from the AdminLogin table
  db.query(`DELETE FROM AdminLogin WHERE ID_Admin = ?`, [id], (err, result1) => {
    if (err) {
      console.error(err);
      return res.status(500).json({ error: true, message: 'Error deleting admin data' });
    }

    // Delete the admin information from the AdminInformation table
    db.query(`DELETE FROM AdminInformation WHERE AdminID = ?`, [id], (err, result2) => {
      if (err) {
        console.error(err);
        return res.status(500).json({ error: true, message: 'Error deleting admin data' });
      }

      // If both deletes were successful, send a success response
      if (result1.affectedRows > 0 && result2.affectedRows > 0) {
        return res.status(200).json({ error: false, message: 'Admin successfully deleted' });
      } else {
        return res.status(404).json({ error: true, message: 'Admin not found' });
      }
    });
  });
});
```

- o prod.js

prod.js is the function for administrator to modify their product in the system consist of:  
no criteria search: type localhost:3030/prod in address bar to display all products data in the system.

```
// router.get /prod this will return all search result since there is no criteria
//Testing: No criteria search
//method: GET
//URL(use this for test): localhost:3030/prod , http://localhost:3030/prod
router.get('/prod', async (req, res) => {
  try {
    const [ProductData] = await db.promise().execute(
      `SELECT * FROM ProductInfomation`,
    );
    console.log(ProductData); // Log the merged data to the console
    return res.status(200).json({ error: false, data: ProductData, message: "Product successfully retrieved" });
  } catch (error) {
    console.error(error);
    return res.status(500).json({ error: true, message: 'Error retrieving user data' });
  }
});
```

Criteria search: Search by ID type localhost:3030/prod /id in address bar to display the products data that match input ID in the system.

```
// router.get /prod/:id this will return all of product data that have matched input ID
//Testing: Criteria Search (Search By ID) : 1st criteria
//method: GET
//URL(use this for test): localhost:3030/prod/001 , http://localhost:3030/prod/001 , localhost:3030/prod/002 , http://localhost:3030/prod/002
router.get('/prod/:id', async (req, res) => {
  try {
    const { id } = req.params;
    const [ProductData] = await db.promise().execute(
      `SELECT * FROM ProductInfomation WHERE ProdID = ?`, [id]
    );
    console.log(ProductData); // Log the merged data to the console
    return res.status(200).json({ error: false, data: ProductData, message: "Product successfully retrieved" });
  } catch (error) {
    console.error(error);
    return res.status(500).json({ error: true, message: 'Error retrieving user data' });
  }
});
```

"error":false,"data":[{"ProdID": "002","ProdName": "Clear soup with onions", "ProdType": "Appetizer", "ProdNutri": "Starch", "ProdFactor": "+ low calories", "ProdDesc": "It's easy to eat. Suitable for everyone, especially those who want to control their diet.", "ProdScript": "2-3 eggs, onions, spring onions, coriander, seasoning powder, fish sauce, soy sauce, sugar, water", "ProdStep": "1.Slice spring onions and onions into small pieces. 2.Crack an egg into a bowl, add sliced onions, spring onions and a little fish sauce\nv3.Heat the pan, add a little oil when the oil is hot, pour the eggs into the pan and fry until the eggstart to cook on both sides.Use a spatula to cut the eggs into pieces. 4.Fill it with water. Season with sugar,nsoy sauce, seasoning powder, and finally add the remaining springonions.Turn off the stove and sprinkle with coriander leaves."}, {"message": "Product successfully retrieved"}]

Criteria search: Search by nutrition: type localhost:3030/prod/nutrition/: nutrition in address bar to display the product data that match input nutrition in the system.

```
// router.get /prod/nutrition/:nutrition this will return all of product data that have matched nutrition
//Testing: Criteria Search (Search By nutrition) : 2nd criteria
//method: GET
//URL(use this for test): localhost:3030/prod/nutrition/Protein , http://localhost:3030/prod/nutrition/Protein , localhost:3030/prod/nutrition/Starch , http://localhost:3030/prod/nutrition/Starch
router.get('/prod/nutrition/:nutrition', async (req, res) => {
  try {
    const { nutrition } = req.params;
    const [productData] = await db.promise().execute(
      `SELECT * FROM ProductInfomation WHERE ProdNeutri = ?`, [nutrition]
    );
    console.log(productData); // Log the merged data to the console
    return res.status(200).json({ error: false, data: productData, message: "Product successfully retrieved" });
  } catch (error) {
    console.error(error);
    return res.status(500).json({ error: true, message: 'Error retrieving user data' });
  }
});
```

{ "error":false,"data":[{"ProdID":"002","ProdName":"Clear soup with omlete","ProdType":"Appertizer","ProdNeutri":"Starch","ProdFactor":"Low calories","ProdDesc":"It's easy to eat. Suitable for everyone, especially those who want to control their diet","ProdRecipe":"2-3 eggs, onions, spring onions, coriander, seasoning powder, fish sauce, soy sauce, sugar, water","ProdStep":"1.Slice spring onions and onions into small pieces. 2.Crack an egg into a bowl, add sliced onions,spring onions and a little fish sauce\n3.Heat the pan, add a little oil when the oil is hot, pour the eggs into the pan and fry until the eggsstart to cook on both sides.Use a spatula to cut the eggs into pieces. 4.Fill it with water. Season with sugar,\nsoy sauce, seasoning powder, and finally add the remaining springonions.Turn off the stove and sprinkle with coriander leaves."}],"message":"Product successfully retrieved"}

Criteria search: Search by Food Type: type localhost:3030/prod/type/: type in address bar to display the product data that match input food type in the system.

```
// router.get /prod/type/:type this will return all of product data that have matched nutrition
//Testing: Criteria Search (Search By nutrition) : 2nd criteria
//method: GET
//URL(use this for test): localhost:3030/prod/type/Appertizer , http://localhost:3030/prod/type/Appertizer , localhost:3030/prod/type/Main Dish , http://localhost:3030/prod/type/Main Dish
router.get('/prod/type/:type', async (req, res) => {
  try {
    const { type } = req.params;
    const [productData] = await db.promise().execute(
      `SELECT * FROM ProductInfomation WHERE Prodtype = ?`, [type]
    );
    console.log(productData); // Log the merged data to the console
    return res.status(200).json({ error: false, data: productData, message: "Product successfully retrieved" });
  } catch (error) {
    console.error(error);
    return res.status(500).json({ error: true, message: 'Error retrieving user data' });
  }
});
```

{ "error":false,"data":[{"["ProdID"]": "001", "["ProdName"]": "Grilled sea bass with miso sauce", "["ProdType"]": "Main dish", "["ProdNeutri"]": "Protein", "["ProdFactor"]": null, "["ProdDesc"]": "It's a healthy food menu. Help nourish the bones. Can help strengthen and relieve rheumatoid arthritis. Reduces high blood pressure", "["ProdRecipe"]": "Sea bass fillet (cut into pieces) 300 g., Miso 80 g, Mirin 1 tbsp ,1 egg (yolk only), Vegetable oil, grated radish, to Japanese steamed rice", "["ProdStep"]": "1.Pour the miso and mirin into a pan over low heat, stir until the miso has dissolved, add the egg yolks and stir until well combined. Set aside. 2.set fire to grill. Spread a table cloth over the surface. Apply miso sauce (from step 1) over the sea bass. 3.Put the sea bass on the grill and cook until the meat is tender and healthy for everyone.", "["ProdFactor"]": "1+1/2 tsp black pepper, Fingerroot 1-2 stems ,3-4 shallots, Shrimp paste 1 teaspoon, 1/3 cup finely ground dried shrimp, 3 cups chicken broth, Chinese okra, cut into bite-sized pieces 150 g, Pumpkin, cut into bite-sized pieces 200 g, Courgette, cut into bite size pieces 150 g, Baby corn, sliced 150 g, 150 g of champion mushroom, 200 g of fresh shrimp, basil 80 g, fish sauce", "["ProdStep"]": "1.Prepare the curry paste first. 2.Pound the pepper with the krachai, then add the shallots, pound them together until they are well combined. 3.Put the chicken broth in the pot. Bring to a boil and add curry paste. When the soup is boiling, add vegetables that are difficult to cook first. Bring to a boil again and add the rest of the vegetables.\nOnce the water is boiling, add the Chinese okra, pumpkin, courgette and mushrooms. When the soup is boiling again, add fresh shrimp. Season with fish sauce as desired.", "["ProdDesc"]": "When the mixture boils again, add the basil leaves. scoop served hot."}], "message": "Product successfully retrieved"}

Insert data: Insert data can proceed by call POST method on localhost:3030/prod in postman application to insert the product data which required attributes that matched the table in mysql to the system.

```
router.post('/prod', async (req, res) => {
  try {
    const { ProdID, ProdName, ProdType, ProdNeutri, ProdFactor, ProdDesc, ProdStep } = req.body; // Destructure the required fields from the request body

    // Insert a new row into the ProductInfomation table
    const [productData] = await db.promise().execute(
      `INSERT INTO ProductInfomation (ProdID, ProdName, ProdType, ProdNeutri, ProdFactor, ProdDesc, ProdStep) VALUES (?, ?, ?, ?, ?, ?, ?, ?)`,
      [ProdID, ProdName, ProdType, ProdNeutri, ProdFactor, ProdDesc, ProdStep]
    );

    return res.status(200).json({ error: false, message: "Product successfully inserted" });
  } catch (error) {
    console.error(error);
    return res.status(500).json({ error: true, message: 'Error inserting product data' });
  }
});
```

Update data: Insert data can proceed by call PUT method on localhost:3030/prod/:id in postman application to insert the product data which required attributes that matched the table as well as matched the id parameter in mysql to the system.

```
router.put('/prod/:id', async (req, res) => {
  try {
    const { ProdName, ProdType, ProdNeutri, ProdFactor, ProdDesc, ProdRecipe, ProdStep } = req.body; // Destructure the required fields from the request body
    const { id } = req.params; // Get the product ID from the request params

    // Update the row in the ProductInformation table with the given ID
    const [ProductData] = await db.promise().execute(
      `UPDATE ProductInformation SET ProdName = ?, ProdType = ?, ProdNeutri = ?, ProdFactor = ?, ProdDesc = ?, ProdRecipe = ?, ProdStep = ? WHERE ProdID = ?`,
      [ProdName, ProdType, ProdNeutri, ProdFactor, ProdDesc, ProdRecipe, ProdStep, id]
    );
  }

  return res.status(200).json({ error: false, message: "Product successfully updated" });
} catch (error) {
  console.error(error);
  return res.status(500).json({ error: true, message: 'Error updating product data' });
}
});
```

Delete data: Delete data can proceed by call DELETE method on localhost:3030/prod/:id in postman application to delete the product data which matched the id parameter in mysql on the system.

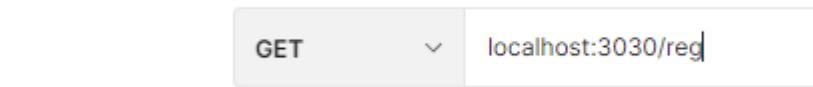
```
// router.delete /prod/:id : this will delete a product data that matched required ID in the system
//Testing: delete a product data in required ID
//method: DELETE
//URL: localhost:3030/prod/001, localhost:3030/prod/005 , or localhost:3030/prod/003
router.delete('/prod/:id', (req, res) => {
  const { id } = req.params;
  db.query(`DELETE FROM ProductInfomation WHERE ProdID = ?`, [id], (err, result1) => {
    if (err) {
      console.error(err);
      return res.status(500).json({ error: true, message: 'Error deleting product data' });
    }

    // If both deletes were successful, send a success response
    if (result1.affectedRows > 0) {
      return res.status(200).json({ error: false, message: "Product successfully deleted" });
    } else {
      return res.status(404).json({ error: true, message: "Product not found" });
    }
  });
});
```

5. The testing results of web services using Postman or any program for testing web service. This is the testing result which running through postman application.

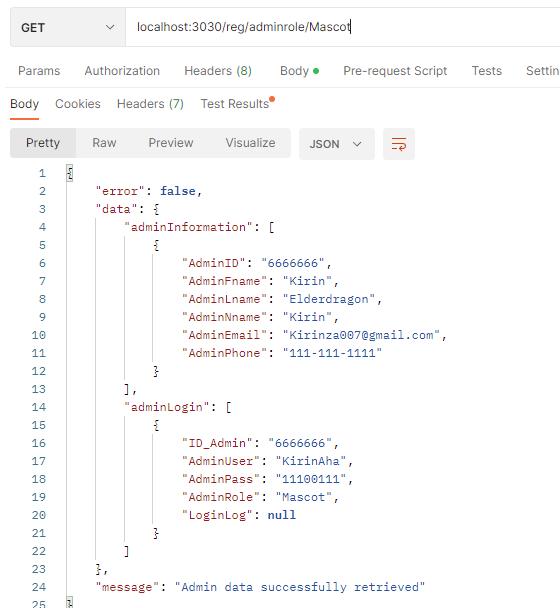
admin.js

no criteria search: GET localhost:3030/reg.

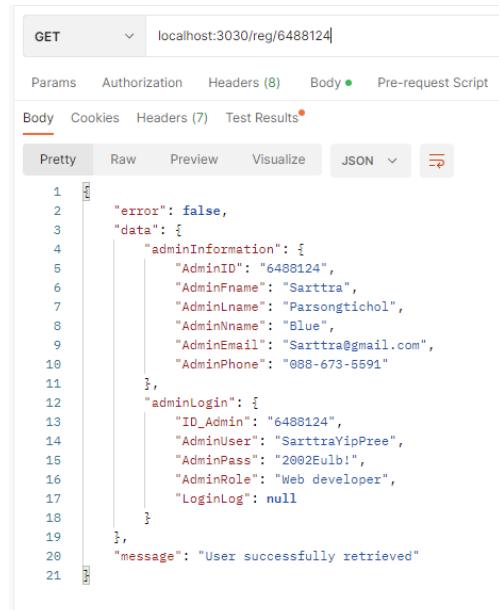


```
1 "error": false,
2 "data": [
3     "adminInformation": [
4         {
5             "AdminID": "6488038",
6             "AdminFname": "Kanyavee",
7             "AdminLname": "Likittawanakij",
8             "AdminName": "Babe",
9             "AdminEmail": "Kanyavee@gmail.com",
10            "AdminPhone": "066-468-2969"
11        },
12        {
13            "AdminID": "6488066",
14            "AdminFname": "Arada",
15            "AdminLname": "Raksapatcharawong",
16            "AdminName": "Aida",
17            "AdminEmail": "Arada@gmail.com",
18            "AdminPhone": "000-000-0000"
19        },
20        {
21            "AdminID": "6488067",
22            "AdminFname": "Tanawat",
23            "AdminLname": "Rawepacharathon",
24            "AdminName": "Pleum",
25            "AdminEmail": "Tanawat@gmail.com",
26            "AdminPhone": "092-625-2924"
27        },
28        {
29            "AdminID": "6488088",
30            "AdminFname": "Ittikorn",
31            "AdminLname": "Suksa",
32            "AdminName": "Nop",
33            "AdminEmail": "Ittikorn@gmail.com",
34            "AdminPhone": "098-584-5035"
35        },
36        {
37            "AdminID": "6488124",
38            "AdminFname": "Sarittra",
39            "AdminLname": "YipPree",
40            "AdminName": "Sarittra",
41            "AdminEmail": "SarittraYipPree@gmail.com",
42            "AdminPhone": "098-202-0011"
43        }
44    ]
45 }
46 }
```

Criteria search: search by Admin role: GET localhost:3030/reg/adminrole/: adminrole

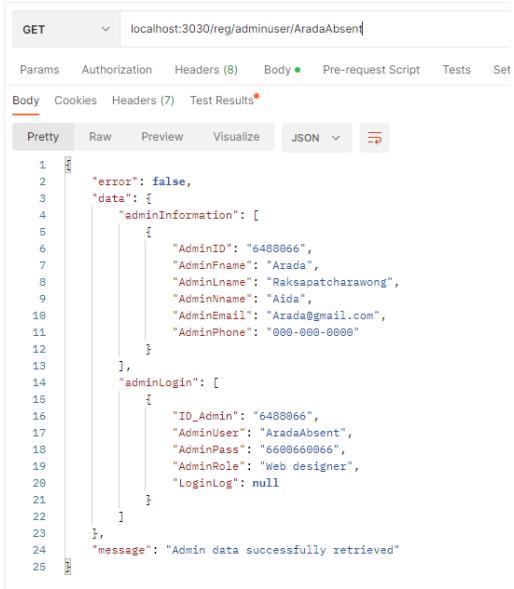


Criteria search: search by ID: GET localhost:3030/reg/:id.



```
1
2   "error": false,
3   "data": {
4     "adminInformation": {
5       "AdminID": "6488124",
6       "AdminFname": "Sarttra",
7       "AdminLname": "Parsongtichol",
8       "AdminName": "Blue",
9       "AdminEmail": "Sarttra@gmail.com",
10      "AdminPhone": "088-673-5591"
11    },
12    "adminLogin": {
13      "ID_Admin": "6488124",
14      "AdminUser": "SarttraYipPre",
15      "AdminPass": "2002Eulb!",
16      "AdminRole": "Web developer",
17      "LoginLog": null
18    }
19  },
20  "message": "User successfully retrieved"
21
```

Criteria search: search by Admin username: GET localhost:3030/reg/adminuser/: adminuser



```
1
2   "error": false,
3   "data": {
4     "adminInformation": [
5       {
6         "AdminID": "648806",
7         "AdminFname": "Arada",
8         "AdminLname": "Rakspatchazawong",
9         "AdminName": "Aida",
10        "AdminEmail": "Arada@gmail.com",
11        "AdminPhone": "000-000-0000"
12      }
13    ],
14    "adminLogin": [
15      {
16        "ID_Admin": "648806",
17        "AdminUser": "AradaAbsent",
18        "AdminPass": "6600660066",
19        "AdminRole": "Web designer",
20        "LoginLog": null
21      }
22    ]
23  },
24  "message": "Admin data successfully retrieved"
25
```

Insert data: POST localhost:3030/reg.

The screenshot shows the Postman interface with a POST request to 'localhost:3030/reg'. The 'Body' tab is selected, displaying the following JSON payload:

```
1 "AdminID": "041269",
2 "AdminFname": "Leon",
3 "AdminLname": "Scott",
4 "AdminNname": "RE4",
5 "AdminEmail": "ashley@hotmail.com",
6 "AdminPhone": "095-566-8551",
7 "AdminUser": "Bio4RE",
8 "AdminPass": "nothankbro",
9 "AdminRole": "Web designer"
```

The 'Body' tab also shows the response from the server:

```
1 "error": false,
2 "message": "Admin registration successful"
```

When using GET localhost:3030/reg there will be the new admin data that was recently inserted.

The screenshot shows the Postman interface with a GET request to 'localhost:3030/reg'. The response body contains two admin objects:

```
"adminLogin": [
  {
    "ID_Admin": "6488066",
    "AdminUser": "AradaAbsent",
    "AdminPass": "6600660066",
    "AdminRole": "Web designer",
    "LoginLog": null
  },
  {
    "ID_Admin": "041269",
    "AdminUser": "Bio4RE",
    "AdminPass": "nothankbro",
    "AdminRole": "Web designer",
    "LoginLog": null
  }
]
```

Update data: we use PUT localhost3030/reg/:id with the remaining administrator. For this case, we want to update the data of Admin Kirin which has ID = '6666666'  
This is the old data.

The screenshot shows the Postman interface with a PUT request to 'localhost:3030/reg/6666666'. The request body is:

```
{
  "ID_Admin": "6666666",
  "AdminUser": "KirinAha",
  "AdminPass": "11100111",
  "AdminRole": "Mascot",
  "LoginLog": null
}
```

The response body is:

```
{
  "AdminID": "6666666",
  "AdminFname": "Kirin",
  "AdminLname": "Elderdragon",
  "AdminNname": "Kirin",
  "AdminEmail": "Kirinza007@gmail.com",
  "AdminPhone": "111-111-1111"
}
```

Then we use PUT localhost:3030/reg/6666666 by following data, when it done the success message will show like this.

The screenshot shows the Postman interface for a PUT request to `localhost:3030/reg/6666666`. The request method is set to `PUT`. The `Body` tab is selected, showing the following JSON payload:

```
1 "AdminName": "Krauser",
2 "AdminName": "Kiki",
3 "AdminName": "Klause",
4 "AdminEmail": "merceneries@hotmail.com",
5 "AdminPhone": "025-647-8845",
6 "AdminUser": "Inwkrkrauser",
7 "AdminPass": "root555",
8 "AdminRole": "Database manager"
9
10
```

The response tab shows the following JSON data:

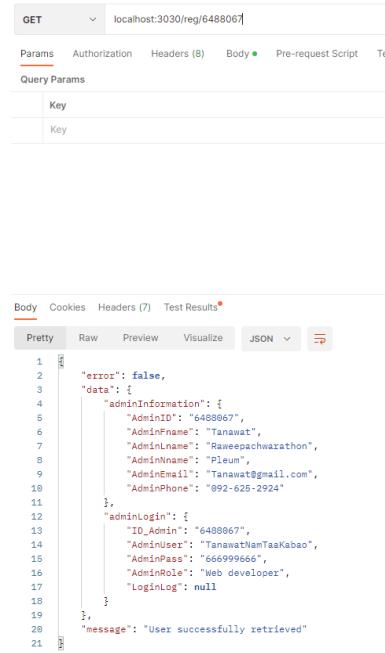
```
1
2   "error": false,
3   "message": "Admin information updated successfully"
4
```

When we check by use GET localhost:3030/reg/6666666 the data inside has been replaced with the new ones.

The screenshot shows the Postman interface for a GET request to `localhost:3030/reg/6666666`. The request method is set to `GET`. The `Body` tab is selected, showing the following JSON response:

```
1
2   "error": false,
3   "data": {
4     "adminInformation": {
5       "AdminID": "6666666",
6       "AdminName": "Krauser",
7       "AdminName": "Kiki",
8       "AdminName": "Klause",
9       "AdminEmail": "merceneries@hotmail.com",
10      "AdminPhone": "025-647-8845"
11    },
12    "adminLogin": {
13      "ID_Admin": "6666666",
14      "AdminUser": "Inwkrkrauser",
15      "AdminPass": "root555",
16      "AdminRole": "Database manager",
17      "LoginLog": null
18    }
19  },
20  "message": "User successfully retrieved"
21
```

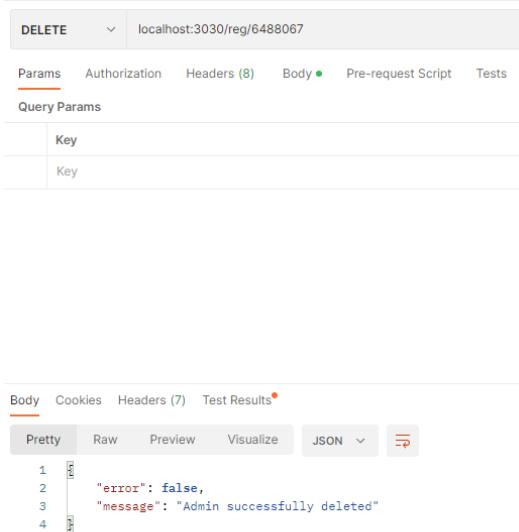
Delete data: we use DELETE localhost:3030/:reg/:id with the remaining administrator. For this case, we want to delete the data of Admin Tanawat which has ID = '6488067'



```
GET      localhost:3030/reg/6488067
Params   Authorization   Headers (8)   Body   Pre-request Script   Tests
Query Params
Key
Key

Body  Cookies  Headers (7)  Test Results*
Pretty  Raw  Preview  Visualize  JSON  ↷
1  [
2    "error": false,
3    "data": {
4      "adminInformation": {
5        "AdminID": "6488067",
6        "AdminFname": "Tanawat",
7        "AdminLname": "Rawepacharathon",
8        "AdminName": "Pleum",
9        "AdminEmail": "tanawat@gmail.com",
10       "AdminPhone": "092-626-2924"
11     },
12     "adminLogin": {
13       "ID_Admin": "6488067",
14       "AdminUser": "TanawatTanTaaKabao",
15       "AdminPass": "66699966",
16       "AdminRole": "Web developer",
17       "LoginLog": null
18     }
19   },
20   "message": "User successfully retrieved"
21 ]
```

After that we use DELETE on this admin.



```
DELETE      localhost:3030/reg/6488067
Params   Authorization   Headers (8)   Body   Pre-request Script   Tests
Query Params
Key
Key

Body  Cookies  Headers (7)  Test Results*
Pretty  Raw  Preview  Visualize  JSON  ↷
1  [
2    "error": false,
3    "message": "Admin successfully deleted"
4  ]
```

And when we check by GET localhost:3030/reg there is no admin Tanawat in the admin data result.

```

        "adminLogin": [
            {
                "ID_Admin": "6488066",
                "AdminUser": "AradaAbsent",
                "AdminPass": "6600660066",
                "AdminRole": "Web designer",
                "LoginLog": null
            },
            {
                "ID_Admin": "041269",
                "AdminUser": "BioARE",
                "AdminPass": "nothankbro",
                "AdminRole": "Web designer",
                "LoginLog": null
            },
            {
                "ID_Admin": "6666666",
                "AdminUser": "Inwkzause",
                "AdminPass": "look555",
                "AdminRole": "Database manager",
                "LoginLog": null
            },
            {
                "ID_Admin": "6488088",
                "AdminUser": "IttikornSukPaiLaeu",
                "AdminPass": "8000800088",
                "AdminRole": "Web developer",
                "LoginLog": null
            },
            {
                "ID_Admin": "6488038",
                "AdminUser": "Kanyaveza",
                "AdminPass": "2468102",
                "AdminRole": "Web developer",
                "LoginLog": null
            },
            {
                "ID_Admin": "6488124",
                "AdminUser": "Tanawat",
                "AdminPass": "1234567890",
                "AdminRole": "Web developer",
                "LoginLog": null
            }
        ]
    }
}

```

## prod.js

no criteria search: GET localhost:3030/prod.

```

localhost:3030/prod
GET localhost:3030/prod
Params Authorization Headers (0) Body Pre-request Script Tests Settings
Body Cookies Headers (7) Test Results*
Pretty Raw Preview Visualize JSON ▾
1 {
2     "errors": false,
3     "exits": [
4         {
5             "prodID": "985",
6             "ProductName": "Grilled sea bass with miso sauce",
7             "ProdType": "Seafood dish",
8             "ProdLevel": "Fried",
9             "ProdFactor": null,
10            "ProdDesc": "It's a healthy menu. Help nourish the bones. Can help strengthen and relieve rheumatoid arthritis. Reduces high blood pressure",
11            "ProdRecipe": "Sea bass fillet (cut into pieces) 300 g., Miso 1 tbsp., 1 egg (yolk only). Vegetable oil, grated radish, to Japanese steamed rice",
12            "ProdSteps": "1.Pour the miso and mirin into a pan over low heat, stir until the consistency, add the egg yolk and stir until homogeneous. Set aside. 2.set fire to grill. Spread vegetable oil over the sieve. Apply miso sauce (as in step 1) over the sea bass. 3.Grilled over low heat until(wcooked). Served with grated radish, and Japanese steamed rice"
13        },
14        {
15            "prodID": "982",
16            "ProductName": "Glest soup with omelete",
17            "ProdType": "Main course",
18            "ProdLevel": "Starch",
19            "ProdFactor": "Low calories",
20            "ProdDesc": "It's easy to eat. Suitable for everyone, especially those who want to control their diet",
21            "ProdRecipe": "2-3 eggs, onions, spring onions, coriander, seasoning powder, fish sauce, soy sauce, sugar, water",
22            "ProdSteps": "1.Slice spring onions and onions into small pieces. 2.Crack an egg into a bowl, add sliced onions,soy onions and a little fish sauce<3.Heat the pan, add a little oil when the oil is hot, pour the eggs into the pan and fry until the eggplant to cook on both sides.<4.Add a spoonful to cut the eggs into pieces. 4.Fill it with water. Season with sugar, soy sauce, seasoning powder, and finally add the remaining spring onions.<5.Turn off the stove and sprinkle with coriander leaves."
23    },
24    {
25        "prodID": "983",
26        "ProductName": "Oodiy miso vegetable soup",
27        "ProdType": "Main dish",
28        "ProdLevel": "Fruit and vegetable",
29        "ProdFactor": "Low calories",
30        "ProdDesc": "Mixed vegetable soup is a traditional Thai dish. It focuses more on vegetables than meat and is healthy for everyone.",
31        "ProdRecipe": "1/2 cup dried shrimp, 1/2 cup finely ground dried shrimp, 2 cups chicken broth, Chinese笛子, cut into bite-sized pieces 100 g., Pumpkin, cut into bite-sized pieces 200 g., Courgette, cut into bite size pieces 100 g., Baby corn, sliced 100 g., 100 g of champion mushrooms, 200 g of fresh shrimp, basil 80 g., Fish sauce",
32        "ProdSteps": "1.Prepare the curry paste first. 2.Pound the popper with the Khachai, then add the shallots, pound thoroughly, followed by the shrimp paste, pound together, then add the ground dried shrimp, pound together again until thoroughly.<3.Put the chicken broth in the pot. Bring to a boil and add curry paste. 4.When the soup is boiling, add vegetables that are difficult to cook first. Bring to a boil again then add the rest of the vegetables.<5.Once the water is boiling add the chinese okra, pumpkin, courgette and"

```

```

    the ground dried shrimp, pound together again until thoroughly.\n3.Put the chicken broth in the pot. Bring to a boil and add curry paste. 4.When the soup is boiling, add
    vegetables that are difficult to cook first. Bring to a boil again then add the rest of the vegetables.\n5.Once the water is boiling add the chinese okra, pumpkin, courgette and
    mushrooms. 6.When the soup is boiling again, add fresh shrimp. Season with fish sauce as desired. 7.When the mixture boils again, \nadd the basil leaves. scoop served hot."
33 },
34 },
35 "ProdID": "084",
36 "ProdName": "Boiled rice with grouper",
37 "ProdType": "Aperitizer",
38 "ProdNeutrl": "Protein",
39 "ProdFactor": "low calories",
40 "ProdDesc": "It's a food that's easy to digest, not sticky. Beneficial to the body. Can be eaten by all genders, ages, suitable for patients.",
41 "ProdRecipe": "Pour bone broth 1,200 mL. Young galangal, thinly sliced 3 tbsp., Shiitake mushrooms (soaked in water until soft) 100 g, 4 tablespoons soy sauce, 1 tablespoon sugar, 2
    teaspoons ground pepper, Grouper fillet (cut into pieces, boiled)\n10 pieces., 300 g cooked jasmine rice., Finely chopped galangal, Braised leeks, Chopped celery",
42 "ProdStep": "1.Bring the bone broth with young galangal until boiling, add shiitake mushrooms, season with light soy sauce,\nsugar and ground pepper, reduce heat and simmer until
    boiling broth. 2.Set aside. 3.Heat the water in another pot until it boils. Add jasmine rice to cook in boiling water until soft, scoop into a bowl, and\nplace blanched grouper.
        4.Put the pork bone broth, sprinkle finely pounded galangal, fried garlic, and celery. Eat while hot with soybean sauce."
43 },
44 },
45 "ProdID": "085",
46 "ProdName": "Spicy chicken soup with mushroom",
47 "ProdType": "Aperitizer",
48 "ProdNeutrl": "Protein",
49 "ProdFactor": "low calories",
50 "ProdDesc": "This spicy nutritious soup is suitable for all ages. It has many benefits and does not cause obesity",
51 "ProdRecipe": "Tom Yum soup consists of galangal, lemongrass, kaffir lime leaves, shallots (peeled) 4-5 heads, Bird's Eye Chilli, Straw Mushroom, Grape Tomatoes, 1 tablespoon wet
    tamari, 1+1/2 tbsp fish sauce, lemon, thinly sliced, chicken breast, parsley",
52 "ProdStep": "1.Boil water 2.Put the Tom Yum ingredients, the galangal is pounded before putting it in. Do not add too much. 3.Smash the lemongrass and cut it into small pieces, put
    it until fragrant. 4.Followed by shallots and tomatoes. 5. When everything falls into place, add the sliced chicken breast.\nwait until cooked, then add the mushrooms.6.
        Season with tamari juice and fish sauce. 6.Remove the pot from the stove. 7.Then tear kaffir lime leaves and sprinkle them on top. 8.Add crushed bird's eye chili to increase
        the spiciness, and \nSprinkle parsley and add seasoning to your liking"
53 },
54 ],
55 "message": "Product successfully retrieved"
56

```

## Criteria search: search by ID: GET localhost:3030/prod/:id.

```

localhost:3030/prod/004
GET localhost:3030/prod/004
Params Authorization Headers (8) Body ● Pre-request Script Tests Settings Cookies
Body Cookies Headers (7) Test Results
Pretty Raw Preview Visualize JSON
Status: 200 OK Time: 6 ms Size: 1.37 KB Save as Example
1 "error": false,
2 "data": [
3     {
4         "ProdID": "084",
5         "ProdName": "Boiled rice with grouper",
6         "ProdType": "Aperitizer",
7         "ProdNeutrl": "Protein",
8         "ProdFactor": "low calories",
9         "ProdDesc": "It's a food that's easy to digest, not sticky. Beneficial to the body. Can be eaten by all genders, ages, suitable for patients.",
10        "ProdRecipe": "Pour bone broth 1,200 mL. Young galangal, thinly sliced 3 tbsp., Shiitake mushrooms (soaked in water until soft) 100 g, 4 tablespoons soy sauce, 1 tablespoon sugar, 2
            teaspoons ground pepper, Grouper fillet (cut into pieces, boiled)\n10 pieces., 300 g cooked jasmine rice., Finely chopped galangal, Braised leeks, Chopped celery",
11        "ProdStep": "1.Bring the bone broth with young galangal until boiling, add shiitake mushrooms, season with light soy sauce,\nsugar and ground pepper, reduce heat and simmer until
            boiling broth. 2.Set aside. 3.Heat the water in another pot until it boils. Add jasmine rice to cook in boiling water until soft, scoop into a bowl, and\nplace blanched grouper.
                4.Put the pork bone broth, sprinkle finely pounded galangal, fried garlic, and celery. Eat while hot with soybean sauce."
12    }
13 ],
14 "message": "Product successfully retrieved"
15
16

```

## Criteria search: search by Product nutritious: GET localhost:3030/prod/nutrition/: nutrition.

```

localhost:3030/prod/nutrition/Fruit and vegetable
GET localhost:3030/prod/nutrition/Fruit and vegetable
Params Authorization Headers (8) Body ● Pre-request Script Tests Settings Cookies
Body Cookies Headers (7) Test Results
Pretty Raw Preview Visualize JSON
Status: 200 OK Time: 5 ms Size: 163 KB Save as Example
1 "error": false,
2 "data": [
3     {
4         "ProdID": "083",
5         "ProdName": "Spicy mixed vegetable soup",
6         "ProdType": "Main dish",
7         "ProdNeutrl": "Fruit and vegetable",
8         "ProdFactor": "low calories",
9         "ProdDesc": "Mixed vegetable soup is a traditional Thai dish. It focuses more on vegetables than meat and is healthy for everyone.",
10        "ProdRecipe": "1+1/2 tbs black pepper, Fingerroot 1-2 stems, 3-4 shallots, Shrimp paste 1 teaspoon, 1/3 cup finely ground dried shrimp, 3 cups chicken broth, Chinese okra, cut into
            bite-sized pieces 100 g, Pumpkin, cut into bite-sized pieces 200 g, Courgette, cut into bite size pieces 150 g, Baby corn, sliced 150 g, 150 g of champion Mushrooms, 200 g of
            fresh shrimp, basil 80 g, fish sauce",
11        "ProdStep": "1.Prepare the curry paste first. 2.Pound the pepper with the Krachai, then add the shallots, pound thoroughly, followed by the shrimp paste, pound together, then add
            the ground dried shrimp, pound together again until thoroughly.\n3.Put the chicken broth in the pot. Bring to a boil and add curry paste. 4.When the soup is boiling, add
            vegetables that are difficult to cook first. Bring to a boil again then add the rest of the vegetables.\n5.Once the water is boiling add the chinese okra, pumpkin, courgette and
            mushrooms. 6.When the soup is boiling again, add fresh shrimp. Season with fish sauce as desired. 7.When the mixture boils again, \nadd the basil leaves. scoop served hot."
12    }
13 ],
14 "message": "Product successfully retrieved"
15
16

```

Criteria search: search by Product Type: GET localhost:3030/prod/type/: type.

```

1  {
2      "error": false,
3      "data": [
4          {
5              "ProdID": "001",
6              "ProdName": "Grilled sea bass with miso sauce",
7              "ProdType": "Main dish",
8              "ProdNutri": "Protein",
9              "ProdFactor": null,
10             "ProdDesc": "It's a healthy food menu. Help nourish the bones. Can help strengthen and relieve rheumatoid arthritis. Reduces high blood pressure",
11             "ProdRecip": "Sea bass fillet (cut into pieces) 300 g., Miso 80 g, Mirin 1 tbsp ,1 egg (yolk only), Vegetable oil, grated radish, to Japanese steamed rice",
12             "ProdStep": "1.Pour the miso and mirin into a pan over low heat, stir until the consistency, add the egg yolks and stir until homogeneous. Set aside. 2.set fire to grill. Spread vegetable oil over the sieve. Apply miso sauce (as in step 1) over the sea bass. 3.Grilled over low heat until\ncooked. Served with grated radish, and Japanese steamed rice"
13         },
14         {
15             "ProdID": "003",
16             "ProdName": "Spicy mixed vegetable soup",
17             "ProdType": "Main dish",
18             "ProdNutri": "Fruit and vegetable",
19             "ProdFactor": "Low calories",
20             "ProdDesc": "Mixed vegetable soup is a traditional Thai dish. It focuses more on vegetables than meat and is healthy for everyone.",
21             "ProdRecip": "1/2 tsp black pepper, Fingernoot 2 stems ,3-4 shallots, Shrimp paste 1 teaspoon, 1/3 cup finely ground dried shrimp, 3 cups chicken broth, Chinese okra, cut into bite-sized pieces 150 g, Pumpkin, cut\ninto bite-sized pieces 200 g, Baby corn, sliced 150 g, 150 g of champion Mushrooms, 200 g of fresh shrimp, basil 80 g, fish sauce",
22             "ProdStep": "1.Prepare the curry paste first. 2.Pound the pepper with the Krachai, then add the shallots, pound thoroughly, followed by the shrimp paste, pound together, then add the ground dried shrimp, pound together again until thoroughly.\n3.Put the chicken broth in the pot. Bring to a boil and add curry paste. 4.When the soup is boiling, add vegetables that are difficult to cook first. Bring to a boil again then add the rest of the vegetables.\n5.Once the water is boiling add the chinese okra, pumpkin, courgette and mushrooms. 6.When the soup is boiling again, add fresh shrimp. Season with fish sauce as desired. 7.When the mixture boils again, \nadd the basil leaves. scoop served hot."
23         }
24     ],
25     "message": "Product successfully retrieved"
26 }
```

Insert data: POST localhost:3030/prod

```

1  {
2      "ProdID": "004",
3      "ProdName": "Tom yum kung",
4      "ProdType": "Main Dish",
5      "ProdNutri": "Fruit and vegetables",
6      "ProdFactor": "Low calories",
7      "ProdDesc": "This hot and sour shrimp soup is an explosion of flavour and is also said to have medicinal properties",
8      "ProdRecip": "16 shrimps or tiger prawns ,100g of mushroom ,600g of chicken stock ,1 tbsp chopped lemongrass ,1 tbsp chopped galangal ,3-4 kaffir lime leaves ,3-4 fresh red chillies, 5 tbsp fish sauce, 4 tbsp Lime Juice, 1 tbsp chilli oil, 1 tbsp chopped coriander",
9      "ProdStep": "1.Clean the shrimp and remove the black thread (the intestine) from the shrimp by cutting it down the back. 2.Slice the galangal and lemongrass into short 3cm pieces and pound it in a mortar to break out the flavours. 3.Remove the stalk from the kaffir lime leaves and tear apart. 4.Put the chicken stock in a pan and bring to the boil. 5.When the water is boiling add the galangal, lemongrass, kaffir lime leaves and wait for it to come back to the boil. 6.Add the shrimps and cook for 2 minutes. 7.Break the chillies into a mortar and pound them for a few moments to a pulp then add to the soup. 8.Add the fish sauce, lime juice and chilli oil to the soup. 9.Turn off the heat and add the coriander leaves."
10 }
```

```

1  {
2      "error": false,
3      "message": "Product successfully inserted"
4 }
```

When using GET localhost:3030/prod there will be the new product data that recently insert.

Update data: we use PUT localhost3030:/prod/:id with the remaining product. For this case, we want to update the data of Clear soup with omelet which has ID = '002'

```
localhost:3030/prod/002

GET /prod/002
Status: 200 OK Time: 8 ms Size: 1.11 KB Save as Example

Param Authorization Headers (8) Body Pre-request Script Tests Settings
Body Cookies Headers (7) Test Results*
Pretty Raw Preview Visualize JSON ▾

1 {
2   "error": false,
3   "data": [
4     {
5       "ProdID": "002",
6       "ProdName": "Clear soup with omelette",
7       "ProdType": "Appetizer",
8       "ProdWtul": "Starch",
9       "ProdFactor": "Low calories",
10      "ProdDesc": "It's easy to eat. Suitable for everyone, especially those who want to control their diet.",
11      "ProdCin": "2 eggs, onions, spring onions, coriander, seasoning powder, fish sauce, soy sauce, sugar, water",
12      "ProdIn": "2 eggs, onions and onions and small pieces 2.000g in a bowl, add sliced onions, adding onions and a little fish sauce)n3.Heat the pan, add a little oil when the oil is hot, then add the eggs into the pan and fry until the eggstart to cook on both sides.Use a scutula to cut the eggs into pieces.4.Fill it with water. Season with sugar, fish sauce, seasoning powder, and finally add the remaining spring onions.Turn off the stove and sprinkle with coriander leaves."
13    }
14  ],
15  "message": "Product successfully retrieved"
}
```

Then we use PUT localhost3030:/reg/002 by following data, when it done the success message will show like this.

When we check by using GET localhost:3030/prod/002 the data inside has been replaced with the new ones.

```

localhost:3030/prod/002
GET localhost:3030/prod/002
Params Authorization Headers (8) Body Pre-request Script Tests Settings
Query Params
Key Value Description
Key Value Description
Body Cookies Headers (7) Test Results*
Pretty Raw Preview Visualize JSON
1
2     "error": false,
3     "data": [
4         {
5             "ProdID": "002",
6             "ProdName": "Seared Salmon With Charred Green Beans",
7             "ProdType": "Main Dish",
8             "ProdNutri": "Protein",
9             "ProdFactor": "low calories",
10            "ProdDesc": "Seared Salmon With Charred Green Beans is food that is easy to make. Complete nutritional value. It is a food that even people who lose weight can eat and it tastes so good that unlike healthy food by just have salmon, green beans, garlic, capers and red chile. That's it!",
11            "ProdRecipe": "1. Toss 2 tbsps olive oil; divided, 1/2 lb. skinless salmon fillet, cut into 4 portions,1 lb. green beans, trimmed, Kosher salt and pepper,4 cloves garlic, smashed and thinly sliced,1 small red chile, thinly sliced,2 tbsp. capers, drained, patted dry,lemon wedges, for serving",
12            "ProdStep": "1. Heat 2 teaspoons oil in a large skillet on medium-high. Season salmon with N teaspoon each salt and pepper, add to skillet, flesh side down, reduce heat to medium, and cook until golden brown and just opaque throughout, 5 to 6 minutes per side. 2. Heat remaining 2 tablespoons oil in a large cast-iron skillet on medium-high. Add green beans and cook until browned, 2N minutes. Turn with tongs and cook until browned and just barely tender, about 3 minutes more. 3. Remove from heat and toss with N teaspoon salt, then garlic, chile, and capers. Return to medium heat and cook, tossing until garlic is golden brown, 1 to 2 minutes. Serve with salmon and lemon wedges if desired."
13        }
14    ],
15    "message": "Product successfully retrieved"
16

```

Delete data: we use DELETE localhost3030:/prod/:id with the remaining product. For this case, we want to delete the data of Grilled Sea bass with miso sauce which has ID = '001'

```

localhost:3030/prod/001
GET localhost:3030/prod/001
Params Authorization Headers (8) Body Pre-request Script Tests Settings
Query Params
Key Value Description
Key Value Description
Body Cookies Headers (7) Test Results*
Pretty Raw Preview Visualize JSON
1
2     "error": false,
3     "data": [
4         {
5             "ProdID": "001",
6             "ProdName": "Grilled sea bass with miso sauce",
7             "ProdType": "Main Dish",
8             "ProdNutri": "Protein",
9             "ProdFactor": "null",
10            "ProdDesc": "It's a healthy food menu. Help nourish the bones. Can help strengthen and relieve rheumatoid arthritis. Reduces high blood pressure",
11            "ProdRecipe": "Sea bass fillet (cut into pieces) 300 g., Miso 80 g. Mirin 1 tbsp ,1 egg (yolk only), Vegetable oil, grated radish, to Japanese steamed rice",
12            "ProdStep": "1.Pour the miso and mirin into a pan over low heat, stir until the consistency, add the egg yolks and stir until homogeneous. Set aside. 2.set fire to grill. Spread vegetable oil over the sieve. Apply miso sauce (as in step 1) over the sea bass. 3.Grilled over low heat until\ncooked. Served with grated radish. and Japanese steamed rice"
13        }
14    ],
15    "message": "Product successfully retrieved"
16

```

After that we use DELETE on this product.

The screenshot shows the Postman interface with the following details:

- URL: `localhost:3030/prod/001`
- Method: `DELETE`
- Request URL: `localhost:3030/prod/001`
- Headers:
  - Params
  - Authorization
  - Headers (8)
  - Body
  - Pre-request Script
  - Tests
  - Settings
- Query Params table:

Key	Value
Key	Value
- Body tab:
  - Pretty
  - Raw
  - Preview
  - Visualize
  - JSON

```
1
2   "error": false,
3   "message": "Product successfully deleted"
4
```

When we use GET `localhost:3030/prod/002` to call admin Grilled Sea bass with miso sauce data it provides an empty array.

The screenshot shows the Postman interface with the following details:

- URL: `localhost:3030/prod/001`
- Method: `GET`
- Request URL: `localhost:3030/prod/001`
- Headers:
  - Params
  - Authorization
  - Headers (8)
  - Body
  - Pre-request Script
  - Tests
  - Settings
- Query Params table:

Key	Value
Key	Value
- Body tab:
  - Pretty
  - Raw
  - Preview
  - Visualize
  - JSON

```
1
2   "error": false,
3   "data": [],
4   "message": "Product successfully retrieved"
5
```

And when we check by GET localhost:3030/prod there is no Grilled Sea bass with miso sauce in the admin data result.

```
  "error": false,
  "data": [
    {
      "ProdID": "002",
      "ProdName": "Seared Salmon With Charred Green Beans",
      "ProdType": "Main dish",
      "ProdNeutri": "Protein",
      "ProdFactor": "low calories",
      "ProdDesc": " Seared Salmon With Charred Green Beans is food that is easy to make. Complete nutritional value. It is a food that even people who lose weight can eat and it tastes so good that unlike healthy food by just have salmon, green beans, garlic, capers and red chile. That's it!",
      "ProdRecipe": " 2 Tbsp. plus 2 tsp. olive oil, divided, 1 1/4 lb. skinless salmon fillet, cut into 4 portions, 1 lb. green beans, trimmed, Kosher salt and pepper, 4 cloves garlic, smashed and thinly sliced, 1 small red chile, thinly sliced, 2 tbsp. capers, drained, patted dry, lemon wedges, for serving",
      "ProdStep": "1. Heat 2 teaspoons oil in a large skillet on medium-high. Season salmon with ½ teaspoon each salt and pepper, add to skillet, flesh side down, reduce heat to medium, and cook until golden brown and just opaque throughout, 5 to 6 minutes per side. 2. Heat remaining 2 tablespoons oil in a large cast-iron skillet on medium-high. Add green beans and cook until browned, 2 minutes. Turn with tongs and cook until browned and just barely tender, about 3 minutes more. 3. Remove from heat and toss with ½ teaspoon salt, then garlic, chile, and capers. Return to medium heat and cook, tossing until garlic is golden brown, 1 to 2 minutes. Serve with salmon and lemon wedges if desired."
    },
    {
      "ProdID": "003",
      "ProdName": "Spicy mixed vegetable soup",
      "ProdType": "Main dish",
      "ProdNeutri": "Fruit and vegetable",
      "ProdFactor": "Low calories",
      "ProdDesc": " Mixed vegetable soup is a traditional Thai dish. It focuses more on vegetables than meat and is healthy for everyone.",
      "ProdRecipe": " 1/2 tsp black pepper, Fingerroot 1-2 stems , 3-4 shallots, Shrimp paste 1 teaspoon, 1/3 cup finely ground dried shrimp, 3 cups chicken broth, Chinese okra, cut into bite-sized pieces 150 g, Pumpkin, cut\ninto bite-sized pieces 200 g, Courgette, cut into bite size pieces 150 g, Baby corn, sliced 150 g, 150 g of championon Mushrooms, 200 g of fresh shrimp, basil 80 g, fish sauce",
      "ProdStep": "1.Prepare the curry paste first. 2.Pound the pepper with the Krachai, then add the shallots, pound thoroughly, followed by the shrimp paste, pound together, then add the ground dried shrimp, pound together again until thoroughly.\n3.Put the chicken broth in the pot. Bring to a boil and add curry paste. 4.When the soup is boiling, add vegetables that are difficult to cook first. Bring to a boil again then add the rest of the vegetables.\n5.Once the water is boiling add the chinese okra, pumpkin, courgette and"
    }
  ]
}
```