



**The Faculty of Information and Communication Technology
Mahidol University**

Project Phase II

BY

Miss Kunruethai	Patimapornchai	6688076
Miss Sunattha	Boonla-or	6688009
Mr. Tinakome	Rasripenngam	6688095
Mr. Teemtat	Weerarattanamanee	6688143
Mr. Teeratat	Kunavoratham	6688198

**A Report Submitted in Partial Fulfillment of
the Requirements for**

ITCS223 Web Development

**Faculty of Information and Communication Technology
Mahidol University
2024**

Table of Contents

Business Domain	1
Wireframe	2 - 3
Data Model	4
Relational Schema	4
Data Dictionary	5 - 6
Web Application	7
Homepage	7 - 11
Login Page for Administrators	12 - 13
Search page for searching product	14 - 20
Detail page for each product result	21 - 23
Product Management page for administrators	24 - 33
Team Page	34 - 36
Web Services	37 - 44
Testing Results	45
Post Method	45 - 46
Put Method	47 - 48
Delete Method	49
References	50

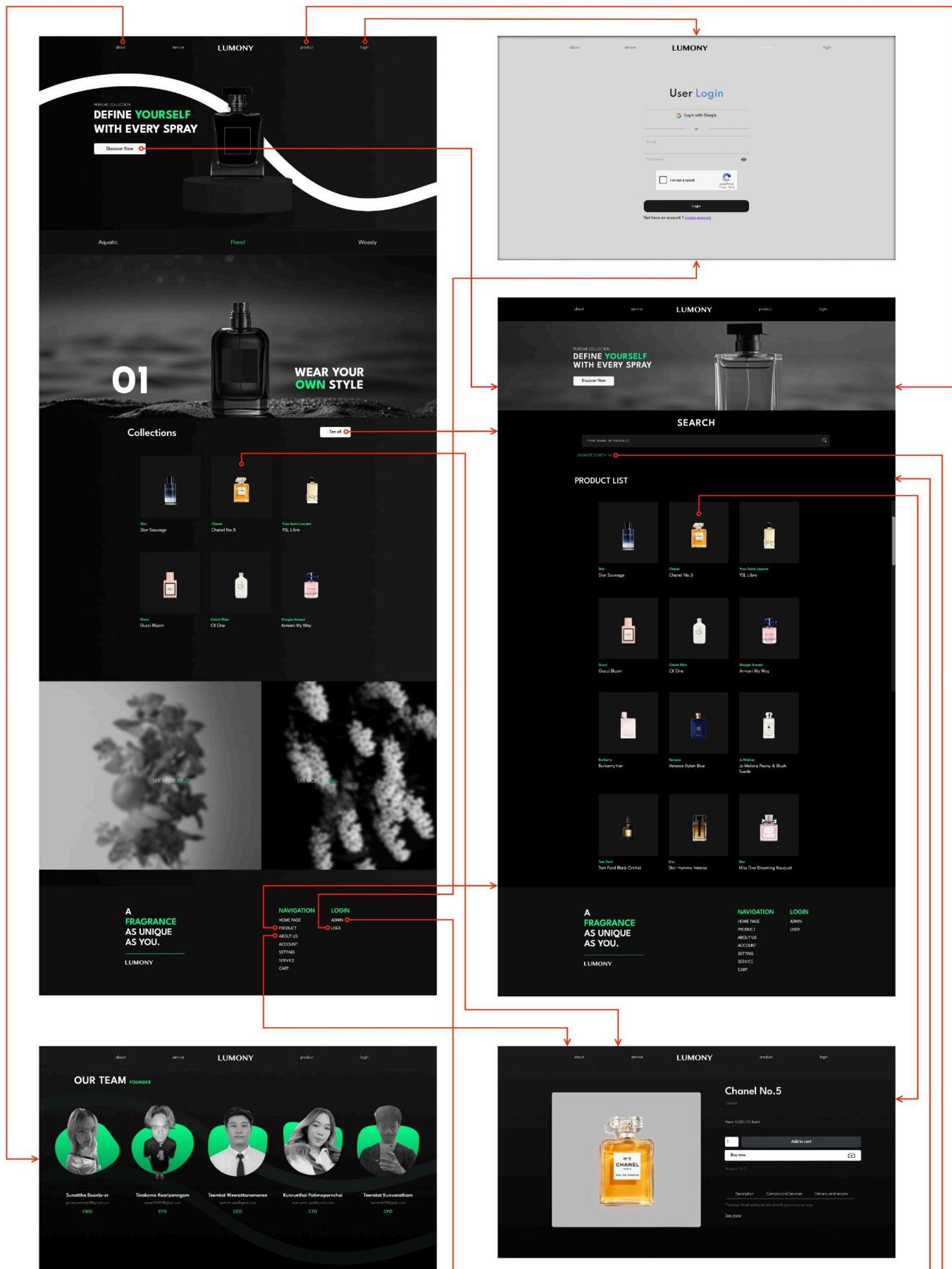
Business Domain

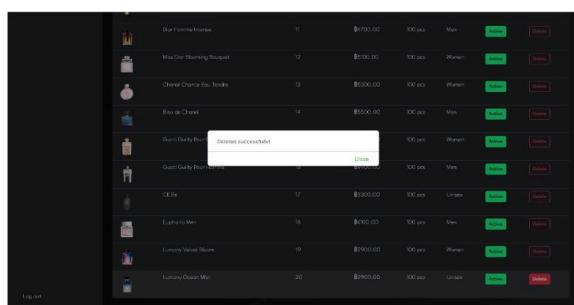
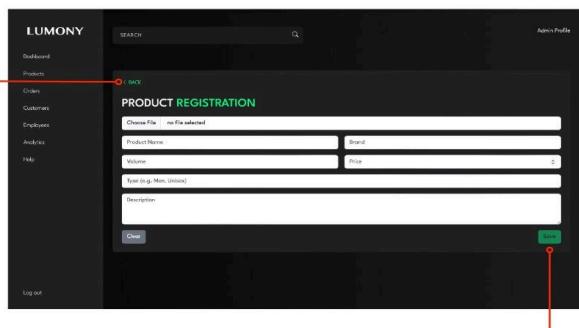
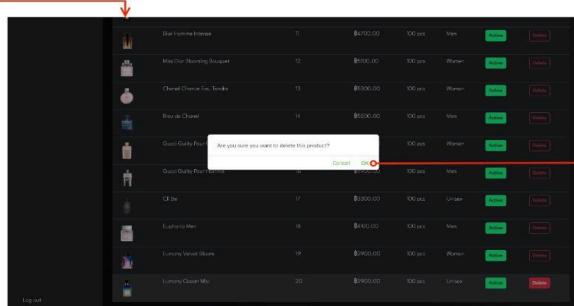
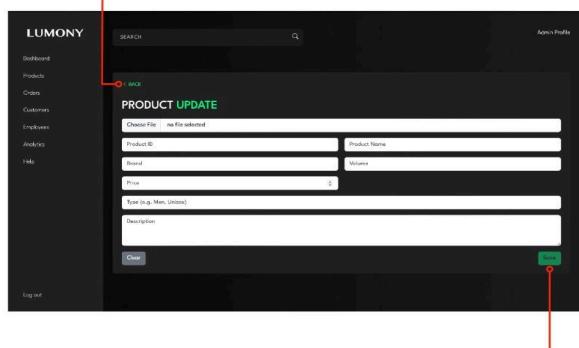
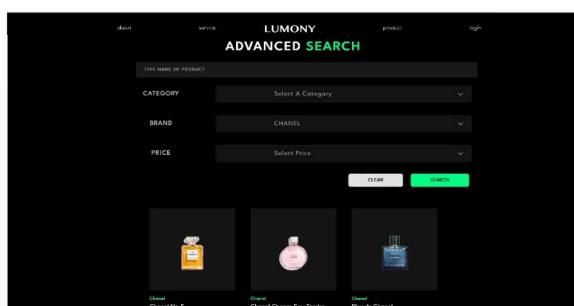
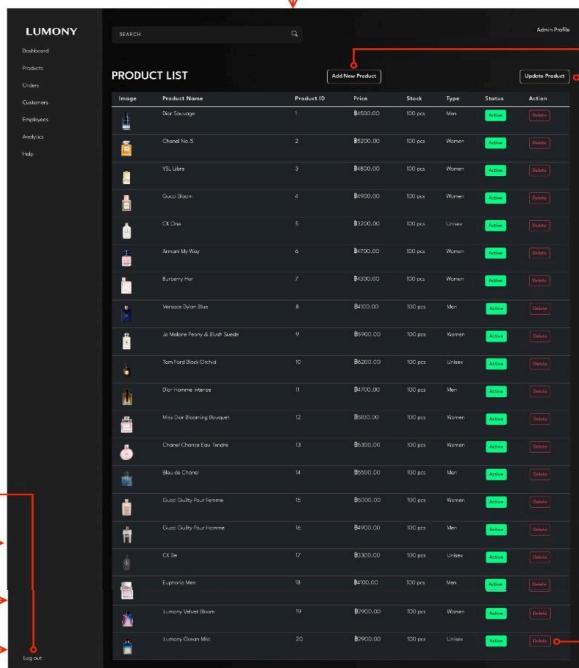
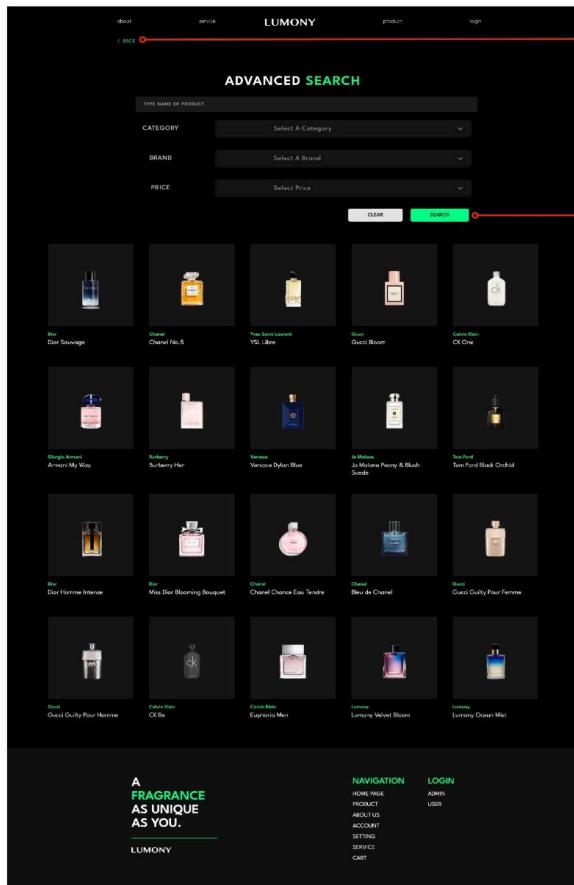
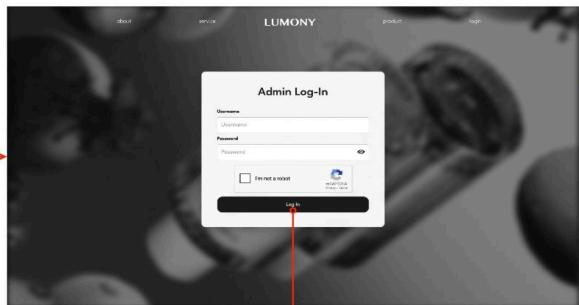
Lumony is a luxury perfume dealer specializing in high-end fragrance brands while also offering our exclusive Lumony collection. We curate a selection of premium perfumes from renowned luxury houses such as Yves Saint Laurent (YSL), Versace, and other exclusive names, guaranteeing genuine quality and elegance in every bottle.

Beyond retailing designer fragrances, we take pride in our Lumony brand, which features uniquely crafted perfumes designed for individuals seeking exclusivity and personalization. Operating solely through our online platform, we provide a seamless and convenient shopping experience, allowing customers to explore and purchase luxury fragrances from anywhere.

Our business scope includes direct online sales, limited-edition releases, and strategic brand collaborations. By combining high-end international perfumes with our own signature creations, we aim to establish Lumony as a distinguished name in the luxury fragrance industry.

Wireframe





Data Model

Relational Schema

The data model is a relational schema which is designed to efficiently manage an administrative product management system, ensuring data consistency, security, and organization. This schema consists of six primary tables that define the relationships between products, administrators, and their respective actions within the system. These tables include:

- **Product:** To store product details.
- **Administrator:** Contains personal information of administrators.
- **Administrator Account:** Manages administrator login credentials, linking to the administrator profile.

Product

Product_ID	Product_name	Product_brand	Product_type	Product_volume	Product_price	Product_image	Product_description
------------	--------------	---------------	--------------	----------------	---------------	---------------	---------------------

Administrator

Admin_ID	Admin_fname	Admin_lname	Admin_email	Admin_address	Admin_DOB
----------	-------------	-------------	-------------	---------------	-----------

Administrator Account

Acc_ID	Acc_username	Acc_password	Admin_ID
--------	--------------	--------------	----------

Data Dictionary

Attribute Name	Content	Type	Format	Nullable	Key	FK Reference
Table: Product						
Product_ID	Product's ID	Char(6)	XXXXXX	No	PK	
Product_name	Product's name	Varchar(50)	Xxxxxx	No		
Product_brand	Product's brand	Varchar(50)	Xxxxxx	No		
Product_type	Product's type/category	Varchar(50)	Xxxxxx	No		
Product_volume	Product's volume	Int	XXX	No		
Product_price	Product's price	Decimal(10,2)	XXX.XX	No		
Product_image	Product's image	Varchar(255)	Xxxxxx	No		
Product_description	Product's description	Varchar(400)	Xxxxxx	Yes		
Table: Administrator						
Admin_ID	Admin's ID	Char(6)	XXXXXX	No	PK	
Admin_fname	Admin's first name	Varchar(50)	Xxxxxx	No		
Admin_lname	Admin's last name	Varchar(50)	Xxxxxx	No		
Admin_email	Admin's email	Varchar(100)	Xxxxxx@x xx	No		
Admin_address	Admin's address	Varchar(100)	Xxxxxx	No		

Admin_DOB	Admin's date of birth	Date	YYYY-MM-DD	No		
Table: Administrator Account						
Acc_ID	Admin account's ID	Char(6)	XXXXXX	No	PK	
Acc_username	Username for admin login	Varchar(50)	Xxxxxx	No		
Acc_password	Password for admin login	Varchar(50)	Xxxxxx	No		
Admin_ID	Admin's ID	Char(6)	XXXXXX	No	FK	Admin_ID [Administrator]

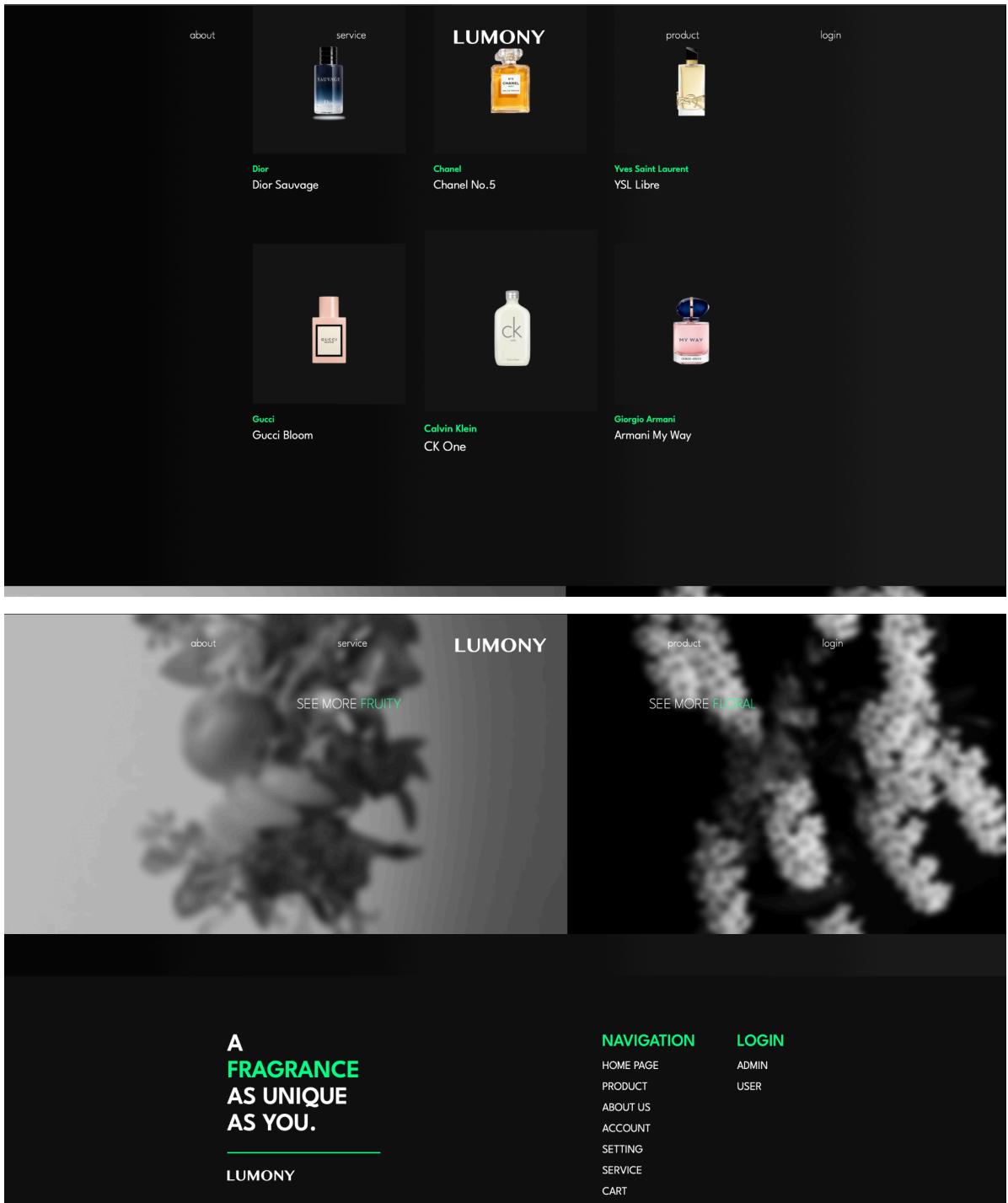
Web Application

Home Page

The image displays two screenshots of a web application's home page, likely for a perfume brand named LUMONY.

Screenshot 1 (Top): This screenshot shows a dark-themed landing page. At the top center is the word "LUMONY". Above the main content area, there are navigation links: "about", "service", "product", and "login". Below these, a large white curved line sweeps across the page. In the center, the text "PERFUME COLLECTION" is followed by "DEFINE YOURSELF WITH EVERY SPRAY" in bold capital letters. A "Discover Now" button is positioned below the text. To the right of the text is a black rectangular perfume bottle with a silver cap, resting on a dark circular base. At the bottom of the page, there are three categories: "Aquatic", "Floral", and "Woody".

Screenshot 2 (Bottom): This screenshot shows a dark-themed landing page with a wavy background texture. At the top center is the word "LUMONY". Above the main content area, there are navigation links: "about", "service", "product", and "login". In the center, a large black rectangular perfume bottle with a silver cap sits on a textured surface. To the left of the bottle is a large white number "01". To the right is the text "WEAR YOUR OWN STYLE" in bold capital letters. Below the bottle, the word "Collections" is displayed in white, along with a "See all" button. At the very bottom of the page, there are three small, partially visible perfume bottles.



Detail: The top menu allows users to navigate through essential sections such as About, Products, Account Login and also return to the landing page by clicking the Lumony logo. Meanwhile, the main banner prominently features effectively highlights key content. Furthermore, the Collections section showcases Lumony's products alongside high-end brands (e.g., YSL, Dior, etc.), ensuring a diverse selection for customers. Additionally, the footer provides extra navigation options, including User Login and Administrator Login, enhancing accessibility and user convenience.

Code: index.html

```
1  <!DOCTYPE html>
2  <html lang="en">
3
4  <head>
5      <meta charset="UTF-8">
6      <meta http-equiv="X-UA-Compatible" content="IE=edge">
7      <meta name="viewport" content="width=device-width, initial-scale=1.0">
8      <title>Lumony | Home-Page</title>
9      <link rel="stylesheet" href="/css/bootstrap.css">
10     <link rel="stylesheet" href="/css/bootstrap.min.css">
11     <link rel="stylesheet" href="/css/bootstrap-grid.css">
12     <link rel="stylesheet" href="/css/mycss.css">
13     <script type="text/javascript" src="/js/bootstrap.bundle.min.js"></script>
14     <script type="text/javascript" src="/js/bootstrap.bundle.js"></script>
15     <script type="text/javascript" src="/js/bootstrap.js"></script>
16     <link rel="shortcut icon" type="image/png" href="/img/logo.png" />
17     <link rel="stylesheet" href="/dist/main.css" />
18     <script src="/dist/main.js"></script>
19     <link rel="preconnect" href="https://fonts.googleapis.com">
20     <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
21     <link href="https://fonts.googleapis.com/css2?family=League+Spartan:wght@100..900&display=swap" rel="stylesheet">
22     <link href="https://fonts.googleapis.com/css2?family=Spartan:wght@400;700&display=swap" rel="stylesheet">
23     <link href="https://fonts.googleapis.com/css2?family=Reem+Kufi:wght@400;700&display=swap" rel="stylesheet">
24  </head>
25
26
27  <body class="bgcolor league-spartan100 text-white">
28
29  <!-- nav -->
30  <div class="sticky-top">
31      <nav class="navbar navbar-expand-lg container">
32          <div class="container-fluid">
33
34              <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNavAltMarkup"
35                  aria-controls="navbarNavAltMarkup" aria-expanded="false" aria-label="Toggle navigation">
36                  <span class="navbar-toggler-icon"></span>
37              </button>
38
39              <ul class="nav-item mt-3">
40                  <a class="nav-link zoom" href="#">team>about</a>
41              </ul>
42
43              <ul class="nav-item mt-3">
44                  <a class="nav-link zoom" href="#">/promotion>service</a>
45              </ul>
46
47              <a class="navbar-brand zoom" href="#">>>
48                  
49              </a>
50
51              <ul class="nav-item mt-3">
52                  <a class="nav-link zoom" href="#">product" target="_blank">product</a>
53              </ul>
54
55              <ul class="nav-item mt-3">
56                  <a class="nav-link zoom" href="#">/login>login</a>
57              </ul>
58
59          </div>
60      </nav>
61  </div>
```

```

63 <section style="position: relative; overflow: hidden">
64   <!-- SVG Wave Background -->
65   <div
66     style="position: absolute; overflow: hidden; top: 0; left: 0; width: 100%; z-index: 0; display: flex; justify-content: center"
67     <svg viewBox="270 0 1700 900" xmlns="http://www.w3.org/2000/svg" style="width: 100%; height: auto;" preserveAspectRatio="xMidYMid meet">
68     <g transform="matrix(1,0,0,1,200,-130)">
69       <path d="M0,350C0,300 620,20 1000,470 C1560,1170 1900,300 2000,900" style="fill:none;stroke: #white;stroke-width:5px" />
70     </g>
71   </svg>
72 </div>
73
74   <!-- Text Content -->
75   <div class="container position-relative" style="z-index: 1; padding-left: 8rem; padding-top: 10rem;">
76     <p class="perfume zoom">PERFUME COLLECTION</p>
77     
78     <h1 class="display-4 fw-bold zoom">
79       DEFINE <span style="color: #06ff87;">YOURSELF</span><br> WITH EVERY SPRAY
80     </h1>
81     <a href="product" class="btn btn-light btn-lg distance-top zoom" style="padding-top: 6px; padding-bottom: 6px; padding-left: 45px; padding-right: 45px;">Discover Now</a>
82   </div>
83
84   <!-- category -->
85   <div class="categories">
86     <div class="zoom" style="position: absolute; left: 380px; top: 20px; height: 710px; z-index: -1;">
87       <a href="#">Aquatic</a>
88     </div>
89     <div class="zoom" style="position: absolute; left: 380px; top: 20px; height: 710px; z-index: -1;">
90       <a href="#" class="active">Floral</a>
91     </div>
92     <div class="zoom" style="position: absolute; left: 380px; top: 20px; height: 710px; z-index: -1;">
93       <a href="#">Woody</a>
94     </div>
95
96   <!-- wear your own -->
97   
98
99   <div class="_01_wear_your_own">
100    <p style="font-family: 'Reem Kufi', sans-serif; font-size: 150px; font-weight: 700;">01</p>
101    <p style="font-size: 47px; font-weight: bold;">WEAR YOUR <br><span style="color: #06ff87;">0WN</span> STYLE</p>
102  </div>
103
104  <!-- collection seeall -->
105  <div class="collection-seeall">
106    <h1>Collections</h1>
107    <a href="product">
108      <button class="see-all zoom">See all</button>
109    </a>
110    <!-- 亂路 -->
111
112    <section class="below-advance">
113      <div id="productListPage" class="py-4">
114        <table class="table table-dark table-hover">
115          <tbody id="productTable" class="grid"></tbody>
116        </table>
117      </div>
118    </section>
119  </div>
120
121  <!-- slogan -->
122  <section class="scent-category">
123    <div style="position: absolute; overflow: hidden; top: 0; left: 0; width: 100%; z-index: -1; display: flex; justify-content: center">
124      <svg viewBox="270 0 1700 900" xmlns="http://www.w3.org/2000/svg" style="width: 100%; height: auto;" preserveAspectRatio="xMidYMid meet">
125        <g transform="matrix(1,0,0,1,200,-200)">
126          <path d="M0,400C0,390 720,20 900,600 C1060,870 1900,50 2000,900" style="fill:none;stroke: #white;stroke-width:5px;" />
127        </g>
128      </svg>
129    </div>
130
131    
132    
133
134    <div class="text-btn">
135      <p class="fruity" href="#">SEE MORE <span style="color: #00ff99;">FRUITY</span></p>
136      <p class="floral" href="#">SEE MORE <span style="color: #00ff99;">FLORAL</span></p>
137    </div>
138  </section>
139
140  </body>
141
142
143
144

```

```

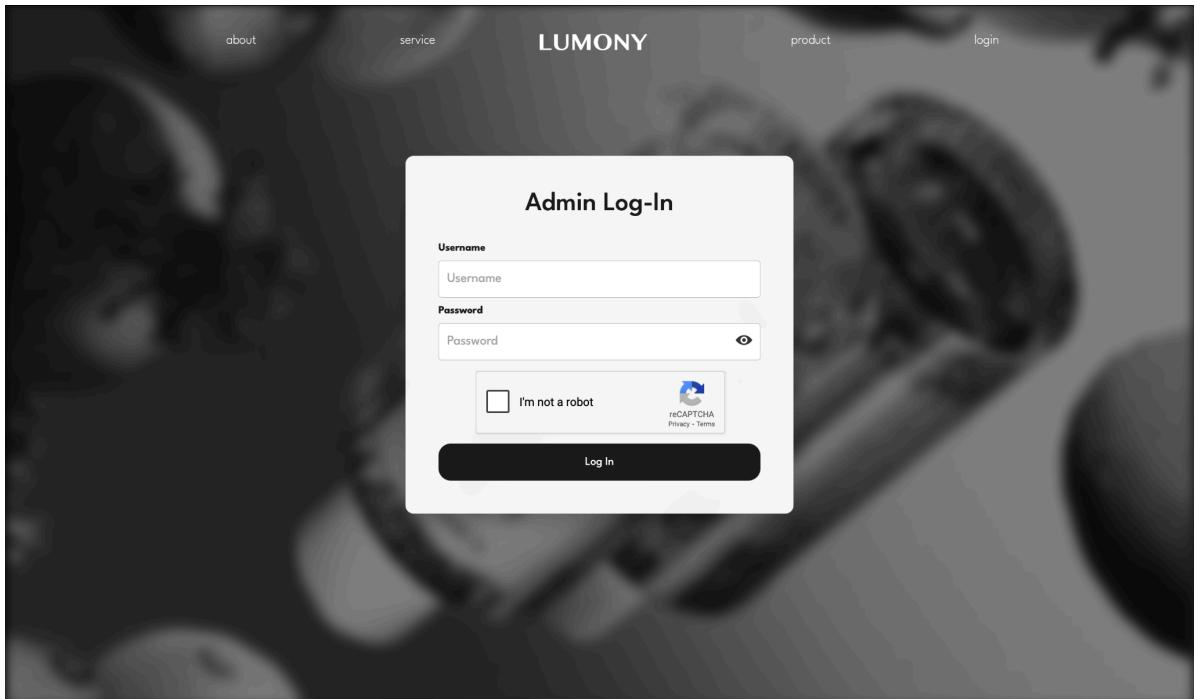
146  <!-- footer -->
147  <footer>
148  <section class="slogan">
149  <div class="slogan-text">
150  | <p>A<br><span>FRAGRANCE</span><br>AS UNIQUE<br>AS YOU.</p>
151  <div class="underline"></div>
152  
153  </div>
154
155  <div class="footer-nav">
156  <div>
157  <h4>NAVIGATION</h4>
158  <ul>
159  | <li class="zoom"><a href="/index">HOME PAGE</a></li>
160  | <li class="zoom"><a href="/product">PRODUCT</a></li>
161  | <li class="zoom"><a href="/team">ABOUT US</a></li>
162  | <li class="zoom"><a href="#">ACCOUNT</a></li>
163  | <li class="zoom"><a href="#">SETTING</a></li>
164  | <li class="zoom"><a href="#">SERVICE</a></li>
165  | <li class="zoom"><a href="#">CART</a></li>
166  </ul>
167  </div>
168  <div>
169  <h4>LOGIN</h4>
170  <ul>
171  | <li class="zoom"><a href="admin_login">ADMIN</a></li>
172  | <li class="zoom"><a href="login">USER</a></li>
173  </ul>
174  </div>
175  </div>
176  </section>
177  </footer>

179 <script>
180
181 // Fetches all products from the server and directly renders them on the page.
182 async function fetchAndRenderProducts() {
183   try{
184     const response = await fetch('/getAllProductsUser');
185     const products = await response.json();
186     renderProductList(products);
187   }
188   catch(error) {
189     console.error('Error fetching products:', error);
190   }
191 }

193 // Renders the first six products as clickable cards linking to their detail pages.
194 function renderProductList(products) {
195   const list = document.getElementById('productTable'); // Make sure this is a <div class="grid">
196   list.innerHTML = ''; // Clear existing content if needed
197   products.slice(0, 6).forEach(element => {
198     const card = document.createElement('div');
199     card.classList.add('card2');
200     card.innerHTML = `
201       <div class="zoom">
202         <div class="img-box" onclick="window.location.href='/product_detail?Product_ID=${element.Product_ID}'">
203           | 
204         </div>
205         <div class="text-each-perfume">
206           | <p class="brand">${element.Product_brand}</p>
207           | <p class="name">${element.Product_name}</p>
208         </div>
209       `;
210     list.appendChild(card);
211   });
212 }
213
214 fetchAndRenderProducts()
215 </script>
216
217 </html>

```

Login Page for Administrators



Detail: The Administrator Login Page is accessed from the footer section of the home page and includes a login form with fields for Username and Password, an "I'm not a robot" reCAPTCHA using an external API, and an authentication button labeled Log In.

Code: admin_login.html

```
1  <!DOCTYPE html>
2  <html lang="en">
3    <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>Lumony | Admin</title>
8      <link rel="stylesheet" href="/css/bootstrap.css">
9      <link rel="stylesheet" href="/css/bootstrap.min.css">
10     <link rel="stylesheet" href="/css/bootstrap-grid.css">
11     <link rel="stylesheet" href="/css/mycss.css">
12     <script type="text/javascript" src="/js/bootstrap.bundle.min.js"></script>
13     <script type="text/javascript" src="/js/bootstrap.bundle.js"></script>
14     <script type="text/javascript" src="/js/bootstrap.js"></script>
15     <link rel="shortcut icon" type="image/jpg" href="/img/logo.png"/>
16     <link rel="stylesheet" href="/dist/main.css" />
17     <script src="/dist/main.js"></script>
18     <link rel="preconnect" href="https://fonts.googleapis.com">
19     <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
20     <link href="https://fonts.googleapis.com/css2?family=League+Spartan:wght@100..900&display=swap" rel="stylesheet">
21     <script src="https://www.google.com/recaptcha/api.js" async defer></script>
22   </head>
```

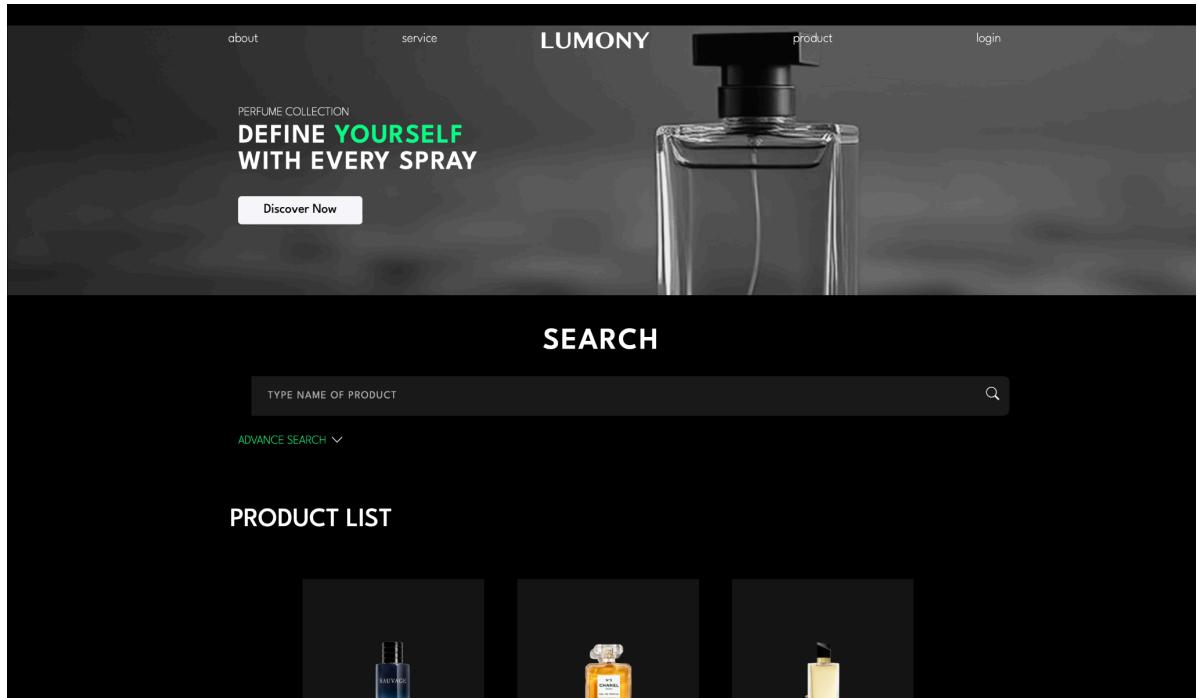
```

24  <body class="bgcolor league-spartan100 text-white">
25    <div class="sticky-top">
26      <nav class="navbar navbar-expand-lg container">
27        <div class="container-fluid">
28          <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNavAltMarkup"
29            aria-controls="navbarNavAltMarkup" aria-expanded="false" aria-label="Toggle navigation">
30              <span class="navbar-toggler-icon"></span>
31            </button>
32
33            <ul class="nav-item mt-3">
34              <a class="nav-link zoom" href="#">team>about</a>
35            </ul>
36
37            <ul class="nav-item mt-3">
38              <a class="nav-link zoom" href="#">/promotion>service</a>
39            </ul>
40
41            <a class="navbar-brand zoom" href="/">
42              
43            </a>
44
45            <ul class="nav-item mt-3">
46              <a class="nav-link zoom" href="#">/product" target="_blank">product</a>
47            </ul>
48
49            <ul class="nav-item mt-3">
50              <a class="nav-link zoom" href="#">/login>login</a>
51            </ul>
52
53        </div>
54      </nav>
55    </div>
56
57
58  <!-- login form -->
59  <div id="loginPage">
60    <section class="admin_log">
61      
62    </section>
63
64    <div class="login-container">
65      <form class="login-box" action="/admin_login_auth" method="POST" onsubmit="return validateCaptcha()">
66        <h2>Admin Log-In</h2>
67
68        <label for="username">Username</label>
69        <input type="text" name="username" placeholder="Username">
70
71        <label for="password">Password</label>
72        <div class="password-wrapper">
73          <input name="password" type="password" placeholder="Password">
74          <span class="eye"></span>
75        </div>
76
77        <div class="g-recaptcha" data-sitekey="6LeEiicrAAAAIWTss-ydcJ0-hhwh0Kj5GjKrxt"></div>
78        <button type="submit" class="login-btn">Log In</button>
79      </form>
80    </div>
81  </div>
82
83  <script>
84    function validateCaptcha() {
85      const response = grecaptcha.getResponse();
86      if (response.length === 0) {
87        alert('Please verify you are not a robot.');
88        return false; // Prevent form submission
89      }
90      return true; // Allow form submission
91    }
92  </script>
93
94 </body>
95 </html>

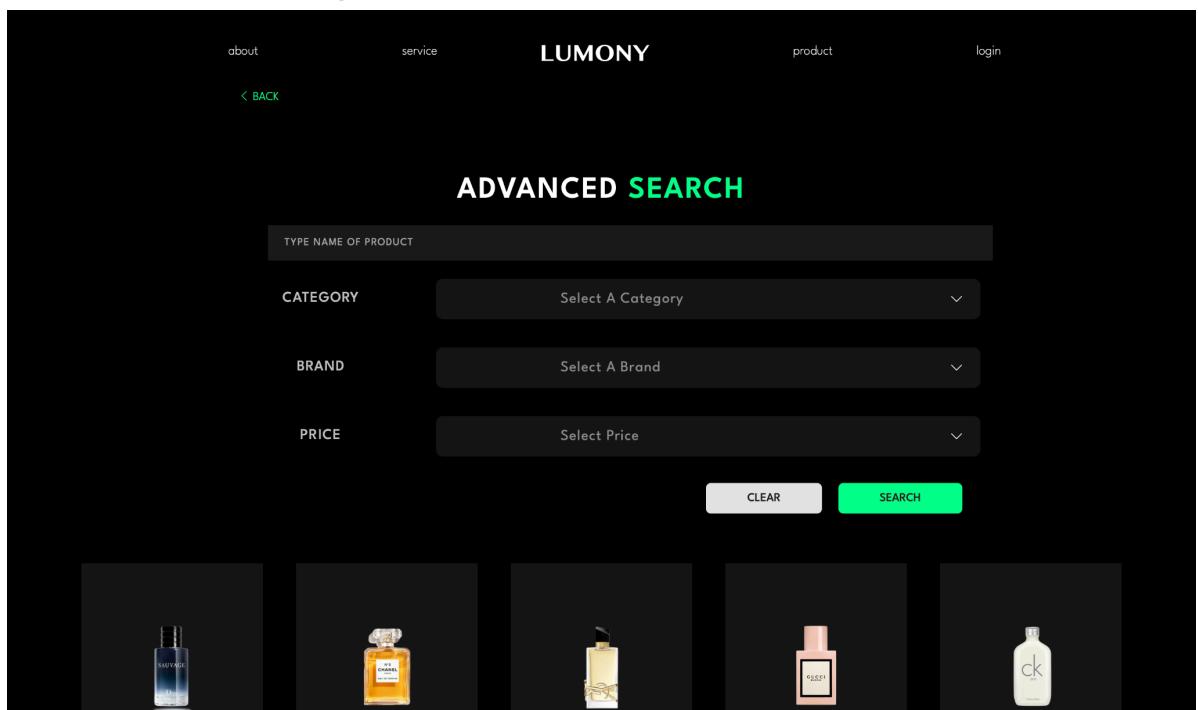
```

Search page for searching product

Product Page



Advanced Search Page



Detail: This product page displays all available products and allows users to search by product name using the search bar. By clicking "Advanced Search," users

are taken to a page where they can refine their search using filters including Product Name, Category, Brand, and Price. Users can input their criteria and use the "Search" button to display matching products or click "Clear" to reset the form. The filtered results are dynamically shown below based on the selected search options.

Code: product.html

```

1   <!DOCTYPE html>
2   <html lang="en">
3
4   <head>
5       <meta charset="UTF-8">
6       <meta http-equiv="X-UA-Compatible" content="IE=edge">
7       <meta name="viewport" content="width=device-width, initial-scale=1.0">
8       <title>Lumony | Product-Page</title>
9       <link rel="stylesheet" href="/css/bootstrap.css">
10      <link rel="stylesheet" href="/css/bootstrap.min.css">
11      <link rel="stylesheet" href="/css/bootstrap-grid.css">
12      <link rel="stylesheet" href="/css/mycss.css">
13      <script type="text/javascript" src="/js/bootstrap.bundle.min.js"></script>
14      <script type="text/javascript" src="/js/bootstrap.bundle.js"></script>
15      <script type="text/javascript" src="/js/bootstrap.js"></script>
16      <link rel="shortcut icon" type="image/jpg" href="/img/logo.png" />
17      <link rel="stylesheet" href="/dist/main.css" />
18      <script src="/dist/main.js"></script>
19      <link rel="preconnect" href="https://fonts.googleapis.com">
20      <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
21      <link href="https://fonts.googleapis.com/css2?family=League+Spartan:wght@100..900&display=swap" rel="stylesheet">
22      <!-- (Wenku) -->
23      <link href="https://fonts.googleapis.com/css2?family=Spartan:wght@400;700&display=swap" rel="stylesheet">
24      <link href="https://fonts.googleapis.com/css2?family=Reem+Kufi:wght@400;700&display=swap" rel="stylesheet">
25      <!-- (Wenicon) -->
26      <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.10.5/font/bootstrap-icons.css">
27  </head>
28
29 <body style="background: #black;" class="league-spartan100 text-white">
30     <!-- nav -->
31     <section class="sticky-top">
32         <nav class="navbar navbar-expand-lg container">
33             <div class="container-fluid">
34
35                 <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
36                     data-bs-target="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup" aria-expanded="false"
37                     aria-label="Toggle navigation">
38                     <span class="navbar-toggler-icon"></span>
39                 </button>
40
41                 <ul class="nav-item mt-3">
42                     <a class="nav-link zoom" href="#team">about </a>
43                 </ul>
44
45                 <ul class="nav-item mt-3">
46                     <a class="nav-link zoom" href="/promotion">service</a>
47                 </ul>
48
49                 <a class="navbar-brand zoom" href="/">
50                     
51                 </a>
52
53                 <ul class="nav-item mt-3">
54                     <a class="nav-link zoom" href="#product" target="_blank">product</a>
55                 </ul>
56
57                 <ul class="nav-item mt-3">
58                     <a class="nav-link zoom" href="/login">login</a>
59                 </ul>
60
61             </div>
62         </nav>
63     </section>

```

```

65  <section class="text_on_pic">
66    
67    <div>
68      <p class="perfume">PERFUME COLLECTION</p>
69      <h1 style="font-size: 2rem; font-weight: 600; line-height: 1; letter-spacing: 1.5px;">
70        DEFINE <span style="color: #06ff87;">YOURSELF</span><br>
71        WITH EVERY SPRAY
72      </h1>
73      <a href="#" class="btn btn-light distance-top">
74        style="padding: 4px 30px 4px 30px;">Discover Now</a>
75    </div>
76  </section>
77
78  <section>
79    <div class="search-section">
80      <h1 class="search-title">SEARCH</h1>
81      <div class="search-bar">
82        <input type="text" id="productSearchInput" placeholder="TYPE NAME OF PRODUCT">
83        <i class="bi bi-search" id="searchIcon" style="cursor: pointer;"></i>
84      </div>
85      <div class="advance">
86        <a href="/advance-search" class="advance-search">ADVANCE SEARCH</a>
87        <i class="bi bi-chevron-down"></i>
88      </div>
89    </div>
90  </section>
91
92  <!-- Creates a product list section with a table where products will be dynamically displayed. -->
93  <section class="below-advance">
94    <div id="productListPage" class="py-4">
95      <div class="d-flex justify-content-between align-items-center mb-4">
96        <h2 style="margin: 0;">PRODUCT LIST</h2>
97      </div>
98      <table class="table table-dark table-hover">
99        <tbody id="productTable" class="grid"></tbody>
100     </table>
101   </div>
102 </section>
103
104 <footer>
105   <section class="slogan">
106     <div class="slogan-text">
107       <p>A<br><span>FRAGRANCE</span><br>AS UNIQUE<br>AS YOU.</p>
108       <div class="underline"></div>
109       
110     </div>
111
112     <div class="footer-nav">
113       <div>
114         <h4>NAVIGATION</h4>
115         <ul>
116           <li><a href="/index">HOME PAGE</a></li>
117           <li><a href="/product">PRODUCT</a></li>
118           <li><a href="/team">ABOUT US</a></li>
119           <li><a href="">ACCOUNT</a></li>
120           <li><a href="">SETTING</a></li>
121           <li><a href="">SERVICE</a></li>
122           <li><a href="">CART</a></li>
123         </ul>
124       </div>
125     </div>
126     <div>
127       <h4>LOGIN</h4>
128       <ul>
129         <li><a href="/admin_login">ADMIN</a></li>
130         <li><a href="/login">USER</a></li>
131       </ul>
132     </div>
133   </div>
134 </section>
135 </footer>
136

```

```

138 <script>
139   let allProducts = [] // global
140
141   async function fetchAndRenderProducts() {
142     try {
143       const response = await fetch('/getAllProductsUser');
144       allProducts = await response.json();
145       renderProductList(allProducts);
146     } catch (error) {
147       console.error('Error fetching products:', error);
148     }
149   }
150
151   // Trigger search on icon click
152   document.getElementById('searchIcon').addEventListener('click', function () {
153     const keyword = document.getElementById('productSearchInput').value.trim().toLowerCase();
154     const filtered = allProducts.filter(product =>
155       product.Product_name.toLowerCase().includes(keyword)
156     );
157     renderProductList(filtered);
158   });
159
160   // Also trigger search when pressing "Enter"
161   document.getElementById('productSearchInput').addEventListener('keypress', function (e) {
162     if (e.key === 'Enter') {
163       document.getElementById('searchIcon').click();
164     }
165   });
166
167   function renderProductList(products) {
168     const list = document.getElementById('productTable');
169     list.innerHTML = '';
170     products.forEach(element => {
171       const card = document.createElement('div');
172       card.classList.add('card2');
173       card.innerHTML =
174         `<div class="zoom">
175           <div class="img-box" onclick="window.location.href='/product_detail?Product_ID=${element.Product_ID}'">
176             
177           </div>
178           <div class="text-each-perfume">
179             <p class="brand">${element.Product_brand}</p>
180             <p class="name">${element.Product_name}</p>
181           </div>
182         </div>
183       `;
184       list.appendChild(card);
185     });
186   }
187
188   fetchAndRenderProducts();
189 </script>
190
191 </html>

```

Advance-search.html

```

1  <!DOCTYPE html>
2  <html lang="en">
3    <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>Lumony | Product-Page</title>
8      <link rel="stylesheet" href="/css/bootstrap.css">
9      <link rel="stylesheet" href="/css/bootstrap.min.css">
10     <link rel="stylesheet" href="/css/bootstrap-grid.css">
11     <link rel="stylesheet" href="/css/mycss.css">
12     <script type="text/javascript" src="/js/bootstrap.bundle.min.js"></script>
13     <script type="text/javascript" src="/js/bootstrap.bundle.js"></script>
14     <script type="text/javascript" src="/js/bootstrap.js"></script>
15     <link rel="shortcut icon" type="image/png" href="/img/logo.png" />
16     <link rel="stylesheet" href="/dist/main.css" />
17     <script src="/dist/main.js"></script>
18     <link rel="preconnect" href="https://fonts.googleapis.com">
19     <link rel="preload" href="https://fonts.googleapis.com/css2?family=League+Spartan:wght@100..900&display=swap" rel="stylesheet">
20     <link href="https://fonts.googleapis.com/css2?family=League+Spartan:wght@100..900&display=swap" rel="stylesheet">
21     <link href="https://fonts.googleapis.com/css2?family=Spartan:wght@400..700&display=swap" rel="stylesheet">
22     <link href="https://fonts.googleapis.com/css2?family=Reem+Kufi:wght@400..700&display=swap" rel="stylesheet">
23     <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.10.5/font/bootstrap-icons.css">
24   </head>

```

```

26 <body style="background: #black;" class="league-spartan100 text-white">
27     <section class="sticky-top">
28         <nav class="navbar navbar-expand-lg container">
29             <div class="container-fluid">
30                 <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
31                     data-bs-target="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup" aria-expanded="false"
32                     aria-label="Toggle navigation">
33                     <span class="navbar-toggler-icon"></span>
34                 </button>
35
36                 <ul class="nav-item mt-3">
37                     <a class="nav-link zoom" href="#">team>about</a>
38                 </ul>
39
40                 <ul class="nav-item mt-3">
41                     <a class="nav-link zoom" href="#">promotion>service</a>
42                 </ul>
43
44                 <a class="navbar-brand zoom" href="/index">
45                     
46                 </a>
47
48                 <ul class="nav-item mt-3">
49                     <a class="nav-link zoom" href="#">product" target="_blank">product</a>
50                 </ul>
51
52                 <ul class="nav-item mt-3">
53                     <a class="nav-link zoom" href="#">login>login</a>
54                 </ul>
55
56             </div>
57         </nav>
58     </section>
59
60     <!-- top text -->
61     <section>
62
63         <div class="back-button">
64             <i class="bi bi-chevron-left"> </i>
65             <a href="#">/product>BACK</a>
66         </div>
67
68         <div class="advance-section">
69             <h1 class="search-title">ADVANCED <span>SEARCH</span></h1>
70             <div class="search-bar">
71                 <input type="text" id="productNameInput" placeholder="TYPE NAME OF PRODUCT">
72             </div>
73         </div>
74     </section>
75
76     <section style="margin-bottom: 180px;">
77         <!-- CATEGORY -->
78         <div class="input-box">
79             <div class="label">CATEGORY</div>
80             <select id="typeSelect" name="type">
81                 <option value="">Select A Category</option>
82                 <option value="Men">Men</option>
83                 <option value="Women">Women</option>
84                 <option value="Unisex">Unisex</option>
85             </select>
86         </div>
87
88         <!-- BRAND -->
89         <div class="input-box">
90             <div class="label">BRAND</div>
91             <select id="brandSelect" name="brand">
92                 <option value="">Select A Brand</option>
93                 <option value="Burberry">BURBERRY</option>
94                 <option value="Calvin Klein">CALVIN KLEIN</option>
95                 <option value="Chanel">CHANEL</option>
96                 <option value="Dior">DIOR</option>
97                 <option value="Giorgio Armani">GIORGIO ARMANI</option>
98                 <option value="Gucci">GUCCI</option>
99                 <option value="Jo Malone">JO MALONE</option>
100                <option value="Tom Ford">TOM FORD</option>
101                <option value="Versace">VERSACE</option>
102                <option value="Yves Saint Laurent">YSL</option>
103            </select>
104        </div>

```

```

105      <!-- PRICE -->
106      <div class="input-box">
107          <div class="label">PRICE</div>
108          <select id="priceSelect" name="price">
109              <option value="">Select Price</option>
110              <option value="2000-3000">2,000-3,000 BAHT</option>
111              <option value="3001-4000">3,001-4,000 BAHT</option>
112              <option value="4001-5000">4,001-5,000 BAHT</option>
113              <option value="5001-6000">5,001-6,000 BAHT</option>
114              <option value="6001-7000">6,001-7,000 BAHT</option>
115          </select>
116      </div>
117
118      <!-- FILTER BUTTONS -->
119      <div class="filter-buttons">
120          <button type="reset" class="clear-btn">CLEAR</button>
121          <button type="submit" class="search-btn" id="searchBtn">SEARCH</button>
122      </div>
123
124
125      <div id="productTable" class="grid" style="margin: 40px 20px;">
126          <!-- Products will show up here -->
127      </div>
128
129  </section>
130
131 </body>

```



```

133  <footer>
134      <section class="slogan">
135          <div class="slogan-text">
136              <p>A<br><span>FRAGRANCE</span><br>AS UNIQUE<br>AS YOU.</p>
137              <div class="underline"></div>
138              
139          </div>
140
141      <div class="footer-nav">
142          <div>
143              <h4>NAVIGATION</h4>
144              <ul>
145                  <li><a href="/index">HOME PAGE</a></li>
146                  <li><a href="/product">PRODUCT</a></li>
147                  <li><a href="/team">ABOUT US</a></li>
148                  <li><a href="#">ACCOUNT</a></li>
149                  <li><a href="#">SETTING</a></li>
150                  <li><a href="#">SERVICE</a></li>
151                  <li><a href="#">CART</a></li>
152              </ul>
153          </div>
154
155          <div>
156              <h4>LOGIN</h4>
157              <ul>
158                  <li><a href="/admin_login">ADMIN</a></li>
159                  <li><a href="/login">USER</a></li>
160              </ul>
161          </div>
162      </div>
163
164  </section>
165 </footer>

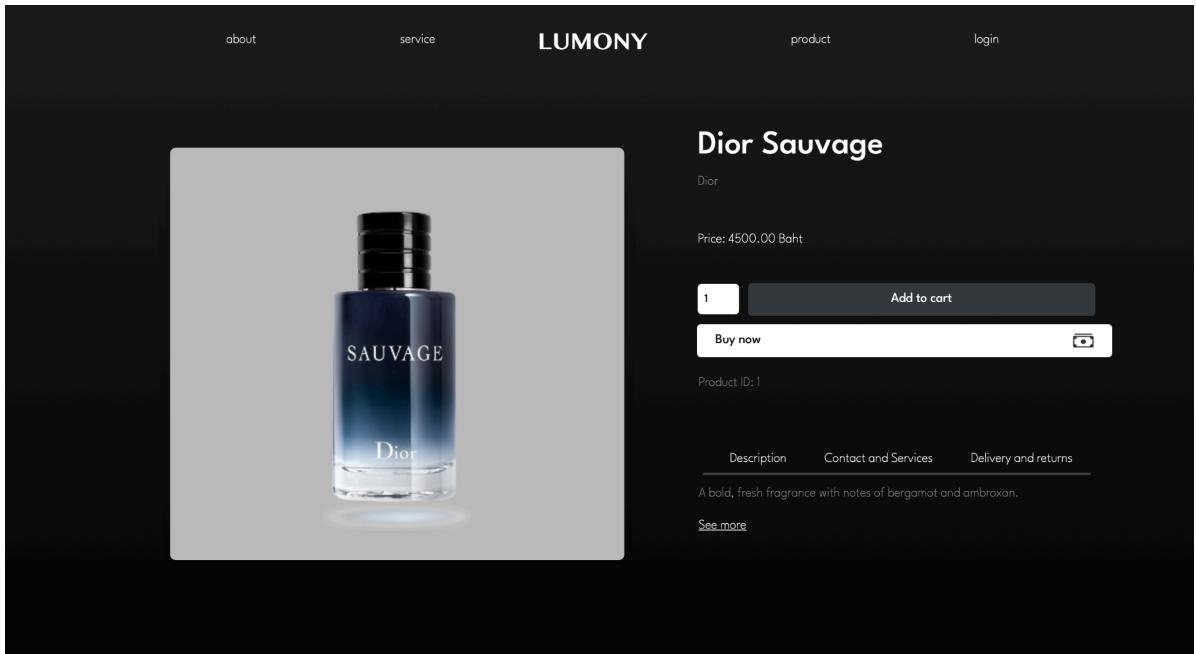
```

```

167 <script>
168
169     // Fetches all products from the server and returns them as a JSON array.
170     let allProducts = [];
171     async function fetchProducts() {
172         try {
173             const response = await fetch('/getAllProductsUser');
174             const products = await response.json();
175             return products;
176         } catch (error) {
177             console.error('Error fetching products:', error);
178             return [];
179         }
180     }
181
182     // Renders a list of products as image cards inside the HTML element with ID productTable.
183     function renderProductList(products) {
184         const list = document.getElementById('productTable');
185         list.innerHTML = '';
186         products.forEach(element => {
187             const card = document.createElement('div');
188             card.classList.add('card2');
189             card.innerHTML =
190                 `<div class="img-box zoom" onclick="window.location.href='/product_detail?Product_ID=${element.Product_ID}'" style=
191                 |   
192                 |>
193                 <div class="text-each-perfume">
194                     <p class="brand">${element.Product_brand}</p>
195                     <p class="name">${element.Product_name}</p>
196                 </div>
197             `;
198             list.appendChild(card);
199         });
200     }
201
202     // Fetches all products and displays them by calling the render function.
203     async function init() {
204         allProducts = await fetchProducts();
205         renderProductList(allProducts);
206     }
207
208
209     // Filters products based on search criteria and updates the displayed product list.
210     document.getElementById('searchBtn').addEventListener('click', async function () {
211         const name = document.getElementById('productNameInput').value.trim().toLowerCase();
212         const type = document.getElementById('typeSelect').value.trim();
213         const brand = document.getElementById('brandSelect').value.trim();
214         const price = document.getElementById('priceSelect').value.trim();
215         const filtered = allProducts.filter(product => {
216             const matchesName = !name || product.Product_name.toLowerCase().includes(name);
217             const matchesType = !type || product.Product_type === type;
218             const matchesBrand = !brand || product.Product_brand === brand;
219             const matchesPrice = !price || product.priceRange === price;
220
221             return matchesName && matchesType && matchesBrand && matchesPrice;
222         });
223         renderProductList(filtered);
224     });
225
226     // Clears search fields, resets the product list, and fetches products again with empty filters.
227     document.querySelector('.clear-btn').addEventListener('click', function () {
228         document.getElementById('productNameInput').value = '';
229         document.getElementById('typeSelect').value = '';
230         document.getElementById('brandSelect').value = '';
231         document.getElementById('priceSelect').value = '';
232         renderProductList(allProducts); // Reset to show all products
233         const params = new URLSearchParams();
234         if (name) params.append('name', name);
235         if (artist) params.append('artist', artist);
236         if (label) params.append('label', label);
237
238         fetchProducts(params.toString());
239     });
240     init();
241
242 </script>
243
244 </html>

```

Detail page for each product result



Detail: The Product Detail Page displays key information about the selected fragrance, including the name and brand, product type, quantity, and price. Additional details such as the Product ID and fragrance description are provided.

Code: product_detail.html

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>Lumony | Product-Page</title>
8      <link rel="stylesheet" href="/css/bootstrap.css">
9      <link rel="stylesheet" href="/css/bootstrap.min.css">
10     <link rel="stylesheet" href="/css/bootstrap-grid.css">
11     <link rel="stylesheet" href="/css/mycss.css">
12     <script type="text/javascript" src="/js/bootstrap.bundle.min.js"></script>
13     <script type="text/javascript" src="/js/bootstrap.bundle.js"></script>
14     <script type="text/javascript" src="/js/bootstrap.js"></script>
15     <link rel="shortcut icon" type="image/png" href="/img/logo.png" />
16     <link rel="stylesheet" href="/dist/main.css" />
17     <script src="/dist/main.js"></script>
18     <link rel="preconnect" href="https://fonts.googleapis.com">
19     <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
20     <link href="https://fonts.googleapis.com/css2?family=League+Spartan:wght@100..900&display=swap" rel="stylesheet">
21     <link href="https://fonts.googleapis.com/css2?family=Spartan:wght@400;700&display=swap" rel="stylesheet">
22     <link href="https://fonts.googleapis.com/css2?family=Reem+Kufi:wght@400;700&display=swap" rel="stylesheet">
23     <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.10.5/font/bootstrap-icons.css">
24
25 </head>
```

```

27 <body class="bgcolor2 league-spartan100 text-white ">
28     <!-- nav -->
29     <section class="sticky-top">
30         <nav class="navbar navbar-expand-lg container">
31             <div class="container-fluid">
32
33             <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
34                 data-bs-target="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup" aria-expanded="false"
35                 aria-label="Toggle navigation">
36                 <span class="navbar-toggler-icon"></span>
37             </button>
38
39             <ul class="nav-item mt-3">
40                 <a class="nav-link zoom" href="#">team>about</a>
41             </ul>
42
43             <ul class="nav-item mt-3">
44                 <a class="nav-link zoom" href="#">/promotion>service</a>
45             </ul>
46
47             <a class="navbar-brand zoom" href="#">/index>
48                 
49             </a>
50
51             <ul class="nav-item mt-3">
52                 <a class="nav-link zoom" href="#">product" target="_blank">product</a>
53             </ul>
54
55             <ul class="nav-item mt-3">
56                 <a class="nav-link zoom" href="#">/login>login</a>
57             </ul>
58
59         </div>
60     </nav>
61 </section>
62
63     <!-- product detail -->
64     <section >
65         <div class="Product_text">
66             <!-- SPACE -->
67         </div>
68         <div class="Product_pretext">
69             <!-- SPACE -->
70         </div>
71         <div class="product-box">
72             <!-- SPACE -->
73         </div>
74         <div class="main_price">
75             <!-- SPACE -->
76         </div>
77     </section>
78
79     <section>
80         <div class="quan">
81             <select name="quantity" id="quantity">
82                 <option value="1">1</option>
83                 <option value="2">2</option>
84                 <option value="3">3</option>
85                 <option value="4">4</option>
86                 <option value="5">5</option>
87             </select>
88             <button class="add_cart">
89                 Add to cart
90                 <!-- SPACE -->
91             </button>
92         </div>
93
94         <button class="buy_now">Buy now
95             <span class="buy_now_split"></span>
96         </button>
97
98         <p class="product_id">
99             <!-- SPACE -->
100            </p>
101     </section>

```

```

103    <!-- description tab -->
104    <section>
105      <div class="d_product">
106        <div class="p-2">Description</div>
107        <div class="p-2">Contact and Services</div>
108        <div class="p-2">Delivery and returns</div>
109      </div>
110      <hr style="width: 470px; height: 3px; border: 1px solid #ccc; margin-left: 845px;">
111      <div class="description">
112        <div class="description_detail">
113          An explosive fragrance clashing powerful iris with spicy black pepper and deep <br>
114          vanilla wood. Givenchy celebrates the free spirit and charismatic charm of the
115        </div>
116      </div>
117      <div class="see_more_S">
118        <u class="see_more">See more</u>
119      </div>
120    </section>
121  </body>
122
123  <script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>

```

```

125  <script>
126    // Fetches product details by ID from the server and displays them on the product detail page.
127    async function loadProductDetail() {
128      const urlParams = new URLSearchParams(window.location.search);
129      const productId = urlParams.get('Product_ID');
130      console.log("Product_ID:", productId); // Debug
131      if (!productId) {
132        console.error("No Product_ID found");
133        return;
134      }
135      try {
136        const response = await axios.get('http://localhost:8083/api/getProductDetail', {
137          params: { Product_ID: productId }
138        });
139        const product = response.data;
140        console.log("Product Data:", product); // Debug API response
141        // Inject data into HTML
142        document.querySelector('.Product_text').innerHTML = `<h1>${product.Product_name}</h1>`;
143        document.querySelector('.Product_pretext').innerHTML = `<p>${product.Product_brand}</p>`;
144        document.querySelector('.product-box').innerHTML =
145          `
146          `;
147        document.querySelector('.main_price').innerHTML = `<p>Price: ${product.Product_price} Baht</p>`;
148        document.querySelector('.description_detail').innerHTML = product.Product_description;
149        document.querySelector('.product_id').innerHTML = `Product ID: ${product.Product_ID}`;
150      } catch (error) {
151        console.error("Error loading product detail:", error);
152      }
153    }
154    loadProductDetail();
155  </script>
156 </html>

```

Product Management page for administrators

LUMONY

Dashboard

Products

Orders

Customers

Employees

Analytics

Help

SEARCH 

Admin Profile

PRODUCT LIST

Add New Product 

Update Product 

Image	Product Name	Product ID	Price	Stock	Type	Status	Action
	Dior Sauvage	1	฿4500.00	100 pcs	Men	Active 	Delete 
	Chanel No.5	2	฿5200.00	100 pcs	Women	Active 	Delete 
	YSL Libre	3	฿4800.00	100 pcs	Women	Active 	Delete 
	Gucci Bloom	4	฿4900.00	100 pcs	Women	Active 	Delete 
	CK One	5	฿3200.00	100 pcs	Unisex	Active 	Delete 
	Armani My Way	6	฿4700.00	100 pcs	Women	Active 	Delete 
	Burberry Her	7	฿4300.00	100 pcs	Women	Active 	Delete 
	Versace Dylan Blue	8	฿4100.00	100 pcs	Men	Active 	Delete 

LUMONY

Dashboard

Products

Orders

Customers

Employees

Analytics

Help

SEARCH 

Admin Profile

< BACK

PRODUCT REGISTRATION

Choose File  no file selected

Product Name  Brand 

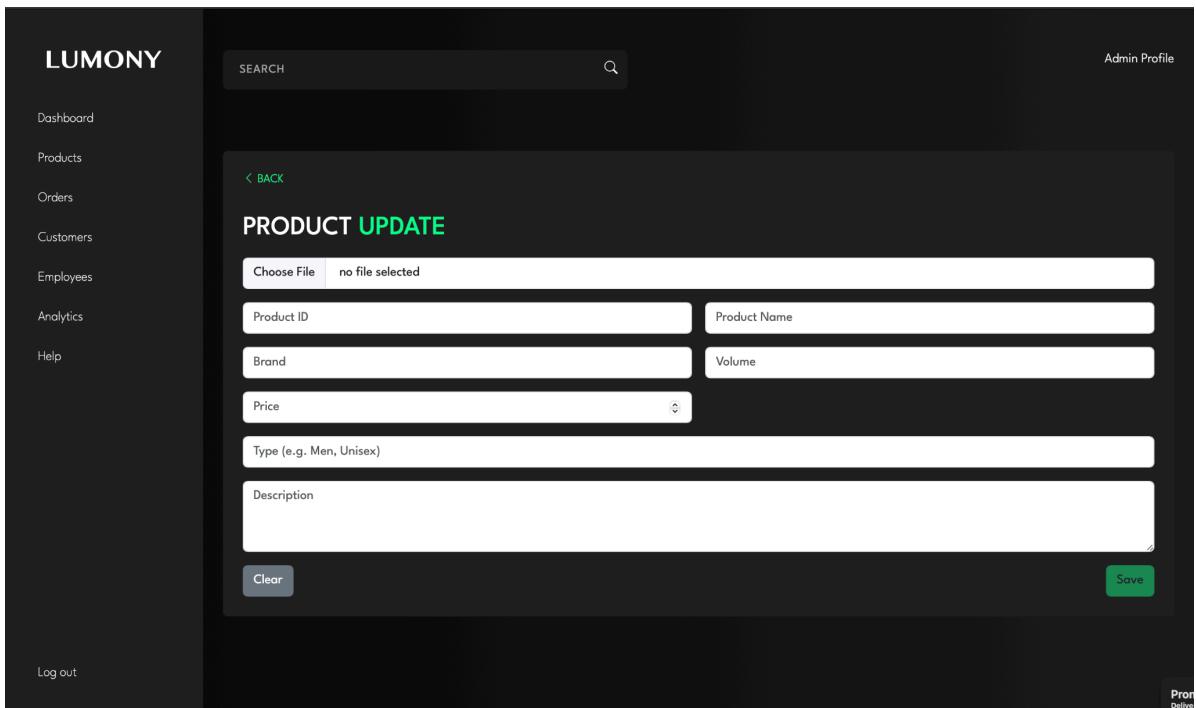
Volume  Price 

Type (e.g. Men, Unisex) 

Description 

Log out



Detail: The Product Management Page is designed for administrators to efficiently add, modify, or delete products in the system.

- **Adding a Product:** Click on the "Add New Product" button which redirects to the Product Registration Page. This page includes fields for Product Name, Brand, Type, Volume, Price, Image, and Description, along with an auto-generated Product ID for tracking. Administrators can upload product images. The interface also features "Clear" and "Save" buttons for seamless updates.
- **Modifying a Product:** Click the "Update Product" button located in the top right corner to navigate to the Product Updating Page. Here, administrators can modify all fields listed on the Product Registration Page. By entering the Product ID, the product details will be retrieved, allowing edits to all fields except the Product ID itself.
- **Deleting a Product:** On the Product Management Page, click the "Delete" button next to the desired product.

Code: service_admin.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5     <meta charset="UTF-8">
6     <meta http-equiv="X-UA-Compatible" content="IE=edge">
7     <meta name="viewport" content="width=device-width, initial-scale=1.0">
8     <title>Lumony | Admin Service</title>
9     <link rel="stylesheet" href="/css/bootstrap.css">
10    <link rel="stylesheet" href="/css/bootstrap.min.css">
11    <link rel="stylesheet" href="/css/bootstrap-grid.css">
12    <link rel="stylesheet" href="/css/mycss.css">
13    <script type="text/javascript" src="/js/bootstrap.bundle.min.js"></script>
14    <script type="text/javascript" src="/js/bootstrap.bundle.js"></script>
15    <script type="text/javascript" src="/js/bootstrap.js"></script>
16    <link rel="shortcut icon" type="image/png" href="/img/logo.png" />
17    <link rel="stylesheet" href="/dist/main.css" />
18    <script src="/dist/main.js"></script>
19    <link rel="preconnect" href="https://fonts.googleapis.com">
20    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
21    <link href="https://fonts.googleapis.com/css2?family=League+Spartan:wght@100..900&display=swap" rel="stylesheet">
22    <link href="https://fonts.googleapis.com/css2?family=Spartan:wght@400;700&display=swap" rel="stylesheet">
23    <link href="https://fonts.googleapis.com/css2?family=Reem+Kufi:wght@400;700&display=swap" rel="stylesheet">
24    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.10.5/font/bootstrap-icons.css">
25    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet">
26
27
28 </head>
29
30 <body class="bgcolor text-white league-spartan100">
31     <div class="container-fluid">
32         <div class="row">
33             <!-- Sidebar -->
34             <nav class="col-md-2 d-none d-md-flex bg-sidenav sidebar p-4">
35                 <div style="margin: 0.7cm;">
36                     <a class="navbar-brand zoom" href="/index">
37                         
38                     </a>
39                 </div>
40
41                 <div>
42                     <ul class="nav flex-column">
43                         <li class="nav-item mb-2">
44                             <a class="nav-link text-white" href="#">Dashboard</a>
45                         </li>
46                         <li class="nav-item mb-2">
47                             <a class="nav-link text-white" href="#">Products</a>
48                         </li>
49                         <li class="nav-item mb-2">
50                             <a class="nav-link text-white" href="#">Orders</a>
51                         </li>
52                         <li class="nav-item mb-2">
53                             <a class="nav-link text-white" href="#">Customers</a>
54                         </li>
55                         <li class="nav-item mb-2">
56                             <a class="nav-link text-white" href="#">Employees</a>
57                         </li>
58                         <li class="nav-item mb-2">
59                             <a class="nav-link text-white" href="#">Analytics</a>
60                         </li>
61                         <li class="nav-item mb-2">
62                             <a class="nav-link text-white" href="#">Help</a>
63                         </li>
64                     </ul>
65                 </div>
66             </nav>
67         </div>
68     </div>
69
70     <div class="row justify-content-center">
71         <div class="col-12 col-md-8 col-lg-6 col-xl-4">
72             <div class="card border-0 shadow rounded-3 p-4">
73                 <div class="card-body">
74                     <div class="mb-3">
75                         <h4>Dashboard</h4>
76                         <p>Overall performance summary and key metrics at a glance.</p>
77                     </div>
78                     <div>
79                         <div class="row">
80                             <div class="col-4">
81                                 <div>
82                                     <h5>Revenue</h5>
83                                     <div>123.456.789.000</div>
84                                     <div>USD</div>
85                                 </div>
86                             </div>
87                             <div class="col-4">
88                                 <div>
89                                     <h5>Profit</h5>
90                                     <div>123.456.789.000</div>
91                                     <div>USD</div>
92                                 </div>
93                             </div>
94                             <div class="col-4">
95                                 <div>
96                                     <h5>Sales</h5>
97                                     <div>123.456.789.000</div>
98                                     <div>USD</div>
99                                 </div>
100                            </div>
101                        </div>
102                    </div>
103                </div>
104            </div>
105        </div>
106    </div>
107
```

```

67      <ul class="nav flex-column logout-link">
68          <li class="nav-item">
69              <a class="nav-link text-white" href="admin_login">Log out</a>
70          </li>
71      </ul>
72
73  </nav>
74
75  <!-- Main Content -->
76  <main class="col-md-10 ms-sm-auto px-md-4">
77      <!-- search bar - profile -->
78      <section style="margin: 30px 0; display: flex; justify-content:space-between;">
79          <div class="search-bar2">
80              <input class="search_bar_d" type="text" placeholder="SEARCH">
81              <i class="bi bi-search"></i>
82          </div>
83          <div>
84              <p id="adminInfo" style="font-weight: 300;" class="admininfo-pro">Admin Profile</p>
85          </div>
86      </section>
87
88  <!-- Admin Product List Page -->
89  <div id="productListPage" class="py-4">
90      <div class="d-flex justify-content-between align-items-center mb-4">
91          <h2 style="margin: 0;">PRODUCT LIST</h2>
92          <a href="/add_product">
93              <button style="text-align: center;" class="btn btn-outline-light">Add New Product</button>
94          </a>
95          <a href="/update_product">
96              <button style="text-align: center;" class="btn btn-outline-light">Update Product</button>
97          </a>
98      </div>
99      <table class="table table-dark table-hover">
100         <thead>
101             <tr>
102                 <th style="padding-left: 0.5cm;">Image</th>
103                 <th>Product Name</th>
104                 <th>Product ID</th>
105                 <th>Price</th>
106                 <th>Stock</th>
107                 <th>Type</th>
108                 <th>Status</th>
109                 <th>Action</th>
110             </tr>
111         </thead>
112         <tbody id="productTable">
113             </tbody>
114         </table>
115     </div>
116
117     </main>
118     </div>
119  </div>
120
121  </div>
122 </body>

```

```

124 <script>
125 // Fetches all products from the server and displays them using the render function.
126 async function fetchAndRenderProducts() {
127   try{
128     const response = await fetch('/getAllProducts');
129     const products = await response.json();
130     renderProductList(products);
131   }
132   catch(error) {
133     console.error('Error fetching products:', error);
134   }
135 }
136
137 // Renders each product as a table row with details and a delete button.
138 function renderProductList(products) {
139   const list = document.getElementById('productTable');
140   products.forEach(element => {
141     const tr = document.createElement('tr');
142     tr.innerHTML =
143       `<td></td>
144       <td>${element.Product_name}</td>
145       <td>${element.Product_ID}</td>
146       <td>${element.Product_price}</td>
147       <td>100 pcs</td> <!-- Placeholder stock -->
148       <td>${element.Product_type}</td>
149       <td><span class="btn btn-sm text-dark" style="background-color: #06ff87;">Active</span></td>
150       <td>
151         <button class="btn btn-sm btn-outline-danger" onclick="deleteProduct(${element.Product_ID})">Delete</button>
152       </td>
153     `;
154     list.appendChild(tr);
155   });
156 }
157
158 // Deletes a product after confirmation and reloads the page to refresh the product list.
159 window.deleteProduct = async function(productId) {
160   if (confirm('Are you sure you want to delete this product?')) {
161     try {
162       const response = await fetch(`/deleteProduct/${productId}`, { method: 'DELETE' });
163       const result = await response.text();
164       console.log('Delete result:', result);
165       alert('Deleted successfully!');
166       location.reload(); // Reload only after delete success
167     } catch (error) {
168       console.error('Error deleting product:', error);
169       alert('Failed to delete product.');
170     }
171   }
172 };
173 fetchAndRenderProducts();
174 </script>
175
176 </html>

```

add_product.html

```
1  <!DOCTYPE html>
2  <html lang="en">
3
4  <head>
5      <meta charset="UTF-8">
6      <meta http-equiv="X-UA-Compatible" content="IE=edge">
7      <meta name="viewport" content="width=device-width, initial-scale=1.0">
8      <title>Lumony | Add Product</title>
9      <link rel="stylesheet" href="/css/bootstrap.css">
10     <link rel="stylesheet" href="/css/bootstrap.min.css">
11     <link rel="stylesheet" href="/css/bootstrap-grid.css">
12     <link rel="stylesheet" href="/css/mycss.css">
13     <script type="text/javascript" src="/js/bootstrap.bundle.min.js"></script>
14     <script type="text/javascript" src="/js/bootstrap.bundle.js"></script>
15     <script type="text/javascript" src="/js/bootstrap.js"></script>
16     <link rel="shortcut icon" type="image/jpg" href="/img/logo.png" />
17     <link rel="stylesheet" href="/dist/main.css" />
18     <script src="/dist/main.js"></script>
19     <link rel="preconnect" href="https://fonts.googleapis.com">
20     <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
21     <link href="https://fonts.googleapis.com/css2?family=League+Spartan:wght@100..900&display=swap" rel="stylesheet">
22     <!-- (ໜົນຊີວຸນ) -->
23     <link href="https://fonts.googleapis.com/css2?family=Spartan:wght@wght@400;700&display=swap" rel="stylesheet">
24     <link href="https://fonts.googleapis.com/css2?family=Reem+Kufi:wght@400;700&display=swap" rel="stylesheet">
25     <!-- (ໜົນຊີວຸນ) -->
26     <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.10.5/font/bootstrap-icons.css">
27     <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet">
28
29
30 </head>
31
32 <body class="bgcolor text-white league-spartan100">
33     <div class="container-fluid">
34         <div class="row">
35             <!-- NAV -->
36             <nav class="col-md-2 d-none d-md-flex bg-sidenav sidebar p-4">
37                 <div style="margin: 0.7cm;">
38                     
39                 </div>
40
41             <!-- left side dashboard -->
42             <div>
43                 <ul class="nav flex-column">
44                     <li class="nav-item mb-2">
45                         <a class="nav-link text-white" href="#">Dashboard</a>
46                     </li>
47                     <li class="nav-item mb-2">
48                         <a class="nav-link text-white" href="#">Products</a>
49                     </li>
50                     <li class="nav-item mb-2">
51                         <a class="nav-link text-white" href="#">Orders</a>
52                     </li>
53                     <li class="nav-item mb-2">
54                         <a class="nav-link text-white" href="#">Customers</a>
55                     </li>
56                     <li class="nav-item mb-2">
57                         <a class="nav-link text-white" href="#">Employees</a>
58                     </li>
59                     <li class="nav-item mb-2">
60                         <a class="nav-link text-white" href="#">Analytics</a>
61                     </li>
62                     <li class="nav-item mb-2">
63                         <a class="nav-link text-white" href="#">Help</a>
64                     </li>
65                 </ul>
66             </div>
67
68             <ul class="nav flex-column logout-link">
69                 <li class="nav-item">
70                     <a class="nav-link text-white" href="admin_login">Log out</a>
71                 </li>
72             </ul>
73         </nav>
```

```

75      <!-- Main Content -->
76      <main class="col-md-10 ms-sm-auto px-md-4">
77
78          <!-- search bar - profile -->
79          <section style="margin: 30px 0; display: flex; justify-content:space-between;">
80              <div class="search-bar2">
81                  <input class="search_bar_d" type="text" placeholder="SEARCH">
82                  <i class="bi bi-search"></i>
83              </div>
84          <div>
85              <p id="adminInfo" style="font-weight: 300;" class="admininfo-pro">Admin Profile</p>
86          </div>
87      </section>
88
89      <!-- Admin Add Product Page -->
90      <div id="productFormPage" class=" p-4 rounded" action="/addNewProduct" method="POST" enctype="multipart/form-data">
91          <form class="bg-sidenav p-4 rounded" action="/addNewProduct" method="POST" enctype="multipart/form-data">
92              <div class="back-button2">
93                  <i class="bi bi-chevron-left"></i>
94                  <a href="/service_admin"> BACK </a>
95              </div>
96              <h2 style="margin-top: 0.8cm;" class="h2 mb-3">PRODUCT <span style="color: #06ff87;">REGISTRATION<span></h2>
97              <div class="row g-3">
98                  <div class="col-md-12">
99                      <input name="productImage" type="file" class="form-control" accept="image/*" onchange="previewImage(event)">
100                     <img id="imagePreview" class="image-preview d-none" alt="Preview" />
101                 </div>
102                 <div class="col-md-6"><input name="productName" class="form-control" placeholder="Product Name" required></div>
103                 <div class="col-md-6"><input name="brand" class="form-control" placeholder="Brand" required></div>
104                 <div class="col-md-6"><input name="volume" class="form-control" placeholder="Volume" required></div>
105                 <div class="col-md-6"><input name="price" class="form-control" type="number" placeholder="Price" required></div>
106                 <div class="col-md-12"><input name="type" class="form-control" placeholder="Type (e.g. Men, Unisex)" required></div>
107                 <div class="col-md-12"><textarea name="description" class="form-control" placeholder="Description" rows="3" required></div>
108             </div>
109             <div class="d-flex justify-content-between mt-3">
110                 <button type="reset" class="btn btn-secondary">Clear</button>
111                 <button type="submit" class="btn btn-success text-dark">Save</button>
112             </div>
113         </form>
114     </div>
115     </main>
116 </div>
117 </body>
118 </html>

```

Update_product.html

```

1  <!DOCTYPE html>
2  <html lang="en">
3
4      <head>
5          <meta charset="UTF-8">
6          <meta http-equiv="X-UA-Compatible" content="IE=edge">
7          <meta name="viewport" content="width=device-width, initial-scale=1.0">
8          <title>Lumony | Update Product</title>
9          <link rel="stylesheet" href="/css/bootstrap.css">
10         <link rel="stylesheet" href="/css/bootstrap.min.css">
11         <link rel="stylesheet" href="/css/bootstrap-grid.css">
12         <link rel="stylesheet" href="/css/mycss.css">
13         <script type="text/javascript" src="/js/bootstrap.bundle.min.js"></script>
14         <script type="text/javascript" src="/js/bootstrap.bundle.js"></script>
15         <script type="text/javascript" src="/js/bootstrap.js"></script>
16         <link rel="shortcut icon" type="image/jpg" href="/img/logo.png" />
17         <link rel="stylesheet" href="/dist/main.css" />
18         <script src="/dist/main.js"></script>
19         <link rel="preconnect" href="https://fonts.googleapis.com">
20         <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
21         <link href="https://fonts.googleapis.com/css2?family=League+Spartan:wght@100..900&display=swap" rel="stylesheet">
22         <!-- iWaiicon -->
23         <link href="https://fonts.googleapis.com/css2?family=Spartan:wght@400;700&display=swap" rel="stylesheet">
24         <link href="https://fonts.googleapis.com/css2?family=Reem+Kufi:wght@400;700&display=swap" rel="stylesheet">
25         <!-- iWaiicon -->
26         <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.10.5/font/bootstrap-icons.css">
27         <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet">
28
29
30     </head>

```

```

32 <body class="bgcolor text-white league-spartan100">
33   <div class="container-fluid">
34     <div class="row">
35       | <!-- Sidebar -->
36       <nav class="col-md-2 d-none d-md-flex bg-sidenav sidebar p-4">
37         <div style="margin: 0.7cm;">
38           | | 
39         </div>
40
41       <div>
42         | | <ul class="nav flex-column">
43           | | | <li class="nav-item mb-2">
44             | | | | <a class="nav-link text-white" href="#">DashboardProductsOrdersCustomersEmployeesAnalyticsHelpadmin\_loginLog outAdmin Profile</p>
83           </div>
84         </section>
85
86       <!-- Admin Add Product Page -->
87       <div id="productFormPage" class=" py-4">
88         <form id="updateProductForm" class="bg-sidenav p-4 rounded" action="/updateProduct" method="POST" enctype="multipart/form-data">
89           <div class="back-button2">
90             | | <i class="bi bi-chevron-left"></i>
91             | | <a href="#">service\_admin BACKClear</button>
110            <button type="submit" class="btn btn-success text-dark">Save</button>
111          </div>
112        </div>
113      </div>
114    </main>
115  </div>
116 </body>

```

```

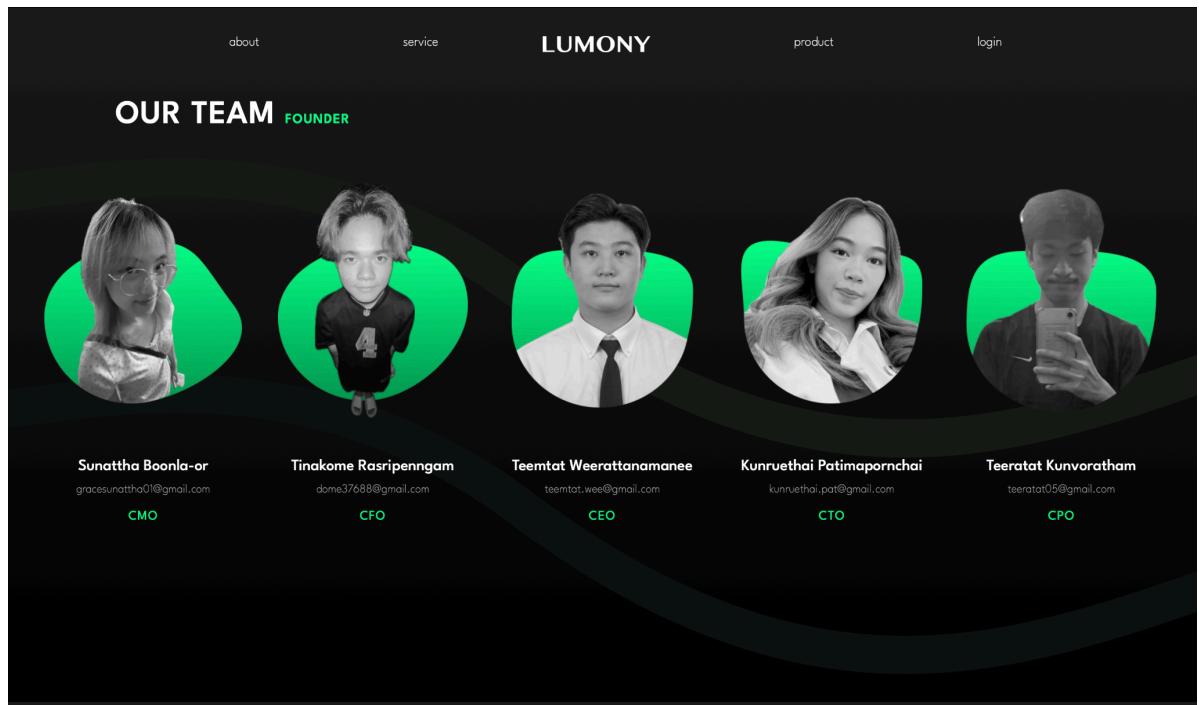
120 <script>
121     // Auto-fill product details when typing product ID
122     document.querySelector('input[name="product_ID"]').addEventListener('blur', function () {
123         const productId = this.value.trim();
124         if (productId) {
125             fetch('/api/getProduct/${productId}')
126                 .then(response => {
127                     if (!response.ok) {
128                         throw new Error('Product not found');
129                     }
130                     return response.json();
131                 })
132                 .then(product => {
133                     document.querySelector('input[name="productName"]').value = product.Product_name || '';
134                     document.querySelector('input[name="brand"]').value = product.Product_brand || '';
135                     document.querySelector('input[name="volume"]').value = product.Product_volume || '';
136                     document.querySelector('input[name="price"]').value = product.Product_price || '';
137                     document.querySelector('input[name="type"]').value = product.Product_type || '';
138                     document.querySelector('textarea[name="description"]').value = product.Product_description || '';
139
140                     if (product.Product_image) {
141                         const imagePreview = document.getElementById('imagePreview');
142                         imagePreview.src = `data:image/jpeg;base64,${product.Product_image}`;
143                         imagePreview.classList.remove('d-none');
144                         document.getElementById('oldImage').value = product.Product_image; // Save old image
145                     }
146                 })
147                 .catch(err => {
148                     alert(err.message);
149                     console.log(err);
150                 });
151             });
152         });
153
154     // Show preview when user chooses a new image
155     function previewImage(event) {
156         const file = event.target.files[0];
157         const preview = document.getElementById('imagePreview');
158         if (file) {
159             const reader = new FileReader();
160             reader.onload = function (e) {
161                 preview.src = e.target.result;
162                 preview.classList.remove('d-none');
163             };
164             reader.readAsDataURL(file);
165         }
166     }
167
168     // Handle form submit manually
169     document.getElementById('updateProductForm').addEventListener('submit', async function (event) {
170         event.preventDefault(); // Prevent normal form POST
171
172         const form = event.target;
173         const formData = new FormData(form);
174
175         const productData = {
176             product_ID: formData.get('product_ID'),
177             productName: formData.get('productName'),
178             brand: formData.get('brand'),
179             volume: formData.get('volume'),
180             price: formData.get('price'),
181             type: formData.get('type'),
182             description: formData.get('description'),
183             imageUrl: '', // Default
184             oldImage: formData.get('oldImage')
185         };

```

```

187     const imageData = formData.get('productImage');
188     if (imageFile && imageFile.size > 0) {
189       const reader = new FileReader();
190       reader.onload = async function (e) {
191         productData.imageUrl = e.target.result.split(',')[1]; // Only Base64 part
192         await updateProduct(productData);
193       };
194       reader.readAsDataURL(imageFile);
195     } else {
196       // No new image, use old
197       await updateProduct(productData);
198     }
199   );
200
201   // Send the updated product data to the server
202   async function updateProduct(productData) {
203     try {
204       const response = await fetch('http://localhost:8083/api/updateProduct', {
205         method: 'PUT',
206         headers: { 'Content-Type': 'application/json' },
207         body: JSON.stringify(productData)
208       });
209
210       const result = await response.json();
211       if (result.success) {
212         alert('Product updated successfully!');
213         window.location.href = '/service_admin'; // Redirect after success
214       } else {
215         alert('Failed to update product.');
216       }
217     } catch (error) {
218       console.error('Error updating product:', error);
219       alert('Error updating product.');
220     }
221   }
222 
```

Team Page



The Team Page can be accessed via the "About Us" link in the footer or the "About" option in the navigation bar. This page provides detailed information about our team members, highlighting their roles and contributions within Lumony.

Code: team.html

```
1  <!DOCTYPE html>
2  <html lang="en">
3
4  <head>
5      <meta charset="UTF-8">
6      <meta http-equiv="X-UA-Compatible" content="IE=edge">
7      <meta name="viewport" content="width=device-width, initial-scale=1.0">
8      <title>Lumony | Home-Page</title>
9      <link rel="stylesheet" href="/css/bootstrap.css">
10     <link rel="stylesheet" href="/css/bootstrap.min.css">
11     <link rel="stylesheet" href="/css/bootstrap-grid.css">
12     <link rel="stylesheet" href="/css/mycss.css">
13     <script type="text/javascript" src="/js/bootstrap.bundle.min.js"></script>
14     <script type="text/javascript" src="/js/bootstrap.bundle.js"></script>
15     <script type="text/javascript" src="/js/bootstrap.js"></script>
16     <link rel="shortcut icon" type="image/jpg" href="/img/logo.png"/>
17     <link rel="stylesheet" href="/dist/main.css" />
18     <script src="/dist/main.js"></script>
19     <link rel="preconnect" href="https://fonts.googleapis.com">
20     <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
21     <link href="https://fonts.googleapis.com/css2?family=League+Spartan:wght@100..900&display=swap" rel="stylesheet">
22 </head>
```

```

24 <body class="bgcolor-forTEAM league-spartan100 text-white">
25   <div class="sticky-top">
26     <nav class="navbar navbar-expand-lg container">
27       <div class="container-fluid">
28
29         <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup" aria-expanded="false" aria-label="Toggle navigation">
30           <span class="navbar-toggler-icon"></span>
31         </button>
32
33         <ul class="nav-item mt-3">
34           <a class="nav-link zoom" href="#">team>about</a>
35         </ul>
36
37         <ul class="nav-item mt-3">
38           <a class="nav-link zoom" href="#">/promotion>service</a>
39         </ul>
40
41         <a class="navbar-brand zoom" href="#">/index>
42           /img/logo.png" alt="" width="auto" height="20" class="d-inline-block align-text-top">
43         </a>
44
45         <ul class="nav-item mt-3">
46           <a class="nav-link zoom" href="#">product" target="_blank">product</a>
47         </ul>
48
49         <ul class="nav-item mt-3">
50           <a class="nav-link zoom" href="#">/login>login</a>
51         </ul>
52
53       </div>
54     </nav>
55   </div>
56
57   <!-- top text -->
58   <section>
59     <div class="our-team">
60       <h1>OUR TEAM <span>FOUNDER</span><h1>
61     </div>
62   </section>
63
64   <section style="position: relative;">
65     <!-- SVG Wave Background -->
66     <div style="position: absolute; width: 100%; height: 100%; display: flex; justify-content: center; align-items: center; color: #white; text-align: center; z-index: 1;">
67       <svg
68         viewBox="270 0 1700 790"
69         xmlns="http://www.w3.org/2000/svg"
70         style="width: 100%; height: auto; preserveAspectRatio='xMidYMid meet'"
71       >
72         <g transform="matrix(1,0,0,1,200,-190)">
73           <path
74             d="M0,238.962C0,330 400,100 800,400 C1600,1000 1900,100 2800,500"
75             style="fill:none;stroke:#rgb(22, 26, 24);stroke-width:55px;">
76           </path>
77         </g>
78       <g transform="matrix(1,0,0,1,200,120)">
79         <path
80           d="M0,238.962C0,330 400,100 800,400 C1600,1000 1900,100 2800,500"
81           style="fill:none;stroke:#rgb(14, 18, 16);stroke-width:55px;">
82         </path>
83       </g>
84     </svg>
85   </div>
86
87
88
89

```

```

90  <section class="image-ad">
91    <div class="team-member zoom">
92      <div class="image-box">
93        
94      </div>
95      <h3>Sunattha Boonla-or</h3>
96      <a href="mailto:gracesunattha01@gmail.com">gracesunattha01@gmail.com</a>
97      <div class="role">CMO</div>
98    </div>
99
100   <div class="team-member zoom">
101     <div class="image-box">
102       
103     </div>
104     <h3>Tinakome Rasripenngam</h3>
105     <a href="mailto:dome37688@gmail.com">dome37688@gmail.com</a>
106     <div class="role">CFO</div>
107   </div>
108
109   <div class="team-member zoom">
110     <div class="image-box">
111       
112     </div>
113     <h3>Teemtat Weerattanamanee</h3>
114     <a href="mailto:teemtat.wee@gmail.com">teemtat.wee@gmail.com</a>
115     <div class="role">CEO</div>
116   </div>
117
118   <div class="team-member zoom">
119     <div class="image-box">
120       
121     </div>
122     <h3>Kunruethai Patimapornchai</h3>
123     <a href="mailto:kunruethai.pat@gmail.com">kunruethai.pat@gmail.com</a>
124     <div class="role">CTO</div>
125   </div>
126
127   <div class="team-member zoom">
128     <div class="image-box">
129       
130     </div>
131     <h3>Teeratat Kunvoratham</h3>
132     <a href="mailto:teeratat05@gmail.com">teeratat05@gmail.com</a>
133     <div class="role">CPO</div>
134   </div>
135 </section>
136 </section>
137 </body>
138 </html>

```

Web Services

server.js

This Express server sets up an application that handles both static file serving and API routing. It imports essential modules like *express* for server creation, *path* for handling filesystem paths, *axios* for making HTTP requests to a backend service, and *multer* for handling file uploads. The server is configured to run on port 8082 and uses a router for organizing routes. Multer is set up with *memoryStorage*, meaning uploaded files are temporarily stored in server memory rather than being saved to disk. The server also uses middleware to parse URL-encoded and JSON bodies, allowing it to handle form data and API requests effectively.

The *BASE_DIR* is set to the current directory, and the server is configured to serve static assets like CSS, JavaScript, images, and built distribution files from corresponding folders. Several routes are defined using *router.get()* and *app.get()* to serve different HTML pages such as the homepage, login page, team page, product page, admin login, and account creation page.

For API operations, the server defines routes for admin login authentication, deleting products, adding new products (with an image upload), and updating existing products. These operations internally use *axios* to communicate with a backend API running on port 8083. Each API route sends a request to the backend and either redirects the user based on the result (for login, add, update) or returns JSON data to the frontend (for fetching products).

Specific product operations include deleting a product by ID, adding a new product by uploading an image and product information, updating a product's information and image, retrieving a single product by ID, fetching all products, fetching only user-visible products, and getting detailed product information.

Additionally, a fallback *404 route* is included to redirect any undefined request back to the homepage (*index.html*). Finally, the server starts and listens on the specified port, logging the server URL once it is running.

Code:

```
1 // Sets up an Express server with Axios, Multer, and routing functionalities.
2 const express = require('express');
3 const path = require('path');
4 const axios = require('axios');
5 const app = express();
6 const PORT = 8082;
7 const router = express.Router();
8 const multer = require('multer');
9
10 // Setup multer to store file uploads
11 const storage = multer.memoryStorage(); // or diskStorage if you want to save files
12 const upload = multer({ storage: storage });
13
14 // Configures the server to use the router and parse URL-encoded and JSON request bodies.
15 app.use(router);
16 router.use(express.urlencoded({ extended: true }));
17 router.use(express.json());
18
19 // Base directory to serve files from
20 const BASE_DIR = path.join(__dirname); // Automatically resolves to the current directory where server.js is located
21 console.log(BASE_DIR);
22
23 // Serve static assets
24 app.use('/css', express.static(path.join(BASE_DIR, 'css')));
25 app.use('/js', express.static(path.join(BASE_DIR, 'js')));
26 app.use('/img', express.static(path.join(BASE_DIR, 'img')));
27 app.use('/dist', express.static(path.join(BASE_DIR, 'dist')));
28
29 // Explicit page routes
30 router.get('/', (req, res) => {
31   // console.log(BASE_DIR);
32   res.sendFile(path.join(BASE_DIR, 'index.html'));
33 }
34 );
35 router.get('/index', (req, res) => {
36   // console.log(BASE_DIR);
37   res.sendFile(path.join(BASE_DIR, 'index.html'));
38 }
39 );
```

```
41 // Defines routes to serve various HTML pages for different parts of the website.
42 app.get('/login', (req, res) => res.sendFile(path.join(BASE_DIR, 'login.html')));
43 app.get('/team', (req, res) => res.sendFile(path.join(BASE_DIR, 'team.html')));
44 app.get('/product', (req, res) => res.sendFile(path.join(BASE_DIR, 'product.html')));
45 app.get('/admin_login', (req, res) => res.sendFile(path.join(BASE_DIR, 'admin_login.html')));
46 app.get('/advance-search', (req, res) => res.sendFile(path.join(BASE_DIR, 'advance-search.html')));
47 app.get('/create_account', (req, res) => res.sendFile(path.join(BASE_DIR, 'create_account.html')));
48 app.get('/product_detail', (req, res) => res.sendFile(path.join(BASE_DIR, 'product_detail.html')));
49 app.get('/service_admin', (req, res) => res.sendFile(path.join(BASE_DIR, 'service_admin.html')));
50 app.get('/add_product', (req, res) => res.sendFile(path.join(BASE_DIR, 'add_product.html')));
51 app.get('/update_product', (req, res) => res.sendFile(path.join(BASE_DIR, 'update_product.html')));
52
53 // Handles admin login authentication by sending credentials to the backend and redirecting based on the result.
54 router.post('/admin_login_auth', async function (req, res) {
55   const username = req.body.username;
56   const password = req.body.password;
57   console.log(username);
58   try{
59     const response = await axios.post('http://localhost:8083/api/admin_login_auth',{
60       username,
61       password
62     });
63
64     if(response.data.success){
65       res.redirect('/service_admin');
66     }
67   }
68   catch(error){
69     console.error('login error:',error);
70     res.redirect('/admin_login');
71   }
72});
```

```

74 // Handles product deletion by forwarding the request to the backend API and returning the result.
75 router.delete('/deleteProduct/:id', async (req, res) => {
76   const { id } = req.params;
77   try {
78     const response = await axios.delete(`http://localhost:8083/api/deleteProduct/${id}`);
79     res.send(response.data); // send back success to frontend
80   } catch (error) {
81     console.error('Delete product error:', error);
82     res.status(500).send('Failed to delete product');
83   }
84 });
85
86 // Handles adding a new product by uploading its image and sending product data to the backend API.
87 router.post('/addNewProduct', upload.single('productImage'), async function (req, res) {
88   const file = req.file; // <-- file upload
89   const { productName, brand, volume, price, type, description } = req.body;
90   console.log(productName);
91   let imageUrl = '';
92   if (file) {
93     imageUrl = file.buffer.toString('base64'); // or upload to storage and get URL
94   }
95   try {
96     const response = await axios.post('http://localhost:8083/api/addNewProduct', {
97       productName,
98       brand,
99       volume,
100      price,
101      type,
102      description,
103      imageUrl
104    });
105    res.redirect('/service_admin');
106  } catch (error) {
107    console.error('Add new product error:', error);
108    res.redirect('/service_admin');
109  }
110 });

```

```

112 // Handles updating a product by sending the updated data and image to the backend API.
113 router.post('/updateProduct', upload.single('productImage'), async (req, res) => {
114   const file = req.file;
115   const { product_ID, productName, brand, volume, price, type, description } = req.body;
116   const imageUrl = file ? file.buffer.toString('base64') : '';
117   try {
118     const response = await axios.put('http://localhost:8083/api/updateProduct', {
119       product_ID,
120       productName,
121       brand,
122       type,
123       volume,
124       price,
125       description,
126       imageUrl
127     });
128     if (response.data.success) {
129       console.log('Update succeeded:', response.data.message);
130       return res.redirect('/service_admin');
131     } else {
132       console.error('Update failed:', response.data);
133       return res.redirect('/service_admin');
134     }
135   } catch (err) {
136     console.error('Update product error:', err.response?.data || err.message);
137     return res.redirect('/service_admin');
138   }
139 });
140
141 // Fetches product details by ID from the backend API and returns them as JSON.
142 router.get('/api/getProduct/:id', async (req, res) => {
143   const { id } = req.params;
144   try {
145     const response = await axios.get(`http://localhost:8083/api/getProduct/${id}`);
146     res.json(response.data);
147   } catch (error) {
148     console.error('Fetch product error:', error.response?.data || error.message);
149     res.status(error.response?.status || 500).send('Product not found');
150   }
151 });

```

```

153 // Fetches all product data from the backend API and returns it as a JSON response.
154 router.get('/getAllProducts', async function (req, res) {
155   try {
156     const response = await axios.get('http://localhost:8083/api/getAllProducts');
157     res.json(response.data); // send back to frontend
158   } catch (error) {
159     console.error('Error fetching products:', error);
160     res.status(500).send('Error fetching products');
161   }
162 });
163
164 // Fetches all user-visible products from the backend API and returns them as JSON to the frontend.
165 router.get('/getAllProductsUser', async function (req, res) {
166   try {
167     const response = await axios.get('http://localhost:8083/api/getAllProductsUser');
168     console.log(response.data);
169     res.json(response.data); // send back to frontend
170   } catch (error) {
171     console.error('Error fetching products:', error);
172     res.status(500).send('Error fetching products');
173   }
174 });
175
176 // Fetches detailed product data from the backend API and returns it as JSON to the frontend.
177 router.get('/getProductDetail', async function (req, res) {
178   try {
179     const response = await axios.get('http://localhost:8083/api/getProductDetail');
180     console.log(response.data);
181     res.json(response.data); // send back to frontend
182   } catch (error) {
183     console.error('Error fetching products:', error);
184     res.status(500).send('Error fetching products');
185   }
186 });

```

```

188 // Fallback 404 route
189 app.use((req, res) => {
190   const notFoundPath = path.join(BASE_DIR, '/index.html');
191   res.status(404).sendFile(notFoundPath);
192 });
193
194 // Start the server
195 app.listen(PORT, () => {
196   console.log(`Server running at http://localhost:${PORT}`);
197 });

```

webservice.js

This Express server initializes a backend API service connected to a MySQL database. It uses essential modules like *express* for creating the server, *mysql2* for database connection, *path* for handling filesystem paths, and *cors* for enabling Cross-Origin Resource Sharing with frontend services (especially from *localhost:8082*). The server listens on port 8083 and accepts URL-encoded and JSON request bodies. The MySQL connection is established with the *Lumony* database, and any connection errors are logged.

Several API endpoints are defined to handle core product management features. A *DELETE* endpoint allows deleting a product by its *Product_ID*, sending back a success response if the operation completes successfully. A *PUT* endpoint updates existing product details, including the ability to keep an old image if a new one is not provided. The *GET /api/getProduct/:id* route fetches and returns a single product's data based on its ID.

The server also includes an *admin_login_auth* endpoint where it verifies the provided username and password against the *Administrator_Account* table to authenticate administrators. New products can be added through a *POST /api/addNewProduct* route by providing product details such as name, brand, type, volume, price, image, and description.

For retrieving products, there are two main routes: one that fetches **all products** (*/api/getAllProducts*) and another (*/api/getAllProductsUser*) that retrieves products based on optional filters like type and brand, while also adding a dynamic *priceRange* field based on the product price. Another route, */api/getProductDetail*, fetches detailed information for a specific product based on the provided *Product_ID*.

Lastly, the server directly serves an HTML file *service_admin.html* at the */service_admin* route for administrative interface access. If all setup is successful, the server starts listening at the designated port and logs its URL.

Code:

```
1 // Initializes an Express server and MySQL connection setup for backend API services.
2 const mysql = require('mysql2');
3 const express = require('express');
4 const app = express();
5 const path = require('path');
6 const PORT = 8083;
7
8 var cors = require('cors');
9 app.use(cors());
10
11 let corsOptions = {
12   origin: 'http://localhost:8082',
13   methods: 'GET,POST,PUT,DELETE'
14 }
15
16 app.use(cors(corsOptions));
17 app.use(express.json({ limit: '50mb' }));
18 app.use(express.urlencoded({ extended: true, limit: '50mb' }));
19
20 const db = mysql.createConnection({
21   host: 'localhost',
22   user: 'root',
23   password: 'Teemmatat_5068',
24   database: 'Lumony'
25 });
26
27 db.connect(function (err) {
28   if(err){
29     console.error("Error", err);
30     return;
31   }
32   console.log('Connected DB:', db.config.database);
33 });
```

```
35 // Deletes a product from the database by its ID and returns a success message.
36 app.delete('/api/deleteProduct/:id', (req, res) => {
37   const { id } = req.params;
38   const sql = 'DELETE FROM Product WHERE Product_ID = ?';
39   db.query(sql, [id], (err, result) => {
40     if (err) {
41       console.error('Delete error:', err);
42       return res.status(500).send('Database delete error');
43     }
44     res.send({ success: true, message: 'Product deleted successfully' });
45   });
46 });
47
48 // Updates a product's details in the database based on the provided information.
49 app.put('/api/updateProduct', (req, res) => {
50   const { product_ID, productName, brand, type, volume, price, description, imageUrl, oldImage } = req.body;
51
52   // If no new image uploaded, use old image
53   const finalImage = imageUrl && imageUrl.trim() !== '' ? imageUrl : oldImage;
54
55   const sql = `
56     UPDATE Product
57     SET Product_name = ?, Product_brand = ?, Product_type = ?, Product_volume = ?, Product_price = ?, Product_image = ?
58     WHERE Product_ID = ?
59   `;
60   const values = [productName, brand, type, volume, price, finalImage, description, product_ID];
61
62   db.query(sql, values, (err, result) => {
63     if (err) {
64       console.error('Update error:', err);
65       return res.status(500).send('Database update error');
66     }
67     console.log('Rows changed:', result.affectedRows);
68     res.send({ success: true, message: 'Product updated successfully' });
69   });
70 });
```

```

73 // Fetches a single product's details from the database by its ID and returns it.
74 app.get('/api/getProduct/:id', (req, res) => [
75   const id = req.params.id;
76   console.log('Fetching product with ID:', id);
77   const sql = 'SELECT * FROM Product WHERE Product_ID = ?';
78   db.query(sql, [id], (err, result) => {
79     if (err) {
80       console.error('Query error:', err);
81       return res.status(500).send('Database query error');
82     }
83     console.log('Query result:', result);
84     if (result.length === 0) {
85       return res.status(404).send('Product not found');
86     }
87     res.send(result[0]);
88   });
89 ]);
90
91 // Authenticates an admin by checking the username and password against the database.
92 app.post('/api/admin_login_auth', (req, res) => {
93   const { username, password } = req.body;
94   console.log(username);
95   db.query(
96     'SELECT * FROM Administrator_Account WHERE Acc_username = ? AND Acc_password = ?',
97     [username, password],
98     (err, results) => {
99       if (err) return res.status(500).send('Error');
100      if (results.length > 0) {
101        res.send({ success: true });
102      } else {
103        res.status(401).send({ success: false });
104      }
105    }
106  );
107});

```

```

109 // Inserts a new product into the database with the provided details and image.
110 app.post('/api/addNewProduct', (req, res) => {
111   const { productName, brand, type, volume, price, imageUrl, description } = req.body;
112   const sql = `
113     INSERT INTO Product (Product_name, Product_brand, Product_type, Product_volume, Product_price, Product_image, Product_descrip
114     VALUES (?, ?, ?, ?, ?, ?, ?)`;
115   const values = [productName, brand, type, volume, price, imageUrl, description];
116   db.query(sql, values, (err, results) => {
117     if (err) {
118       console.error('Insert error:', err);
119       return res.status(500).send('Database insert error');
120     }
121     res.send({ success: true, message: 'Product added successfully' });
122   });
123 });
124
125 // Fetches and returns all products from the database as a JSON array.
126 app.get('/api/getAllProducts', (req, res) => {
127   const sql = 'SELECT * FROM Product';
128   db.query(sql, (err, results) => {
129     if (err) {
130       console.error('Error fetching products:', err);
131       return res.status(500).send('Database error');
132     }
133     res.json(results);
134   });
135 });

```

```

137 // Fetches user-filtered products from the database, adds price ranges, and returns them as JSON.
138 app.get('/api/getAllProductsUser', (req, res) => {
139   const { type, brand } = req.query;
140   let sql = 'SELECT * FROM Product WHERE 1=1';
141   const params = [];
142   if (type) {
143     sql += ` AND Product_type = ?`;
144     params.push(type);
145   }
146   if (brand) {
147     sql += ` AND Product_brand = ?`;
148     params.push(brand);
149   }
150   db.query(sql, params, (err, results) => {
151     if (err) {
152       console.error('Error fetching products:', err);
153       return res.status(500).send('Database error');
154     }
155     const productsWithRange = results.map(product => {
156       let priceRange = '';
157       if (product.Product_price >= 2000 && product.Product_price <= 3000) {
158         priceRange = '2000-3000';
159       } else if (product.Product_price <= 4000) {
160         priceRange = '3001-4000';
161       } else if (product.Product_price <= 5000) {
162         priceRange = '4001-5000';
163       } else if (product.Product_price <= 6000) {
164         priceRange = '5001-6000';
165       } else if (product.Product_price <= 7000) {
166         priceRange = '6001-7000';
167       }
168       return {
169         ...product,
170         priceRange: priceRange
171       };
172     });
173     res.json(productsWithRange);
174   });
175 });

```

```

177 // Fetches detailed information for a specific product by ID and returns it as JSON.
178 app.get('/api/getProductDetail', (req, res) => {
179   const productId = req.query.Product_ID;
180   const query = 'SELECT * FROM Product WHERE Product_ID = ?';
181   db.query(query, [productId], (err, results) => {
182     if (err) {
183       console.error('Database query error:', err);
184       return res.status(500).json({ error: "Database error" });
185     }
186     if (results.length > 0) {
187       res.json(results[0]); // ເສັ້ນມູນຄືນຕໍ່າ
188     } else {
189       res.status(404).json({ error: "Product not found" });
190     }
191   });
192 });
193
194 // Serves the 'service_admin.html' page when the '/service_admin' route is accessed.
195 app.get('/service_admin', (req, res) => {
196   res.sendFile(path.join(__dirname, 'service_admin.html'));
197 });
198
199 app.listen(PORT, () => {
200   console.log(`http://localhost:${PORT}`);
201 });

```

Testing Results

POST method

Post Method 1:

The screenshot shows the Postman application interface. At the top, it says "HTTP webservice / POST1". Below that is a "POST" dropdown and the URL "http://localhost:8083/api/addNewProduct". The "Body" tab is selected, showing a raw JSON payload:

```
1 {
2   "productName": "Blooming Bouquet",
3   "brand": "Dior",
4   "type": "Perfume",
5   "volume": 50,
6   "price": 120.00,
7   "imageUrl": "base64-image-or-image-url-here",
8   "description": "A fresh floral fragrance perfect for daily wear."
9 }
10
```

Below the body, the response is shown under the "Body" tab, which is highlighted in blue. It shows a 200 OK status with a response time of 6 ms, a response size of 356 B, and a save response button. The response JSON is:

```
{}
200 OK 6 ms 356 B Save Response
1 {
2   "success": true,
3   "message": "Product added successfully"
4 }
```

```
// Testing Insert a New Product
// method: POST
// URL: http://localhost:8083/api/addNewProduct
// body: raw JSON
// {
//   "productName": "Blooming Bouquet",
//   "brand": "Dior",
//   "type": "Perfume",
//   "volume": 50,
//   "price": 120.00,
//   "imageUrl": "base64-image-or-image-url-here",
//   "description": "A fresh floral fragrance perfect for daily wear."
// }
```

Post Method 2:

The screenshot shows the Postman application interface. At the top, it says "HTTP webservice / POST2". Below that is a search bar with "POST" selected and the URL "http://localhost:8083/api/addNewProduct". To the right are "Save" and "Share" buttons. The main area has tabs for "Params", "Authorization", "Headers (8)", "Body", "Scripts", and "Settings". "Body" is currently selected and has a dropdown menu with options: "none", "form-data", "x-www-form-urlencoded", "raw" (which is selected), "binary", "GraphQL", and "JSON". There is also a "Beautify" button. The "Body" section contains the following JSON payload:

```
1 {
2   "productName": "Chance Eau Tendre",
3   "brand": "Chanel",
4   "type": "Perfume",
5   "volume": 100,
6   "price": 135.50,
7   "imageUrl": "sample-base64-image-or-url",
8   "description": "A fruity-floral fragrance that is delicate and fresh, perfect for everyday elegance."
9 }
10
```

Below the body, there are tabs for "Body", "Cookies", "Headers (9)", and "Test Results". The "Test Results" tab is selected and shows a green "200 OK" status with a response time of "6 ms" and a size of "356 B". It also includes a "Save Response" button and some icons. The response body is displayed as:

```
{}
1 {
2   "success": true,
3   "message": "Product added successfully"
4 }
```

```
// Testing Insert Another New Product
// method: POST
// URL: http://localhost:8083/api/addNewProduct
// body: raw JSON
// {
//   "productName": "Chance Eau Tendre",
//   "brand": "Chanel",
//   "type": "Perfume",
//   "volume": 100,
//   "price": 135.50,
//   "imageUrl": "sample-base64-image-or-url",
//   "description": "A fruity-floral fragrance that is delicate and fresh, perfect
for everyday elegance."
// }
```

PUT method

Put Method 1:

The screenshot shows the Postman application interface. At the top, it says "webservice / PUT1". Below that, the method is set to "PUT" and the URL is "http://localhost:8083/api/updateProduct". The "Body" tab is selected, showing the raw JSON payload:

```
1 {
2   "productName": "Chance Eau Tendre",
3   "brand": "Chanel",
4   "type": "Perfume",
5   "volume": 100,
6   "price": 135.50,
7   "imageUrl": "sample-base64-image-or-url",
8   "description": "A fruity-floral fragrance that is delicate and fresh, perfect for everyday elegance."
9 }
10
```

Below the body, the response is shown with a status of "200 OK", a duration of "6 ms", and a size of "358 B". The response body is also JSON:

```
1 {
2   "success": true,
3   "message": "Product updated successfully"
4 }
```

```
// Testing Update a Product
// method: PUT
// URL: http://localhost:8083/api/updateProduct
// body: raw JSON
// {
//   "product_ID": 1,
//   "productName": "Blooming Bouquet Updated",
//   "brand": "Dior",
//   "type": "Perfume",
//   "volume": 50,
//   "price": 125.00,
//   "description": "Updated description: Fresh floral fragrance with a soft
// finish.",
//   "imageUrl": "new-base64-or-image-url-here",
//   "oldImage": "old-image-url"
// }
```

Put Method 1:

The screenshot shows the Postman application interface. At the top, it says "HTTP webservice / PUT2". Below that, the method is set to "PUT" and the URL is "http://localhost:8083/api/updateProduct". The "Body" tab is selected, showing a raw JSON payload with line numbers 1 through 12. The payload is a product update object. Below the body, the response status is "200 OK" with a response time of 6 ms and a size of 358 B. The response body is also shown with line numbers 1 through 4, indicating a successful update message.

```
1 {
2   "product_ID": 22,
3   "productName": "Eros Updated",
4   "brand": "Versace",
5   "type": "Cologne",
6   "volume": 100,
7   "price": 95.00,
8   "description": "Updated description: Bold fragrance with notes of mint and vanilla.",
9   "imageUrl": "",
10  "oldImage": "existing-old-image-url-or-base64"
11 }
12
```

Body Cookies Headers (9) Test Results ⚡

{ } JSON ▾ Preview Visualize

```
1 {
2   "success": true,
3   "message": "Product updated successfully"
4 }
```

200 OK • 6 ms • 358 B • Save Response ⚡

```
// Testing Update Another Product (Keep Old Image)
// method: PUT
// URL: http://localhost:8083/api/updateProduct
// body: raw JSON
// {
//   "product_ID": 2,
//   "productName": "Eros Updated",
//   "brand": "Versace",
//   "type": "Cologne",
//   "volume": 100,
//   "price": 95.00,
//   "description": "Updated description: Bold fragrance with notes of mint and
//   vanilla.",
//   "imageUrl": "",
//   "oldImage": "existing-old-image-url-or-base64"
// }
```

DELETE method

Delete Method 1:

The screenshot shows the Postman interface with the following details:

- Method:** DELETE
- URL:** <http://localhost:8083/api/deleteProduct/21>
- Body:** Raw JSON response:

```
1 {
2     "success": true,
3     "message": "Product deleted successfully"
4 }
```
- Status:** 200 OK
- Time:** 4 ms
- Size:** 358 B

```
// Testing Delete a Product
// method: DELETE
// URL: http://localhost:8083/api/deleteProduct/21
// body: none
```

Delete Method 2:

The screenshot shows the Postman interface with the following details:

- Method:** DELETE
- URL:** <http://localhost:8083/api/deleteProduct/22>
- Query Params:** Key: Value
- Body:** Raw JSON response:

```
1 {
2     "success": true,
3     "message": "Product deleted successfully"
4 }
```
- Status:** 200 OK
- Time:** 4 ms
- Size:** 358 B

```
// Testing Delete Another Product
// method: DELETE
// URL: http://localhost:8083/api/deleteProduct/22
// body: none
```

References

- [1] Versace, "Fragrances," Versace, 2025. [Online]. Available: <https://www.versace.com/row/en/fragrances/>. [Accessed: 20-Feb-2025].
- [2] Givenchy Beauty, "Fragrance," Givenchy, 2025. [Online]. Available: <https://www.givenchybeauty.com/us/fragrance/>. [Accessed: 21-Feb-2025].
- [3] Dior, "Fragrances for Men and Women," Dior Beauty, 2025. [Online]. Available: https://www.dior.com/en_int/fragrance. [Accessed: 29-Apr-2025].
- [4] Chanel, "Fragrance," Chanel Official Site, 2025. [Online]. Available: <https://www.chanel.com/us/fragrance/>. [Accessed: 29-Apr-2025].
- [5] Yves Saint Laurent, "Fragrance," YSL Beauty, 2025. [Online]. Available: <https://www.yslbeautyus.com/fragrance/>. [Accessed: 29-Apr-2025].
- [6] Gucci, "Fragrances," Gucci Beauty, 2025. [Online]. Available: <https://www.gucci.com/us/en/ca/beauty/fragrances>. [Accessed: 29-Apr-2025].
- [7] Calvin Klein, "Fragrance," Calvin Klein, 2025. [Online]. Available: <https://www.calvinklein.us/en/fragrance>. [Accessed: 29-Apr-2025].
- [8] Giorgio Armani Beauty, "Fragrance," Armani Beauty, 2025. [Online]. Available: <https://www.armani-beauty.com/fragrances>. [Accessed: 29-Apr-2025].
- [9] Burberry, "Fragrance Collection," Burberry, 2025. [Online]. Available: <https://www.burberry.com/fragrance/>. [Accessed: 29-Apr-2025].
- [10] Jo Malone London, "Fragrance Collection," Jo Malone, 2025. [Online]. Available: <https://www.jomalone.com/fragrances>. [Accessed: 29-Apr-2025].
- [11] Tom Ford, "Fragrances," Tom Ford Beauty, 2025. [Online]. Available: <https://www.tomford.com/beauty/fragrance/>. [Accessed: 29-Apr-2025].