



Faculty of Information and Communication Technology
Mahidol University

Project Phase II

By

Miss Russarin	Eaimrittikrai	6488021
Miss Chaninan	Phetpangun	6488061
Miss Penpitchapa	Pantaraksakul	6488175
Miss Warangkhana	Srichan	6188096

ITCS212 Web Programming
2023

Table of Contents

Project Overview	2
Navigation Diagram	3
Database	4
Details of web application and code	6
Details of web services and code	17
Testing results of web services	29
Test Cases for admin login authentication	29
Test Cases for product management	31
Test Cases for admin information management	42

Link to Presentation Video:

https://drive.google.com/file/d/1qn28AZ4JhSeJ-rFWYabip6CL3TzvyVmv/view?usp=share_link

Project Overview

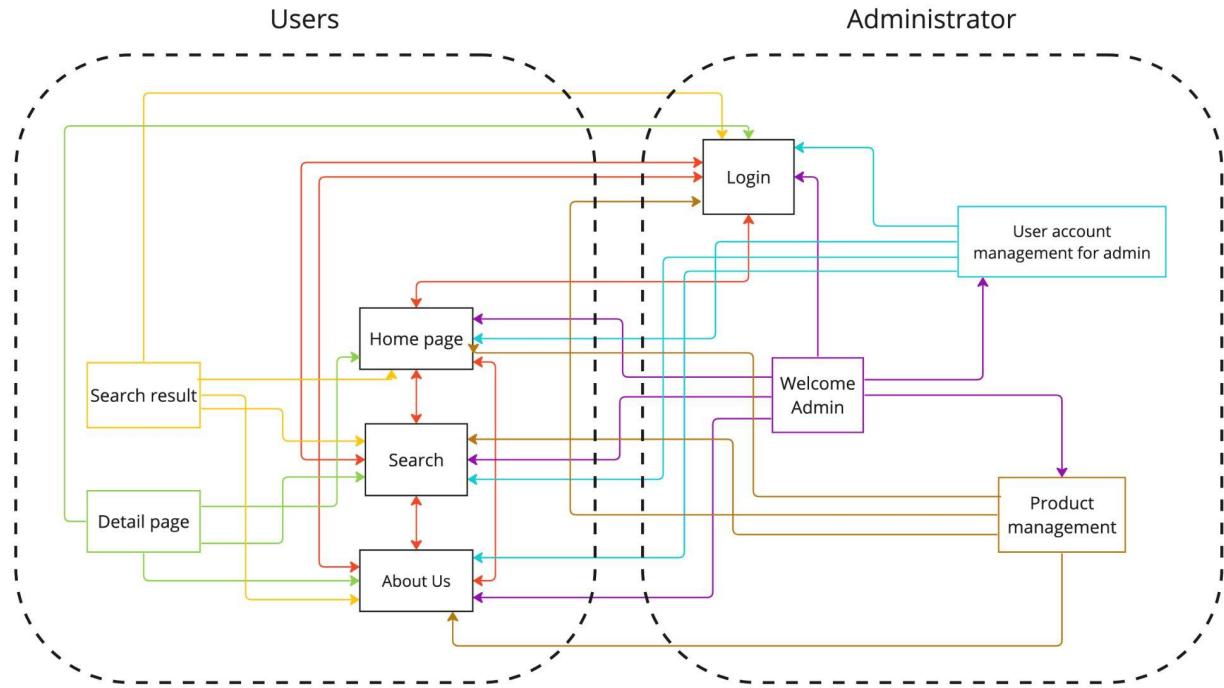
The business domain that was selected by the team members is an online clothes-shopping site that allows users to explore products from various brands; therefore, the name of this website is MultiClothes Shopping. In this project, the members are assigned to design and develop a front-ended section of the web application (UI) using the combination of HTML and CSS, and a back-ended section (web services) using the combination of Node.js and JavaScript.

The objective of the first phase of the project is to plan out the navigation pages and implement the pages by utilizing HTML and CSS to create front-end web pages for the business which consist of a Home Page, Search Page, Search Result Page, Product Detail Page, About Us Page, Log in for Administrator Page, Welcome Admin Page, Product Management for Administrator Page, and Admin Information Management for Administrator Page.

The objective of the second phase of the project is to create a database to keep the administrators' information, administrators' login information, and product information. Then, the members have to create web services to handle functions: authentication for administrators to log in, products modifications for administrators (search or view, insert, update, delete product), and administrators' information modification for administrators (search or view, insert, update, delete administrators' information). Finally, the members create web services interaction by connecting the web services to the interface of the web application and connecting the web application to a public API. In the implementation, the selected web API is Currency Data API to convert currency in real time

(https://apilayer.com/marketplace/currency_data-api?e=Sign+Up&l=Success)

Navigation Diagram



The navigation diagram for this project is divided into two major sections: users and administrators. The users' section consists of a Home page, Search page, Search result, Details page of the product, and About Us. The administrators' section consists of a Login page, a Welcome Admin page, a Product Management page, and a User account management page for admin. Since the navigation bar is allocated on every page, the Home page, Search page, About Us page, and Login page for admin are accessible on every page. On the other hand, the Welcome Admin page, Product Management page, and User account management page are only accessible after the administrators successfully log in through the Login page.

Database

Name: multiclothes

Table admin information: admin_info

```
--  
14 • CREATE TABLE IF NOT EXISTS `admin_info` (  
15     `AD_NUM` CHAR(9) PRIMARY KEY,  
16     `AD_FNAME` VARCHAR(50) NOT NULL,  
17     `AD_LNAME` VARCHAR(50) NOT NULL,  
18     `AD_ADDRESS` VARCHAR(500) NOT NULL,  
19     `AD_BD` DATE NOT NULL,  
20     `AD_EMAIL` VARCHAR(50) NOT NULL,  
21     `AD_PHONE` CHAR(10)  
22 );  
23  
24 • INSERT INTO `admin_info` VALUES  
25     ('0000000001','Somsri','Mayang','999 Phuttamonthon 4 Road, Salaya, Nakhon Pathom 73170 Thailand','1988-11-22','somsri.may@multiclothes.com','0869144407'),  
26     ('0000000002','Yang','Nonyu','23, Trok Rongnamkeang, Yotha Rd., Bangkok, 10100 Thailand','1991-01-17','yang.non@multiclothes.com','0972187739'),  
27     ('0000000003','Suaysood','Maisoy','Khlong Luang 25 KhlongLuang, Chang Wat Pathum Thani 12120','2001-05-03','Suaysood.nai@multiclothes.com','0978394783'),  
28     ('000885600','Waimaiii','Yaknon','17 Henri Dunant Rd, Khaeng Pathum Wan, Khet Pathum Wan, Krung Thep Maha Nakhon 10330','1970-07-29','Waimaiii.yak@multiclothes.com','0622785590'),  
29     ('000032191','Chanom','Kaimook','62 Moo 1, Rangsit-Ongkarak Road (Km.7) Thanyaburi, Pathum Thani','1999-09-12','Chanom.kai@multiclothes.com','0974328801');  
30 -----
```

Table admin login information: admin_login

```
36 • CREATE TABLE IF NOT EXISTS `admin_login` (  
37     `AD_USERNAME` VARCHAR(50) NOT NULL,  
38     `AD_PASSWORD` VARCHAR(20) NOT NULL,  
39     `AD_LOGIN_TIMESTAMP` DATETIME NOT NULL,  
40     `AD_NUM` CHAR(9) NOT NULL,  
41     CONSTRAINT `FK_adminAD_NUM` FOREIGN KEY (`AD_NUM`)  
42     REFERENCES `admin_info` (`AD_NUM`)  
43 );  
44  
45 • INSERT INTO `admin_login` VALUES  
46     ('somsri.may@multiclothes.com', 'Somchai555', '2022-11-08 22:02:22', '0000000001'),  
47     ('yang.non@multiclothes.com', 'nyaungyooka', '2022-03-18 18:29:30', '006000202'),  
48     ('Waimaiii.yak@multiclothes.com', 'Maiwai', '2022-01-04 10:20:35', '000885600'),  
49     ('Chanom.kai@multiclothes.com', 'Maiwann0', '2022-10-01 15:29:09', '000032191'),  
50     ('Suaysood.nai@multiclothes.com', 'blursabb', '2022-12-15 01:45:27', '000020031');
```

Table product information: product

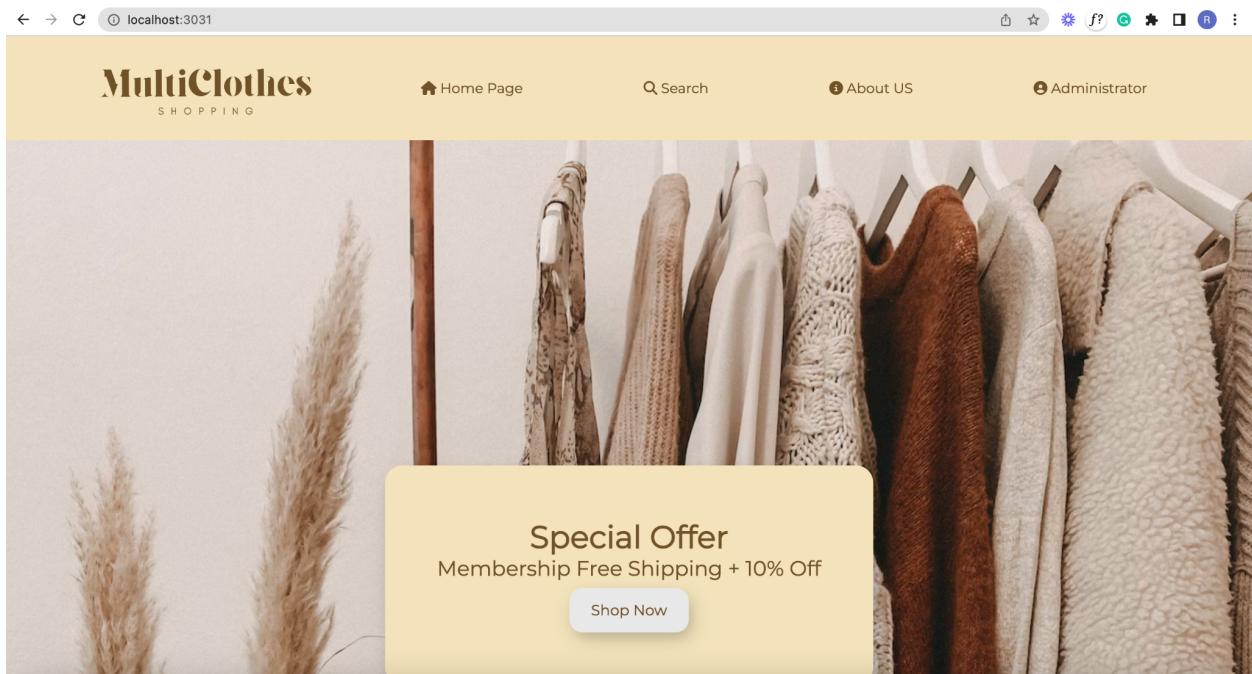
```

57 • ◇ CREATE TABLE IF NOT EXISTS `product` (
58     `PROD_ID` CHAR(12) PRIMARY KEY,
59     `PROD_NAME` VARCHAR(50) NOT NULL,
60     `PROD_BRAID` VARCHAR(50) NULL,
61     `PROD_DES` VARCHAR(500) NOT NULL,
62     `PROD_TYPE` VARCHAR(20) DEFAULT NULL,
63     `PROD_COLOR` VARCHAR(100) DEFAULT NULL,
64     `PROD_PRICE` DECIMAL(9,2) NOT NULL,
65     `PROD_QUANTITY` INT NOT NULL,
66     `PROD_SIZE` VARCHAR(10) DEFAULT NULL
67 )
68 • ◇ INSERT INTO `product` VALUES
69     ('AB1234567890', 'Oversized V-neck Sweatshirt', 'H&M',
70      'CONSCIOUS CHOICE Oversized top in sweatshirt fabric made from a cotton blend with a soft brushed inside. V-neck, low dropped shoulders and long sleeves. Ribbing around the neckline and cuffs and a raw, roll-edge hem.
71      Composition
72      BiC Cotton 95%
73      Elastane 5%
74      Shell Cotton 62%
75      Polyester 38%',
76      'Cloth', 'Black-White', '29.99', '100', 'S'),
77     ('AB1134567890', 'Canvas Hi-top Trainers', 'H&M',
78      'CONSCIOUS CHOICE Hi-tops in sturdy cotton canvas with a tongue and lacing at the front, and metal eyelets in one side. Cotton canvas linings and insoles and rubber soles that are patterned underneath. Height of soles 3.1 cm.
79      Composition
80      Upper Cotton 100%
81      Outersole Rubber 100%
82      Liningsock Cotton 100%',
83      'Footwear', 'Black', '33.00', '50', '35'),
84     ('AB1114567890', 'Fitted Double-Breasted Blazer', 'ZARA',
85      'Tailored blazer with a lapel collar and long sleeves. Front flap pockets and a welt chest pocket. Matching lining. Double-breasted button fastening at the front.
86      Composition
87      Polyester 68%
88      Viscose 29%
89      Elastane 3%',
90      'Cloth', 'White', '120.00', '20', 'XXS'),
91     ('AB1111567890', 'Heart Print T-Shirt', 'ZARA',
92      'T-shirt with a round neckline and short sleeves. Contrast prints on the front and back.
93      Composition
94      Cotton 100%',
95      'Cloth', 'Crimson', '29.99', '150', 'XL'),
96     ('AB1111167890', 'Mens MB', 'FILA',
97      'Iconic midsole design and printed graphic motif on quarter with embroidered FILA logos on quarter, tongue, toebox, and heel.
98      Composition
99      Leather
100      Synthetic',
101      'Footwear', '107 EGRET-GRAY MIST', '95.00', '130', '40');

```

Details of web application and code

Web server of the homepage: <http://localhost:3031/>



Code that connects this page to the port:

```
// Home Page
router.get('/', (req, res) => {
  res.sendFile(path.join(`__dirname}/html/HomePage.html`))
  res.status(200)
```

This is the homepage of the web application which is running on port 3031. On this page, there is the connected public API at the end of the page that the users can convert USD to THB in real-time.

API TO CONVERT USD TO THB REAL TIME

Amount To Convert From USD To THB: Convert



Code that connects the web application to the Currency Data API:

```
<script>
    // set endpoint and your access key
    const endpoint = 'convert'
    const access_key = 'ti3FllknYhicQb25iSRJrTVkEAXib9Gy';

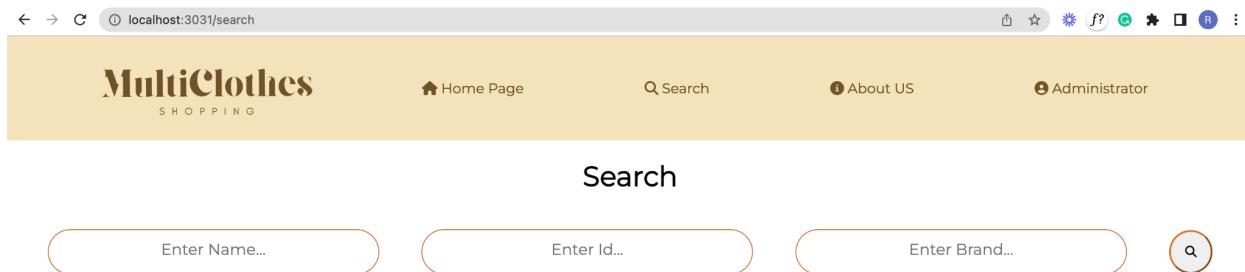
    // define from currency, to currency, and amount
    const from = 'USD';
    const to = 'THB';
    const amount = document.getElementById('amount-input');
    const convertButton = document.getElementById('convert-btn');
    convertButton.addEventListener('click', async (event) => {
        event.preventDefault();
        var myHeaders = new Headers();
        myHeaders.append("apikey", "ti3FllknYhicQb25iSRJrTVkEAXib9Gy");
        const query = amount.value;

        var requestOptions = {
            method: 'GET',
            redirect: 'follow',
            headers: myHeaders
        };

        await fetch(`https://api.apilayer.com/currency_data/convert?to=${to}&from=${from}&amount=${query}`)
            .then(response => response.text())
            .then(result => alert(result))
            .catch(error => console.log('error', error));
    });
</script>
```

This code connects to the Currency Data API by using the Free API key from the website (The limitation of use is 100 a month), and then executes fetch to acquire the information from the link provided in the website to calculate the currency from USD to THB in real-time. The return data will alert on the web page in the form of JSON.

Web server of the search page: <http://localhost:3031/search>



Code that connects this page to the port:

```
// Search Page
router.get('/search', (req, res) => {
  res.sendFile(path.join(`__dirname}/html/Search.html`))
  res.status(200)

});
```

This is the search page of the web application. On this page, the users can view all of the products and search for the products by entering either name, id, or brand of the product.

Web server of the about us page: <http://localhost:3031/about-us>

About us

Meet The Team



Russarin Eaimrittikrai
6488021
russarin.eai@multiclothes.com

[Facebook](#) [Twitter](#) [Instagram](#)



Chaninan Phetpangun
6488061
chaninan.phe@multiclothes.com

[Facebook](#) [Twitter](#) [Instagram](#)



Penpitchapa Pantaraksakul
6488175
penpitchapa.pan@multiclothes.com

[Facebook](#) [Twitter](#) [Instagram](#)



Warangkhana Srichan
6188096
warangkhana.sri@multiclothes.com

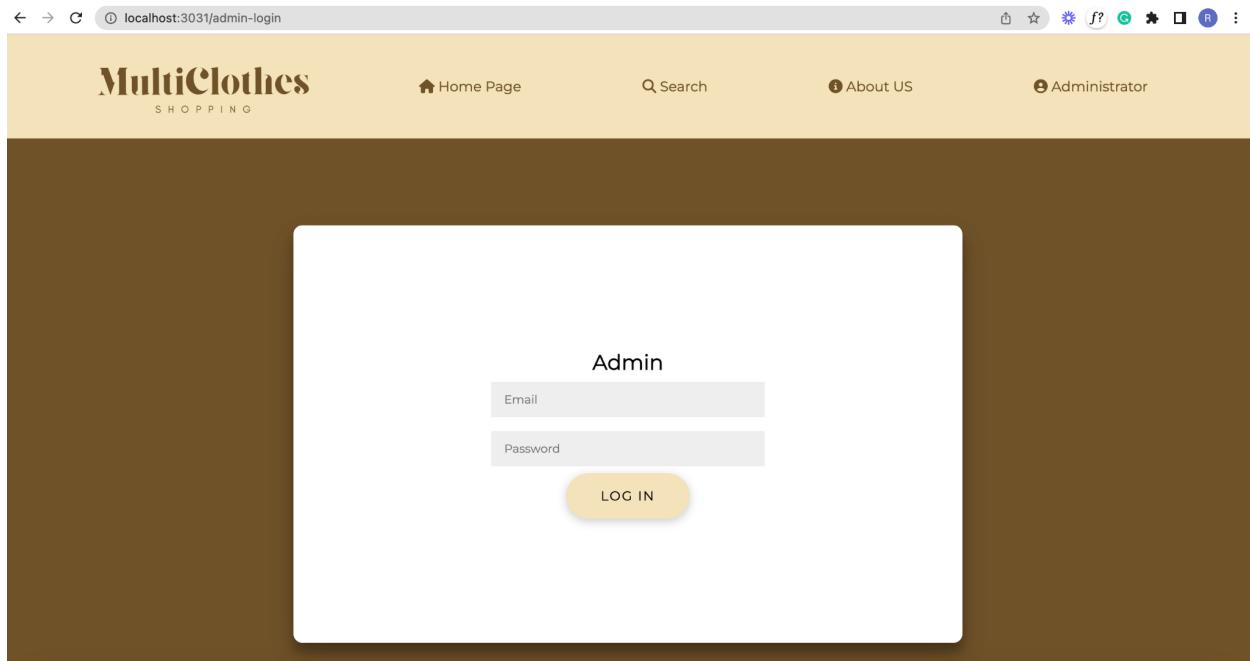
[Facebook](#) [Twitter](#) [Instagram](#)

Code that connects this page to the port:

```
// About Us Page
router.get('/about-us', (req, res) => {
  res.sendFile(path.join(`__dirname}/html/About_us.html`))
  res.status(200)
});
```

This is the about us page of the web application. On this page, the users can view the name and contact information of the members of the team.

Web server of the administrator login page: <http://localhost:3031/admin-login>



Code that connects this page to the port:

```
// Admin login Page
router.get('/admin-login', (req, res) => {
  res.sendFile(path.join(`__dirname)/html/Admin.html`))
  res.status(200)
});

router.post('/form-submit', function (req, res) {
  console.log(req.method);
  console.log(req.body.email);
  res.status(200);
});
```

This is the login page for administrators. The administrators who can log in to this page already have their email (username) and password information in the database. Otherwise, it will alert invalid login.

Web server of the login page for administrators: <http://localhost:3031/product-all>

The screenshot shows a web browser window with the URL localhost:3031/product-all. The page displays a grid of three product cards. Each card features a photograph of a model wearing the product, a product name, and a rating. The first card is for an 'H&M Oversized V-neck Sweatshirt' featuring a black and white striped top, rated 4.7 stars. The second card is for 'H&M Canvas Hi-top Trainers' showing a person sitting cross-legged in dark pants and a green top, rated 4.1 stars. The third card is for a 'ZARA Fitted Double-Breasted Blazer' in cream-colored, double-breasted suit, rated 5.0 stars.

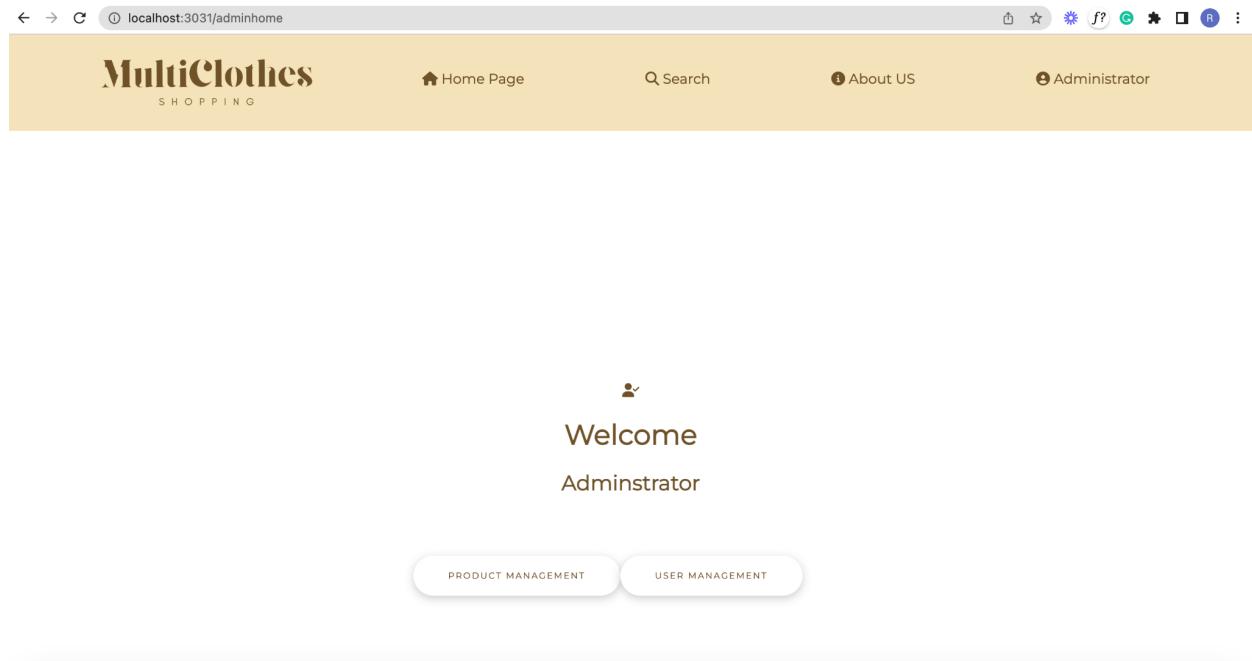
This is the page of the web application to show the list of products with pictures and information. On this page, the users can click the Buy Now button to see detailed information about each product. For instance, this page will show detailed information about the first product.

The screenshot shows a web browser window with the URL localhost:3031/product-one. The page is titled 'MultiClothes SHOPPING'. It displays a detailed product page for the 'H&M Oversized V-neck Sweatshirt'. The product image shows a woman wearing the sweatshirt. The product title is 'H&M Oversized V-neck Sweatshirt'. A description states: 'CONSCIOUS CHOICE Oversized top in sweatshirt fabric made from a cotton blend with a soft brushed inside. V-neck, low dropped shoulders and long sleeves. Ribbing around the neckline and cuffs and a raw, roll-edge hem.' The product ID is AB1234567890. The composition is listed as: 'Rib Cotton 95%, Elastane 5%, Shell Cotton 62%, Polyester 38%'. The price is \$29.99. There are dropdown menus for 'Available Sizes' and 'Available Colors', both set to 'Please select size' and 'Please select color'. Below these are 'Add to Cart' and a heart-shaped 'Like' button.

Code that connects this page to the port:

```
// page to show all product with picture
router.get('/product-all', (req, res) => {
    res.sendFile(path.join(`$__dirname__/html/DetailPage.html`))
    res.status(200)
});
// Detail page of product 1
router.get('/product-one', (req, res) => {
    res.sendFile(path.join(`$__dirname__/html/DetailPageOne.html`))
    res.status(200)
});
// Detail page of product 2
router.get('/product-two', (req, res) => {
    res.sendFile(path.join(`$__dirname__/html/DetailPageTwo.html`))
    res.status(200)
});
// Detail page of product 3
router.get('/product-three', (req, res) => {
    res.sendFile(path.join(`$__dirname__/html/DetailPageThree.html`))
    res.status(200)
});
// Detail page of product 4
router.get('/product-four', (req, res) => {
    res.sendFile(path.join(`$__dirname__/html/DetailPageFour.html`))
    res.status(200)
});
// Detail page of product 5
router.get('/product-five', (req, res) => {
    res.sendFile(path.join(`$__dirname__/html/DetailPageFive.html`))
    res.status(200)
});
// Detail page of product 6
router.get('/product-six', (req, res) => {
    res.sendFile(path.join(`$__dirname__/html/DetailPageSix.html`))
    res.status(200)
});
```

Web server of the page to welcome administrators: <http://localhost:3031/adminhome>



Code that connects this page to the port:

```
// Welcome Admin Page
router.get('/adminhome', (req, res) => {
  res.sendFile(path.join(`${__dirname}/html/WelcomePageAdmin.html`))
  res.status(200)
});
```

Once the administrator successfully logs in to the system, they will see this page that has two buttons that will navigate to the product management page and the user management page.

Web server of the product management page:<http://localhost:3031/product-lists>

The screenshot shows a web application interface for managing products. At the top, a navigation bar includes a back button, a refresh button, and a URL field showing 'localhost:3031/product-lists'. Below the navigation is a header 'Your Products'. The main content area displays three products in a grid:

- Mens MB**
 - ID: AB1111167890
 - Brand: FILA
 - Description: CIconic midsole design and printed graphic motif on quarter with embroidered FILA logos on quarter, tongue, toebox, and heel.
 - Composition: Leather Synthetic
 - Type: Footwear
 - Color: 107 EGRET-GRAY MIST
 - Price: 95.00
 - Quantity: 130
 - Size: 40
 - Edit
- Heart Print T-Shirt**
 - ID: AB11111567890
 - Brand: ZARA
 - Description: T-shirt with a round neckline and short sleeves. Contrast prints on the front and back.
 - Composition: Cotton 100%
 - Type: Cloth
 - Color: Crimson
 - Price: 29.99
 - Quantity: 150
 - Size: XL
 - Edit
- Fitted Double-Breasted Blazer**
 - ID: AB11114567890
 - Brand: ZARA
 - Description: Tailored blazer with a lapel collar and long sleeves. Front flap pockets and a welt chest pocket. Matching lining. Double-breasted button fastening at the front.
 - Composition: Polyester 68% Viscose 29% Elastane 3%
 - Type: Cloth
 - Color: White
 - Price: 120.00
 - Quantity: 20
 - Size: XXS
 - Edit

Below the main grid, there are two more product cards:

- Canvas Hi-top Trainers**
- Oversized V-neck Sweatshirt**

Code that connects this page to the port:

```
// List of products in the store
router.get('/product-lists', (req, res) => {
  res.sendFile(path.join(`__dirname)/html/ProductManage1.html`))
  res.status(200)
});
// Form to add more product
router.get('/product-add', (req, res) => {
  res.sendFile(path.join(`__dirname)/html/ProductManage.html`))
  res.status(200)
});
// Form to update product
router.get('/product-update', (req, res) => {
  res.sendFile(path.join(`__dirname)/html/ProductUpdate.html`))
  res.status(200)
});
```

After the administrator clicked on the Product management button, they will see this page that contains the information about the products. They can click the trash can button on the product to delete the product, click Edit to modify the product information, and click Add More which will lead to the form to fill in the detail of the new product (<http://localhost:3031/product-add>).

Product Management

Product Id*

Product Name*

Product Brand*

Product Description

Product Type: *

Clothing Footwear

Available Sizes:

XXS XS S M L XL XXL Free Size
 35 EU 36 EU 37 EU 38 EU 39 EU 40 EU 41 EU 42 EU

Web server of the administrators' account management page:<http://localhost:3031/admin-manage>

The screenshot shows a web browser window with the URL localhost:3031/admin-manage. The title bar says "localhost:3031/admin-manage". The page content is titled "Add a New Account". There is a back button labeled "Back to previous". The main area contains a form with the following fields:

- Username: [input field]
- Password: [input field]
- First Name: [input field]
- Last Name: [input field]
- Address: [input field]
- birthday: [input field] (dd/mm/yyyy)
- Email: [input field]
- Mobile Phone: [input field]
- ID: [input field]

At the bottom of the form are two buttons: "ADD ACCOUNT" and "CLEAR".

Code that connects this page to the port:

```
// Form to add more admin info
router.get('/admin-manage', (req, res) => {
  res.sendFile(path.join(`__dirname}/html/User_Account_Management.html`))
  res.status(200)
});
```

After the administrator clicked on the Account management button, they will see this page that contains the form to add a new account. They can view all the administrators' details, search for the administrator's information, edit the administrator's information, and delete the administrator's account.

Details of web services and code

Port of the backend.js that keeps all the web services: <http://localhost:3030>

There is a total of three routers in the node:

- Router 1: product management service (<http://localhost:3030/product>)
- Router 2: login authentication service (<http://localhost:3030/admin-login>)
- Router 3: admin information management service (<http://localhost:3030/admin>)

1. Authentication web service for administrators

```
router2.post('/', (req, res) => {
  let admin_username = req.body.username;
  let admin_password = req.body.password;
  console.log('admin login:', admin_username, `at ${Date()} ${Date.now()}`);
  if (!admin_username || !admin_password) {
    return res.status(400).send({
      error: true,
      message: 'Please enter username and password'
    });
  }
  dbConn.query("SELECT AD_NUM FROM admin_login WHERE AD_USERNAME = ? AND AD_PASSWORD = ?", [admin_username, admin_password], function (error, results) {
    if (error) throw error;
    // If username or password doesn't match database
    if (results.length == 0) {
      return res.status(401).send({
        error: true,
        message: 'Invalid log in',
        status: 401
      });
    } else {
      return res.status(200).send({
        error: false,
        data: results,
        message: 'Successfully log in',
        status: 200
      });
    }
  });
});
```

HTTP method for login authentication is POST. The elements that are required for authentication are username and password. The query checks if the username and password exist in the database and if they are matched.

2. Functions for administrators to search/view, insert, update, and delete products' information

- View all products

```
// Select All (No criteria search)
router1.get('/products', (req, res) => {
  dbConn.query("SELECT * FROM product", function (error, results) {
    if (error) throw error;
    return res.send({
      error: false,
      data: results,
      message: 'Product list.'
    });
  });
});
```

Method: GET

URL: <http://localhost:3030/product/products>

Search all the products in the database.

- Search by three criteria: Product Id, Product Name, and Product Brand

Search by Id	<pre>router1.get('/id/:id', (req, res) => { let product_id = req.params.id; console.log(product_id); if (!product_id) { return res.status(400).send({ error: true, message: 'Please provide product ID' }); } dbConn.query(`SELECT * FROM product WHERE PROD_ID LIKE "%\${product_id}%"`, function (error, results) { if (error) throw error; return res.send({ error: false, data: results, message: 'Product retrieved' }); }); });</pre>	Method: GET URL: http://localhost:3030/product/id/ Recieve ID as parameter and check if there is any product ID that contains the input ID.
Search by Name	<pre>// Search by name router1.get('/name/:name', (req, res) => { let product_name = req.params.name; console.log(product_name); if (!product_name) { return res.status(400).send({ error: true, message: 'Please provide product name' }); } dbConn.query('SELECT * FROM product WHERE PROD_NAME LIKE "%" + product_name + "%"', function (error, results) { if (error) throw error; return res.send({ error: false, data: results, message: 'Product retrieved' }); }); });</pre>	Method: GET URL: http://localhost:3030/product/name/ Recieve Name as parameter and check if there is any product Name that contains the input Name.
Search by Brand	<pre>// Search by brand router1.get('/brand/:brand', (req, res) => { let product_brand = req.params.brand; console.log(product_brand); if (!product_brand) { return res.status(400).send({ error: true, message: 'Please provide product Brand' }); } dbConn.query(`SELECT * FROM product WHERE PROD_BRAND LIKE "%\${product_brand}%"`, function (error, results) { if (error) throw error; return res.send({ error: false, data: results, message: 'Product retrieved' }); }); });</pre>	Method: GET URL: http://localhost:3030/product/brand/ Recieve Brand as parameter and check if there is any product Brand that contains the input Brand.

Search by Id and Name	<pre>router.get('/idname/:id/:name', (req, res) => { let product_id = req.params.id; console.log(product_id); let product_name = req.params.name; console.log(product_name); if (!product_id) { return res.status(400).send({ error: true, message: 'Please provide product ID' }); } if (!product_name) { return res.status(400).send({ error: true, message: 'Please provide product name' }); } dbConn.query('SELECT * FROM product WHERE PROD_ID LIKE "%\${product_id}%" OR PROD_NAME LIKE "%\${product_name}%"', function (error, results) { if (error) throw error; return res.send({ error: false, data: results, message: 'Product retrieved' }); }); });</pre>	<p>Method: GET URL: http://localhost:3030/product/idname/</p> <p>Recieve ID and Name as parameters and check if there is any product that contains the input ID or Name.</p>
Search by Id and Brand	<pre>/ Search by id and brand router.get('/idbrand/:id/:brand', (req, res) => { let product_id = req.params.id; console.log(product_id); let product_brand = req.params.brand; console.log(product_brand); if (!product_id) { return res.status(400).send({ error: true, message: 'Please provide product ID' }); } if (!product_brand) { return res.status(400).send({ error: true, message: 'Please provide product Brand' }); } dbConn.query('SELECT * FROM product WHERE PROD_ID LIKE "%\${product_id}%" OR PROD_NAME LIKE "%\${product_brand}%"', function (error, results) { if (error) throw error; return res.send({ error: false, data: results, message: 'Product retrieved' }); }); });</pre>	<p>Method: GET URL: http://localhost:3030/product/idbrand/</p> <p>Recieve ID and Brand as parameters and check if there is any product that contains the input ID or Brand.</p>
Search by Name and Brand	<pre>// Search by name and brand router.get('/namebrand/:name/:brand', (req, res) => { let product_name = req.params.name; console.log(product_name); let product_brand = req.params.brand; console.log(product_brand); if (!product_name) { return res.status(400).send({ error: true, message: 'Please provide product name' }); } if (!product_brand) { return res.status(400).send({ error: true, message: 'Please provide product Brand' }); } dbConn.query('SELECT * FROM product WHERE PROD_BRAND LIKE "%\${product_brand}%" OR PROD_NAME LIKE "%\${product_name}%"', function (error, results) { if (error) throw error; return res.send({ error: false, data: results, message: 'Product retrieved' }); }); });</pre>	<p>Method: GET URL: http://localhost:3030/product/namebrand/</p> <p>Recieve Name and Brand as parameters and check if there is any product that contains the input Name or Brand.</p>
Search by Id, Name, and Brand		

```

// Search by id, name, and brand
router1.get('/all/:id/:name/:brand', (req, res) => {
  let product_id = req.params.id;
  console.log(product_id);
  let product_name = req.params.name;
  console.log(product_name);
  let product_brand = req.params.brand;
  console.log(product_brand);
  if (!product_id) {
    return res.status(400).send({
      error: true,
      message: 'Please provide product ID'
    });
  }
  if (!product_name) {
    return res.status(400).send({
      error: true,
      message: 'Please provide product name'
    });
  }
  if (!product_brand) {
    return res.status(400).send({
      error: true,
      message: 'Please provide product Brand'
    });
  }
  dbConn.query(`SELECT * FROM product WHERE PROD_ID LIKE "%${product_id}%" OR PROD_BRAND LIKE "%${product_brand}%" OR PROD_NAME LIKE "%${product_name}"%` ,
  if (error) throw error;
  return res.send({
    error: false,
    data: results,
    message: 'Product retrieved'
  });
});
});

```

Method: GET
URL:
<http://localhost:3030/product/all/>

Recieve ID, Name and Brand as parameters and check if there is any product that contains the input ID or Name or Brand.

- Insert a product

```

// Insert
router1.post('/add', (req, res) => {
  let product = req.body.product;
  console.log(product);
  if (!product) {
    return res.status(400).send({
      error: true,
      message: 'Please provide product information'
    });
  }
  dbConn.query("INSERT INTO product SET ?", [product], function (error, results) {
    if (error) throw error;
    return res.send({
      error: false,
      data: results.affectedRows,
      message: 'New product has been created successfully'
    });
  });
});

```

Method: POST

URL: <http://localhost:3030/product/add>

Insert a product into the database. Receive the information in the body of product.

- Update a product

```
// Update
router1.put('/update', (req, res) => [
  let product_id = req.body.product.PROD_ID;
  let product = req.body.product;
  console.log(product_id);
  console.log(product);
  if (!product || !product_id) {
    return res.status(400).send({
      error: true,
      message: 'Please provide Product information'
    });
  }
  dbConn.query("UPDATE product SET ? WHERE PROD_ID = ?", [product, product_id], function (error, results) {
    if (error) throw error;
    return res.send({
      error: false,
      data: results.affectedRows,
      message: 'Product has been updated successfully'
    });
  });
]);
```

Method: PUT

URL: <http://localhost:3030/product/update>

Update product information of the products that are already exist in the database based on the given product ID. Receive the information in the body of product.

- Delete a product

```
// Delete
router1.delete('/delete', (req, res) => {
  let product_id = req.body.product.PROD_ID;
  console.log(product_id);
  if (!product_id) {
    return res.status(400).send({
      error: true,
      message: 'Please provide Product ID'
    });
  }
  dbConn.query("DELETE FROM product WHERE PROD_ID = ?", [product_id], function (error, results) {
    if (error) throw error;
    return res.send({
      error: false,
      data: results.affectedRows,
      message: 'Product has been deleted successfully'
    });
  });
});
```

Method: DELETE

URL: <http://localhost:3030/product/delete>

Delete a product that are already exist in the database based on the given product ID. Receive the product ID in the body of product.

3. Functions for administrators to search/view, insert, update, and delete administrators' information

- View all admin information

```
router3.get('/admins', (req, res) => {
  dbConn.query("SELECT * FROM admin_info", function (error, results) {
    if (error) throw error;
    return res.send({
      error: false,
      data: results,
      message: 'Administrator list.'
    });
  });
});
```

Method: GET

URL: <http://localhost:3030/admin/admins>

Search all the admin information in the database.

- Search by three criteria: Admin Number, Admin First Name, and Admin Last Name

Search by Number	
<pre>router3.get('/Id/:num', (req, res) => { let ad_num = req.params.num; console.log(ad_num); if (!ad_num) { return res.status(400).send({ error: true, message: 'Please provide admin number' }); } dbConn.query(`SELECT * FROM admin_info WHERE AD_NUM LIKE "%\${ad_num}%"`, function (error, results) { if (error) throw error; return res.send({ error: false, data: results, message: 'Admin information retrieved' }); }); });</pre>	<p>Method: GET URL: http://localhost:3030/admin/Id/</p> <p>Recieve Number as parameter and check if there is any admin ID that contains the input ID.</p>
Search by First Name	

```
// Search by fname
router3.get('/fname/:fname', (req, res) => {
  let admin_fname = req.params.fname;
  console.log(admin_fname);
  if (!admin_fname) {
    return res.status(400).send({
      error: true,
      message: 'Please provide admin first name'
    });
  }
  dbConn.query(`SELECT * FROM admin_info WHERE AD_FNAME LIKE "%${admin_fname}%"`, function (error, results) {
    if (error) throw error;
    return res.send({
      error: false,
      data: results,
      message: 'Admin information retrieved'
    });
  });
});
```

Method: GET
 URL:
<http://localhost:3030/admin/fname/>

Recieve First Name as parameter and check if there is any admin First Name that contains the input First Name.

Search by Last Name

```
// Search by lname
router3.get('/lname/:lname', (req, res) => {
  let admin_lname = req.params.lname;
  console.log(admin_lname);
  if (!admin_lname) {
    return res.status(400).send({
      error: true,
      message: 'Please provide admin last name'
    });
  }
  dbConn.query(`SELECT * FROM admin_info WHERE AD_LNAME LIKE "%${admin_lname}%"`, function (error, results) {
    if (error) throw error;
    return res.send({
      error: false,
      data: results,
      message: 'Admin information retrieved'
    });
  });
});
```

Method: GET
 URL:
<http://localhost:3030/admin/lname/>

Recieve Last Name as parameter and check if there is any admin Last Name that contains the input Last Name.

Search by Num and First name

```
router3.get('/IdFName/:num/:fname', (req, res) => {
  let ad_num = req.params.num;
  console.log(ad_num);
  let ad_fname = req.params.fname;
  console.log(ad_fname);
  if (!ad_num) {
    return res.status(400).send({
      error: true,
      message: 'Please provide admin number'
    });
  }
  if (!ad_fname) {
    return res.status(400).send({
      error: true,
      message: 'Please provide admin first name'
    });
  }
  dbConn.query(`SELECT * FROM admin_info WHERE AD_NUM LIKE "%${ad_num}%" OR AD_FNAME LIKE "%${ad_fname}%"`, function (error, results) {
    if (error) throw error;
    return res.send({
      error: false,
      data: results,
      message: 'Admin information retrieved'
    });
  });
});
```

Method: GET
 URL:
<http://localhost:3030/admin/IdFName/>

Recieve Num and First Name as parameters and check if there is any admin that contains the input Num or First Name.

Search by Num and Last name

```

router3.get('/IdLName/:num/:lname', (req, res) => {
  let ad_num = req.params.num;
  console.log(ad_num);
  let ad_lname = req.params.lname;
  console.log(ad_lname);
  if (!ad_num) {
    return res.status(400).send({
      error: true,
      message: 'Please provide admin number'
    });
  }
  if (!ad_lname) {
    return res.status(400).send({
      error: true,
      message: 'Please provide admin last name'
    });
  }
  dbConn.query('SELECT * FROM admin_info WHERE AD_NUM LIKE "%${ad_num}%" OR AD_LNAME LIKE "%${ad_lname}%"', function (error, results) {
    if (error) throw error;
    return res.send({
      error: false,
      data: results,
      message: 'Admin information retrieved'
    });
  });
});

```

Method: GET
URL:
<http://localhost:3030/admin/IdLName/>

Recieve Num and Last Name as parameters and check if there is any admin that contains the input Num or Last Name.

Search by First name and Last name

```

router3.get('/FNameLName/:fname/:lname', (req, res) => {
  let ad_fname = req.params.fname;
  console.log(ad_fname);
  let ad_lname = req.params.lname;
  console.log(ad_lname);
  if (!ad_fname) {
    return res.status(400).send({
      error: true,
      message: 'Please provide admin first name'
    });
  }
  if (!ad_lname) {
    return res.status(400).send({
      error: true,
      message: 'Please provide admin last name'
    });
  }
  dbConn.query('SELECT * FROM admin_info WHERE AD_FNAME LIKE "%${ad_fname}%" OR AD_LNAME LIKE "%${ad_lname}%"', function (error, results) {
    if (error) throw error;
    return res.send({
      error: false,
      data: results,
      message: 'Admin information retrieved'
    });
  });
});

```

Method: GET
URL:
<http://localhost:3030/admin/FNameLName/>

Recieve First Name and Last Name as parameters and check if there is any admin that contains the input First Name or Last Name.

Search by Num, First name, and Last name

```

router3.get('/IdFNameLName/:num/:fname/:lname', (req, res) => {
  let ad_num = req.params.num;
  console.log(ad_num);
  let ad_fname = req.params.fname;
  console.log(ad_fname);
  let ad_lname = req.params.lname;
  console.log(ad_lname);
  if (!ad_num) {
    return res.status(400).send({
      error: true,
      message: 'Please provide admin number'
    });
  }
  if (!ad_fname) {
    return res.status(400).send({
      error: true,
      message: 'Please provide admin first name'
    });
  }
  if (!ad_lname) {
    return res.status(400).send({
      error: true,
      message: 'Please provide admin last name'
    });
  }
  dbConn.query('SELECT * FROM admin_info WHERE AD_NUM LIKE "%${ad_num}%" OR AD_FNAME LIKE "%${ad_fname}%" OR AD_LNAME LIKE "%${ad_lname}%"', function (error, results) {
    if (error) throw error;
    return res.send({
      error: false,
      data: results,
      message: 'Admin information retrieved'
    });
  });
});

```

Method: GET
URL:
<http://localhost:3030/admin/IdFNameLName/>

Recieve Num, First Name, and Last Name as parameters and check if there is any admin that contains the input Num or First Name or Last Name.

- Insert new admin

```
router3.post('/add', (req, res) => {
  let adminInfo = req.body.Admin_info;
  let adminLogin = req.body.Admin_login;
  if (!adminInfo || !adminLogin) {
    return res.status(400).send({
      error: true,
      message: 'Please provide new admin information'
    });
  }
  dbConn.query("INSERT INTO admin_info SET ?", [adminInfo], function (error, results) {
    if (error) throw error;
  });
  dbConn.query("INSERT INTO admin_login SET ?", [adminLogin], function (error, results) {
    if (error) throw error;
  });
  return res.send({
    error: false,
    message: 'New administrator information has been created successfully'
  });
});
```

Method: POST

URL: <http://localhost:3030/admin/add>

Insert a new administrator into the database. Receive the information in the body of Admin_info and Admin_login. The information have to be separate because the information of admin is divided into two database tables.

- Update an admin

```
// Update
router3.put('/update', (req, res) => {
  let admin_num = req.body.Admin_info.AD_NUM;
  let admin = req.body.Admin_info;
  console.log(admin_num);
  console.log(admin);
  if (!admin || !admin_num) {
    return res.status(400).send({
      error: true,
      message: 'Please provide admin information'
    });
  }
  dbConn.query("UPDATE admin_info SET ? WHERE AD_NUM = ?", [admin, admin_num], function (error, results) {
    if (error) throw error;
    return res.send({
      error: false,
      data: results.affectedRows,
      message: 'Administrator information has been updated successfully'
    });
});
```

Method: PUT

URL: <http://localhost:3030/admin/update>

Update information of administrator that already existed in the database based on the given Admin Number. Receive the information in the body of Admin_info.

- Delete an admin

```
// Delete
router3.delete('/delete', (req, res) => {
  let admin_num = req.body.Admin.AD_NUM;
  console.log(admin_num);
  if (!admin_num) {
    return res.status(400).send({
      error: true,
      message: 'Please provide Admin number'
    });
  }
  dbConn.query("DELETE FROM admin_login WHERE AD_NUM = ?", [admin_num], function (error, results) {
    if (error) throw error;
  });
  dbConn.query("DELETE FROM admin_info WHERE AD_NUM = ?", [admin_num], function (error, results) {
    if (error) throw error;
  });
  return res.send({
    error: false,
    message: 'Administrator information has been deleted successfully'
  });
});
```

Method: DELETE

URL: <http://localhost:3030/admin/delete>

Delete information of administrator based on the given Admin Number. Receive the information in the body of Admin.

Javascript to call web services from backend to be interactive in the frontend

```
async function callWS(url, method, sentData = {}) {
    console.log("Calling backend web service awdawd")
    let data;
    if (method == "select") {
        console.log("select")
        let response = await fetch(url, {
            method: 'GET'
        });
        data = await response.json();
    }
    else if (method == "insert") {
        let response = await fetch(url, {
            method: 'POST',
            headers: { 'Content-Type': 'application/json' },
            body: JSON.stringify(sentData)
        });
        data = await response.json();
    }
    else if (method == "update") {
        let response = await fetch(url, {
            method: 'PUT',
            headers: { 'Content-Type': 'application/json' },
            body: JSON.stringify(sentData)
        });
        data = await response.json();
    }
    else if (method == "delete") {
        let response = await fetch(url, {
            method: 'DELETE',
            headers: { 'Content-Type': 'application/json' },
            body: JSON.stringify(sentData)
        });
        data = await response.json();
    }
    return data;
}
```

Function to call web service to execute select, insert, update, and delete.

```

const rootURLAdminLogin = "http://localhost:3030/admin-login";
async function loginHandler() {
    const adminEmail = document.querySelector("#email").value;
    const adminPass = document.querySelector("#password").value;
    const data = {
        username: adminEmail,
        password: adminPass
    };
    const response = await callWS(rootURLAdminLogin, "insert", data);
    if (response.status === 200) {
        console.log("Login successful");
        alert("Login successful");
        window.location.href = "/adminhome";
    } else if (response.status === 401) {
        console.log(response.message);
        alert('Invalid, Please try again');
    }
    else{
        alert('Please enter username and password');
    }
}

```

This is an example for using function callWS() to call web service to perform login authentication. Once the response status is sent from the backend, it will check if the response is successful (200). If it is successful, it will alert “Login successful and direct to Welcome Admin Page”. If not it will alert “Invalid, Please try again”.

```

const rootURLPd = "http://localhost:3030/product";
let searchBtnRef = document.querySelector("#Submit");
searchBtnRef.addEventListener("click", () => [
    const prodId = document.querySelector("#Id").value;
    const prodName = document.querySelector("#Name").value;
    const prodBrand = document.querySelector("#Brand").value;
    const outputHtml = document.getElementById("output");
    // Select all
    if (!prodId && !prodName && !prodBrand) {
        callWS(rootURLPd + "/products", "select").then((data) => {
            console.log(data);
            outputHtml.innerHTML = '';
            if (data.data.length > 0) {
                alert(data.message);
                let output;
                output = "<h1 class='list'>Product List<h1>";
                output += "<article class='container'>";
                data.data.forEach(element) => {
                    output += "<article class='column'>";
                    output += "<h2>" + element.PROD_NAME + "</h2>";
                    output += "<p> ID: " + element.PROD_ID + "</p>";
                    output += "<p> Brand: " + element.PROD_BRAND + "</p>";
                    output += "<p> Description: " + element.PROD_DES + "</p>";
                    output += "<p> Type: " + element.PROD_TYPE + "</p>";
                    output += "<p> Color: " + element.PROD_COLOR + "</p>";
                    output += "<p class='price'>Price: " + element.PROD_PRICE + "</p>";
                    output += "<p> Quantity: " + element.PROD_QUANTITY + "</p>";
                    output += "<p> Size: " + element.PROD_SIZE + "</p>";
                    output += "</article>";
                });
                output += "</article>";
                console.log(data.data.length);
                outputHtml.innerHTML = output;
            }
        });
    }
}

```

This is an example for using function callWS() to call web service to perform select. Once the response is sent from the backend, it will appear in the web page by using innerHTML.

Testing results of web services

Test Cases for admin login authentication

The screenshot shows the Postman application interface. At the top, there is a header bar with 'POST' selected as the method, the URL 'http://localhost:3030/admin-login/' entered, and a 'Send' button. Below the header are tabs for 'Params', 'Authorization', 'Headers (8)', 'Body' (which is currently selected), 'Pre-request Script', 'Tests', and 'Settings'. Under the 'Body' tab, there are options for 'none', 'form-data', 'x-www-form-urlencoded', 'raw' (which is selected), 'binary', 'GraphQL', and 'JSON'. The 'JSON' dropdown is set to 'Pretty'. The raw body content is:

```
1 {  
2   "username": "ress.aaa@gmail.com",  
3   "password": "Somchai655"  
4 }
```

At the bottom of the interface, there are tabs for 'Body', 'Cookies', 'Headers (8)', and 'Test Results'. The 'Test Results' tab is selected, showing the response details. The status is '401 Unauthorized' with a response time of '10 ms' and a size of '331 B'. There is a 'Save as Example' button and a 'More' button. Below the status, there are tabs for 'Pretty' (selected), 'Raw', 'Preview', 'Visualize', and 'JSON' (with a dropdown arrow). The JSON response is:

```
1 {  
2   "error": true,  
3   "message": "Invalid log in",  
4   "status": 401  
5 }
```

The screenshot shows a Postman interface with the following details:

- Method:** POST
- URL:** <http://localhost:3030/admin-login/>
- Body (JSON):**

```
1 {  
2   ... "username": "yang.non@multiclothes.com",  
3   ... "password": "nyaungyooka"  
4 }
```
- Response Status:** 200 OK
- Response Time:** 7 ms
- Response Size:** 359 B
- Response Content:**

```
1 "error": false,  
2 "data": [  
3   {  
4     "AD_NUM": "006000202"  
5   }  
6 ],  
7 "message": "Successfully log in",  
8 "status": 200  
9  
10
```

Test Cases for product management

Search All

GET http://localhost:3030/product/products

Params	Authorization	Headers (6)	Body	Pre-request Script	Tests	Settings	Cookies
Body	Cookies	Headers (8)	Test Results				
				200 OK 16 ms 2.38 KB	Save as Example	...	
Pretty	Raw	Preview	Visualize	JSON			
<pre> 1 2 "error": false, 3 "data": [4 5 { 6 "PROD_ID": "AB1111167890", 7 "PROD_NAME": "Mens MB", 8 "PROD_BRAND": "FILA", 9 "PROD_DES": "CIconic midsole design and printed graphic motif on quarter with embroidered FILA 10 logos on quarter, tongue, toebox, and heel.\n\tComposition\n\tLeather\n\tSynthetic", 11 "PROD_TYPE": "Footwear", 12 "PROD_COLOR": "107 EGRET-GRAY MIST", 13 "PROD_PRICE": "95.00", 14 "PROD_QUANTITY": 130, 15 "PROD_SIZE": "40" 16 }, 17 { 18 "PROD_ID": "AB1111567890", 19 "PROD_NAME": "Heart Print T-Shirt", 20 "PROD_BRAND": "ZARA", 21 "PROD_DES": "T-shirt with a round neckline and short sleeves. Contrast prints on the front and 22 back.\n\tComposition\n\tCotton 100%", 23 "PROD_TYPE": "Cloth", 24 "PROD_COLOR": "Crimson", 25 "PROD_PRICE": "29.99", 26 "PROD_QUANTITY": 150, 27 "PROD_SIZE": "XL" 28] </pre>							

Search by ID

http://localhost:3030/product/id/AB1134567890

Params	Authorization	Headers (6)	Body	Pre-request Script	Tests	Settings	Cookies
Body	Cookies	Headers (8)	Test Results				
				200 OK 10 ms 832 B	Save as Example	...	
Pretty	Raw	Preview	Visualize	JSON			
<pre> 1 2 "error": false, 3 "data": [4 5 { 6 "PROD_ID": "AB1134567890", 7 "PROD_NAME": "Canvas Hi-top Trainers", 8 "PROD_BRAND": "H&M", 9 "PROD_DES": "CONSCIOUS CHOICE Hi-tops in sturdy cotton canvas with a tongue and lacing at the 10 front, and metal eyelets in one side. Cotton canvas linings and insoles and rubber soles 11 that are patterned underneath. Height of soles 3.1 cm.\n\tComposition\n\tUpper Cotton 12 100%\n\tOutersole Rubber 100%\n\tLining sock Cotton 100%", 13 "PROD_TYPE": "Footwear", 14 "PROD_COLOR": "Black", 15 "PROD_PRICE": "33.00", 16 "PROD_QUANTITY": 50, 17 "PROD_SIZE": "36" 18], 19 "message": "Product retrieved" 20] </pre>							

	<pre> GET http://localhost:3030/product/id/AB1111167890 Send Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies Body Cookies Headers (8) Test Results 200 OK 6 ms 686 B Save as Example ... Pretty Raw Preview Visualize JSON 1 2 "error": false, 3 "data": [4 { 5 "PROD_ID": "AB1111167890", 6 "PROD_NAME": "Mens MB", 7 "PROD_BRAND": "FILA", 8 "PROD_DESC": "CIconic midssole design and printed graphic motif on quarter with embroidered FILA logos on quarter, tongue, toebox, and heel.\n\tComposition\n\tLeather\n\tSynthetic", 9 "PROD_TYPE": "Footwear", 10 "PROD_COLOR": "107 EGRET-GRAY MIST", 11 "PROD_PRICE": "95.00", 12 "PROD_QUANTITY": 130, 13 "PROD_SIZE": "40" 14 }, 15], 16 "message": "Product retrieved" 17 </pre>
Search by Brand	<pre> GET http://localhost:3030/product/brand/zara Send Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies Body Cookies Headers (8) Test Results 200 OK 5 ms 1.04 KB Save as Example ... Pretty Raw Preview Visualize JSON 1 2 "error": false, 3 "data": [4 { 5 "PROD_ID": "AB111167890", 6 "PROD_NAME": "Heart Print T-Shirt", 7 "PROD_BRAND": "ZARA", 8 "PROD_DESC": "T-shirt with a round neckline and short sleeves. Contrast prints on the front and back.\n\tComposition\n\tCotton 100%", 9 "PROD_TYPE": "Cloth", 10 "PROD_COLOR": "Crimson", 11 "PROD_PRICE": "29.99", 12 "PROD_QUANTITY": 150, 13 "PROD_SIZE": "XL" 14 }, 15 { 16 "PROD_ID": "AB1114567890", 17 "PROD_NAME": "Fitted Double-Breasted Blazer", 18 "PROD_BRAND": "ZARA", 19 "PROD_DESC": "Tailored blazer with a lapel collar and long sleeves. Front flap pockets and a welt chest pocket. Matching lining. Double-breasted button fastening at the front.\n\tComposition\n\tPolyester 68%\n\tViscose 29%\n\tElastane 3%", 20 "PROD_TYPE": "Cloth", 21 "PROD_COLOR": "White", 22 "PROD_PRICE": "120.00", 23 "PROD_QUANTITY": 20, 24 "PROD_SIZE": "XXS" 25 } 26], 27 "message": "Product retrieved" 28 </pre>

	<p>GET http://localhost:3030/product/brand/fila</p> <p>Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies</p> <p>Body Cookies Headers (8) Test Results</p> <p>Pretty Raw Preview Visualize JSON </p> <pre> 1 "error": false, 2 "data": [3 { 4 "PROD_ID": "AB1111167890", 5 "PROD_NAME": "Mens MB", 6 "PROD_BRAND": "FILA", 7 "PROD_DESC": "CIconic midsole design and printed graphic motif on quarter with embroidered FILA logos on quarter, tongue, toebox, and heel.\n\tComposition\n\tLeather\n\tSynthetic", 8 "PROD_TYPE": "Footwear", 9 "PROD_COLOR": "107 EGRET-GRAY MIST", 10 "PROD_PRICE": "95.00", 11 "PROD_QUANTITY": 130, 12 "PROD_SIZE": "40" 13 }, 14], 15], 16 "message": "Product retrieved" 17 </pre>
Search by name	<p>GET http://localhost:3030/product/name/Heart Print T-Shirt</p> <p>Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies</p> <p>Body Cookies Headers (8) Test Results</p> <p>Pretty Raw Preview Visualize JSON </p> <pre> 1 "error": false, 2 "data": [3 { 4 "PROD_ID": "AB11111567890", 5 "PROD_NAME": "Heart Print T-Shirt", 6 "PROD_BRAND": "ZARA", 7 "PROD_DESC": "T-shirt with a round neckline and short sleeves. Contrast prints on the front and back.\n\tComposition\n\tCotton 100%", 8 "PROD_TYPE": "Cloth", 9 "PROD_COLOR": "Crimson", 10 "PROD_PRICE": "29.99", 11 "PROD_QUANTITY": 160, 12 "PROD_SIZE": "XL" 13 }, 14], 15], 16 "message": "Product retrieved" 17 </pre>

	<pre> GET http://localhost:3030/product/name/Oversized V-neck Sweatshirt { "error": false, "data": [{ "PROD_ID": "AB1234567890", "PROD_NAME": "Oversized V-neck Sweatshirt", "PROD_BRAND": "H&M", "PROD_DES": "CONSCIOUS CHOICE Oversized top in sweatshirt fabric made from a cotton blend with a soft brushed inside. V-neck, low dropped shoulders and long sleeves. Ribbing around the neckline and cuffs and a raw, roll-edge hem.\nComposition\nRib Cotton 95%\nElastane 5%\\ntShell Cotton 62%\nPolyester 38%", "PROD_TYPE": "Cloth", "PROD_COLOR": "Black-White", "PROD_PRICE": "29.99", "PROD_QUANTITY": 100, "PROD_SIZE": "S" }], "message": "Product retrieved" } </pre>
Search by ID and name	<pre> GET http://localhost:3030/product/idname/AB1234567890/Canvas { "error": false, "data": [{ "PROD_ID": "AB1134567890", "PROD_NAME": "Canvas Hi-top Trainers", "PROD_BRAND": "H&M", "PROD_DES": "CONSCIOUS CHOICE Hi-tops in sturdy cotton canvas with a tongue and lacing at the front, and metal eyelets in one side. Cotton canvas linings and insoles and rubber soles that are patterned underneath. Height of soles 3.1 cm.\nComposition\nUpper Cotton 100%\nOutersole Rubber 100%\nLiningsock Cotton 100%", "PROD_TYPE": "Footwear", "PROD_COLOR": "Black", "PROD_PRICE": "33.00", "PROD_QUANTITY": 50, "PROD_SIZE": "35" }], "message": "Product retrieved" } </pre>

Search by Id and brand

The screenshot shows the Postman interface with the following details:

- Method:** GET
- URL:** http://localhost:3030/product/idbrand/1567890/ZARA
- Body:** JSON (Pretty printed)


```

1
2 "error": false,
3 "data": [
4   {
5     "PROD_ID": "AB1114567890",
6     "PROD_NAME": "Fitted Double-Breasted Blazer",
7     "PROD_BRAND": "ZARA",
8     "PROD_DESC": "Tailored blazer with a lapel collar and long sleeves. Front flap pockets and a welt chest pocket. Matching lining.\nDouble-breasted button fastening at the front.\n\\tComposition\\n\\tPolyester 68%\\nViscose 29%\\n\\tElastane 3%",
9     "PROD_TYPE": "Cloth",
10    "PROD_COLOR": "White",
11    "PROD_PRICE": "120.00",
12    "PROD_QUANTITY": 20,
13    "PROD_SIZE": "XXS"
14  },
15  {
16    "PROD_ID": "AB1134567890",
17    "PROD_NAME": "Canvas Hi-top Trainers".
      
```
- Response Status:** 200 OK
- Time:** 17 ms
- Size:** 636 B

	<p>GET http://localhost:3030/product/idbrand/AB111116/H&M</p> <p>Send</p> <p>Params Auth Headers (6) Body Pre-req. Tests Settings Cookies</p> <p>Query Params</p> <table border="1"> <thead> <tr> <th>Key</th> <th>Value</th> <th>Description</th> <th>...</th> <th>Bulk Edit</th> </tr> <tr> <th>Key</th> <th>Value</th> <th colspan="3">Description</th> </tr> </thead> <tbody> <tr> <td></td> <td></td> <td colspan="3"></td> </tr> </tbody> </table> <p>Body Pretty Raw Preview Visualize JSON Copy Save as Example</p> <pre> 1 2 "error": false, 3 "data": [4 { 5 "PROD_ID": "AB1111167890", 6 "PROD_NAME": "Mens MB", 7 "PROD_BRAND": "FILA", 8 "PROD_DES": "CIconic midsole design and printed graphic motif on quarter with embroidered FILA logos on quarter, tongue, toebox, and heel. \n\tComposition\n\tLeather\n\tSynthetic", 9 "PROD_TYPE": "Footwear", 10 "PROD_COLOR": "107 EGRET-GRAY MIST", 11 "PROD_PRICE": "95.00", 12 "PROD_QUANTITY": 130, 13 "PROD_SIZE": "40" 14 }, 15], 16 "message": "Product retrieved" 17 </pre>	Key	Value	Description	...	Bulk Edit	Key	Value	Description							
Key	Value	Description	...	Bulk Edit												
Key	Value	Description														
Search by name and brand	<p>GET http://localhost:3030/product/namebrand/Double/H&M</p> <p>Send</p> <p>Params Auth Headers (6) Body Pre-req. Tests Settings Cookies</p> <p>Query Params</p> <table border="1"> <thead> <tr> <th>Key</th> <th>Value</th> <th>Description</th> <th>...</th> <th>Bulk Edit</th> </tr> <tr> <th>Key</th> <th>Value</th> <th colspan="3">Description</th> </tr> </thead> <tbody> <tr> <td></td> <td></td> <td colspan="3"></td> </tr> </tbody> </table> <p>Body Pretty Raw Preview Visualize JSON Copy Save as Example</p> <pre> 1 2 "error": false, 3 "data": [4 { 5 "PROD_ID": "AB1114567890", 6 "PROD_NAME": "Fitted Double-Breasted Blazer", 7 "PROD_BRAND": "ZARA", 8 "PROD_DES": "Tailored blazer with a lapel collar and long sleeves. Front flap pockets and a welt chest pocket. Matching lining. Double-breasted button fastening at the front. \n\tComposition\n\tPolyester 68%\n\tViscose 29%\n\tElastane 3%", 9 "PROD_TYPE": "Cloth", 10 "PROD_COLOR": "White", 11 "PROD_PRICE": "120.00", 12 "PROD_QUANTITY": 20, 13 "PROD_SIZE": "Xxs" 14 }, 15 { 16 "PROD_ID": "AB1134567890", 17 "PROD_NAME": "Canvas Hi-top Trainers". </pre>	Key	Value	Description	...	Bulk Edit	Key	Value	Description							
Key	Value	Description	...	Bulk Edit												
Key	Value	Description														

	<p>GET http://localhost:3030/product/namebrand/Mens/FILA</p> <p>Send Cookies</p> <p>Params Auth Headers (6) Body Pre-req. Tests Settings</p> <p>Query Params</p> <table border="1"> <thead> <tr> <th>Key</th> <th>Value</th> <th>Description</th> <th>...</th> <th>Bulk Edit</th> </tr> </thead> <tbody> <tr> <td>Key</td> <td>Value</td> <td>Description</td> <td>...</td> <td></td> </tr> </tbody> </table> <p>Body Pretty Raw Preview Visualize JSON Copy Save as Example ...</p> <pre> 1 2 "error": false, 3 "data": [4 { 5 "PROD_ID": "AB1111167890", 6 "PROD_NAME": "Mens MB", 7 "PROD_BRAND": "FILA", 8 "PROD_DESC": "CIconic midsole design and printed graphic motif on quarter with embroidered FILA logos on quarter, tongue, toebox, and heel. \n\tComposition\n\tLeather\n\tSynthetic", 9 "PROD_TYPE": "Footwear", 10 "PROD_COLOR": "107 EGRET-GRAY MIST", 11 "PROD_PRICE": "95.00", 12 "PROD_QUANTITY": 130, 13 "PROD_SIZE": "40" 14 }, 15], 16 "message": "Product retrieved" 17 </pre>	Key	Value	Description	...	Bulk Edit	Key	Value	Description	...	
Key	Value	Description	...	Bulk Edit							
Key	Value	Description	...								
Search by Id, name, and brand	<p>GET http://localhost:3030/product/all/AB1111167890/Double/H&M</p> <p>Send Cookies</p> <p>Params Auth Headers (6) Body Pre-req. Tests Settings</p> <p>Query Params</p> <table border="1"> <thead> <tr> <th>Key</th> <th>Value</th> <th>Description</th> <th>...</th> <th>Bulk Edit</th> </tr> </thead> <tbody> <tr> <td>Key</td> <td>Value</td> <td>Description</td> <td>...</td> <td></td> </tr> </tbody> </table> <p>Body Pretty Raw Preview Visualize JSON Copy Save as Example ...</p> <pre> 1 2 "error": false, 3 "data": [4 { 5 "PROD_ID": "AB1111167890", 6 "PROD_NAME": "Mens MB", 7 "PROD_BRAND": "FILA", 8 "PROD_DESC": "CIconic midsole design and printed graphic motif on quarter with embroidered FILA logos on quarter, tongue, toebox, and heel. \n\tComposition\n\tLeather\n\tSynthetic", 9 "PROD_TYPE": "Footwear", 10 "PROD_COLOR": "107 EGRET-GRAY MIST", 11 "PROD_PRICE": "95.00", 12 "PROD_QUANTITY": 130, 13 "PROD_SIZE": "40" 14 }, 15 { 16 "PROD_ID": "AB1114567890", 17 "PROD_NAME": "Fitted Double-Breasted Blazer", 18 "PROD_BRAND": "ZARA". </pre>	Key	Value	Description	...	Bulk Edit	Key	Value	Description	...	
Key	Value	Description	...	Bulk Edit							
Key	Value	Description	...								

Insert

GET Send http://localhost:3030/product/all/4567890/Oversized/ZARA

Params Auth Headers (6) Body Pre-req. Tests Settings Cookies

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description	...	

Body 200 OK 35 ms 2.03 KB Save as Example

Pretty Raw Preview Visualize JSON View more ac

```

1
2 "error": false,
3 "data": [
4   {
5     "PROD_ID": "AB1111567890",
6     "PROD_NAME": "Heart Print T-Shirt",
7     "PROD_BRAND": "ZARA",
8     "PROD_DESC": "T-shirt with a round neckline and short sleeves. Contrast prints on the front and back.\n\tComposition\n\tCotton 100%",
9     "PROD_TYPE": "Cloth",
10    "PROD_COLOR": "Crimson",
11    "PROD_PRICE": "29.99",
12    "PROD_QUANTITY": 150,
13    "PROD_SIZE": "XL"
14  },
15  {
16    "PROD_ID": "AB1114567890",
17    "PROD_NAME": "Fitted Double-Breasted Blazer",
18    "PROD_BRAND": "ZARA",
19    "PROD_DESC": "Tailored blazer with a lapel collar and long sleeves. Front = "
20  }
]

```

POST Send http://localhost:3030/product/add

Params Authorization Headers (8) Body ● Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

Body Cookies Headers (8) Test Results 200 OK 20 ms 345 B Save as Example

Pretty Raw Preview Visualize JSON □ □

```

1 {
2   "product": {
3     "PROD_ID": "AB2224567890",
4     "PROD_NAME": "Oversized Sweatshirt",
5     "PROD_BRAND": "H&M",
6     "PROD_DESC": "Oversized top in sweatshirt fabric made from a cotton blend with a soft brushed inside",
7     "PROD_TYPE": "Cloth",
8     "PROD_COLOR": "White",
9     "PROD_PRICE": "55.00",
10    "PROD_QUANTITY": 120,
11    "PROD_SIZE": "S"
12  }
13 }

```

```

1
2 "error": false,
3 "data": 1,
4 "message": "New product has been created successfully"
5

```

38

	<p>POST <input type="text" value="http://localhost:3030/product/add"/> Send</p> <p>Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies</p> <p>none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify</p> <pre> 1 { 2 ... 3 "product": { 4 ... 5 "PROD_ID": "AB2221234590", 6 "PROD_NAME": "pants", 7 "PROD_BRAND": "nike", 8 "PROD_DESC": "lightweight, stretchy knit fabric perfect for warmer months.", 9 "PROD_TYPE": "Cloth", 10 "PROD_COLOR": "black", 11 "PROD_PRICE": "80.00", 12 "PROD_QUANTITY": 70, 13 "PROD_SIZE": "M" 14 } 15 }</pre> <p>Body Cookies Headers (8) Test Results ⌚ 200 OK 9 ms 345 B 📁 Save as Example ⚙️</p> <p>Pretty Raw Preview Visualize JSON JSON</p> <pre> 1 { 2 ... 3 "error": false, 4 "data": 1, 5 "message": "New product has been created successfully" 6 }</pre>
Update	<p>PUT <input type="text" value="http://localhost:3030/product/update"/> Send</p> <p>Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies</p> <p>none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify</p> <pre> 1 { 2 ... 3 "product": { 4 ... 5 "PROD_ID": "AB2224567890", 6 "PROD_NAME": "Oversized T-shirt" 7 } 8 }</pre> <p>Body Cookies Headers (8) Test Results ⌚ 200 OK 7 ms 341 B 📁 Save as Example ⚙️</p> <p>Pretty Raw Preview Visualize JSON JSON</p> <pre> 1 { 2 ... 3 "error": false, 4 "data": 1, 5 "message": "Product has been updated successfully" 6 }</pre>

Delete

PUT Send http://localhost:3030/product/update

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```

1 {
2     "product": {
3         "PROD_ID": "AB1111167890",
4         "PROD_NAME": "Men's sport training shoes",
5         "PROD_BRAND": "FILA",
6         "PROD_QUANTITY": 120,
7         "PROD_SIZE": "41"
8     }
9 }
```

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON Save as Example

```

1
2     "error": false,
3     "data": 1,
4     "message": "Product has been updated successfully"
5 }
```

The screenshot shows a Postman interface with the following details:

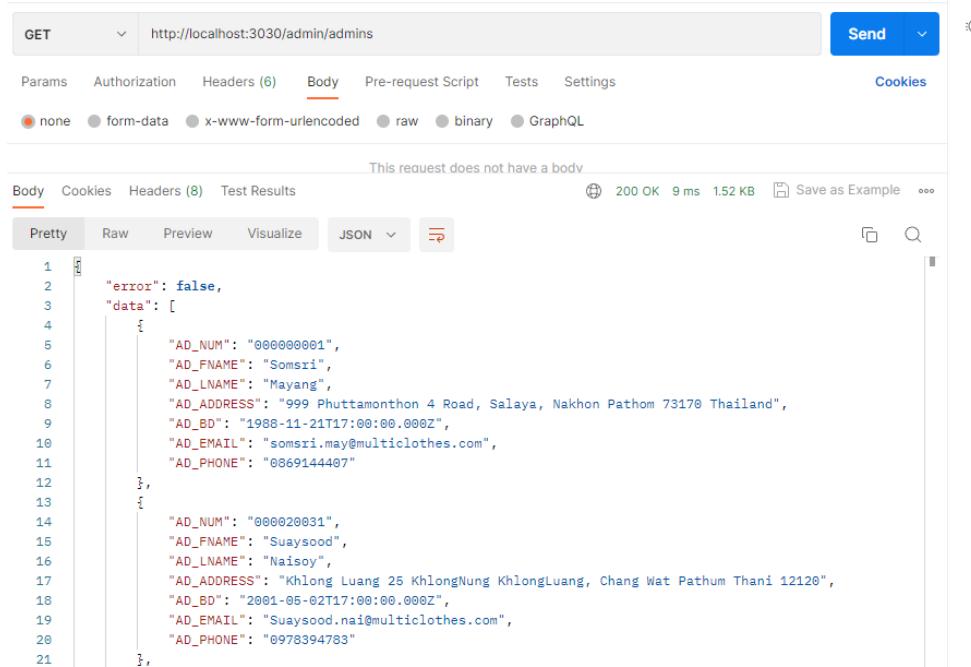
- Method:** DELETE
- URL:** <http://localhost:3030/product/delete>
- Body (JSON):**

```
1 {
2   "product": {
3     "PROD_ID": "AB2221234590"
4   }
5 }
```
- Response Status:** 200 OK
- Response Time:** 10 ms
- Response Size:** 341 B
- Response Body (Pretty JSON):**

```
1 {
2   "error": false,
3   "data": 1,
4   "message": "Product has been deleted successfully"
5 }
```

Test Cases for admin information management

Search All



GET <http://localhost:3030/admin/admins>

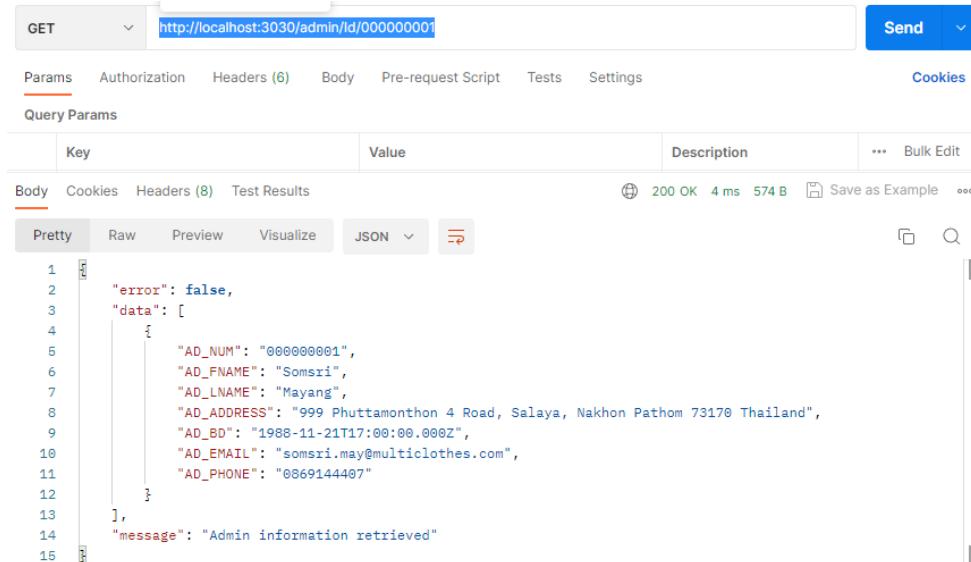
Body

```

1
2   "error": false,
3   "data": [
4     {
5       "AD_NUM": "00000001",
6       "AD_FNAME": "Somsri",
7       "AD_LNAME": "Mayang",
8       "AD_ADDRESS": "999 Phuttamonthon 4 Road, Salaya, Nakhon Pathom 73170 Thailand",
9       "AD_BD": "1988-11-21T17:00:00.000Z",
10      "AD_EMAIL": "somsri.may@multiclothes.com",
11      "AD_PHONE": "0869144407"
12    },
13    {
14      "AD_NUM": "000020031",
15      "AD_FNAME": "Suaysood",
16      "AD_LNAME": "Naisoy",
17      "AD_ADDRESS": "Khlong Luang 25 KhlongNung KhlongLuang, Chang Wat Pathum Thani 12120",
18      "AD_BD": "2001-05-02T17:00:00.000Z",
19      "AD_EMAIL": "Suaysood.nai@multiclothes.com",
20      "AD_PHONE": "0978394783"
21    }
]

```

Search by Num



GET <http://localhost:3030/admin/id/000000001>

Query Params

Key	Value	Description	...	Bulk Edit

Body

```

1
2   "error": false,
3   "data": [
4     {
5       "AD_NUM": "00000001",
6       "AD_FNAME": "Somsri",
7       "AD_LNAME": "Mayang",
8       "AD_ADDRESS": "999 Phuttamonthon 4 Road, Salaya, Nakhon Pathom 73170 Thailand",
9       "AD_BD": "1988-11-21T17:00:00.000Z",
10      "AD_EMAIL": "somsri.may@multiclothes.com",
11      "AD_PHONE": "0869144407"
12    }
],
13   "message": "Admin information retrieved"
14
15

```

	<p>GET http://localhost:3030/admin/Id/006000202</p> <p>Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies</p> <p>Query Params</p> <table border="1"> <thead> <tr> <th>Key</th> <th>Value</th> <th>Description</th> <th>...</th> <th>Bulk Edit</th> </tr> </thead> <tbody> <tr> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> </tbody> </table> <p>Body Cookies Headers (8) Test Results</p> <p>Pretty Raw Preview Visualize JSON</p> <pre> 1 2 "error": false, 3 "data": [4 { 5 "AD_NUM": "006000202", 6 "AD_FNAME": "Yang", 7 "AD_LNAME": "Nonyu", 8 "AD_ADDRESS": "23, Trok Rongnamkeang, Yotha Rd., Bangkok, 10100 Thailand", 9 "AD_BD": "1991-01-16T17:00:00.000Z", 10 "AD_EMAIL": "yang.non@multiclothes.com", 11 "AD_PHONE": "0972187739" 12 } 13], 14 "message": "Admin information retrieved" 15 </pre>	Key	Value	Description	...	Bulk Edit					
Key	Value	Description	...	Bulk Edit							
Search by First Name	<p>GET http://localhost:3030/admin/fname/somsri</p> <p>Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies</p> <p>Query Params</p> <table border="1"> <thead> <tr> <th>Key</th> <th>Value</th> <th>Description</th> <th>...</th> <th>Bulk Edit</th> </tr> </thead> <tbody> <tr> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> </tbody> </table> <p>Body Cookies Headers (8) Test Results</p> <p>Pretty Raw Preview Visualize JSON</p> <pre> 1 2 "error": false, 3 "data": [4 { 5 "AD_NUM": "000000001", 6 "AD_FNAME": "Somsri", 7 "AD_LNAME": "Mayang", 8 "AD_ADDRESS": "999 Phuttamonthon 4 Road, Salaya, Nakhon Pathom 73170 Thailand", 9 "AD_BD": "1988-11-21T17:00:00.000Z", 10 "AD_EMAIL": "somsri.may@multiclothes.com", 11 "AD_PHONE": "0869144407" 12 } 13], 14 "message": "Admin information retrieved" 15 </pre>	Key	Value	Description	...	Bulk Edit					
Key	Value	Description	...	Bulk Edit							

GET http://localhost:3030/admin/fname/Chanom

Key	Value	Description	...	Bulk Edit

Body Cookies Headers (8) Test Results

```

1
2   "error": false,
3   "data": [
4     {
5       "AD_NUM": "000032191",
6       "AD_FNAME": "Chanom",
7       "AD_LNAME": "Kaimook",
8       "AD_ADDRESS": "62 Moo 1, Rangsit-Ongkhazak Road (Km.7) Thanyaburi, Pathum Thani",
9       "AD_BD": "1999-09-11T17:00:00.000Z",
10      "AD_EMAIL": "Chanom.kai@multiclothes.com",
11      "AD_PHONE": "0974328801"
12    },
13  ],
14  "message": "Admin information retrieved"
15

```

Search by Last Name

GET http://localhost:3030/admin/lname/Naisoy

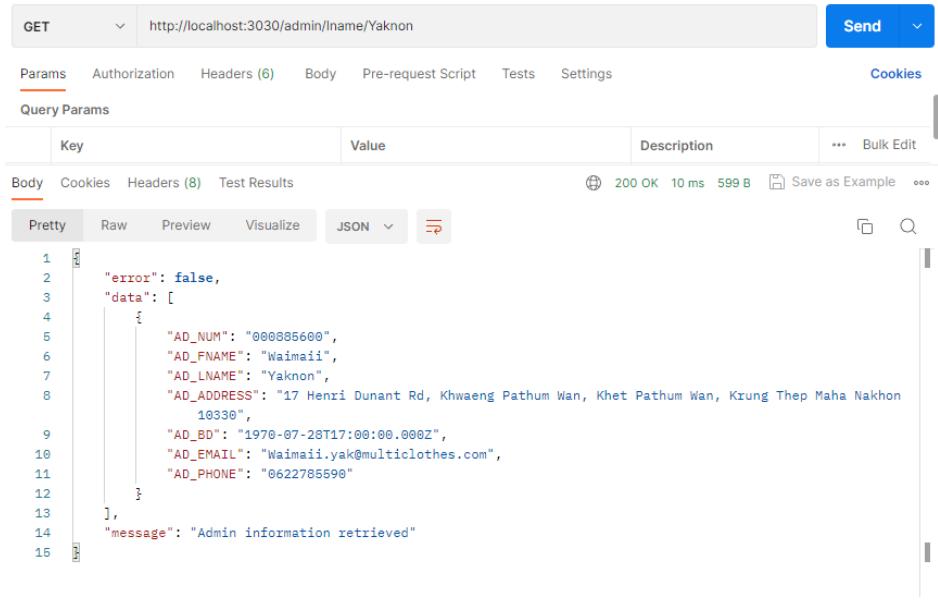
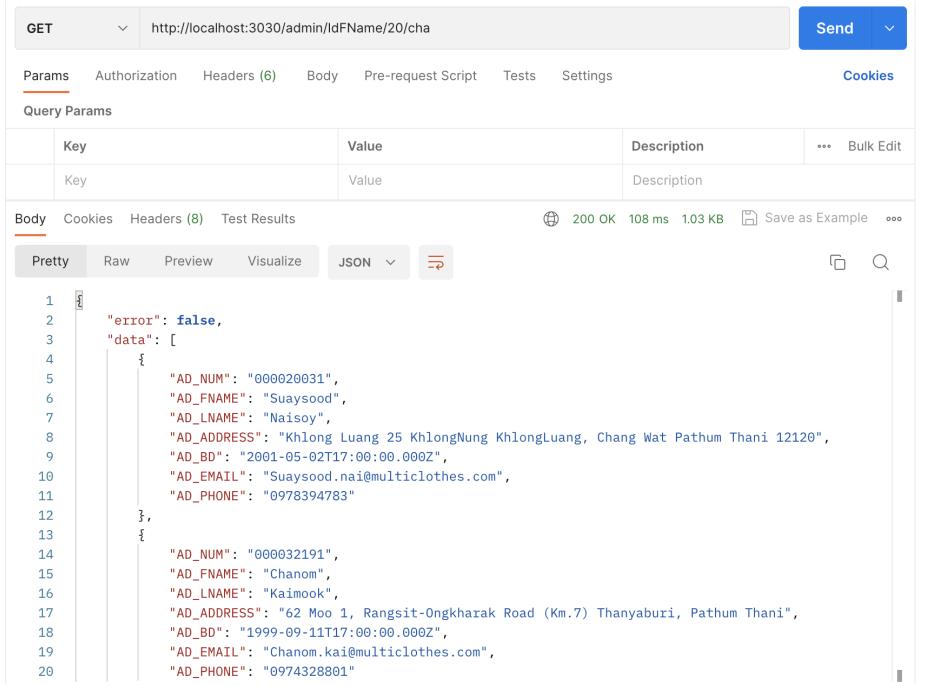
Key	Value	Description	...	Bulk Edit

Body Cookies Headers (8) Test Results

```

1
2   "error": false,
3   "data": [
4     {
5       "AD_NUM": "000020031",
6       "AD_FNAME": "Suaysood",
7       "AD_LNAME": "Naisoy",
8       "AD_ADDRESS": "Khlong Luang 25 KhlongNung KhlongLuang, Chang Wat Pathum Thani 12120",
9       "AD_BD": "2001-05-02T17:00:00.000Z",
10      "AD_EMAIL": "Suaysood.nai@multiclothes.com",
11      "AD_PHONE": "0978394783"
12    },
13  ],
14  "message": "Admin information retrieved"
15

```

	 <pre> 1 2 "error": false, 3 "data": [4 { 5 "AD_NUM": "000085600", 6 "AD_FNAME": "Waimaii", 7 "AD_LNAME": "Yaknon", 8 "AD_ADDRESS": "17 Henzi Dunant Rd, Khwaeng Pathum Wan, Khet Pathum Wan, Krung Thep Maha Nakhon 10330", 9 "AD_BD": "1970-07-28T17:00:00.000Z", 10 "AD_EMAIL": "Waimaii.yak@multipclothes.com", 11 "AD_PHONE": "0622785590" 12 } 13], 14 "message": "Admin information retrieved" 15 </pre>
Search by ID and First Name	 <pre> 1 2 "error": false, 3 "data": [4 { 5 "AD_NUM": "000020031", 6 "AD_FNAME": "Suaysood", 7 "AD_LNAME": "Naisoy", 8 "AD_ADDRESS": "Khlong Luang 25 KhlongNung KhlongLuang, Chang Wat Pathum Thani 12120", 9 "AD_BD": "2001-05-02T17:00:00.000Z", 10 "AD_EMAIL": "Suaysood.nai@multipclothes.com", 11 "AD_PHONE": "0978394783" 12 }, 13 { 14 "AD_NUM": "000032191", 15 "AD_FNAME": "Chanom", 16 "AD_LNAME": "Kaimook", 17 "AD_ADDRESS": "62 Moo 1, Rangsit-Ongkhazak Road (Km.7) Thanyaburi, Pathum Thani", 18 "AD_BD": "1999-09-11T17:00:00.000Z", 19 "AD_EMAIL": "Chanom.kai@multipclothes.com", 20 "AD_PHONE": "0974328801" 21 } 22] 23 </pre>

Search by ID and Last Name

GET Send <http://localhost:3030/admin/idFName/006000202/Somsri>

Params	Auth	Headers (6)	Body	Pre-req.	Tests	Settings	Cookies										
Query Params																	
<table border="1" style="width: 100%; border-collapse: collapse; text-align: center;"> <thead> <tr> <th>Key</th> <th>Value</th> <th>Description</th> <th>...</th> <th>Bulk Edit</th> </tr> </thead> <tbody> <tr> <td>Key</td> <td>Value</td> <td>Description</td> <td></td> <td></td> </tr> </tbody> </table>								Key	Value	Description	...	Bulk Edit	Key	Value	Description		
Key	Value	Description	...	Bulk Edit													
Key	Value	Description															

Body Cookies Headers (8) Test Results 200 OK 19 ms 805 B Save as Example ...

Pretty Raw Preview Visualize JSON Copy Search

```

1
2   "error": false,
3   "data": [
4     {
5       "AD_NUM": "000000001",
6       "AD_FNAME": "Somsri",
7       "AD_LNAME": "Mayang",
8       "AD_ADDRESS": "999 Phuttamonthon 4 Road, Salaya, Nakhon Pathom 73170 Thailand",
9       "AD_BD": "1988-11-21T17:00:00.000Z",
10      "AD_EMAIL": "somsri.may@multiclothes.com",
11      "AD_PHONE": "0869144407"
12    },
13    {
14      "AD_NUM": "006000202",
15      "AD_FNAME": "Yang",
16      "AD_LNAME": "Nonyu",
17      "AD_ADDRESS": "23, Trok Rongnamkeang, Yotha Rd., Bangkok, 10100 Thailand",
18      "AD_BD": "1991-01-16T17:00:00.000Z",
19      "AD_EMAIL": "yang.non@multiclothes.com",
20      "AD_PHONE": "0972187739"

```

GET Send <http://localhost:3030/admin/idLName/32191/non>

Params	Auth	Headers (6)	Body	Pre-req.	Tests	Settings	Cookies										
Query Params																	
<table border="1" style="width: 100%; border-collapse: collapse; text-align: center;"> <thead> <tr> <th>Key</th> <th>Value</th> <th>Description</th> <th>...</th> <th>Bulk Edit</th> </tr> </thead> <tbody> <tr> <td>Key</td> <td>Value</td> <td>Description</td> <td></td> <td></td> </tr> </tbody> </table>								Key	Value	Description	...	Bulk Edit	Key	Value	Description		
Key	Value	Description	...	Bulk Edit													
Key	Value	Description															

Body Cookies Headers (8) Test Results 200 OK 15 ms 1.05 KB Save as Example ...

Pretty Raw Preview Visualize JSON Copy Search

```

1
2   "error": false,
3   "data": [
4     {
5       "AD_NUM": "000032191",
6       "AD_FNAME": "Chanom",
7       "AD_LNAME": "Kaimook",
8       "AD_ADDRESS": "62 Moo 1, Rangsit-Ongkharak Road (Km.7) Thanyaburi, Pathum Thani",
9       "AD_BD": "1999-09-11T17:00:00.000Z",
10      "AD_EMAIL": "Chanom.kai@multiclothes.com",
11      "AD_PHONE": "0974328801"
12    },
13    {
14      "AD_NUM": "000885600",
15      "AD_FNAME": "Waimaii",
16      "AD_LNAME": "Yaknon",
17      "AD_ADDRESS": "17 Henri Dunant Rd, Khwaeng Pathum Wan, Khet Pathum Wan, Krung Thep Maha Nakhon 10330",
18      "AD_BD": "1970-07-28T17:00:00.000Z",
19      "AD_EMAIL": "Waimaii.yak@multiclothes.com",

```

	<p>GET http://localhost:3030/admin/ldLName/006000202/Ma Send</p> <p>Params Auth Headers (6) Body Pre-req. Tests Settings Cookies</p> <p>Query Params</p> <table border="1"> <thead> <tr> <th>Key</th> <th>Value</th> <th>Description</th> <th>... Bulk Edit</th> </tr> </thead> <tbody> <tr> <td>Key</td> <td>Value</td> <td>Description</td> <td></td> </tr> </tbody> </table> <p>Body Cookies Headers (8) Test Results 200 OK 16 ms 805 B Save as Example</p> <p>Pretty Raw Preview Visualize JSON </p> <pre> 1 { 2 "error": false, 3 "data": [4 { 5 "AD_NUM": "000000001", 6 "AD_FNAME": "Somsri", 7 "AD_LNAME": "Mayang", 8 "AD_ADDRESS": "999 Phuttamonthon 4 Road, Salaya, Nakhon Pathom 73170 Thailand", 9 "AD_BD": "1988-11-21T17:00:00.000Z", 10 "AD_EMAIL": "somsri.may@multiclothes.com", 11 "AD_PHONE": "0869144407" 12 }, 13 { 14 "AD_NUM": "006000202", 15 "AD_FNAME": "Yang", 16 "AD_LNAME": "Nonyu", 17 "AD_ADDRESS": "23, Trok Rongnamkeang, Yotha Rd., Bangkok, 10100 Thailand", 18 "AD_BD": "1991-01-16T17:00:00.000Z", 19 "AD_EMAIL": "yang.non@multiclothes.com", 20 "AD_PHONE": "0972187739" </pre>	Key	Value	Description	... Bulk Edit	Key	Value	Description	
Key	Value	Description	... Bulk Edit						
Key	Value	Description							
Search by First Name and Last Name	<p>GET http://localhost:3030/admin/FNameLName/Som/non Send</p> <p>Params Auth Headers (6) Body Pre-req. Tests Settings Cookies</p> <p>Query Params</p> <table border="1"> <thead> <tr> <th>Key</th> <th>Value</th> <th>Description</th> <th>... Bulk Edit</th> </tr> </thead> <tbody> <tr> <td>Key</td> <td>Value</td> <td>Description</td> <td></td> </tr> </tbody> </table> <p>Body Cookies Headers (8) Test Results 200 OK 16 ms 1.05 KB Save as Example</p> <p>Pretty Raw Preview Visualize JSON </p> <pre> 1 { 2 "error": false, 3 "data": [4 { 5 "AD_NUM": "000000001", 6 "AD_FNAME": "Somsri", 7 "AD_LNAME": "Mayang", 8 "AD_ADDRESS": "999 Phuttamonthon 4 Road, Salaya, Nakhon Pathom 73170 Thailand", 9 "AD_BD": "1988-11-21T17:00:00.000Z", 10 "AD_EMAIL": "somsri.may@multiclothes.com", 11 "AD_PHONE": "0869144407" 12 }, 13 { 14 "AD_NUM": "000885600", 15 "AD_FNAME": "Waimaiii", 16 "AD_LNAME": "Yaknon", 17 "AD_ADDRESS": "17 Henri Dunant Rd, Khwaeng Pathum Wan, Khet Pathum Wan, Krung Thep Maha Nakhon 10330", 18 "AD_BD": "1970-07-28T17:00:00.000Z", 19 "AD_EMAIL": "Waimaiii.yak@multiclothes.com", </pre>	Key	Value	Description	... Bulk Edit	Key	Value	Description	
Key	Value	Description	... Bulk Edit						
Key	Value	Description							

	<p>GET http://localhost:3030/admin/FNameLName/Chanom/Naisoy Send</p> <p>Params Auth Headers (6) Body Pre-req. Tests Settings Cookies</p> <p>Query Params</p> <table border="1"> <thead> <tr> <th>Key</th> <th>Value</th> <th>Description</th> <th>Bulk Edit</th> </tr> <tr> <th>Key</th> <th>Value</th> <th colspan="2">Description</th> </tr> </thead> <tbody> <tr> <td></td> <td></td> <td colspan="2"></td> </tr> </tbody> </table> <p>Body Cookies Headers (8) Test Results 200 OK 14 ms 828 B Save as Example</p> <p>Pretty Raw Preview Visualize JSON</p> <pre> 1 { 2 "error": false, 3 "data": [4 { 5 "AD_NUM": "000020031", 6 "AD_FNAME": "Suaysood", 7 "AD_LNAME": "Naisoy", 8 "AD_ADDRESS": "Khlong Luang 25 KhlongNung KhlongLuang, Chang Wat Pathum Thani 12120", 9 "AD_BD": "2001-05-02T17:00:00.000Z", 10 "AD_EMAIL": "Suaysood.nai@multiclothes.com", 11 "AD_PHONE": "0978394783" 12 }, 13 { 14 "AD_NUM": "000032191", 15 "AD_FNAME": "Chanom", 16 "AD_LNAME": "Kaimook", 17 "AD_ADDRESS": "62 Moo 1, Rangsit-Ongkharak Road (Km.7) Thanyaburi, Pathum Thani", 18 "AD_BD": "1999-09-11T17:00:00.000Z", 19 "AD_EMAIL": "Chanom.kai@multiclothes.com", </pre>	Key	Value	Description	Bulk Edit	Key	Value	Description					
Key	Value	Description	Bulk Edit										
Key	Value	Description											
Search by ID, First Name, and Last Name	<p>GET http://localhost:3030/admin/IdFNameLName/02/sood/oo Send</p> <p>Params Auth Headers (6) Body Pre-req. Tests Settings Cookies</p> <p>Query Params</p> <table border="1"> <thead> <tr> <th>Key</th> <th>Value</th> <th>Description</th> <th>Bulk Edit</th> </tr> <tr> <th>Key</th> <th>Value</th> <th colspan="2">Description</th> </tr> </thead> <tbody> <tr> <td></td> <td></td> <td colspan="2"></td> </tr> </tbody> </table> <p>Body Cookies Headers (8) Test Results 200 OK 7 ms 1.03 KB Save as Example</p> <p>Pretty Raw Preview Visualize JSON</p> <pre> 1 { 2 "error": false, 3 "data": [4 { 5 "AD_NUM": "000020031", 6 "AD_FNAME": "Suaysood", 7 "AD_LNAME": "Naisoy", 8 "AD_ADDRESS": "Khlong Luang 25 KhlongNung KhlongLuang, Chang Wat Pathum Thani 12120", 9 "AD_BD": "2001-05-02T17:00:00.000Z", 10 "AD_EMAIL": "Suaysood.nai@multiclothes.com", 11 "AD_PHONE": "0978394783" 12 }, 13 { 14 "AD_NUM": "000032191", 15 "AD_FNAME": "Chanom", 16 "AD_LNAME": "Kaimook", 17 "AD_ADDRESS": "62 Moo 1, Rangsit-Ongkharak Road (Km.7) Thanyaburi, Pathum Thani", 18 "AD_BD": "1999-09-11T17:00:00.000Z", 19 "AD_EMAIL": "Chanom.kai@multiclothes.com", </pre>	Key	Value	Description	Bulk Edit	Key	Value	Description					
Key	Value	Description	Bulk Edit										
Key	Value	Description											

Insert

http://localhost:3030/admin/IdFNameLName/91/So/Mayang

GET http://localhost:3030/admin/IdFNameLName/91/So/Mayang

Params Auth Headers (6) Body Pre-req. Tests Settings Cookies

Query Params

Key	Value	Description	... Bulk Edit
Key	Value	Description	

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```

1
2 "error": false,
3 "data": [
4     {
5         "AD_NUM": "00000001",
6         "AD_FNAME": "Somsri",
7         "AD_LNAME": "Mayang",
8         "AD_ADDRESS": "999 Phutthamonthon 4 Road, Salaya, Nakhon Pathom 73170 Thailand",
9         "AD_BD": "1988-11-21T17:00:00.000Z",
10        "AD_EMAIL": "somsri.may@multiclothes.com",
11        "AD_PHONE": "0869144407"
12    },
13    {
14        "AD_NUM": "000020031",
15        "AD_FNAME": "Suaysood",
16        "AD_LNAME": "Naisoy",
17        "AD_ADDRESS": "Khlong Luang 25 KhlongNung KhlongLuang, Chang Wat Pathum Thani
12120",
18        "AD_BD": "2001-05-02T17:00:00.000Z",
19        "AD_EMAIL": "Suaysood.nai@multiclothes.com",

```


POST http://localhost:3030/admin/add

Params Authorization Headers (8) Body **Body** Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

Body Cookies Headers (8) Test Results

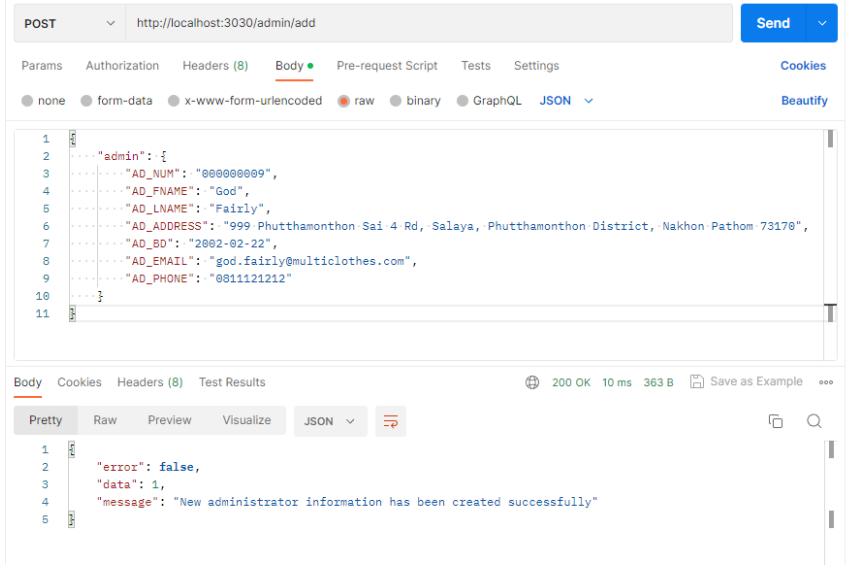
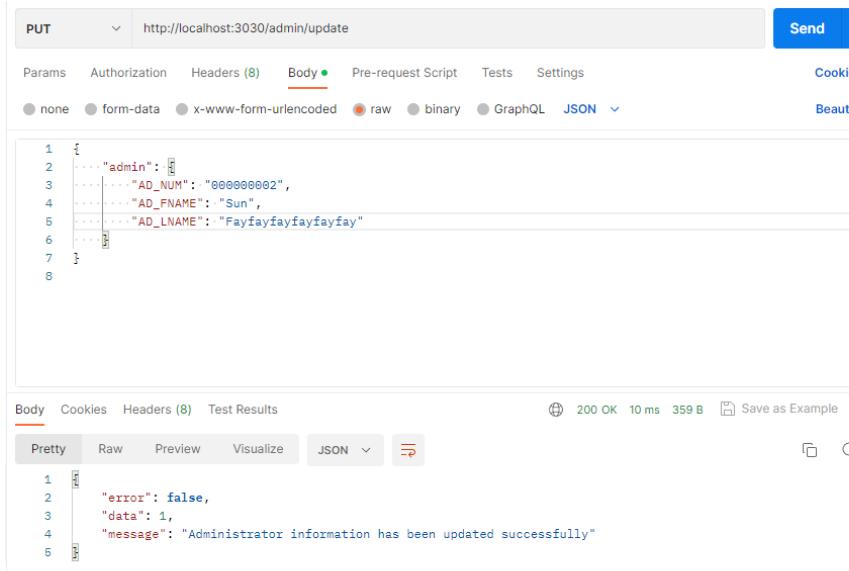
Pretty Raw Preview Visualize JSON

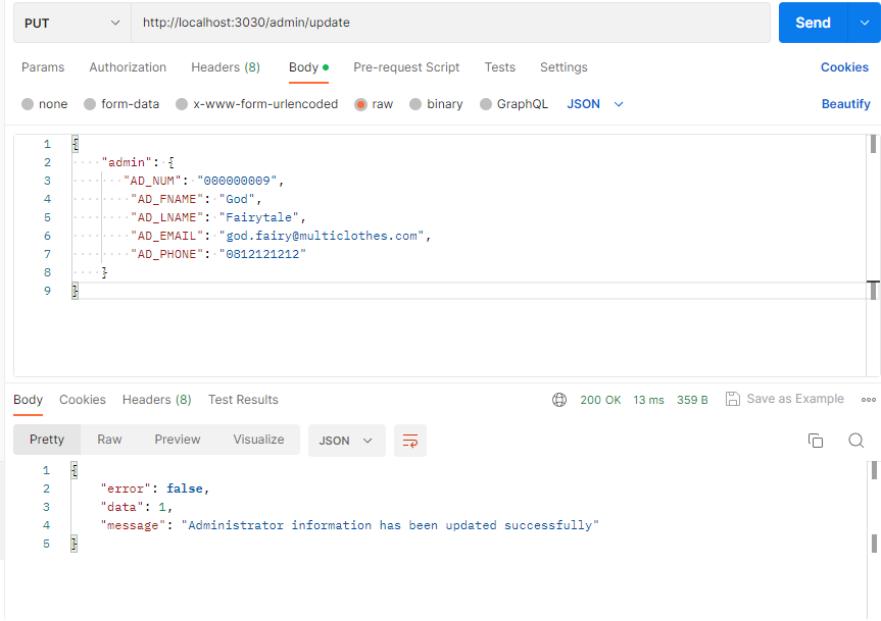
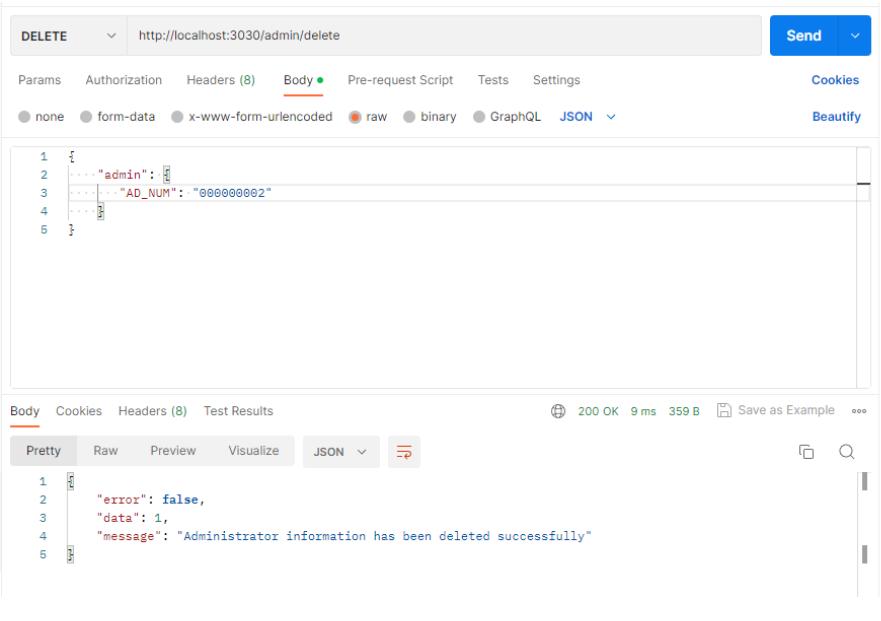
```

1 {
2     "admin": {
3         "AD_NUM": "00000002",
4         "AD_FNAME": "Sun",
5         "AD_LNAME": "Fayfay",
6         "AD_ADDRESS": "999 Phutthamonthon Sai 4 Rd, Salaya, Phutthamonthon District, Nakhon Pathom 73170",
7         "AD_BD": "2003-04-18",
8         "AD_EMAIL": "sun.fay@multiclothes.com",
9         "AD_PHONE": "0822356489"
10    }
11 }

```

200 OK 11 ms 363 B Save as Example

	 <pre> 1 2 "admin": { 3 "AD_NUM": "00000009", 4 "AD_FNAME": "God", 5 "AD_LNAME": "Fairly", 6 "AD_ADDRESS": "999 Phutthamonthon Sai 4 Rd, Salaya, Phutthamonthon District, Nakhon Pathom 73170", 7 "AD_BD": "2002-02-22", 8 "AD_EMAIL": "god.fairly@multiclothes.com", 9 "AD_PHONE": "0811121212" 10 } 11 </pre> <p>Body Cookies Headers (8) Test Results</p> <p>Pretty Raw Preview Visualize JSON</p> <pre> 1 2 "error": false, 3 "data": 1, 4 "message": "New administrator information has been created successfully" 5 </pre>
Update	 <pre> 1 2 "admin": { 3 "AD_NUM": "00000002", 4 "AD_FNAME": "Sun", 5 "AD_LNAME": "Fayfayfayfayfayfay" 6 } 7 8 </pre> <p>Body Cookies Headers (8) Test Results</p> <p>Pretty Raw Preview Visualize JSON</p> <pre> 1 2 "error": false, 3 "data": 1, 4 "message": "Administrator information has been updated successfully" 5 </pre>

	 <pre> 1 2 "admin": { 3 "AD_NUM": "00000009", 4 "AD_FNAME": "God", 5 "AD_LNAME": "Fairytale", 6 "AD_EMAIL": "god.fairy@multiclothes.com", 7 "AD_PHONE": "0812121212" 8 } 9 </pre> <pre> 1 2 "error": false, 3 "data": 1, 4 "message": "Administrator information has been updated successfully" 5 </pre>
Delete	 <pre> 1 2 "admin": { 3 "AD_NUM": "00000002" 4 } 5 </pre> <pre> 1 2 "error": false, 3 "data": 1, 4 "message": "Administrator information has been deleted successfully" 5 </pre>

The screenshot shows the Postman interface for a DELETE request to `http://localhost:3030/admin/delete`. The request body is set to JSON and contains the following data:

```
1 {  
2   ... "admin": ...,  
3   ... "AD_NUM": "00000009"  
4 }  
5
```

The response status is 200 OK, with a response time of 5 ms and a response size of 359 B. The response body is:

```
1 {  
2   "error": false,  
3   "data": 1,  
4   "message": "Administrator information has been deleted successfully"  
5 }
```