



## โครงการระดับที่ 2

เรื่องการวิเคราะห์และออกแบบเว็บ โดยใช้ Business Model ร้านค้าล่องสุ่ม(กลุ่มที่ 9)

### จัดทำโดย

นาย เกียรติคุณ กล่อมเกลี้ยง 6687001  
นาย เตชิต ฐิติธรรมจริยา 6687002  
นางสาว พربีณ์ ปัญมพรวัฒน์ 6687031  
นาย ชนสันต์ บุญมาก 6687088

### เสนอ

ผู้ช่วยศาสตราจารย์ ดร. จิตาภา ไกรสังข์  
อาจารย์ ดร. วุฒิชาติ แสงผล

โครงการนี้เป็นส่วนหนึ่งของรายวิชา ITDS241 Web Technologies and Applications  
ภาคเรียนที่ 1 ปีการศึกษา 2567

คณะเทคโนโลยีสารสนเทศและการสื่อสาร มหาวิทยาลัยมหิดล

## คำอธิบายเกี่ยวกับธุรกิจ

ในปัจจุบันธุรกิจกล่องสุ่มเป็นที่นิยมสำหรับทุกๆ ช่วงวัยด้วยการออกแบบที่ทันสมัยและการขายความตื่นเต้นให้กับผู้บริโภคด้วยการซื้อสินค้าโดยภายในบรรจุสินค้าที่ไม่ทราบรายละเอียดที่ชัดเจนจนกว่าจะเปิดออกมากำทำให้ธุรกิจนี้เติบโตอย่างรวดเร็วในตลาดโลกและในหลาย ๆ ประเทศจึงมักเน้นการขายกล่องสุ่มเป็นสินค้าออนไลน์ทางกลุ่มของคนละผู้จัดทำเห็นถึงความน่าสนใจของธุรกิจจึงได้ออกแบบ Web Application ที่มีชื่อว่า Blind Fortune สำหรับการจำหน่ายโมเดลธุรกิจกล่องสุ่ม เช่น สินค้าจาก Popmart ที่จำหน่ายสินค้าประเภทกล่องสุ่ม โมเดลสินค้า Figure ต่างๆ และ Accessory สำหรับสินค้า เช่น เคสโทรศัพท์มือถือ กระเพาในเครื่องของ Popmart อีกด้วยโดยมีขอบเขตการใช้งานทางด้านการค้นหาสินค้า เลือกชื่อสินค้า การดูรายละเอียดสินค้า จนไปถึงการลงทะเบียนและการจัดการแก้ไขข้อมูลของสินค้า และ Administrators

## Wireframe Design

เริ่มต้นที่ Navigator bar จะอยู่ทางด้านบนของ Wireframe ใช้ในการเชื่อมโยงไปยังหน้าอื่นๆ

- ปุ่ม Blind Fortune เมื่อ Click เข้าไปจะแสดงกลับไปที่หน้า Home Page

- ปุ่ม **MENU** เมื่อ Click เข้าไปจะแสดงไปยังหน้า MENU Page ของสินค้า

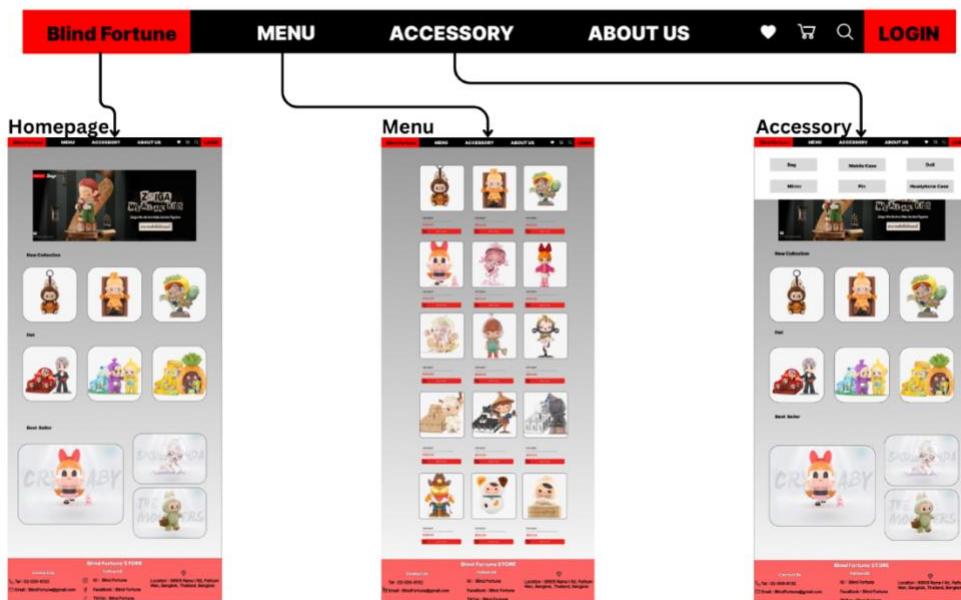
- ปุ่ม **ACCESSORY** เมื่อนำ Mouse ไปวางจะแสดงแถบ ACCESSORY และจะแสดงสินค้าเพิ่มเติม ได้แก่

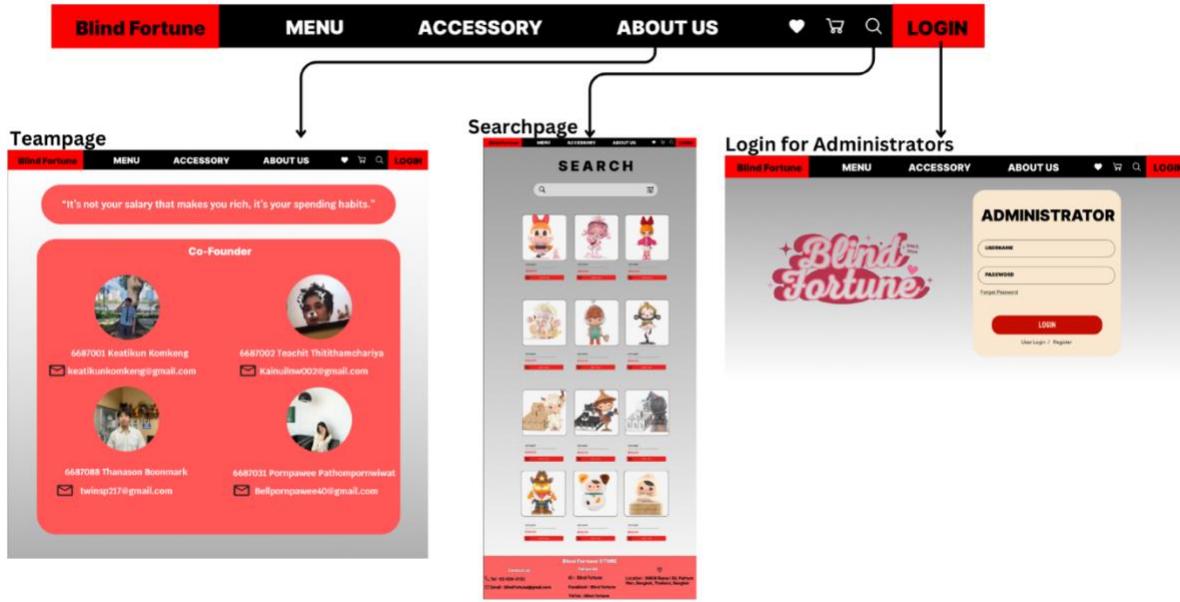
Bag, Mobile Case, Doll, Mirror, Pin, Other

- ปุ่ม **ABOUT US** เมื่อ Click เข้าไปจะแสดงไปยังหน้า ABOUT US จะแสดงข้อมูล รูปภาพของสมาชิกในกลุ่ม และช่องทางการติดต่อของสมาชิก

- ปุ่ม  เมื่อ Click เข้าไปจะแสดงไปยังหน้า SEARCH ใช้ในการค้นหาสินค้า

- ปุ่ม **LOGIN** เมื่อ Click เข้าไปจะแสดงไปยังหน้า Login ของ Administrator





**FOOTER** จะแสดงอยู่ล้วนท้ายของหน้า Wireframe



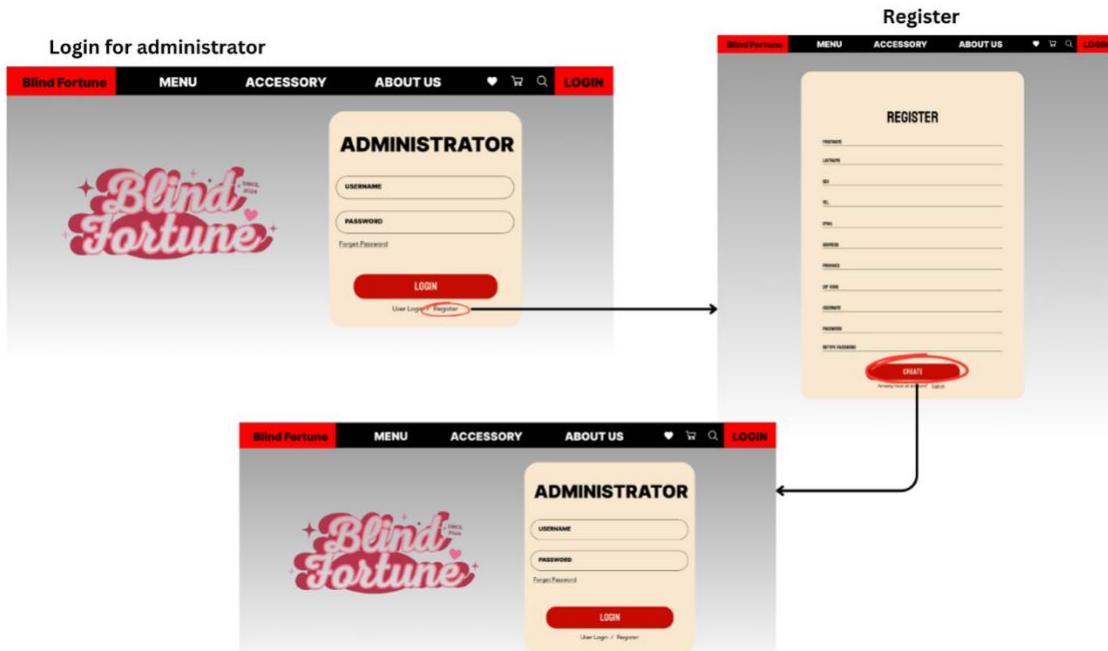
1. Homepage เป็นหน้าแรกของ Web Application จะแสดง Navigator bar อยู่ทางด้านบน 左 ภายใน Homepage จะแสดงรูปภาพสินค้าตัวอย่าง มีการแสดงรูปภาพของสินค้า ได้แก่ สินค้า New Collection สินค้า Hot และสินค้า Best Seller ภายในแบบ Navigator จะมีปุ่ม Login เพื่อเชื่อมกับหน้า Login ของ Administrators

The diagram illustrates the user flow between two web pages:

- Homepage:** The left side shows the main website interface. It features a top navigation bar with "Blind Fortune", "MENU", "ACCESSORY", and "ABOUT US". Below this is a banner for "ZIGA WEBSITE KIDS". The main content area is divided into sections: "New Collection" (with three items), "Hot" (with three items), and "Best Seller" (with three items). At the bottom, there's a "Blind Fortune STORE" section with contact information (Tel: 012-000-0002, Email: info@blindfortune.com) and social media links (Facebook, YouTube).
- Login for Administrators:** The right side shows the login page for administrators. It has a top navigation bar with "Blind Fortune", "MENU", "ACCESSORY", and "ABOUT US". The main area is titled "ADMINISTRATOR" and contains fields for "USERNAME" and "PASSWORD". There is also a "Forgot Password?" link and a large red "LOGIN" button.

A curved arrow points from the "Login" button on the Homepage to the "LOGIN" button on the Login for Administrators page, indicating the transition between the two screens.

2. หน้า Administrators จะแสดง Navigator อ่ายทางด้านบน ในหน้านี้จะมีการกรอก Username และ Password ของ Admin หากยังไม่มีบัญชีของ Admin จะต้องลงทะเบียนที่หน้า Register ที่แสดงอยู่ทางด้านล่างของปุ่ม Login ในหน้า Register เมื่อกรอกข้อมูลเสร็จสิ้นแล้วให้กดCREATE และจะกลับมาหน้าLogin เพื่อเป็นการแสดงว่า มีการลงทะเบียนเป็น Admin เสร็จสิ้นแล้ว ให้กดปุ่ม Login และจะไปยังหน้าการจัดการข้อมูลของ Admin ในหน้า Admin Management และสามารถกดปุ่มLogout ได้เพื่อไปยังหน้าHomepage



Adminที่ยังไม่ได้เป็นลงทะเบียน จะต้องทำการลงทะเบียนก่อนที่หน้าRegister โดยการกรอกFIRSTNAME LASTNAME SEX TEL EMAIL ADDRESS PROVINCE ZIPCODE USERNAME PASSWORD และ RETYPE PASSWORD และทำการ Upload Profile Image เมื่อทำการCREATEเสร็จสิ้นแล้วจะกลับไปยังหน้า Login for Administrator และสามารถกรอกUSERNAME และ PASSWORD เพื่อ LOGIN ไปยังหน้า Admin management ได้

### Login for administrator

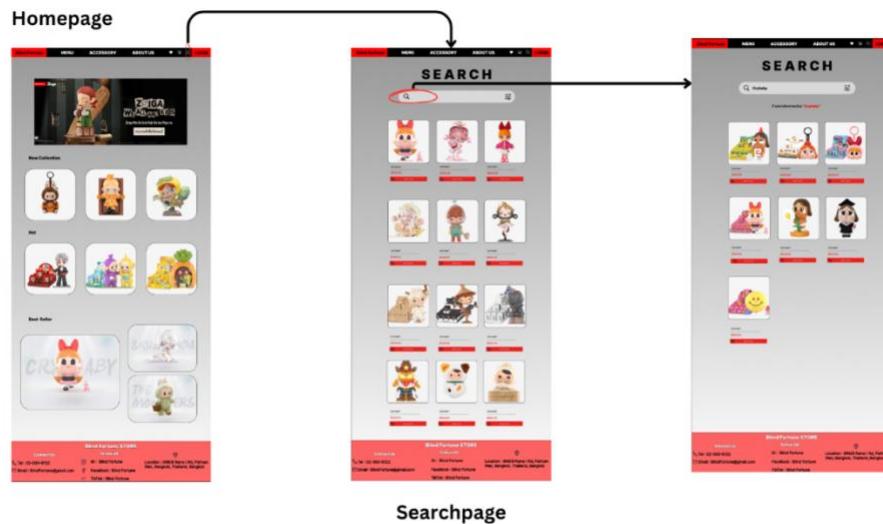


Admin ที่มีบัญชีข้อมูลเดิ๋วจะสามารถกรอก USERNAME และ PASSWORD ได้เมื่อกดปุ่ม LOGIN จะไปยังหน้า Admin management



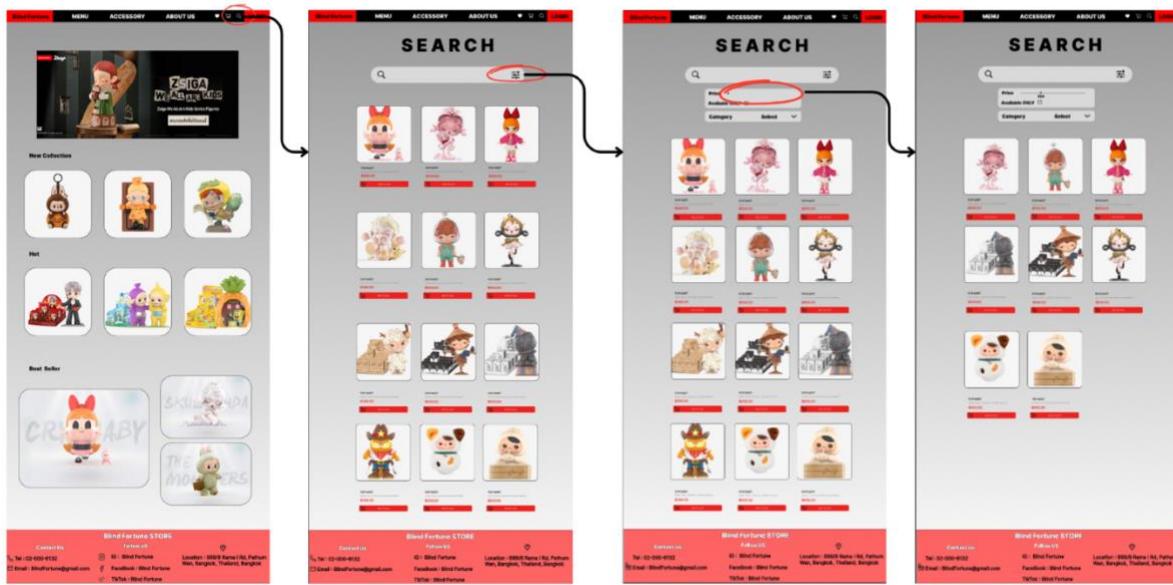
3. Search page และ Navigator bar จะมีปุ่ม เพื่อไปยังหน้า Search และในหน้านี้จะมีวิธีการค้นหาสินค้าได้หลายรูปแบบ ได้แก่

3.1 การค้นหาสินค้าแบบพิมพ์ได้ที่แนบค้นหา เมื่อค้นหาสินค้าแล้วจะแสดงสินค้าที่เกี่ยวข้องทั้งหมด

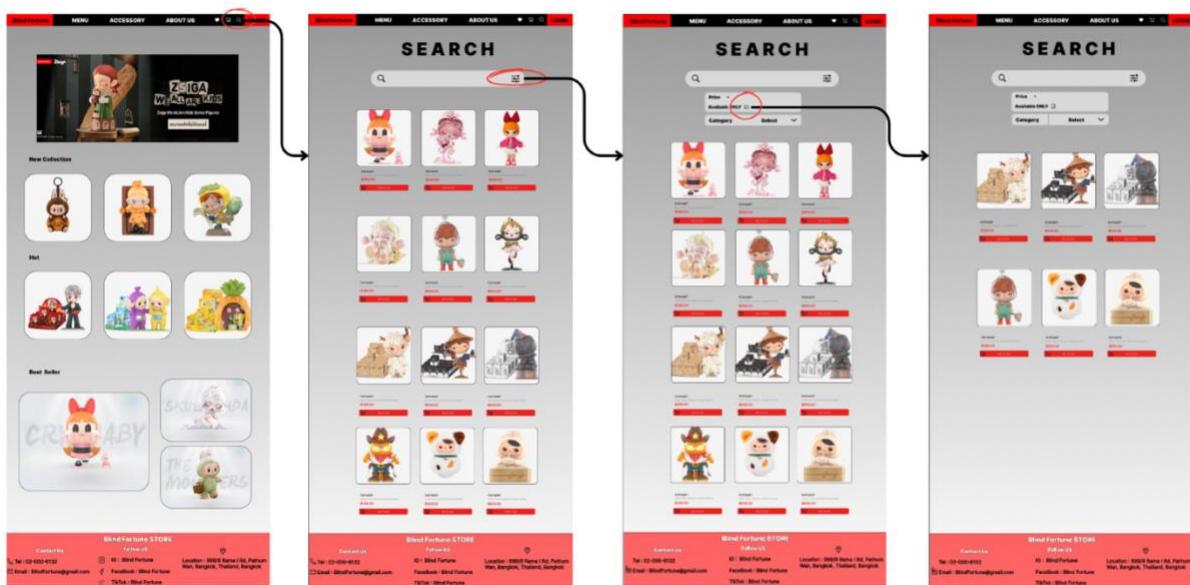


3.2 การค้นหาสินค้าแบบการใช้ **Search Filter** จะมีปุ่ม Filter อยู่ทางด้านขวา เมื่อ Click ปุ่มนี้แสดงวิธีการเลือกสินค้า

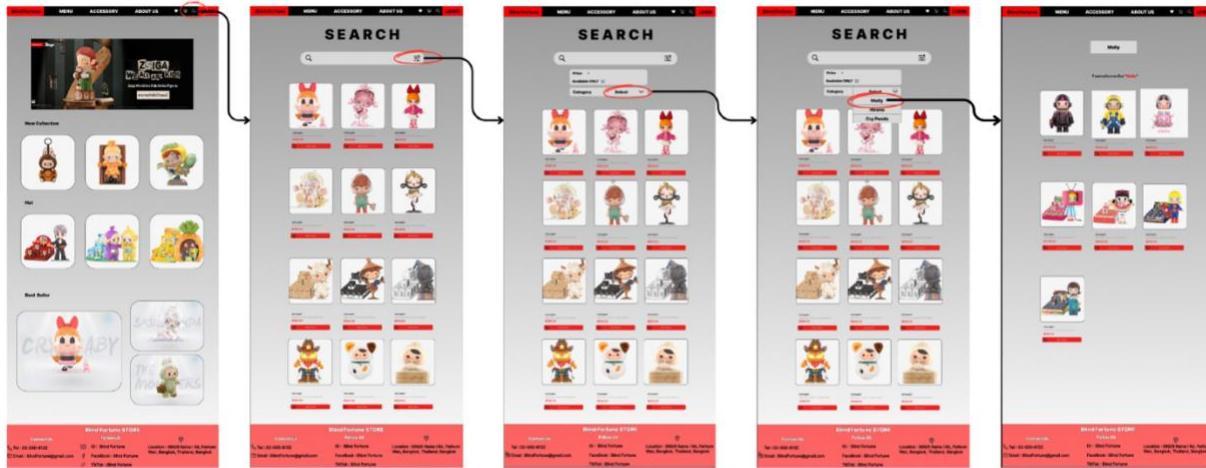
3.2.1 การค้นหาสินค้าโดยการเช็คราคาที่ **Rating Price** เมื่อเลื่อนไปทางขวาจะแสดงราคาสินค้าที่มากที่สุด และทางซ้ายจะแสดงราคาสินค้าที่น้อยที่สุด



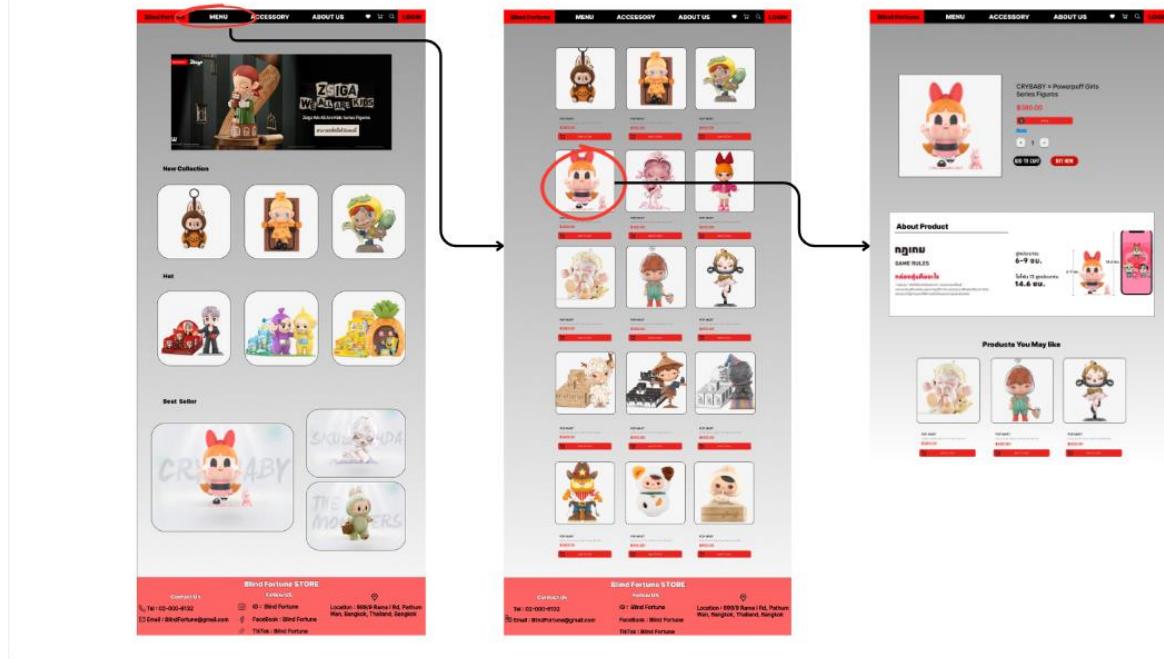
3.2.2 การค้นหาสินค้าแบบ Check box โดยสามารถ Check สินค้าได้ที่ปุ่ม Available จะแสดงสินค้าที่พร้อมส่ง



3.2.3 การค้นหาสินค้าแบบ Drop down ที่มีชื่อว่า Category จะมีปุ่มSelect ให้เลือกหมวดหมู่ของสินค้า ได้แก่ Molly , Hirono ,Cry Panda เมื่อClick ไปที่หมวดหมู่ของสินค้าจะแสดงสินค้าและจำนวนของสินค้าที่ค้นพบ ทั้งหมด



4. Detail Page เมื่อClickมาที่รูปภาพของสินค้า ในหน้านี้ จะแสดงรายละเอียดของสินค้าที่เราเลือกไว้ มีการแสดงรูปของสินค้า ชื่อเต็มของสินค้า ราคา และยังมีปุ่มสำหรับสินค้าที่สามารถยก Box ได้ หรือ สามารถเพิ่ม/ลดจำนวนสินค้าได้ เมื่อทำการเลือกสินค้าและจำนวนแล้ว จะสามารถนำสินค้าไปที่ ADD TO CART คือการเพิ่มสินค้าลงไว้ในตะกร้าสินค้า หรือการBUY NOW คือการที่เราสามารถซื้อสินค้าได้เลย และมีการแสดงสินค้าอื่นๆ ทำให้ลูกค้าสามารถเลือกดูหรือซื้อสินค้าอื่นได้จากที่เราเลือกไว้(Products You May Like)



5. User Account Management page เมื่อทำการ Login มาที่หน้า Administrators และจะแสดงมายังหน้านี้ ทางด้านซ้ายจะแสดง Admin ในการบริหารจัดการบัญชีของผู้ดูแลระบบ และสามารถเลือกแก้ไข Category และ Accessory ของสินค้าได้ ในเนื้อหาของหน้านี้ จะสามารถ Search รายชื่อ Admin ที่ต้องการจะค้นหาได้ จากนั้นจะแสดง ID USERNAME Email Phone ของ Admin และยังสามารถแก้ไข และลบรายชื่อของ Admin ที่ไม่ต้องการได้ และสามารถออกจากราชานี้ได้เมื่อกดปุ่ม LOGOUT เพื่อไปยังหน้า Homepage

#### Login for administrator

Admin ID	Username	Email	Phone
100	Iswong	Kettikung@gmail.com	0824543578
200	Pengaza	Penglove@hotmail.com	0958459999
300	Wibitabla	Wibitabla@gmail.com	0610973295
400	Pluemhe	Pluemlunir@gmail.com	0605946521
500	Faisali	Faisalilunip@gmail.com	0624986521
600	Norponr	Norponrufah@gmail.com	0934399215
700	Blackew	Blackew@gmail.com	0823204864

6. Product/Service Management page เมื่อทำการ Login มาที่หน้า Administrators แล้วจะแสดงรายการนี้  
ทางด้านซ้ายจะแสดง Admin ในการบริหารจัดการบัญชีของผู้ดูแลระบบ และสามารถเลือกแก้ไข Category และ Accessory ของสินค้าได้

เมื่อ Click มาที่ Category จะแสดง ID Name จำนวนสินค้าในคลัง สถานะของสินค้า และสามารถแก้ไข ID และชื่อของสินค้าได้

The figure consists of three screenshots of the 'Blind Fortune' application:

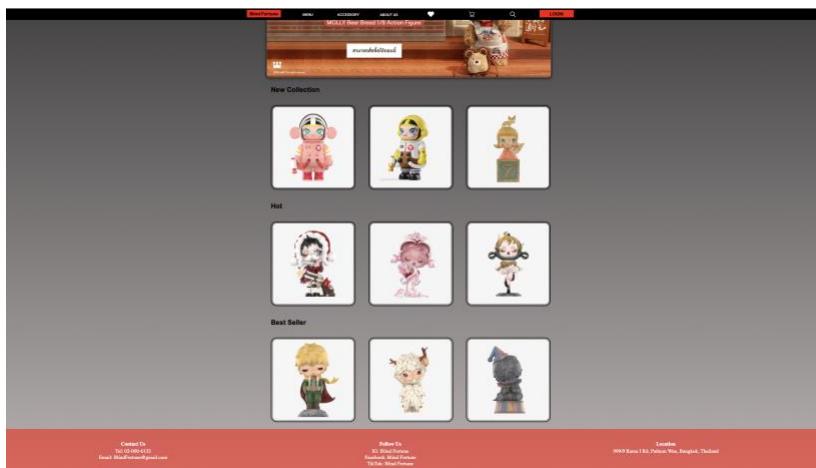
- Login Screen:** Shows the logo 'Blind Fortune' and a central 'ADMINISTRATOR' section with fields for 'USERNAME' and 'PASSWORD'. Below it is a 'LOGIN' button and a 'User Login / Register' link.
- Administrator Dashboard:** Shows a sidebar with 'Admin', 'Category', and 'Accessory' sections. The 'Category' section is highlighted with a red oval. A large arrow points from the 'Category' section of the sidebar to the 'Category' section of the next screenshot.
- Category Management Screen:** Shows a table of categories with columns: ID, Name, and Status (represented by a switch). The table contains six rows with names like Molly, Dimoo, etc. Each row has edit and delete icons.

เมื่อ Click มา Accessory จะแสดง ID Name จำนวนสินค้าในคลัง สถานะของสินค้า และสามารถแก้ไข ID และชื่อของสินค้าได้

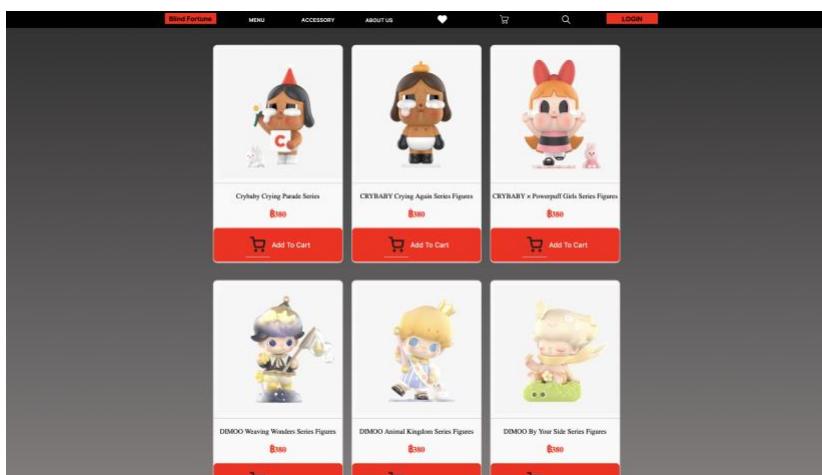
7. Team page ในแถบ Navigator bar จะมีปุ่มABOUT US เพื่อไปยังหน้า Team page จะแสดงข้อมูลและรูปภาพของสมาชิกในกลุ่ม

## รายละเอียดหน้าเว็บต่างๆ

### 1.HOME PAGE และ Menu page



Homepage



Menupage

```
<script>
  document.addEventListener('DOMContentLoaded', () => {
    fetch('http://localhost:3000/products')
      .then(response => response.json())
      .then(data => {
        const newCollection = data.filter(item => item.category === 'Molly').slice(0, 3);
        const hotItems = data.filter(item => item.category === 'SkullPanda').slice(0, 3);
        const bestSellers = data.filter(item => item.category === 'Hirono').slice(0, 3);

        updateProductGrid('new-collection', newCollection);
        updateProductGrid('hot', hotItems);
        updateProductGrid('best-seller', bestSellers);
      })
      .catch(err => console.error('Error fetching products:', err));
  });

  function updateProductGrid(sectionId, products) {
    const container = document.getElementById(sectionId);
    container.innerHTML = '';

    products.forEach(product => {
      const productCard = `
        <div class="product-card">
          
        </div>`;
      container.innerHTML += productCard;
    });
  }
</script>
```

```
<h2>New Collection</h2>
<section class="product-grid1" id="new-collection"></section>

<h2>Hot</h2>
<section class="product-grid1" id="hot"></section>

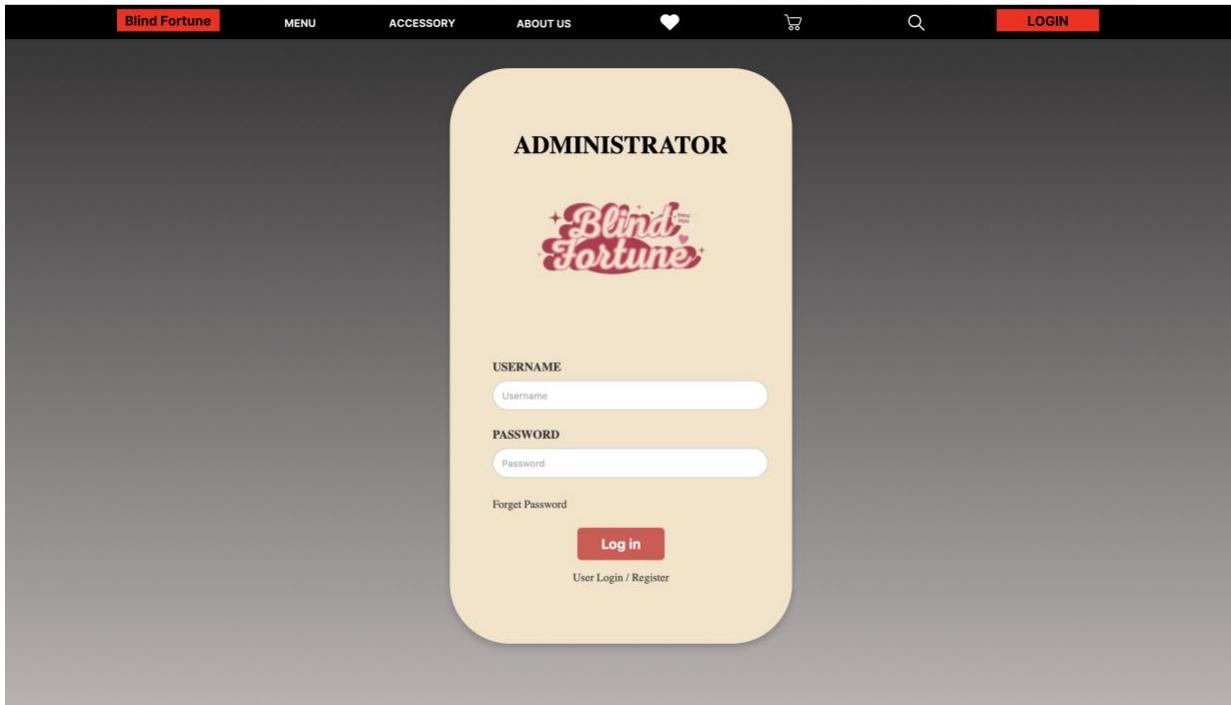
<h2>Best Seller</h2>
<section class="product-grid1" id="best-seller"></section>
```

ในหน้า home.html นี้ใช้การ fetch จาก path products ใน server.js มีการดึงข้อมูลเมื่อเรียกใช้ ได้แก่ Molly Skullpanda Hirono เพื่อนำไปแสดงด้าน frontend และแสดงอุปกรณ์มาเป็นรูปจาก link

```
// API สำหรับดึงข้อมูลสินค้า
app.get('/products', (req, res) => {
  const sql = `
    SELECT
      product.product_id AS id,
      product.product_name AS name,
      product.product_price AS price,
      product.product_quantity AS quantity,
      categories.cat_name AS category,
      product.product_size AS size,
      product.product_material AS material,
      images_product.image_path AS image
    FROM product
    JOIN categories ON product.cat_id = categories.cat_id
    LEFT JOIN images_product ON product.product_id = images_product.id
  `;
  db.query(sql, (err, results) => {
    if (err) throw err;
    res.json(results);
  });
});
```

ในหน้า server.js ได้มีการใช้ get เพื่อดึงข้อมูลสินค้า จาก sql table product มาแสดงบน path products

## 2. Login for Administrator



```
<section class="login-container">
<!-- អាជីវការណ៍ដែលមានបញ្ហា -->
<form id="loginForm">
    <h1>ADMINISTRATOR</h1>
     <!-- ទាត់កំចរចនប -->

    <!-- សម្រាប់ឈ្មោះ (Username) -->
    <label for="admin_id">USERNAME</label>
    <input type="text" id="admin_id" name="admin_id" placeholder="Username" required>
    <br>

    <!-- សម្រាប់ពាក្យដោយ (Password) -->
    <label for="admin_password">PASSWORD</label>
    <input type="password" id="admin_password" name="admin_password" placeholder="Password" required>
    <!-- សម្រាប់ពាក្យដោយ -->
    <br>

    <!-- ឲ្យកំណត់សម្រាប់បាន -->
    <p class="forget-password"> <a href="login.html">Forget Password</a> </p> |

    <!-- បញ្ជីពិនិត្យដែលត្រូវ -->
    <button type="submit" id="formBT">Log in</button>
    <!-- ពិនិត្យការងារដែលបានដោយខ្សោយបន្ថីត្រូវបាន -->
    <p class="register-login"> <a href="Regist.html">User Login / Register</a> </p>
</form>
<p id="error-message" style="color: red; display: none;">Invalid Admin ID or Password</p>
</section>
```

ໃນទាំងlogin.html នេះ គឺជាអាជីវការ ឬ section class="login-container" និង id="loginForm" ដែលជាផ្លូវការសំរាប់បញ្ចូនឈ្មោះ username និង password ដើម្បីបញ្ចប់ពាក្យដោយ។

```

<script>
    document.getElementById('loginForm').addEventListener('submit', function (e) {
        e.preventDefault(); // ข้องกันการรีเฟรชหน้าเมื่อส่งฟอร์ม

        const username = document.getElementById('admin_id').value;
        const password = document.getElementById('admin_password').value;

        // ส่งข้อมูลไปที่เซิร์ฟเวอร์ผ่าน POST
        fetch('http://localhost:3000/login', {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json'
            },
            body: JSON.stringify({
                username: username,
                password: password
            })
        })
        .then(response => response.text())
        .then(data => {
            if (data === 'Login successful!') {
                alert('Login successful!');
                window.location.href = 'Manage_admin.html'; // ไปที่หน้าดังไปเมื่อ login สำเร็จ
            } else {
                alert('Invalid username or password!');
            }
        })
        .catch(error => {
            console.error('Error:', error);
            alert('There was an error logging in');
        });
    });
</script>

```

การใช้ javascript มีการดึงข้อมูลจากDom ที่มี id= loginform สำหรับ api และมีการเก็บค่าจากform จากนั้น

fetch('http://localhost:3000/login')

คือการส่งข้อมูลserver ผ่าน http requestแบบ POST ในรูปแบบ json ไปยัง localhost:3000/login และมีการรับ  
ข้อมูลจากserver .then(response => response.text()) คือการอ่านข้อมูลแบบ text และส่งไปที่ dataถ้าผลลัพธ์ตรง  
จะไปยังหน้า Manage\_admin.html

```

app.post('/login', (req, res) => {
  const { username, password } = req.body;

  const query = 'SELECT * FROM admins WHERE admin_username = ? AND admin_password = ?';
  db.execute(query, [username, password], (err, result) => {
    if (err) {
      console.error('Database error:', err.message);
      return res.status(500).send('Database error: ' + err.message);
    }

    if (result.length > 0) {
      const insertQuery = 'INSERT INTO login (Username, admin_password) VALUES (?, ?)';
      db.execute(insertQuery, [username, password], (err, insertResult) => {
        if (err) {
          console.error('Error logging login attempt:', err.message);
          return res.status(500).send('Error logging login attempt: ' + err.message);
        }
        res.status(200).send('Login successful!');
      });
    } else {
      res.send('Invalid username or password!');
    }
  });
});

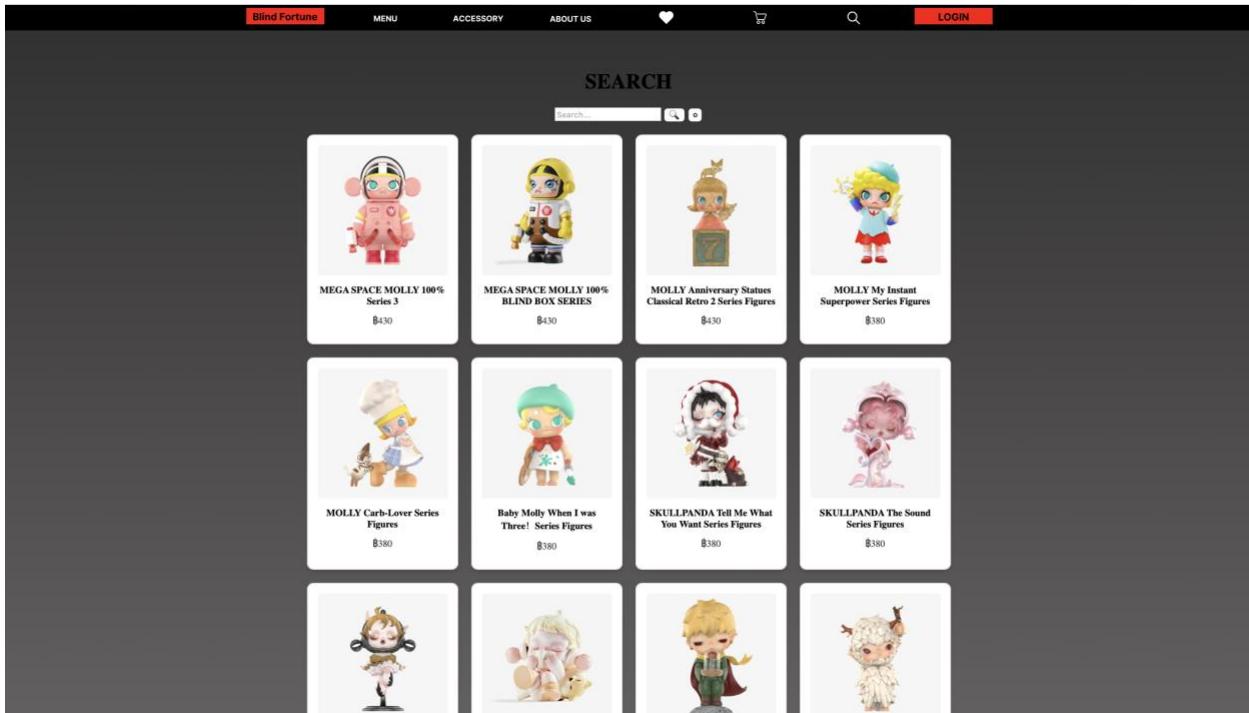
app.get("/login", (req, res) => {
  console.log("req select all")
  let sql = `select * from login`;
  db.query(sql, function (err, result) {
    if (err) {
      throw err;
    }
    res.json(result)
  });
});

```

ในหน้าserver.js ได้มีการรับคำขอจาก Post ใน path login มีการดึงข้อมูลผ่าน form ในรูปแบบ json และนำไปเก็บใน username, password ถ้าถูกต้องจะสามารถรันในคำสั่ง sql ได้เพื่อนำไปบันทึกใน table login

Get /login คือการรับคำขอไปที่ path login มีการดึงข้อมูลจากdb ถ้าถูกต้องจะส่งกลับไปยังclient รูปแบบ json

### 3. Search Page



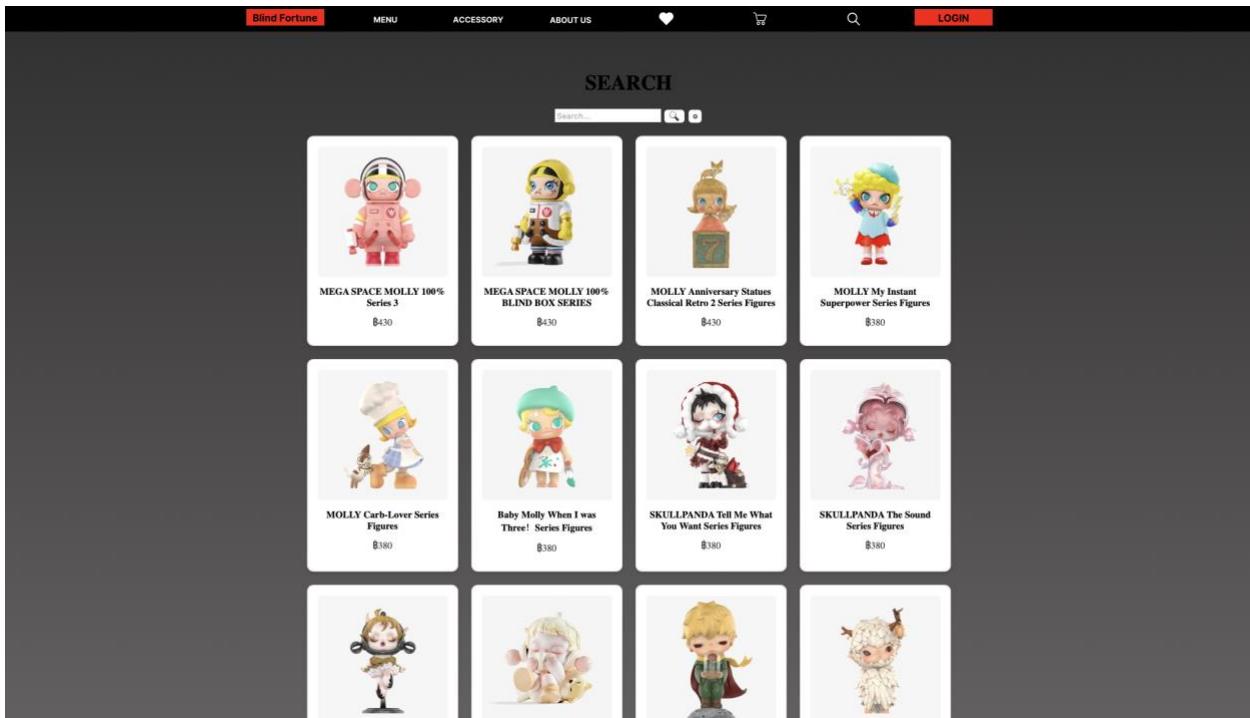
```
<script>
  document.addEventListener('DOMContentLoaded', () => {
    const searchButton = document.getElementById('search-button');
    const searchInput = document.getElementById('search-input');
    const priceRange = document.getElementById('price-range');
    const categoryFilter = document.getElementById('category-filter');
    const availableOnly = document.getElementById('available-only');

    // ดักจับการกดปุ่มค้นหา
    searchButton.addEventListener('click', () => {
      performSearch(); // เรียกฟังก์ชันค้นหาหลัก
    });

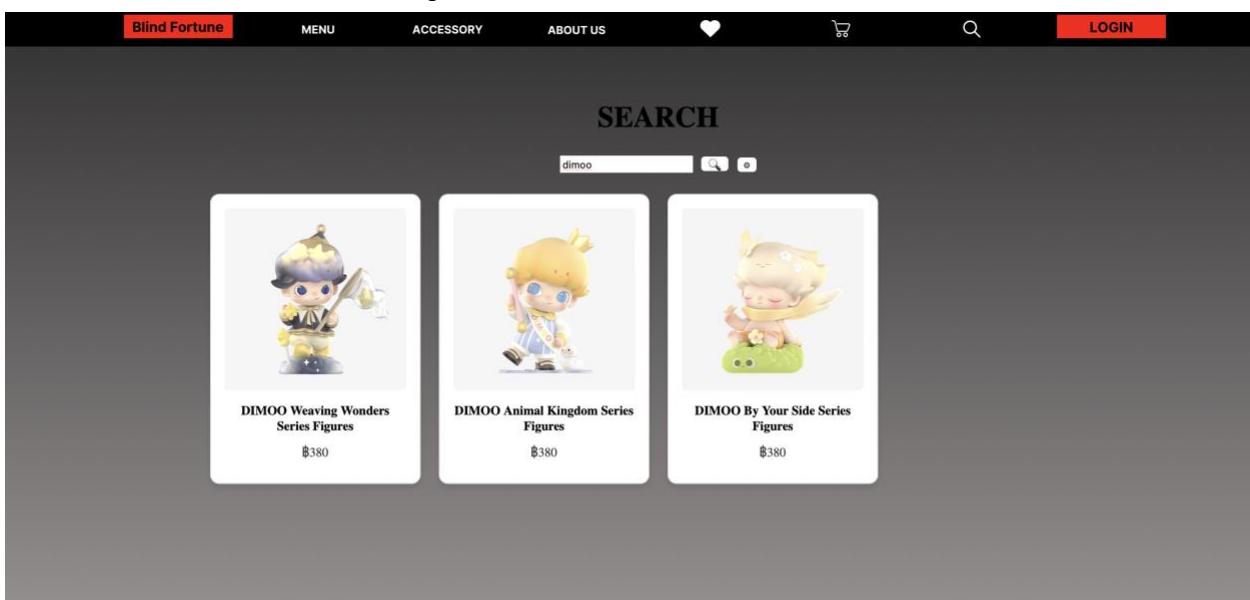
    // ดักจับการเปลี่ยนแปลงของตัวกรอง
    priceRange.addEventListener('input', performSearch);
    categoryFilter.addEventListener('change', performSearch);
    availableOnly.addEventListener('change', performSearch);
  });
</script>
```

Search page จะมีช่องรับ input ทั้งหมด 3 ช่อง ได้แก่ search-input , search-button และ category-filter ใช้สำหรับ search filter ทั้งหมด 3 แบบ รวมไปถึงหน้าการใช้ accessory search

3.1. การค้นหาสินค้าทั้งหมดจากปุ่ม  จะแสดงผลลัพธ์ทั้งหมดที่มีในdatabase ของสินค้า



3.2. การค้นหาสินค้าจากช่อง search-input



จะใช้ผลลัพธ์จากหน้า search.html ที่มีค่าจากในquery และจะให้ผลลัพธ์ที่ผ่านการรับค่าจาก search-input

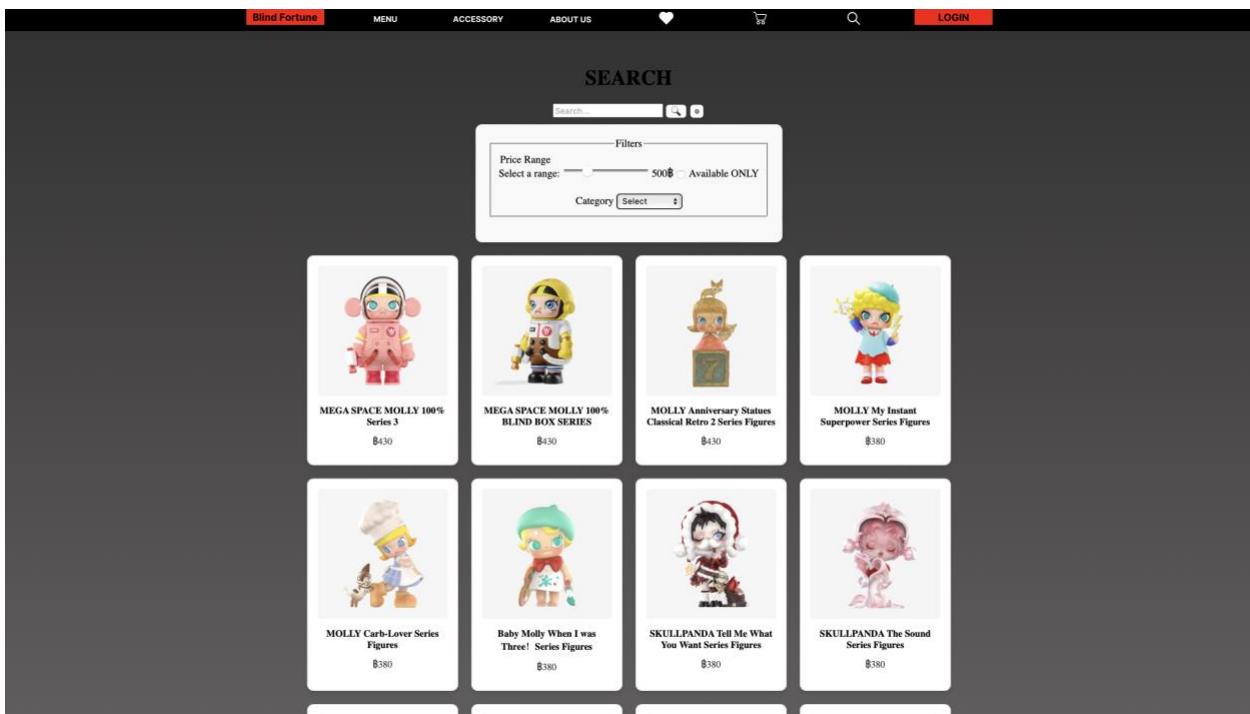
```

// กรองด้วยคำค้นหา
if (query) {
    filteredProducts = filteredProducts.filter(product =>
        product.name.toLowerCase().includes(query) ||
        product.category.toLowerCase().includes(query)
    );
}

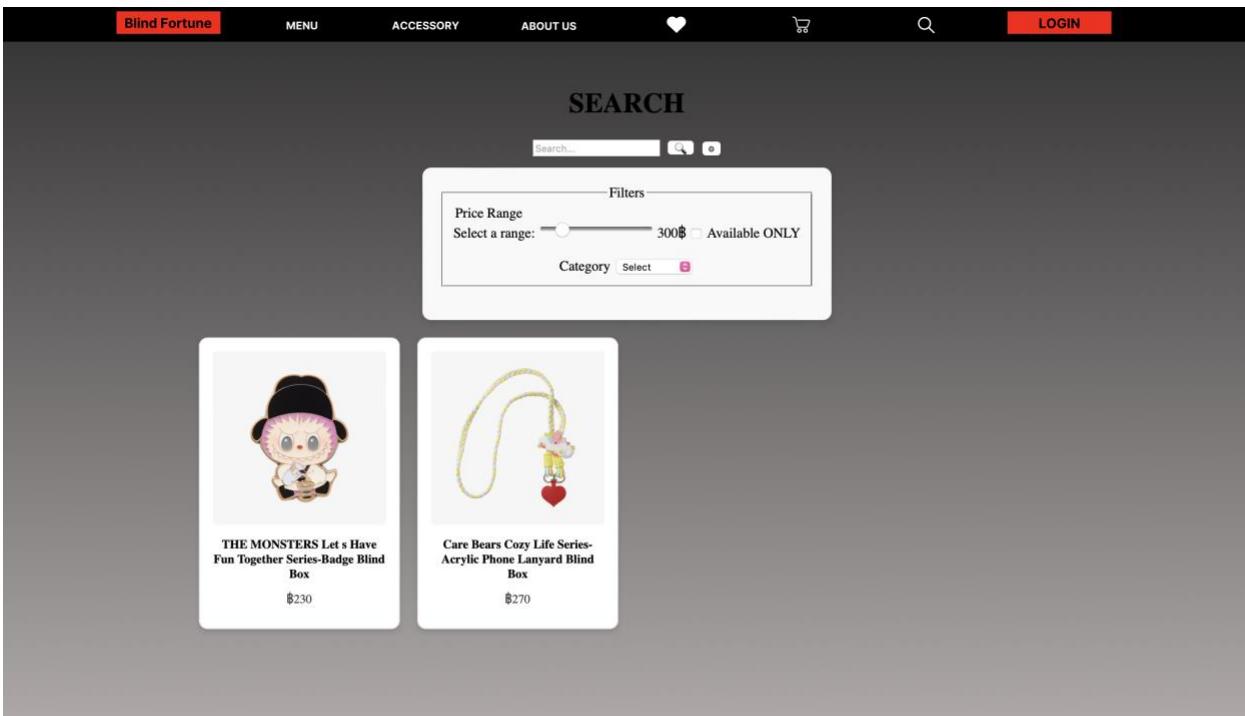
```

ฟังก์ชัน filter จะกรองสินค้าใน filteredProducts และสร้างอาร์เรย์ใหม่ที่ตรงกับเงื่อนไข ถ้าเป็นจริงจะเก็บไว้ใน array ที่ผ่านการกรองสินค้าแล้ว

### 3.3 Serach filter ทั้งหมด 3 แบบ



### 3.3.1.serach แบบ sidebar



การกรองราคาจากค่า price-range มาใช้ในการเทียบกับราคางานค้าที่ผ่านการ fetch มา

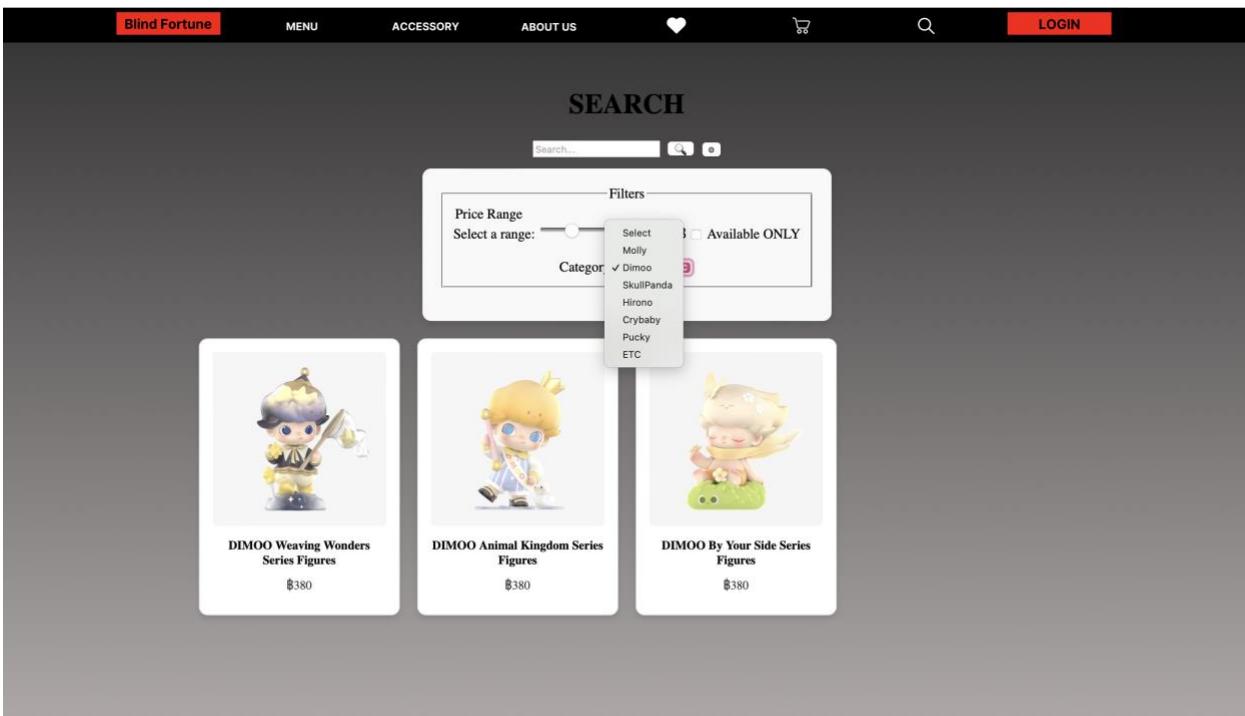
```
<!-- Slide Bar -->
<legend>Price Range</legend>
<label for="price-range">Select a range:</label>
<input type="range" id="price-range" min="0" max="2000" step="50" value="500" oninput="updatePriceValue(this.value)">
500฿ <!-- แสดงค่าที่เลือก -->
```

ในหน้า serach.html จะสร้าง Slide bar โดยกำหนดค่าต่ำสุดคือ 0 เพิ่มทีละ 50 และค่ามากสุดคือ 2000 จากนั้นจะเรียกใช้ฟังก์ชัน updatePriceValue ทุกครั้งที่เลื่อนແบบ เพื่ออัปเดตค่าราคาใหม่

```
// กรองด้วย Price Range
filteredProducts = filteredProducts.filter(product => product.price <= maxPrice);
```

ฟังก์ชัน filter จะทำการกรองสินค้า filteredProducts โดยจะสร้าง Array ใหม่ที่มีเฉพาะสินค้าตามเงื่อนไข

### 3.3.2. search ด้วย category



จะใช้การกรองcategoryจากค่าcategory filter มาใช้ในการเทียบกับราคางานค้าที่ผ่านการ fetchมา

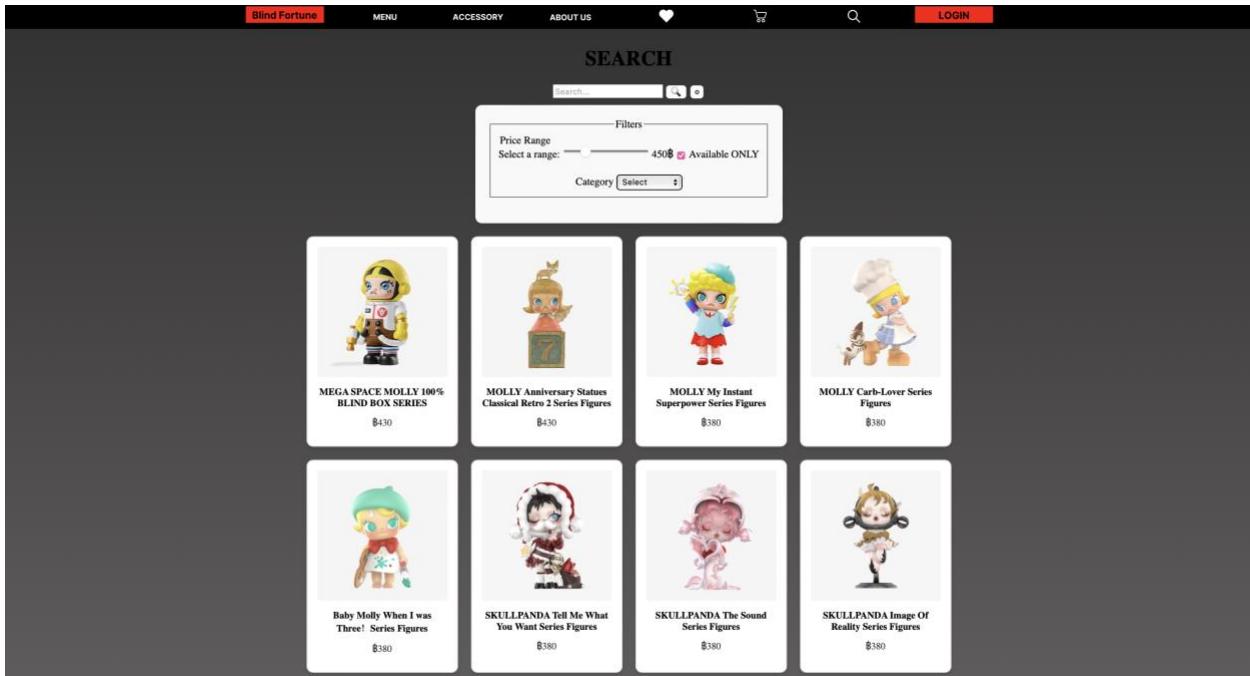
```
<label>Category</label>
<select id="category-filter">
    <option value="">Select</option>
    <option value="Molly">Molly</option>
    <option value="Dimoo">Dimoo</option>
    <option value="SkullPanda">SkullPanda</option>
    <option value="Hirono">Hirono</option>
    <option value="Crybaby">Crybaby</option>
    <option value="Pucky">Pucky</option>
    <option value="ETC">ETC</option>
</select>
```

ในหน้าsearch.html ใช้การสร้างdropdown menu เพื่อเก็บค่าProducts แต่ละ Category

```
// กรองด้วย Category
if (category) {
    filteredProducts = filteredProducts.filter(product => product.category === category);
```

การใช้ condition if สำหรับ เพื่อตรวจสอบค่าในcategory มีค่าหรือไม่และใช้ function filter !เพื่อกรองข้อมูลใน filterProducts ถ้า product.category ตรงกับ category จะเก็บไว้ใน arrayที่ผ่านการกรองแล้ว

### 3.3.3.search แบบการเลือกปุ่ม available only



จะใช้การกรองสินค้าที่ available จาก condition if-else ถ้า product.quality > 0 จะแสดงสินค้า available only

```
<!-- Check box -->
<label>
    <input type="checkbox" id="available-only"> Available ONLY
</label><br><br>
```

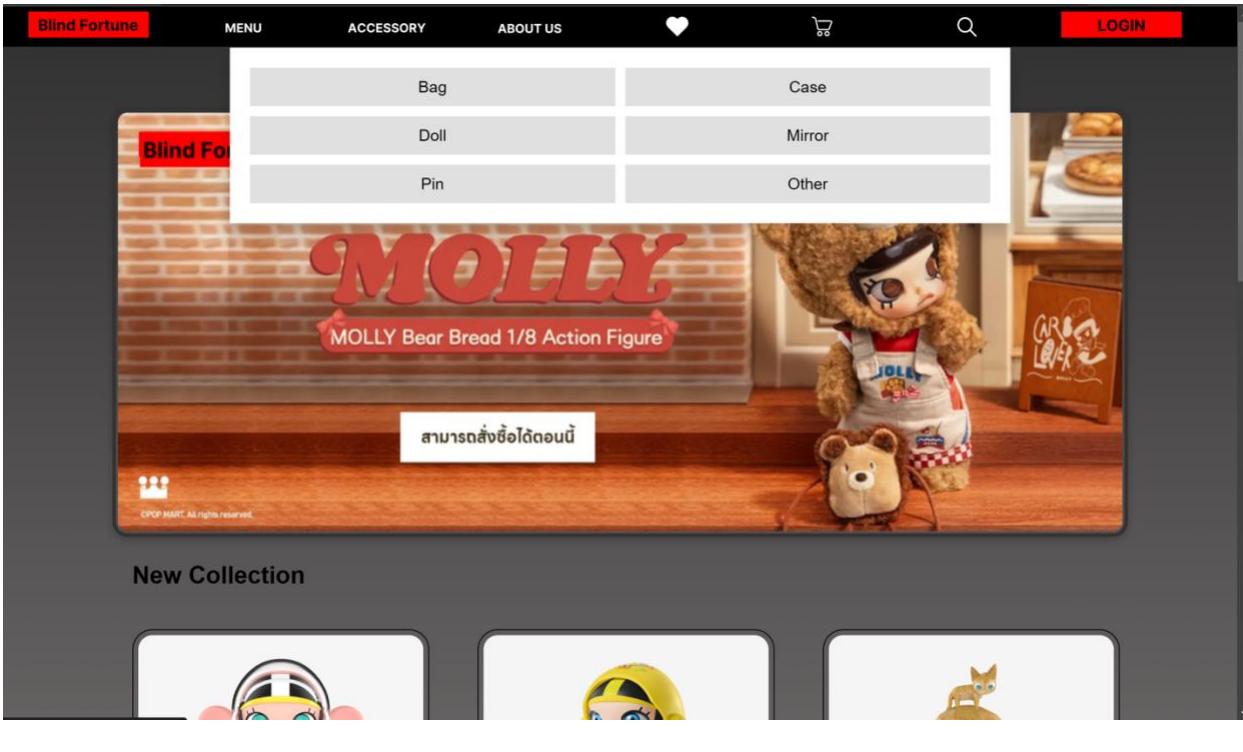
ในหน้า serach.html จะมีปุ่ม checkbox สำหรับสินค้าที่มี id="available-only"

```
// กรองด้วย Available ONLY (สมมติว่า JSON มี property `available`)
if (isAvailable) {
    filteredProducts = filteredProducts.filter(product => product.quantity > 0);
```

การใช้ condition if สำหรับเพื่อตรวจสอบสินค้าที่ Available และใช้ function filteredProducts เพื่อกรองข้อมูลในfilterProducts ถ้า quality ของ product มีค่ามากกว่า 0 จะเก็บไว้ใน array ที่ผ่านการกรองแล้ว

### 3.4 Accessory search

มาจากการค้นหาในส่วนของ nav bar



```
<div class="accessory-popup">
  <div class="accessory-options">
    <!-- ตัวเลือกของ Accessories จะพากันแบบ select -->
    <button class="accessory-option" id="accessory-bag" onclick="window.location.href='accessorysearch.html?category=Bag'">Bag</button>
    <button class="accessory-option" id="accessory-case" onclick="window.location.href='accessorysearch.html?category=Case'">Case</button>
    <button class="accessory-option" id="accessory-doll" onclick="window.location.href='accessorysearch.html?category=Doll'">Doll</button>
    <button class="accessory-option" id="accessory-mirror" onclick="window.location.href='accessorysearch.html?category=Mirror'">Mirror</button>
    <button class="accessory-option" id="accessory-pin" onclick="window.location.href='accessorysearch.html?category=Pin'">Pin</button>
    <button class="accessory-option" id="accessory-other" onclick="window.location.href='accessorysearch.html?category=Other'">Other</button>
  </div>
</div>
```

ในหน้า navbar.html จะใช้การ onclick และเมื่อ click ไปที่ accessory จะเปลี่ยนเป็น query ของหน้านั้นๆ

```
<!-- Dynamic Script Data -->
<script>
  document.addEventListener('DOMContentLoaded', () => {
    // รับค่า category จาก URL
    const urlParams = new URLSearchParams(window.location.search);
    const category = urlParams.get('category'); // ค่า category ที่ได้รับ

    if (category) {
      // ตั้งค่า category ในช่อง select
      document.getElementById('category-title').textContent = category;

      // ตั้งค่า category ที่ต้องการในหน้าผลลัพธ์
      searchProducts(category);
    }
  });

  // โหลดข้อมูลจาก API
  function searchProducts(category) {
    fetch('http://localhost:3000/products')
      .then(response => response.json())
      .then(data => {
        // กรองผลลัพธ์ตามค่า category
        const filteredProducts = data.filter(item => item.category.toLowerCase() === category.toLowerCase());

        // แสดงผลลัพธ์
        updateProductGrid('Row1', filteredProducts);
      })
      .catch(err => console.error('Error fetching products:', err));
  }
</script>
```

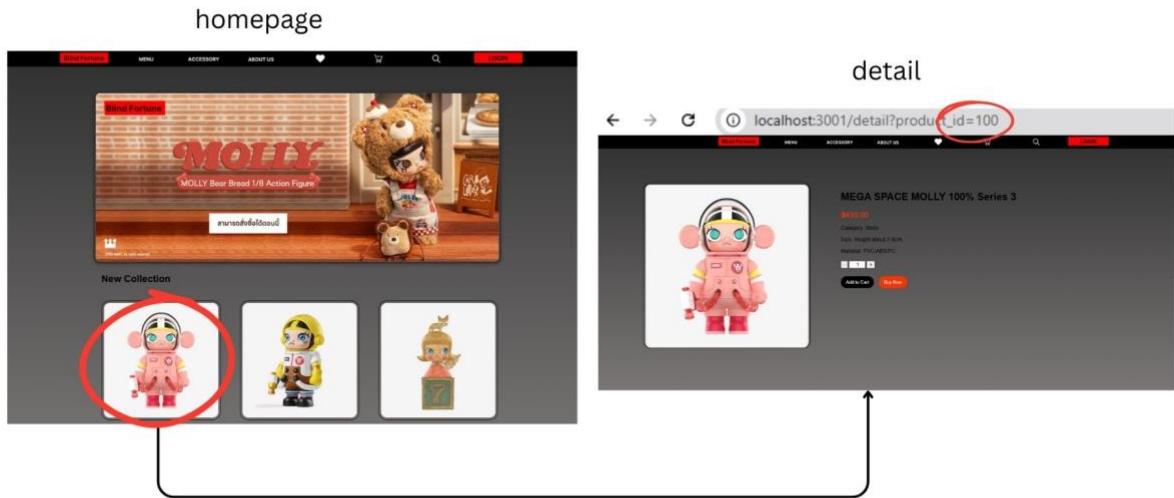
```
// ฟังก์ชันเพื่อเก็บผลลัพธ์
function updateProductGrid(sectionId, products) {
  const container = document.getElementById(sectionId);
  container.innerHTML = ''; // ล้างตัวแปรนี้

  if (products.length === 0) {
    container.innerHTML = '<p>No products found.</p>'; // แจ้งว่าไม่พบผลลัพธ์
    return;
  }

  products.forEach(product => {
    const productCard = `
      <div class="product-card">
        
        <p class="product-description">${product.name}</p>
        <p class="product-price">${product.price}</p>
        <button class="add-to-cart">
           Add To Cart
        </button>
      </div>`;
    container.innerHTML += productCard;
  });
}
```

ในหน้า accessorysearch.html จะใช้ script ในการดึง url ของนั้นๆ และค่าquery จะถูกนำมา check กับข้อมูลที่ fetch อกมามาจาก <http://localhost:3000/products> ถ้ามีค่าหมู่ตรงกันจะนำอุปกรณ์แต่งผลผ่าน function updategrid

## 5.Detail Page



```
document.addEventListener('DOMContentLoaded', () => {
  const urlParams = new URLSearchParams(window.location.search); // ดึง query string จาก URL
  const productId = urlParams.get('product_id'); // ดึงค่า product_id จาก URL

  if (productId) {
    // ดึงข้อมูลรายละเอียดสินค้า
    fetch(`http://localhost:3000/detail/${productId}`)
      .then(response => response.json())
      .then(product => {
        displayProductDetails(product); // พิ้งก์ชิ้นที่แสดงรายละเอียดสินค้า
        const category = product.category; // ดึงค่า category

        fetch('http://localhost:3000/products')
          .then(response => response.json())
          .then(data => {
            console.log('Fetched Data:', data); // ดูข้อมูลที่ fetch ได้
            const newCollection = data.filter(item => item.category === category).slice(0, 3);
            updateProductGrid('like', newCollection);
          })
          .catch(err => console.error('Error fetching products:', err));
      })
      .catch(err => console.error('Error fetching product details:', err));

  } else {
    console.error('Product ID is missing in the URL');
  }
});
```

ในหน้า detail.html มีการดึง url จากหน้าอื่นๆ และส่ง product id นั้นๆ ไปที่ link ซึ่งในหน้า detail จะเก็บ product id ที่เลือกมา และจะ fetch ข้อมูล จากserver.js

```
//หน้า detail
app.get("/detail/:product_id", (req, res) => {
  const { product_id } = req.params; // รับค่า product_id จาก URL
  const query = `
    SELECT
      product.product_id AS id,
      product.product_name AS name,
      product.product_price AS price,
      product.product_quantity AS quantity,
      categories.cat_name AS category,
      product.product_size AS size,
      product.product_material AS material,
      images_product.image_path AS image
    FROM product
    JOIN categories ON product.cat_id = categories.cat_id
    LEFT JOIN images_product ON product.product_id = images_product.id
    WHERE product.product_id = ?
    LIMIT 1 -- ดึงมาแค่รูปเดียว
  `;

  db.query(query, [product_id], (err, results) => {
    if (err) {
      console.error("Error fetching product:", err);
      res.status(500).send("Error fetching product");
    } else if (results.length === 0) {
      res.status(404).send("Product not found"); // กรุณามิ่งเพบลินค้า
    } else { // นำร่างดัวแปรที่เลือกไว้
      const image = results[0].image;
      const product = {
        product_id: results[0].id,
        product_name: results[0].name,
        product_price: results[0].price,
        product_quantity: results[0].quantity,
        category: results[0].category,
        product_size: results[0].size,
        product_material: results[0].material,
        image_path: image // ส่งคืนรูปภาพ
      };
      res.json(product);
    }
  });
});
```

และค่าที่ได้จาก /detail จะไปถูกเรียกใช้ใน productID

## 6. Management Page

### 6.1. Admin Management

Picture	ID	First Name	Last Name	Sex	Phone Number	Actions
	6687001	Keatikun	Komkeng	Male	082-5204864	
	6687002	Techit	Thititammajiaraya	Male	085-41438022	
	6687031	Pompawee	Pathompornwiwat	Female	063-1973255	
	6687088	Thanason	Boonmark	Male	080-5354455	

LOGOUT

```
// API สำหรับดึงข้อมูล admins
app.get('/admins', (req, res) => {
  const sql = `SELECT
    admin_id, admin_firstname,
    admin_lastname, admin_sex,
    admin_tel, admin_email,
    admin_address, admin_province,
    admin_zipcode, admin_username, admin_password, admin_retypepassword, image_path FROM admins`;

  db.query(sql, (err, results) => {
    if (err) throw err;
    res.json(results); // ส่งข้อมูล JSON ไปยัง client
  });
});
```

ในหน้า server.js จะใช้ get ในการดึงข้อมูล admins และจะมาแสดงข้อมูลดังกล่าว และเรียกใช้ในหน้า Management\_admin.html และใช้การ fetch เพื่อดึงข้อมูลอุปกรณ์มาแสดงผลลัพธ์ออกมา

```

const row = `

<div class="table-row" data-id="${admin.admin_id}">
    
    <span>${admin.admin_id}</span>
    <span>${admin.admin_firstname}</span>
    <span>${admin.admin_lastname}</span>
    <span>${admin.admin_sex}</span>
    <span>${admin.admin_tel}</span>
    <div class="action-buttons">
        
        
    </div>
</div>
`;
container.innerHTML += row;
}

.catch(err => {
    console.error(err);
    alert('Error retrieving admin data.');
});

```

ในหน้า Management\_admin.html จะสร้างและแสดงข้อมูลของ Admin และมีปุ่มสำหรับ edit และปุ่มสำหรับลบ

```

fetch('http://localhost:3000/admins') // ดึงข้อมูล Admin ทั้งหมด
    .then(response => {
        if (!response.ok) {
            throw new Error('Failed to fetch admin data.');
        }
        return response.json();
    })

```

การใช้ fetch เพื่อดึงข้อมูล admin ออกมายากเชิร์ฟเวอร์ผ่าน API

ទិន្នន័យ

```
[  
  {  
    "admin_id": 6687001,  
    "admin_firstname": "Keatikun",  
    "admin_lastname": "Komkeng",  
    "admin_sex": "Male",  
    "admin_tel": "082-5204864",  
    "admin_email": "Kettikungkom@gmail.com",  
    "admin_address": "លេខែង53 អ៊ូរបាលពុទ្ធនា",  
    "admin_province": "ក្រុងពេណមានគ្រ",  
    "admin_zipcode": 10160,  
    "admin_username": "Inwong",  
    "admin_password": "ong6687001",  
    "admin_retypepassword": "ong6687001",  
    "image_path": "ong.jpg"  
  },  
  {  
    "admin_id": 6687002,  
    "admin_firstname": "Techit",  
    "admin_lastname": "Thittitammajiaraya",  
    "admin_sex": "Male",  
    "admin_tel": "085-41438022",  
    "admin_email": "techit.thi@student.mahidol.edu",  
    "admin_address": "បានចាប់",  
    "admin_province": "សាកថ្មី",  
    "admin_zipcode": 73110,  
    "admin_username": "k4inui",  
    "admin_password": "pluem6687002",  
    "admin_retypepassword": "pluem6687002",  
    "image_path": "1732818410640-101598713.jpg"  
  },  
  {  
    "admin_id": 6687031,  
    "admin_firstname": "Pornpawee",  
    "admin_lastname": "Pathompornwiwat",  
    "admin_sex": "Female",  
    "admin_tel": "063-1973255",  
    "admin_email": "Bellpornpawee40@gmail.com",  
    "admin_address": "95តាមឃើញបោត់",  
    "admin_province": "សាកថ្មី",  
    "admin_zipcode": 73000,  
    "admin_username": "Bellbell",  
    "admin_password": "bell6687031",  
    "admin_retypepassword": "bell6687031",  
    "image_path": "bell.jpg"  
  },  
  {  
    "admin_id": 6687088,  
    "admin_firstname": "Thanason",  
    "admin_lastname": "Boonmark",  
    "admin_sex": "Male",  
    "admin_tel": "080-5354455",  
    "admin_email": "Twinsp217@gmail.com",  
    "admin_address": "ផែនរាតាណ",  
    "admin_province": "ព្រះនរោត្តម្ពោះ",  
    "admin_zipcode": 13180,  
    "admin_username": "Pengdeadeye",  
    "admin_password": "peng6687088",  
    "admin_retypepassword": "peng6687088",  
    "image_path": "peng.jpg"  
  }]  
]
```

Data ទិន្នន័យ admins

## edit admin

The screenshot shows a web application interface. On the left, there's a sidebar with 'Blind Fortune' logo, 'Admin' (selected), 'Category' (with sub-options: Moly, Dimoo, SkullPanda, Hirono, Crybab, Pucky, ETC), and 'Accessory' (with sub-options: Bag, Case, Doll, Mirror, Pin, Mirror, Other). A 'LOGOUT' button is at the bottom. The main area has a header 'Blind Fortune' with 'MENU', 'ACCESSION', 'ABOUT US', a search bar, and a 'LOGIN' button. Below is a table with columns: Picture, ID, First Name, Last Name, Sex, Phone Number, and Actions (Edit, Delete). The table contains four rows of administrator data. To the right is an 'EDIT ADMIN' form with fields for First Name, Last Name, Gender, Phone Number, Email (set to 'bellbell'), Address, Province, Zip Code, Password, Retype Password, and an Upload Profile Image field (no file chosen). Buttons for 'Update Admin' (green) and 'Cancel' (red) are at the bottom.

หน้าสำหรับปูม edit admin

จะใช้ค่าจาก admin\_id และจะแสดงไปยังหน้า Manage\_admin.html

```
function editAdmin(adminId) {
    window.location.href = `edit_admin.html?adminId=${adminId}`;
}
```

จากนั้นในหน้า edit\_admin.html จะดึงค่าจาก admin ที่เราเรียก มาเก็บไว้ในตัวแปร adminId

```
// ดึงค่า adminId จาก URL และเติมเข้าในฟอร์ม
const urlParams = new URLSearchParams(window.location.search);
const adminId = urlParams.get('adminId'); // จะได้ค่า adminId ที่ถูกส่งมาจาก URL
if (adminId) {
    document.getElementById('admin_id').value = adminId; // ใส่ค่า adminId เข้าในพิล็อต hidden
}
```

และนำค่า adminId ไป update ค่าใน server.js ในส่วนของ put

```

// API สำหรับบันทึกข้อมูล Admin
app.put('/editAdmin/:admin_id', upload.single('image_path'), (req, res) => {
  const admin_id = req.params.admin_id;
  const {
    admin_firstname,
    admin_lastname,
    admin_sex,
    admin_tel,
    admin_email,
    admin_address,
    admin_province,
    admin_zipcode,
    admin_password
  } = req.body;

  const image_path = req.file ? req.file.filename : null; // รูปไฟล์ที่อัปโหลด

  // สร้างรายการ SQL สำหรับอัปเดตข้อมูล
  const updates = [];
  const values = [];

  if (admin_firstname) {
    updates.push("admin_firstname = ?");
    values.push(admin_firstname);
  }
  if (admin_lastname) {
    updates.push("admin_lastname = ?");
    values.push(admin_lastname);
  }
  if (admin_sex) {
    updates.push("admin_sex = ?");
    values.push(admin_sex);
  }
  if (admin_tel) {
    updates.push("admin_tel = ?");
    values.push(admin_tel);
  }
  if (admin_email) {
    updates.push("admin_email = ?");
    values.push(admin_email);
  }
  if (admin_address) {
    updates.push("admin_address = ?");
    values.push(admin_address);
  }
  if (admin_province) {
    updates.push("admin_province = ?");
    values.push(admin_province);
  }
  if (admin_zipcode) {
    updates.push("admin_zipcode = ?");
    values.push(admin_zipcode);
  }
  if (admin_password) {
    updates.push("admin_password = ?");
    values.push(admin_password);
  }
  if (image_path) {
    updates.push("image_path = ?");
    values.push(image_path);
  }

  if (updates.length === 0) {
    return res.status(400).json({ message: "No data to update." });
  }

  values.push(admin_id);

  const sql = `UPDATE admins SET ${updates.join(", ")} WHERE admin_id = ?`;

  db.query(sql, values, (err, result) => {
    if (err) {
      console.error("Error:", err);
      return res.status(500).json({ message: "Error updating admin data", error: err });
    }
    res.json({ message: "Admin data updated successfully.", result });
  });
});

```

จากนั้นจะส่งข้อมูลกลับมาที่ edit\_admin.html โดยใช้ method put

```
// ส่ง PUT request พร้อมข้อมูล
fetch(`http://localhost:3000/editAdmin/${formData.get("admin_id")}`, {
  method: "PUT",
  body: formData,
})
```

เมื่อคุณส่งปุ่ม submit data จะส่งไปที่ form editAdminform

```
<section class="edit_admin">
  <form id="editAdminForm" action="/path_to_your_backend" method="POST" enctype="multipart/form-data">
    <fieldset>
      <h1>EDIT ADMIN</h1>

      <!-- Input hidden ผ่าน adminId -->
      <input type="hidden" id="admin_id" name="admin_id">

      <label for="admin_firstname">First Name</label>
      <input type="text" id="admin_firstname" name="admin_firstname" placeholder="Enter First Name"><br>

      <label for="admin_lastname">Last Name</label>
      <input type="text" id="admin_lastname" name="admin_lastname" placeholder="Enter Last Name"><br>

      <label for="admin_sex">Gender</label>
      <select id="admin_sex" name="admin_sex">
        <option value="">Select Gender</option>
        <option value="Male">Male</option>
        <option value="Female">Female</option>
      </select><br>

      <label for="admin_tel">Phone Number</label>
      <input type="tel" id="admin_tel" name="admin_tel" placeholder="Enter Phone Number" pattern="[0-9]{10}" maxlength="10"><br>

      <label for="admin_email">Email</label>
      <input type="email" id="admin_email" name="admin_email" placeholder="Enter Email Address"><br>

      <label for="admin_address">Address</label>
      <input type="text" id="admin_address" name="admin_address" placeholder="Enter Address"><br>

      <label for="admin_province">Province</label>
      <input type="text" id="admin_province" name="admin_province" placeholder="Enter Province"><br>

      <label for="admin_zipcode">Zip Code</label>
      <input type="text" id="admin_zipcode" name="admin_zipcode" placeholder="Enter Zip Code" pattern="[0-9]{5}" maxlength="5"><br>

      <label for="admin_password">Password</label>
      <input type="password" id="admin_password" name="admin_password" placeholder="Enter New Password"><br>

      <label for="retype_password">Retype Password</label>
      <input type="password" id="retype_password" name="admin_retypepassword" placeholder="Retype New Password"><br>

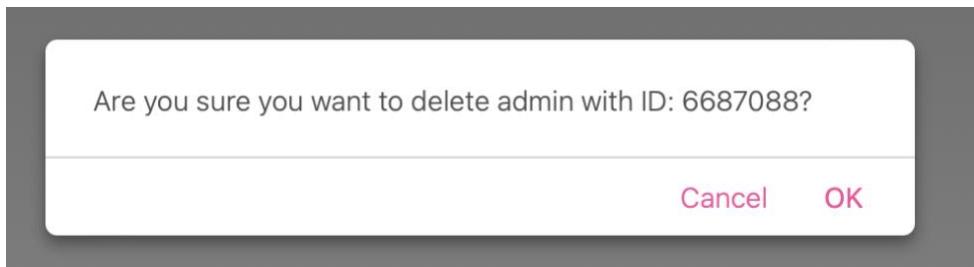
      <label for="image_path">Upload Profile Image</label>
      <input type="file" id="image_path" name="image_path" accept="image/*"><br>

      <input type="submit" value="Update Admin">
      <button type="button" id="cancelButton" onclick="window.location.href='Manage_admin.html';">Cancel</button>

      <p id="responseMessage" style="color: green;"></p>
    </fieldset>
  </form>
</section>
```

จากนั้นค่าที่ได้จะส่งไปใน file json และอัปเดตค่าโดยใช้ method ใหม่อันเดิม

```
// ส่ง PUT request พร้อมข้อมูล
fetch(`http://localhost:3000/editAdmin/${formData.get("admin_id")}`, {
  method: "PUT",
  body: formData,
})
.then((response) => {
  if (!response.ok) throw new Error("Failed to update admin.");
  return response.json();
})
```



หน้าสำหรับปุ่ม delete admin จะใช้ function deleteAdmin ในหน้า Manage\_admin.html โดยการลบ adminID 000

```
function deleteAdmin(adminId) {
  if (confirm('Are you sure you want to delete admin with ID: ${adminId}?')) {
    fetch(`http://localhost:3000/deleteAdmin/${adminId}`, {
      method: 'DELETE',
    })
    .then(response => {
      if (response.ok) {
        alert('Admin deleted successfully.');
        document.querySelector(`[data-id='${adminId}']`).remove();
      } else {
        alert('Failed to delete admin.');
      }
    })
    .catch(err => console.error('Error deleting admin:', err));
  }
}
```

และเรียกใช้ method delete ในหน้า server.js โดยให้ adminId ส่งค่ามาและลบค่าออกในsql

```

// DELETE Endpoint
app.delete('/deleteAdmin/:id', (req, res) => {
  const adminId = req.params.id;
  const sql = 'DELETE FROM admins WHERE admin_id = ?';

  db.query(sql, [adminId], (err, result) => {
    if (err) {
      console.error(err);
      return res.status(500).send('Error deleting admin.');
    }

    if (result.affectedRows === 0) {
      return res.status(404).send('Admin not found.');
    }

    res.status(200).send('Admin deleted successfully.');
  });
});

```

add admin

The screenshot shows two forms side-by-side. On the left is a table titled 'Blind Fortune' with columns: Picture, ID, First Name, Last Name, Sex, Phone Number, and Actions. A red circle highlights the 'Actions' column header. On the right is a 'REGISTER' form with fields for FIRSTNAME, LASTNAME, SEX (Male or Female), TEL, EMAIL, ADDRESS, PROVINCE, ZIP CODE, USERNAME (bellbell), PASSWORD, RETYPE PASSWORD, and an 'Upload Profile Image' input field. Below the form are 'CREATE' and 'Cancel' buttons.

Picture	ID	First Name	Last Name	Sex	Phone Number	Actions
	6687001	Keatkun	Komkong	Male	082-5204964	
	6687002	Techit	Thitthamajaraya	Male	085-41430022	
	6687031	Pompawee	Pathompomrueaw	Female	063-1973255	
	6687088	Thanason	Boonmark	Male	080-5354455	

**REGISTER**

FIRSTNAME  
LASTNAME  
SEX  
Male or Female  
TEL  
EMAIL  
ADDRESS  
PROVINCE  
ZIP CODE  
USERNAME  
bellbell  
PASSWORD  
RETYPE PASSWORD  
Upload Profile Image  
Choose File | No file chosen  
CREATE Cancel

Already have an account? [Sign in](#)

หน้าสำหรับการ add admin

ปุ่ม สำหรับแสดงไฟล์ในหน้า Register.html จะมีการสร้าง form และส่งข้อมูลโดยใช้ POST ในหน้า server.js ของ API admin คล้ายกับการทำงานในหน้า edit

```

// API ສ່ວນໃຈ Admin
app.post('/addAdmin', upload.single('image_path'), (req, res) => {
  const { admin_firstname, admin_lastname, admin_sex, admin_tel, admin_email, admin_address, admin_province, admin_zipcode, admin_username, admin_password, admin_retypepassword } = req.body;

  if (admin_password !== admin_retypepassword) {
    return res.status(400).send('Password and Retype Password do not match.');
  }

  const image_path = req.file ? req.file.filename : 'default.jpg'; // ພົມລູ້ລົມໄວ້ ຂົດ 'default.jpg'

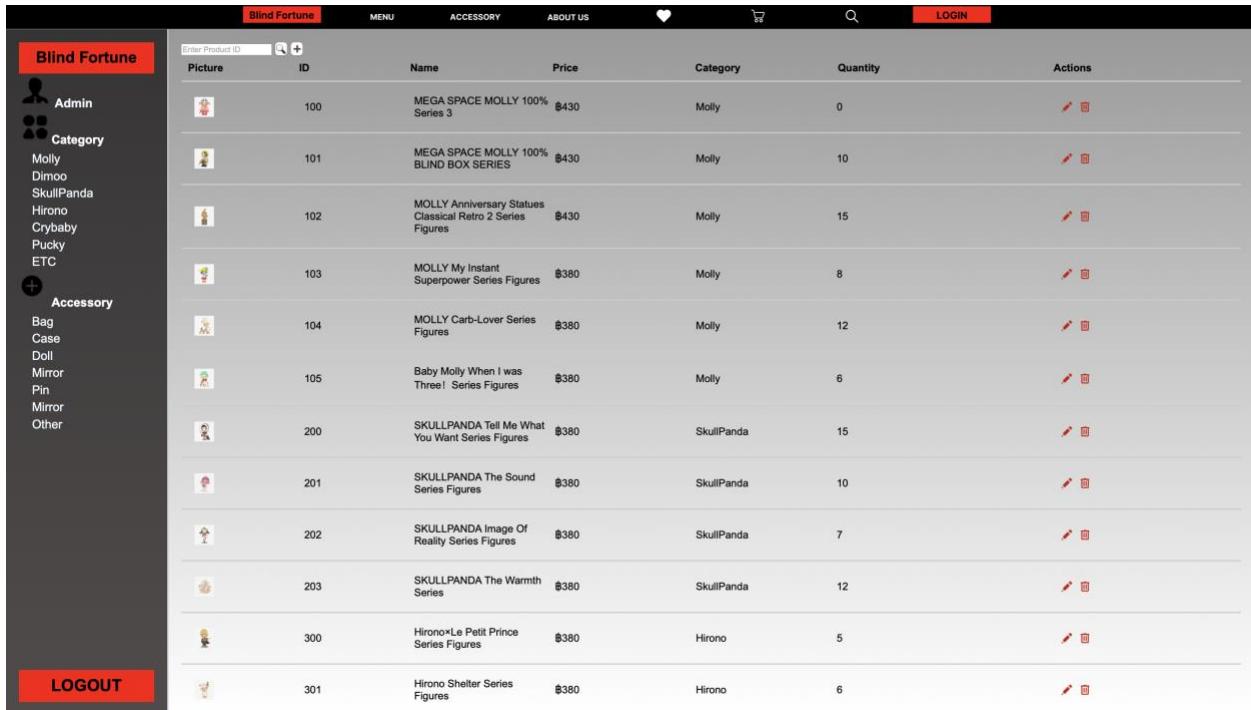
  const sql = `
    INSERT INTO admins
    (admin_firstname, admin_lastname, admin_sex, admin_tel, admin_email, admin_address, admin_province, admin_zipcode, admin_username, admin_password, admin_retypepassword, image_path)
    VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
  `;

  db.query(sql, [admin_firstname, admin_lastname, admin_sex, admin_tel, admin_email, admin_address, admin_province, admin_zipcode, admin_username, admin_password, admin_retypepassword, image_path], (err, result) => {
    if (err) {
      console.error(err);
      return res.status(500).send('Error adding admin.');
    }

    res.status(200).send('Admin added successfully.');
  });
});

```

## 6.2. Manage Product ໃຊ້ Accessory



The screenshot shows the Admin dashboard of the Blind Fortune website. The left sidebar has a dark theme with white text. It includes sections for Admin (Profile icon), Category (Molly, Dimoo, SkullPanda, Hirono, Crybabu, Pucky, ETC), and Accessory (Bag, Case, Doll, Mirror, Pin, Mirror, Other). A red 'LOGOUT' button is at the bottom of the sidebar.

The main content area has a header with 'Blind Fortune' and navigation links for MENU, ACCESSORY, ABOUT US, and LOGIN. Below is a search bar and a table of products:

Picture	ID	Name	Price	Category	Quantity	Actions
	100	MEGA SPACE MOLLY 100% Series 3	\$430	Molly	0	
	101	MEGA SPACE MOLLY 100% BLIND BOX SERIES	\$430	Molly	10	
	102	MOLLY Anniversary Statues Classical Retro 2 Series Figures	\$430	Molly	15	
	103	MOLLY My Instant Superpower Series Figures	\$380	Molly	8	
	104	MOLLY Carb-Lover Series Figures	\$380	Molly	12	
	105	Baby Molly When I was Three! Series Figures	\$380	Molly	6	
	200	SKULLPANDA Tell Me What You Want Series Figures	\$380	SkullPanda	15	
	201	SKULLPANDA The Sound Series Figures	\$380	SkullPanda	10	
	202	SKULLPANDA Image Of Reality Series Figures	\$380	SkullPanda	7	
	203	SKULLPANDA The Warmth Series	\$380	SkullPanda	12	
	300	Hirono Le Petit Prince Series Figures	\$380	Hirono	5	
	301	Hirono Shelter Series Figures	\$380	Hirono	6	

```
// API สำหรับดึงข้อมูลสินค้า
app.get('/products', (req, res) => {
  const sql = `
    SELECT
      product.product_id AS id,
      product.product_name AS name,
      product.product_price AS price,
      product.product_quantity AS quantity,
      categories.cat_name AS category,
      product.product_size AS size,
      product.product_material AS material,
      images_product.image_path AS image
    FROM product
    JOIN categories ON product.cat_id = categories.cat_id
    LEFT JOIN images_product ON product.product_id = images_product.id
  `;
  db.query(sql, (err, results) => {
    if (err) throw err;
    res.json(results);
  });
});
```

ในหน้า server.js จะใช้ get ในการดึงข้อมูล products และจะมาแสดงข้อมูลดังกล่าว และเรียกใช้ในหน้า Management\_product.html และใช้การ fetch เพื่อดึงข้อมูลจากมาแสดงผลลัพธ์ออกมา

```

// ผังสำนักอัปเดตกริดสินค้า
function updateProductGrid(sectionId, products) {
    const container = document.getElementById(sectionId);
    container.innerHTML = ''; // ล้างข้อมูลเก่า

    if (products.length === 0) {
        container.innerHTML = '<p>No products found.</p>'; // แจ้งว่าไม่พบสินค้า
        return;
    }

    products.forEach(product => {
        const row = `
            <div class="table-row" data-id="${product.id}">
                
                <span>${product.id}</span>
                <span>${product.name}</span>
                <span>฿${product.price}</span>
                <span>${product.category}</span>
                <span>${product.quantity}</span>
                <div class="action-buttons">
                    
                    
                </div>
            </div>
        `;
        container.innerHTML += row;
    });
}

```

ในหน้า Management\_product.html จะสร้างและแสดงข้อมูลของ Admin และมีปุ่มสำหรับ edit และปุ่ม

สำหรับลบ

```

function searchProducts(filterValue, filterType) {
    fetch('http://localhost:3000/products') // เรียก API ลินค้าทั้งหมด
        .then(response => {
            if (!response.ok) throw new Error('Failed to fetch products.');
            return response.json();
        })
}

```

การใช้ fetch เพื่อดึงข้อมูล admin ออกมายากเชิร์ฟเวอร์ผ่าน API

```

Pretty-print ▾
[
  {
    "id": 100,
    "name": "MEGA SPACE MOLLY 100% Series 3",
    "price": 430,
    "quantity": 9,
    "category": "Molly",
    "size": "Height about 7-8cm",
    "material": "PVC/ABS/PC",
    "image": "https://prod-thailand-res.popmart.com/default/20240731_171817_505119_1_1200x1200.jpg?x-oss-process=image/resize,p_40,format,webp,format,webp"
  },
  {
    "id": 101,
    "name": "MEGA SPACE MOLLY 100% BLIND BOX SERIES",
    "price": 430,
    "quantity": 10,
    "category": "Molly",
    "size": "Height about 7-8cm",
    "material": "PVC/ABS",
    "image": "https://prod-eurasian-res.popmart.com/default/1_vw0DGJiw2a_1200x1200.jpg?x-oss-process=image/resize,p_40,format,webp,format,webp"
  },
  {
    "id": 102,
    "name": "MOLLY Anniversary Statues Classical Retro 2 Series Figures",
    "price": 430,
    "quantity": 15,
    "category": "Molly",
    "size": "Height about 12 cm",
    "material": "ABS/PVC",
    "image": "https://prod-thailand-res.popmart.com/default/20240701_115750_825046_1_1200x1200.jpg?x-oss-process=image/resize,p_40,format,webp,format,webp"
  },
  {
    "id": 103,
    "name": "MOLLY My Instant Superpower Series Figures",
    "price": 380,
    "quantity": 8,
    "category": "Molly",
    "size": "Height about 7.8-9.9cm",
    "material": "ABS/PVC",
    "image": "https://prod-thailand-res.popmart.com/default/20241029_153000_491907_1_1200x1200.jpg?x-oss-process=image/resize,p_40,format,webp,format,webp"
  },
  {
    "id": 104,
    "name": "MOLLY Carb-Lover Series Figures",
    "price": 380,
    "quantity": 12,
    "category": "Molly"
  }
]

```

Data ข้อมูลของ Products

## edit product-category

Picture	ID	Name	Price	Category	Quantity	Actions
	100	MEGA SPACE MOLLY 100% Series 2	\$430	Molly	0	
	101	MEGA SPACE MOLLY 100% BLIND BOX SERIES	\$430	Molly	10	
	102	MOLLY Anniversary Statues Classical Retro 2 Series Figures	\$430	Molly	15	
	103	MOLLY My Instant Superpower Series Figures	\$380	Molly	8	
	104	MOLLY Carb-Lover Series Figures	\$380	Molly	12	
	105	Baby Molly When I was Three! Series Figures	\$380	Molly	6	

## edit product-accessory

Picture	ID	Name	Price	Category	Quantity	Actions
	700	THE MONSTERS FALL IN WILD SERIES 5-Denim Messenger Bag	\$1050	Bag	12	
	701	THE MONSTERS FALL IN WILD SERIES Denim Apron Bag	\$900	Bag	8	
	702	THE MONSTERS FALL IN WILD SERIES Bucket Hat Mini Bag Blind Box	\$850	Bag	15	
	703	KUBO JEANS SERIES Messenger Bag	\$850	Bag	10	

หน้าสำหรับปุ่ม edit ทั้ง product และ accessory

หน้าสำหรับปุ่ม edit product



จะใช้ค่าจาก product\_id และจะแสดงไปยังหน้า Manage\_product.html

```
function editProduct(productId) {
    window.location.href = `edit_product.html?productId=${productId}`;
}
```

จากนั้นในหน้า edit\_product.html จะดึงค่าจาก product ที่เราเรียก มาเก็บไว้ในตัวแปร productId

```

document.addEventListener('DOMContentLoaded', function () {
    // ตั้งค่า productId จาก URL และตั้งค่าให้กับฟอร์ม
    const urlParams = new URLSearchParams(window.location.search);
    const productId = urlParams.get('productId'); // ตั้งค่า productId จาก URL
    if (productId) {
        document.getElementById('product_id').value = productId; // ตั้งค่า product_id ในฟอร์ม
    }
}

```

ແລະนำค่า productId ไป update ค่าในserver.js ในส่วนของput

```

app.put('/editProduct/:product_id', upload.single('image_path'), (req, res) => {
    const product_id = req.params.product_id;
    const {
        product_name,
        product_price,
        product_quantity,
        product_size,
        product_material,
        cat_id
    } = req.body;

    const image_path = req.file ? req.file.filename : null; // ถ้าไม่มีไฟล์แนบ

    // สร้างคำสั่ง SQL สำหรับอัปเดตข้อมูล
    const updates = [];
    const values = [];

    // ตรวจสอบชื่อและเพิ่มลงในคำสั่ง UPDATE
    if (product_name) {
        updates.push("product_name = ?");
        values.push(product_name);
    }
    if (product_price) {
        updates.push("product_price = ?");
        values.push(product_price);
    }
    if (product_quantity) {
        updates.push("product_quantity = ?");
        values.push(product_quantity);
    }
    if (product_size) {
        updates.push("product_size = ?");
        values.push(product_size);
    }
    if (product_material) {
        updates.push("product_material = ?");
        values.push(product_material);
    }
    if (cat_id) {
        updates.push("cat_id = ?");
        values.push(cat_id);
    }
    if (image_path) {
        updates.push("image_path = ?");
        values.push(image_path);
    }

    // ถ้าไม่มีข้อมูลที่จะอัปเดต
    if (updates.length === 0) {
        return res.status(400).json({ message: "No data to update." });
    }

    // เพิ่ม product_id เข้ามาด้วย
    values.push(product_id);

    // สร้างคำสั่ง SQL สำหรับอัปเดตข้อมูลที่มี
    const sql = `UPDATE product SET ${updates.join(", ")} WHERE product_id = ?`;

    // Log SQL Query ลง Debug
    console.log("SQL Query:", sql, values);

    // ดำเนินการอัปเดตในฐานข้อมูล
    db.query(sql, values, (err, result) => {
        if (err) {
            console.error("Error:", err);
            return res.status(500).json({ message: "Error updating product data", error: err });
        }

        // ตรวจสอบว่ามีการอัปเดตข้อมูลหรือไม่
        if (result.affectedRows === 0) {
            return res.status(404).json({ message: "No product found with the given ID." });
        }

        res.json({ message: "Product data updated successfully.", result });
    });
}

```

จากนั้นจะส่งข้อมูลกลับมาที่ edit\_product.html โดยใช้ method put

```
// ส่งข้อมูลทั้งหมดไปยังเซิร์ฟเวอร์
fetch(`http://localhost:3000/editProduct/${formData.get('product_id')}` , {
  method: 'PUT',
  body: formData
})
```

เมื่อกดส่งปุ่ม submit data จะส่งไปที่ form editProductform

```
<div class="edit-product-form">
  <h1>Edit Product</h1>
  <form id="editProductForm" enctype="multipart/form-data">

    <input type="hidden" id="product_id" name="product_id">

    <label for="product_name">Product Name</label>
    <input type="text" id="product_name" name="product_name" placeholder="Enter Product Name">

    <label for="product_price">Price</label>
    <input type="number" id="product_price" name="product_price" placeholder="Enter Product Price">

    <label for="product_quantity">Quantity</label>
    <input type="number" id="product_quantity" name="product_quantity" placeholder="Enter Product Quantity">

    <label for="category">Category</label>
    <select id="category" name="category" required>
      <option value="Molly">Molly</option>
      <option value="SkullPanda">SkullPanda</option>
      <option value="Hirono">Hirono</option>
      <option value="Crybaby">Crybaby</option>
      <option value="Pucky">Pucky</option>
      <option value="ETC">ETC</option>
      <option value="Case">Case</option>
      <option value="Doll">Doll</option>
      <option value="Mirror">Mirror</option>
      <option value="Pin">Pin</option>
      <option value="Other">Other</option>
    </select>
    <br>

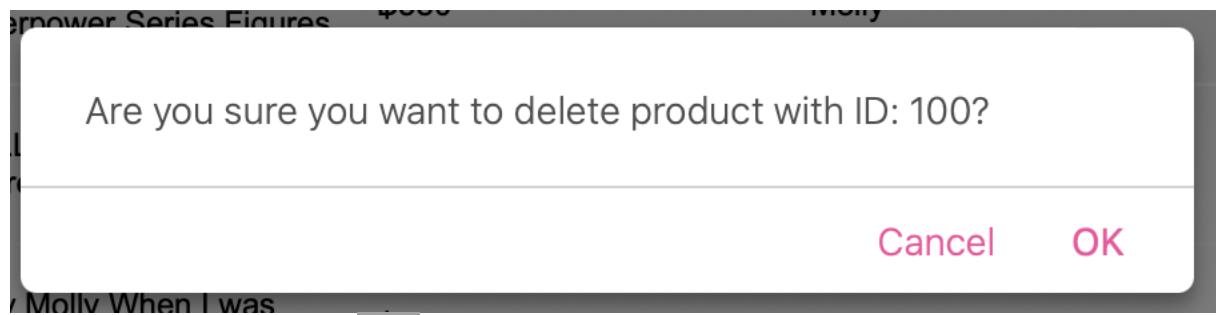
    <label for="product_size">Size</label>
    <input type="text" id="product_size" name="product_size" placeholder="Enter Product Size">

    <label for="product_material">Material</label>
    <input type="text" id="product_material" name="product_material" placeholder="Enter Product Material">

    <label for="image_path">Product Image</label>
    <input type="file" id="image_path" name="image_path">

    <button type="submit">Update Product</button>
    <button type="button" id="cancelButton"
      |   onclick="window.location.href='Manage_product.html';">Cancel</button>
  </form>
  <p id="responseMessage" class="success"></p>
```

จากนั้นค่าที่ได้จะส่งไปใน file json และอัปเดตค่าโดยใช้ method เหมือนเดิม



หน้าสำหรับปุ่ม delete product  จะใช้ function deleteProduct ในหน้า Manage\_Product.html โดยการรับ ProductID ออก

```
// พังก์ชันลบสินค้าจากตาราง
function deleteProduct(productId) {
    if (confirm(`Are you sure you want to delete product with ID: ${productId}?`)) {
        // ส่งคำขอ DELETE ไปยัง server (backend)
        fetch(`http://localhost:3000/deleteProduct/${productId}`, {
            method: 'DELETE', // ใช้ HTTP method DELETE
        })
        .then(response => {
            // ตรวจสอบสถานะการตอบกลับ
            if (response.ok) {
                alert('Product deleted successfully.');
                // ลบแท็กที่ตรงกับ productId ออกจากหน้า HTML
                document.querySelector(`[data-id='${productId}']`).remove();
            } else {
                // ถ้าไม่สามารถลบได้
                response.text().then(errorMessage => {
                    alert(`Failed to delete product: ${errorMessage}`);
                });
            }
        })
        .catch(err => {
            // แสดงข้อผิดพลาดที่เกิดขึ้น
            console.error('Error deleting product:', err);
            alert('Error deleting product. Please try again.');
        });
    }
}
```

และเรียกใช้ method delete ในหน้า server.js โดยให้ ProductId ส่งค่ามาและลบค่าออกในsql

```
// DELETE Endpoint สำหรับลบข้อมูลสินค้า
app.delete('/deleteProduct/:id', (req, res) => {
  const productId = req.params.id; // รับ product_id จาก URL parameter
  const sql = 'DELETE FROM product WHERE product_id = ?'; // สั่งลบข้อมูลสินค้าในตาราง

  db.query(sql, [productId], (err, result) => {
    if (err) {
      console.error(err);
      return res.status(500).send('Error deleting product.');
    }

    // ตรวจสอบว่าไม่มีข้อมูลที่ถูกลบ (กรณีไม่พบ ID ที่ต้องการลบ)
    if (result.affectedRows === 0) {
      return res.status(404).send('Product not found.');
    }

    // ส่งข้อความเมื่อสินค้าถูกลบสำเร็จ
    res.status(200).send('Product deleted successfully.');
  });
});

//-----END PRODUCT-----//
```

### add product-category

The screenshot shows a web application interface for managing products. On the left, there's a sidebar with navigation links for Admin, Category (Molly, Dimoo, SkulPanda, Huggers, Crybab, Pucky, ETC), and Accessory (Bag, Case, Dal, Mirror, Pin, Mirror, Other). The main area displays a table of products with columns: Picture, ID, Name, Price, Category, Quantity, and Actions. An arrow points from the 'Actions' column to an 'ADD Product' modal window on the right.

**Product Table Data:**

ID	Name	Price	Category	Quantity	Actions	
100	MEGA SPACE MOLLY 100% Series 3	\$450	Molly	0		
101	MEGA SPACE MOLLY 100% BLIND BOX SERIES	\$450	Molly	15		
102	MOLLY Anniversary Status Classical Retro 2 Series Figures	\$450	Molly	8		
103	MOLLY My Instant Superpower Series Figures	\$380	Molly	12		
104	MOLLY Cars-Lover Series Figures	\$380	Molly	6		
105	Baby Molly When I was Three! Series Figures	\$380	Molly	0		

**ADD Product Modal Fields:**

- Product Name
- Product Price
- Product Quantity
- Category (Molly)
- Size
- Material
- Upload Product Image (Choose File: no file selected)
- CREATE
- Cancel

At the bottom of the modal, it says "Already have an account? Sign in".

## add product-accessory

The screenshot shows a web application interface. On the left, there's a sidebar with a user profile ('Admin'), categories ('Category' like Molly, Demco, etc.), and a 'Logout' button. The main area has a header with 'Blind Fortune' and navigation links. A table lists four products (700, 701, 702, 703) with columns for Picture, Name, Price, Category, Quantity, and Actions. An arrow points from the 'Actions' column of product 700 to a modal window titled 'ADD Product'. The modal contains fields for Product Name, Product Price, Product Quantity, Category (set to 'Molly'), Size, Material, and an 'Upload Product Image' section with a file input. It also has 'CREATE' and 'Cancel' buttons.

หน้าสำหรับการ add product !! และ accessory

ปุ่ม สำหรับแสดงไปยังหน้า Add\_product ซึ่งในหน้า Add\_product.html จะมีการสร้าง form และส่งข้อมูลโดยใช้ POST ในหน้า server.js ของ API product คล้ายกับการทำงานในหน้า edit

```
// API สำหรับการอัปโหลดรูปภาพ
app.post('/uploadProduct', productUpload.single('image'), (req, res) => {
  if (!req.file) {
    return res.status(400).send('ไม่มีการอัปโหลดไฟล์.');
  }

  const imageUrl = `http://localhost:3000/picture_product/${req.file.filename}`;

  res.json({
    message: 'อัปโหลดไฟล์สำเร็จ!',
    imageUrl: imageUrl
  });
});
```

```

app.post('/products', upload.single('image_path'), (req, res) => {
  const {product_name, product_price, product_quantity, category, product_size, product_material} = req.body;

  // ສ້າງ URL ສໍາພັບ imagePath
  const imagePath = req.file ? `http://localhost:3000/picture_product/${req.file.filename}` : null;

  // ດ້ວຍສອນວ່າເມວຄຫຼຸມອີ່ນຕາງໆ categories
  const checkCategorySql = `
    SELECT cat_id FROM categories WHERE cat_name = ?
  `;
  db.query(checkCategorySql, [category], (err, results) => {
    if (err) {
      console.error('Error checking category:', err);
      return res.status(500).json({ message: 'Error checking category.' });
    }

    if (results.length === 0) {
      console.log(`Category "${category}" not found.`);
      return res.status(400).json({ message: `Category "${category}" does not exist.` });
    }

    const cat_id = results[0].cat_id; // ຕິດ cat_id ຈາກຕາງໆ categories

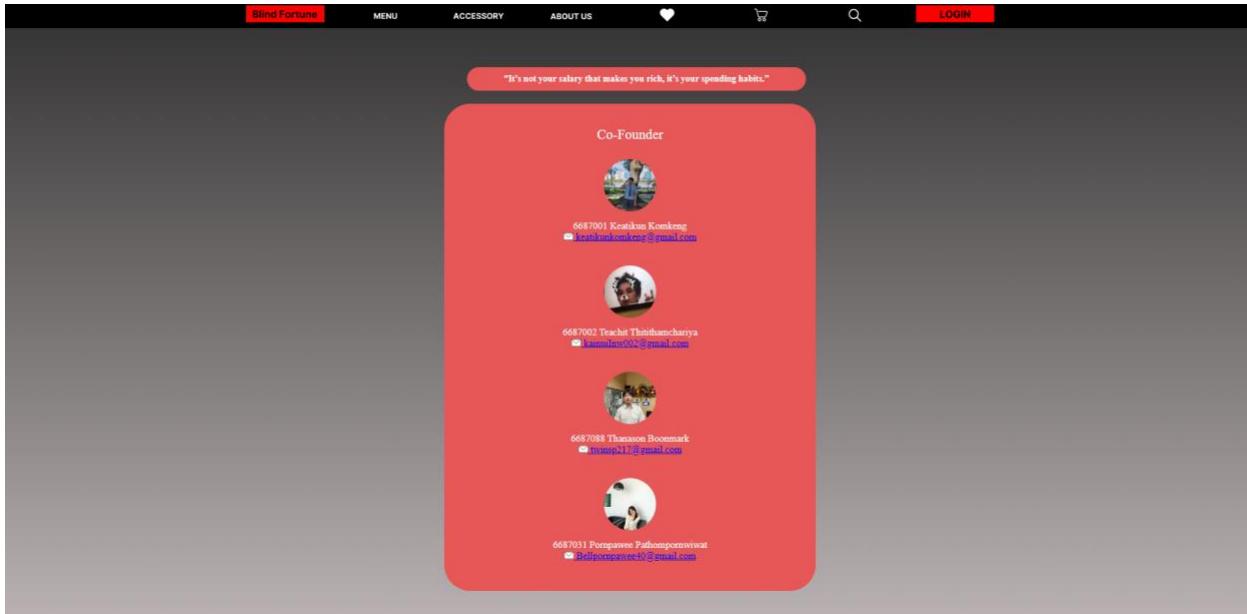
    // ເພີ່ມຂໍ້ມູນລິນຕ້າໃນຕາງໆ product
    const productSql = `
      INSERT INTO product (product_name, product_price, product_quantity, cat_id, product_size, product_material)
      VALUES (?, ?, ?, ?, ?, ?)
    `;
    db.query(productSql, [product_name, product_price, product_quantity, cat_id, product_size, product_material], (err, result) => {
      if (err) {
        console.error('Error inserting product:', err);
        return res.status(500).json({ message: 'Failed to add product.' });
      }

      const productId = result.insertId;

      // ເພີ່ມຂໍ້ມູນຮູບໃນຕາງໆ images_product
      if (imagePath) {
        const imageSql = `
          INSERT INTO images_product (id, image_path)
          VALUES (?, ?)
        `;
        db.query(imageSql, [productId, imagePath], (err, result) => {
          if (err) {
            console.error('Error inserting image:', err);
            return res.status(500).json({ message: 'Failed to add product image.' });
          }
          res.status(200).json({ message: 'Product and image added successfully.' });
        });
      } else {
        res.status(200).json({ message: 'Product added without image.' });
      }
    });
  });
});

```

## 7. Team page



```
<div class="quote">"It's not your salary that makes you rich, it's your spending habits."</div> <!-- ข้อความ Quote -->
<div class="member-container"> <!-- ส่วนแสดงสมาชิก -->
    <div class="cofounder">Co-Founder</div> <!-- พี่น้อง Co-Founder -->
    <div class="member-item"> <!-- รำยานเด็กสมาร์ทเก็ตที่ 1 -->
        <br>
        6687001 Keatikun Komkeng<br>
        <a href="mailto:keatikunkomkeng@gmail.com" class="mailong">
             keatikunkomkeng@gmail.com</a>
    </div>

    <div class="member-item"> <!-- รำยานเด็กสมาร์ทเก็ตที่ 2 -->
        <br>
        6687002 Teachit Thitithamchariya<br>
        <a href="mailto:kainuiw002@gmail.com" class="mailtae">
             kainuiw002@gmail.com</a>
    </div>

    <div class="member-item"> <!-- รำยานเด็กสมาร์ทเก็ตที่ 3 -->
        <br>
        6687088 Thanason Boonmark<br>
        <a href="mailto:twinsp217@gmail.com" class="mailpeng">
             twinsp217@gmail.com</a>
    </div>

    <div class="member-item"> <!-- รำยานเด็กสมาร์ทเก็ตที่ 4 -->
        <br>
        6687031 Pornpawee Pathompornwiwat<br>
        <a href="mailto:Bellpornpawee40@gmail.com" class="mailbell">
             Bellpornpawee40@gmail.com</a>
    </div>
</div>
```

ในหน้า aboutus.html มีการแสดงรายละเอียดของสมาชิกผู้ร่วมก่อตั้ง (Co-Founder) ขององค์กร โดยประกอบด้วยข้อมูลสำคัญของแต่ละคน ซึ่งรวมถึง รูปภาพ, ชื่อเต็ม และ url ของอีเมลที่สามารถกดเพื่อเข้าถึงได้

## รายละเอียดของ Web Service

```
// API สำหรับดึงข้อมูลสินค้า
app.get('/products', (req, res) => {
  const sql = `
    SELECT
      product.product_id AS id,
      product.product_name AS name,
      product.product_price AS price,
      product.product_quantity AS quantity,
      categories.cat_name AS category,
      product.product_size AS size,
      product.product_material AS material,
      images_product.image_path AS image
    FROM product
    JOIN categories ON product.cat_id = categories.cat_id
    LEFT JOIN images_product ON product.product_id = images_product.id
  `;
  db.query(sql, (err, results) => {
    if (err) throw err;
    res.json(results);
  });
});
```

การใช้ API โดยใช้ method get เพื่อดึงข้อมูลสินค้าจากฐานข้อมูลและส่งข้อมูลกลับไปในรูปแบบ json

```
// API สำหรับการอัปโหลดรูปภาพ
app.post('/uploadProduct', productUpload.single('image'), (req, res) => {
  if (!req.file) {
    return res.status(400).send('ไม่มีการอัปโหลดไฟล์.');
  }

  const imageUrl = `http://localhost:3000/picture_product/${req.file.filename}`;

  res.json({
    message: 'อัปโหลดไฟล์สำเร็จ!',
    imageUrl: imageUrl
  });
});
```

การใช้ API โดยใช้ method post สำหรับการอัปโหลดรูปภาพสินค้าโดยรับรูปภาพจาก form และบันทึก

```

app.put('/editProduct/:product_id', upload.single('image_path'), (req, res) => {
  const product_id = req.params.product_id;
  const {
    product_name,
    product_price,
    product_quantity,
    product_size,
    product_material,
    cat_id
  } = req.body;

  const image_path = req.file ? req.file.filename : null; // ชื่อไฟล์ที่อัปโหลด
  // สั่งค่าสิ่ง SQL สำหรับอัปเดตข้อมูล
  const updates = [];
  const values = [];

  // ตรวจสอบข้อมูลและเพิ่มลงในค่าสิ่ง UPDATE
  if (product_name) {
    updates.push("product_name = ?");
    values.push(product_name);
  }
  if (product_price) {
    updates.push("product_price = ?");
    values.push(product_price);
  }
  if (product_quantity) {
    updates.push("product_quantity = ?");
    values.push(product_quantity);
  }
  if (product_size) {
    updates.push("product_size = ?");
    values.push(product_size);
  }
  if (product_material) {
    updates.push("product_material = ?");
    values.push(product_material);
  }
  if (cat_id) {
    updates.push("cat_id = ?");
    values.push(cat_id);
  }
  if (image_path) {
    updates.push("image_path = ?");
    values.push(image_path);
  }

  // ถ้าไม่มีข้อมูลที่จะอัปเดต
  if (updates.length === 0) {
    return res.status(400).json({ message: "No data to update." });
  }

  // เพิ่ม product_id เป็นค่าปลอกหาด
  values.push(product_id);

  // สั่งค่าสิ่ง SQL สำหรับอัปเดตข้อมูลสินค้า
  const sql = `UPDATE product SET ${updates.join(", ")} WHERE product_id = ?`;

  // Log SQL Query เฟื่อง Debug
  console.log("SQL Query:", sql, values);

  // ดำเนินการอัปเดตในฐานข้อมูล
  db.query(sql, values, (err, result) => {
    if (err) {
      console.error("Error:", err);
      return res.status(500).json({ message: "Error updating product data", error: err });
    }

    // ตรวจสอบว่ามีการอัปเดตข้อมูลหรือไม่
    if (result.affectedRows === 0) {
      return res.status(404).json({ message: "No product found with the given ID." });
    }

    res.json({ message: "Product data updated successfully.", result });
  });
});

```

การใช้ API โดยใช้ method put สำหรับแก้ไขข้อมูลสินค้าและupdateสินค้าในฐานข้อมูล

```

app.post('/products', upload.single('image_path'), (req, res) => {
  const {product_name, product_price, product_quantity, category, product_size, product_material} = req.body;

  // สร้าง URL สำหรับ imagePath
  const imagePath = req.file ? `http://localhost:3000/picture_product/${req.file.filename}` : null;

  // ตรวจสอบว่าหมวดหมู่มีอยู่ในตาราง categories
  const checkCategorySql = `
    SELECT cat_id FROM categories WHERE cat_name = ?
  `;
  db.query(checkCategorySql, [category], (err, results) => {
    if (err) {
      console.error('Error checking category:', err);
      return res.status(500).json({ message: 'Error checking category.' });
    }

    if (results.length === 0) {
      console.log(`Category "${category}" not found.`);
      return res.status(400).json({ message: `Category "${category}" does not exist.` });
    }

    const cat_id = results[0].cat_id; // ตั้ง cat_id จากตาราง categories

    // เพิ่มข้อมูลสินค้าในตาราง product
    const productService = `
      INSERT INTO product (product_name, product_price, product_quantity, cat_id, product_size, product_material)
      VALUES (?, ?, ?, ?, ?, ?)
    `;
    db.query(productService, [product_name, product_price, product_quantity, cat_id, product_size, product_material], (err, result) => {
      if (err) {
        console.error('Error inserting product:', err);
        return res.status(500).json({ message: 'Failed to add product.' });
      }

      const productId = result.insertId;

      // เพิ่มรูปภาพในตาราง images_product
      if (imagePath) {
        const imageSql = `
          INSERT INTO images_product (id, image_path)
          VALUES (?, ?)
        `;
        db.query(imageSql, [productId, imagePath], (err, result) => {
          if (err) {
            console.error('Error inserting image:', err);
            return res.status(500).json({ message: 'Failed to add product image.' });
          }
          res.status(200).json({ message: 'Product and image added successfully.' });
        });
      } else {
        res.status(200).json({ message: 'Product added without image.' });
      }
    });
  });
});

```

การใช้ API โดยใช้ method post เพื่อเพิ่มข้อมูลสินค้าลงในฐานข้อมูลและบันทึกรูปภาพของสินค้าที่เพิ่มเข้ามา

```

// DELETE Endpoint สำหรับลบข้อมูลสินค้า
app.delete('/deleteProduct/:id', (req, res) => {
  const productId = req.params.id; // รับ product_id จาก URL parameter
  const sql = 'DELETE FROM product WHERE product_id = ?'; // สั่งลบข้อมูลสินค้าในตาราง

  db.query(sql, [productId], (err, result) => {
    if (err) {
      console.error(err);
      return res.status(500).send('Error deleting product.');
    }

    // ตรวจสอบว่าไม่มีข้อมูลที่ถูกลบ (กรณีไม่พบ ID ที่ต้องการลบ)
    if (result.affectedRows === 0) {
      return res.status(404).send('Product not found.');
    }

    // แสดงความเมื่อเพิ่งลบสินค้า
    res.status(200).send('Product deleted successfully.');
  });
});

```

การใช้ API โดยใช้ method delete สำหรับการลบข้อมูลสินค้าใน table product มีการระบุ id ของ product ที่จะลบผ่าน URL parameter

```

// API สำหรับดึงข้อมูลadmins
app.get('/admins', (req, res) => {
  const sql = `SELECT
    admin_id, admin_firstname,
    admin_lastname, admin_sex,
    admin_tel, admin_email,
    admin_address, admin_province,
    admin_zipcode, admin_username, admin_password, admin_retypepassword, image_path FROM admins`;

  db.query(sql, (err, results) => {
    if (err) throw err;
    res.json(results); // ส่งข้อมูล JSON ไปยัง client
  });
});

```

การใช้ API โดยใช้ method get สำหรับการดึงข้อมูล admins จากฐานข้อมูลและส่งข้อมูลที่ได้กลับไปในรูปแบบ json

```

// API สำหรับเพิ่ม Admin
app.post('/addAdmin', upload.single('image_path'), (req, res) => {
  const { admin_firstname, admin_lastname, admin_sex, admin_tel, admin_email, admin_address, admin_province, admin_zipcode, admin_username, admin_password, admin_retypepassword } = req.body;

  if (admin_password !== admin_retypepassword) {
    return res.status(400).send('Password and Retype Password do not match.');
  }

  const image_path = req.file ? req.file.filename : 'default.jpg'; // กรณีไม่มีไฟล์ 'default.jpg'

  const sql = `
    INSERT INTO admins
    (admin_firstname, admin_lastname, admin_sex, admin_tel, admin_email, admin_address, admin_province, admin_zipcode, admin_username, admin_password, admin_retypepassword, image_path)
    VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
  `;

  db.query(sql, [admin_firstname, admin_lastname, admin_sex, admin_tel, admin_email, admin_address, admin_province, admin_zipcode, admin_username, admin_password, admin_retypepassword, image_path], (err, result) => {
    if (err) {
      console.error(err);
      return res.status(500).send('Error adding admin.');
    }

    res.status(200).send('Admin added successfully.');
  });
});

```

การใช้ API โดยใช้ method post สำหรับการเพิ่มข้อมูล Admin ลงในฐานข้อมูลและบันทึกลงในตาราง admins

```

// API สำหรับแก้ไขข้อมูล Admin
app.put('/editadmin/:admin_id', upload.single('image_path'), (req, res) => {
  const admin_id = req.params.admin_id;
  const {
    admin_firstname,
    admin_lastname,
    admin_sex,
    admin_tel,
    admin_email,
    admin_address,
    admin_province,
    admin_zipcode,
    admin_password
  } = req.body;

  const image_path = req.file ? req.file.filename : null; // ชื่อไฟล์ที่อัปโหลด

  // สร้างภาษาตัว SQL สำหรับอัปเดตข้อมูล
  const updates = [];
  const values = [];

  if (admin_firstname) {
    updates.push("admin_firstname = ?");
    values.push(admin_firstname);
  }
  if (admin_lastname) {
    updates.push("admin_lastname = ?");
    values.push(admin_lastname);
  }
  if (admin_sex) {
    updates.push("admin_sex = ?");
    values.push(admin_sex);
  }
  if (admin_tel) {
    updates.push("admin_tel = ?");
    values.push(admin_tel);
  }
  if (admin_email) {
    updates.push("admin_email = ?");
    values.push(admin_email);
  }
  if (admin_address) {
    updates.push("admin_address = ?");
    values.push(admin_address);
  }
  if (admin_province) {
    updates.push("admin_province = ?");
    values.push(admin_province);
  }
  if (admin_zipcode) {
    updates.push("admin_zipcode = ?");
    values.push(admin_zipcode);
  }
  if (admin_password) {
    updates.push("admin_password = ?");
    values.push(admin_password);
  }
  if (image_path) {
    updates.push("image_path = ?");
    values.push(image_path);
  }

  if (updates.length === 0) {
    return res.status(400).json({ message: "No data to update." });
  }

  values.push(admin_id);

  const sql = `UPDATE admins SET ${updates.join(", ")} WHERE admin_id = ?`;

  db.query(sql, values, (err, result) => {
    if (err) {
      console.error("Error:", err);
      return res.status(500).json({ message: "Error updating admin data", error: err });
    }
    res.json({ message: "Admin data updated successfully.", result });
  });
});

```

การใช้ API โดยใช้ method put สำหรับการแก้ไขข้อมูล Admin ในฐานข้อมูล

```

// DELETE Endpoint
app.delete('/deleteAdmin/:id', (req, res) => {
  const adminId = req.params.id;
  const sql = 'DELETE FROM admins WHERE admin_id = ?';

  db.query(sql, [adminId], (err, result) => {
    if (err) {
      console.error(err);
      return res.status(500).send('Error deleting admin.');
    }

    if (result.affectedRows === 0) {
      return res.status(404).send('Admin not found.');
    }

    res.status(200).send('Admin deleted successfully.');
  });
});

```

การใช้ API โดยใช้ method delete สำหรับการลบข้อมูล Admin จากฐานข้อมูลและระบุ id ของ admin ผ่าน

#### URL parameter

```

app.post('/login', (req, res) => {
  const { username, password } = req.body;

  const query = 'SELECT * FROM admins WHERE admin_username = ? AND admin_password = ?';
  db.execute(query, [username, password], (err, result) => {
    if (err) {
      console.error('Database error:', err.message);
      return res.status(500).send('Database error: ' + err.message);
    }

    if (result.length > 0) {
      const insertQuery = 'INSERT INTO login (Username, admin_password) VALUES (?, ?)';
      db.execute(insertQuery, [username, password], (err, insertResult) => {
        if (err) {
          console.error('Error logging login attempt:', err.message);
          return res.status(500).send('Error logging login attempt: ' + err.message);
        }
        res.status(200).send('Login successful!');
      });
    } else {
      res.send('Invalid username or password!');
    }
  });
});

```

การใช้ API โดยใช้ method post สำหรับการเข้าสู่ระบบของ Admin และบันทึกลงในฐานข้อมูล

```
app.get('/login',(req,res)=>{
  console.log('req select all')
  let sql = `select * from login`;
  db.query(sql, function(err,result){
    if (err){
      throw err;
    }
    res.json(result)
  });
});
```

การใช้ API โดยใช้ method get ที่ใช้ query จากตาราง login ในฐานข้อมูล ส่งข้อมูลกลับไปในรูปแบบ json

```
app.post('/send', async (req, res) => {
  const { message } = req.body;

  try {
    const response = await axios.post(
      'https://notify-api.line.me/api/notify',
      `message=${encodeURIComponent(message)}`,
      {
        headers: {
          'Authorization': `Bearer ${LINE_NOTIFY_TOKEN}`,
          'Content-Type': 'application/x-www-form-urlencoded',
        },
      }
    );

    if (response.status === 200) {
      res.json({ status: 'success', message: 'ส่งข้อความสำเร็จ!' });
    } else {
      res.status(400).json({ status: 'error', message: 'ส่งข้อความล้มเหลว!' });
    }
  } catch (error) {
    console.error(error);
    res.status(500).json({ status: 'error', message: 'เกิดข้อผิดพลาด!' });
  }
});
```

การใช้ API โดยใช้ method post สำหรับรับคำขอจากผู้ใช้ และส่งข้อความไปยัง LINE Notify API เพื่อส่งข้อความแจ้งเตือนใน LINE โดยการใช้ axios ในการทำ HTTP request ไปยัง API ของ LINE Notify ต่อ

```

//หน้า detail
app.get("/detail/:product_id", (req, res) => {
  const { product_id } = req.params; // รับค่า product_id จาก URL
  const query = `
    SELECT
      product.product_id AS id,
      product.product_name AS name,
      product.product_price AS price,
      product.product_quantity AS quantity,
      categories.cat_name AS category,
      product.product_size AS size,
      product.product_material AS material,
      images_product.image_path AS image
    FROM product
    JOIN categories ON product.cat_id = categories.cat_id
    LEFT JOIN images_product ON product.product_id = images_product.id
    WHERE product.product_id = ?
    LIMIT 1 -- ดึงมาแค่รูปเดียว
  `;
  db.query(query, [product_id], (err, results) => {
    if (err) {
      console.error("Error fetching product:", err);
      res.status(500).send("Error fetching product");
    } else if (results.length === 0) {
      res.status(404).send("Product not found"); // กรณีไม่พบสินค้า
    } else { //มาร่างตัวแปรที่เลือกไว้
      const image = results[0].image;
      const product = {
        product_id: results[0].id,
        product_name: results[0].name,
        product_price: results[0].price,
        product_quantity: results[0].quantity,
        category: results[0].category,
        product_size: results[0].size,
        product_material: results[0].material,
        image_path: image // ส่งคืนรูปภาพ
      };
      res.json(product);
    }
  });
});

```

การใช้ API โดยใช้ method get สำหรับการดึงรายละเอียดข้อมูลสินค้าจากฐานข้อมูล โดยมี id ของ product ที่ส่งมาจาก url และส่งข้อมูลกลับไปในรูปแบบ json

## การใช้ API สำหรับ

ในหน้า server.js ใช้ path send ของ line notification และใช้ post ในการส่ง

```
app.post('/send', async (req, res) => {
  const { message } = req.body;

  try {
    const response = await axios.post(
      'https://notify-api.line.me/api/notify',
      `message=${encodeURIComponent(message)}`,
      {
        headers: {
          'Authorization': `Bearer ${LINE_NOTIFY_TOKEN}`,
          'Content-Type': 'application/x-www-form-urlencoded',
        },
      }
    );

    if (response.status === 200) {
      res.json({ status: 'success', message: 'ส่งข้อความสำเร็จ!' });
    } else {
      res.status(400).json({ status: 'error', message: 'ส่งข้อความล้มเหลว!' });
    }
  } catch (error) {
    console.error(error);
    res.status(500).json({ status: 'error', message: 'เกิดข้อผิดพลาด!' });
  }
});
```

ในหน้าAdd\_product จะส่ง data จาก formData และproduct\_name จะส่งชื่อของสินค้าไปที่หน้า

Add\_product.html ซึ่งจะส่งไปที่ port localhost3000 path send และระบบจะแจ้งเตือนสินค้าที่ส่งไปแล้ว

```
// ส่งข้อความร่องรอยผ่าน LINE Notify
const message = `Product added successfully:\n- Name: ${formData.get('product_name')}\n- Category: ${formData.get('category')}`;
const notifyResponse = await fetch('http://localhost:3000/send', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify({ message })
});

const notifyResult = await notifyResponse.json();
if (notifyResult.status !== 'success') {
  throw new Error(notifyResult.message);
}

statusElement.textContent = '✅ Product added and notification sent successfully!';
statusElement.style.color = 'green';
} catch (error) {
  console.error('Error:', error);
  statusElement.textContent = '✖ ${error.message}';
  statusElement.style.color = 'red';
}
```

ผลลัพธ์จากการส่งสินค้าที่หน้า Add\_product.html

## ADD Product

Product Name

---

Product Price

Product Quantity

Category

Size

---

Material

---

Upload Product Image

no file selected

CREATE Cancel

Already have an account? [Sign in](#)

และในหน้าserver.jsจะต้องใช้ เพื่อที่จะเก็บ Access Token สำหรับใช้งาน LINE Notify API

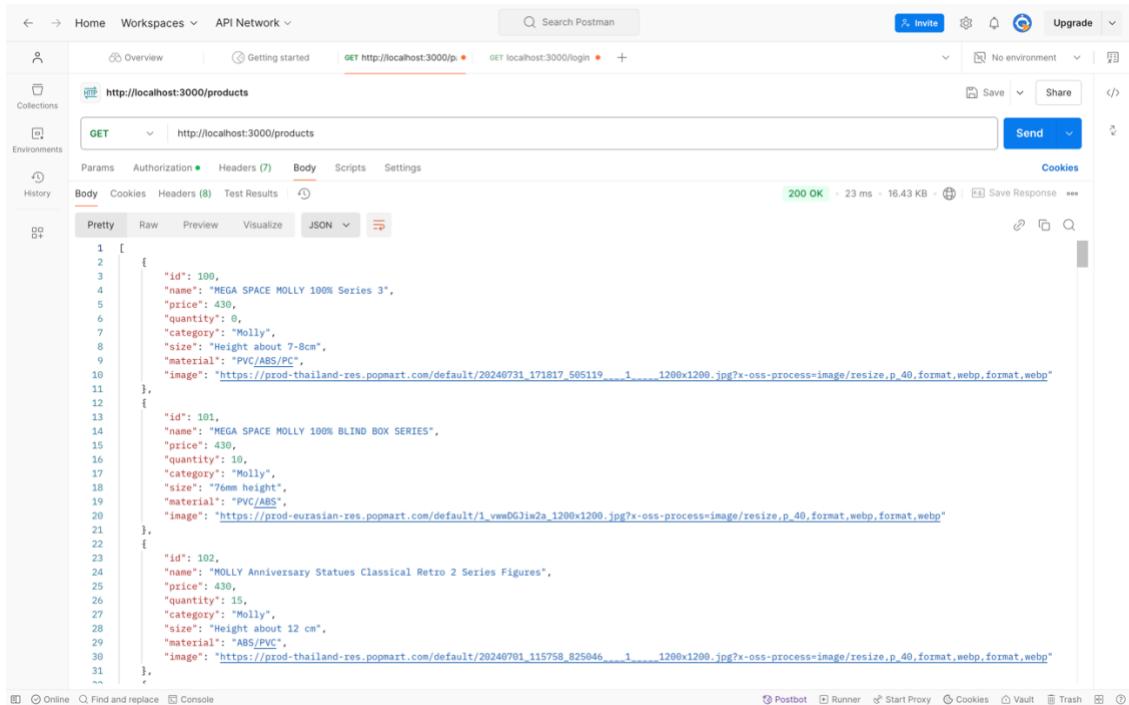
```
const LINE_NOTIFY_TOKEN = 'dec5WgC31a2IN84avCw1sN3NvCEf04bV4IOzzbsYWp7';
```

ผลลัพธ์จากการใช้ line Notify จะแสดงข้อมูลสินค้าที่ add ไว้ และส่งแจ้งเตือนพร้อมรายละเอียดสินค้า



## ผลการทดสอบของเว็บซอฟต์แวร์วิสทั้งหมด โดยใช้โปรแกรม Postman

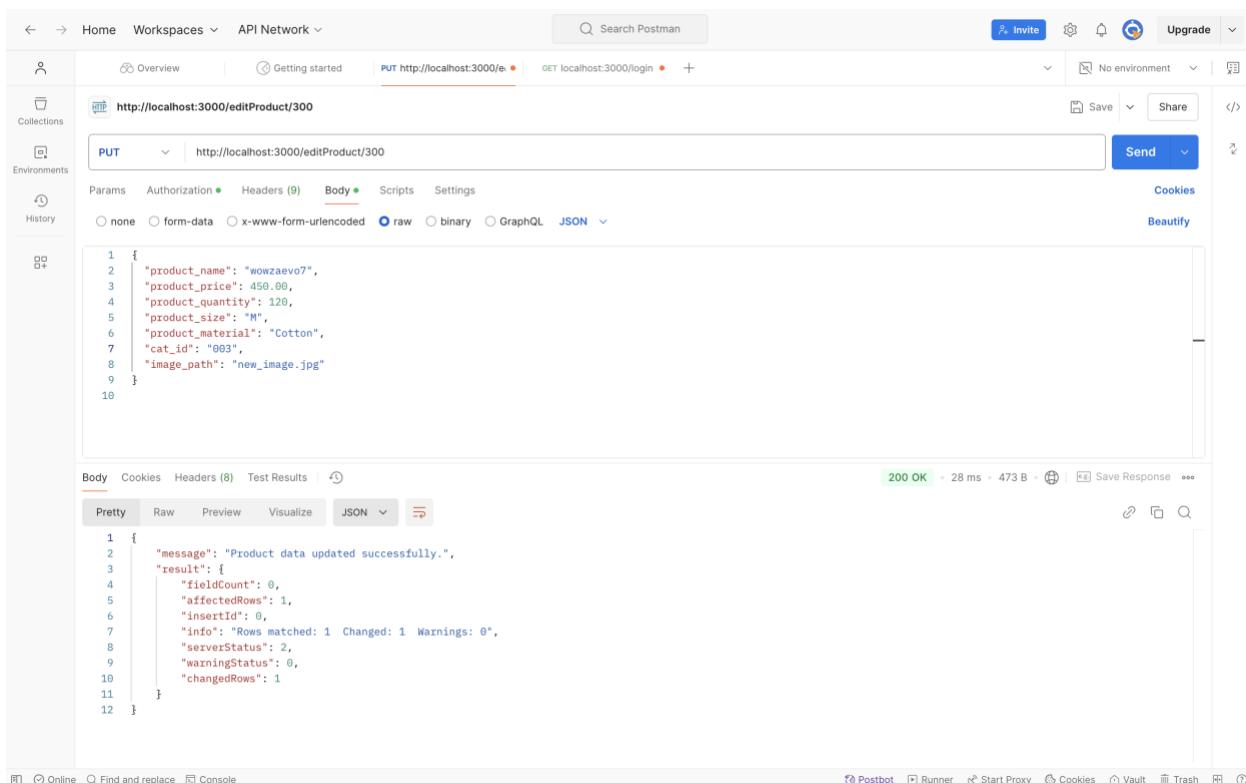
Method Get - http://localhost:3000/products



The screenshot shows the Postman interface with a successful GET request to `http://localhost:3000/products`. The response status is 200 OK, and the response body is a JSON array of products:

```
[{"id": 1, "name": "MEGA SPACE MOLLY 100% Series 3", "price": 430, "quantity": 8, "category": "Molly", "size": "Height about 7-8cm", "material": "PVC/ABS/PC", "image": "https://prod-thailand-res.popmart.com/default/20240731_171817_505119..._1.....1200x1200.jpg?x-oss-process=image/resize,p_40/format,webp/format,webp"}, {"id": 101, "name": "MEGA SPACE MOLLY 100% BLIND BOX SERIES", "price": 430, "quantity": 10, "category": "Molly", "size": "76mm height", "material": "PVC/ABS", "image": "https://prod-eurasian-res.popmart.com/default/1_vvmDGJiwm2a_1200x1200.jpg?x-oss-process=image/resize,p_40/format,webp/format,webp"}, {"id": 102, "name": "MOLLY Anniversary Statues Classical Retro 2 Series Figures", "price": 430, "quantity": 15, "category": "Molly", "size": "Height about 12 cm", "material": "ABS/PVC", "image": "https://prod-thailand-res.popmart.com/default/20240701_115758_825046..._1.....1200x1200.jpg?x-oss-process=image/resize,p_40/format,webp/format,webp"}]
```

Method Put - http://localhost:3000/editProduct/300



The screenshot shows the Postman interface with a successful PUT request to `http://localhost:3000/editProduct/300`. The response status is 200 OK, and the response body is a JSON object:

```
{"message": "Product data updated successfully.", "result": { "fieldCount": 0, "affectedRows": 1, "insertId": 0, "info": "Rows matched: 1 Changed: 1 Warnings: 0", "serverStatus": 2, "warningStatus": 0, "changedRows": 1 }}
```

## Method Delete- <http://localhost:3000/deleteProduct/100>

The screenshot shows the Postman interface for a DELETE request. The URL is <http://localhost:3000/deleteProduct/100>. The response status is 200 OK, and the body contains the message "Product deleted successfully."

## Method Post- <http://localhost:3000/add-product>

The screenshot shows the Postman interface for a POST request. The URL is <http://localhost:3000/add-product>. The body is a JSON object with the following content:

```
1 "product_name": "Moodeng",
2 "product_price": 1500,
3 "product_quantity": 10,
4 "product_size": "Medium",
5 "product_material": "SVG",
6 "cat_id": "001"
```

The response status is 200 OK, and the body contains the message "เพิ่มสินค้าสำเร็จ!"

GET - http://localhost:3000/admins

The screenshot shows the Postman interface with a collection named "My first collection". Inside this collection are two folders: "First folder inside collection" and "Second folder inside collection", each containing several requests. A modal window is open for a GET request to "http://localhost:3000/admins". The "Body" tab is selected, showing a JSON response:

```
1 [  
2   {  
3     "admin_id": 6687001,  
4     "admin_firstname": "Keatikun",  
5     "admin_lastname": "Komkeng",  
6     "admin_sex": "Female",  
7     "admin_tel": "082-5284864",  
8     "admin_email": "Kettikungkom@gmail.com",  
9     "admin_address": "เลขที่ 123 หมู่บ้านทิพย์",  
10    "admin_province": "กรุงเทพมหานคร",  
11    "admin_zipcode": 10160,  
12    "admin_username": "Tawand"
```

PUT - http://localhost:3000/editAdmin/6687031

The screenshot shows the Postman interface with a collection named "My first collection". Inside this collection is a single request to "http://localhost:3000/editAdmin/6687031". The "Body" tab is selected, showing a JSON payload:

```
1 {  
2   "admin_firstname": "John",  
3   "admin_lastname": "Doe",  
4   "admin_sex": "Male",  
5   "admin_tel": "1234567890",  
6   "admin_email": "john.doe@example.com",  
7   "admin_address": "123 Main St",  
8   "admin_province": "Bangkok",  
9   "admin_zipcode": "10110",  
10  "admin_password": "newpassword123"  
11 }  
12
```

The response body is a JSON object:

```
1 {  
2   "message": "Admin data updated successfully.",  
3   "result": {  
4     "fieldCount": 0,  
5     "affectedRows": 1,  
6     "insertId": 0,  
7     "info": "Rows matched: 1  Changed: 1  Warnings: 0",  
8     "serverStatus": 2,  
9     "warningStatus": 0,  
10    "changedRows": 1  
11  }  
12 }
```

## POST- http://localhost:3000/addAdmin

The screenshot shows the Postman interface with a successful POST request to `http://localhost:3000/addAdmin`. The request body contains the following JSON:

```
1 {
2   "admin_firstname": "Peng",
3   "admin_lastname": "Deadeye",
4   "admin_sex": "Male",
5   "admin_tel": "0834444199",
6   "admin_email": "peng.eye@example.com",
7   "admin_address": "97 wow street",
8   "admin_province": "Bangkok",
9   "admin_zipcode": "10900",
10  "admin_username": "pengdie",
11  "admin_password": "wow123",
12  "admin_retypepassword": "wow123",
13  "image_path": "file_example.jpg"
14 }
```

The response status is **200 OK** with a response time of 72 ms and a response size of 285 B. The response body is: **1 Admin added successfully.**

## DELETE- http://localhost:3000/deleteAdmin/6687002

The screenshot shows the Postman interface with a successful DELETE request to `http://localhost:3000/deleteAdmin/6687002`. The response status is **200 OK** with a response time of 20 ms and a response size of 287 B. The response body is: **1 Admin deleted successfully.**

Reference

<https://www.popmart.com/th>