



รายงาน

Project Web Phase 2

โดย

6687015	นางสาวฐิตารีย์	กองแก้ว
6687019	นางสาวดารากร	แซ่บ
6687031	นางสาวพรปวิณ	ปฐมพรวิวัฒน์
6687057	นางสาวสุธาราทพิพิร্য	หลวงทิพย์
6687093	นางสาวภัทรกร	ต้นสมบูรณ์

เสนอ

ผศ. ดร. จิตาภา ไกรสังข์

ดร. วุฒิชาติ แสงผล

รายงานนี้เป็นส่วนหนึ่งของรายวิชา ITDS242 Web Technologies Lab

คณะเทคโนโลยีสารสนเทศและการสื่อสาร มหาวิทยาลัยมหิดล

ภาคเรียนที่ 1 ปีการศึกษา 2567

## Phase 1

### คำอธิบายธุรกิจ

บริษัทที่จำหน่ายสินค้าเกี่ยวกับอนิเมะ "Haikyu!!"

ก่อตั้งขึ้นจากความนิยมของอนิเมะและมังงะเรื่องนี้ในกลุ่มแฟนคลับทั่วโลก

โดยเริ่มต้นจากการจำหน่ายสินค้าเล็ก ๆ น้อย ๆ

เพื่อให้แฟนคลับสามารถซื้อสินค้าที่รักได้โดยตรง

ต่อมา ทางเราได้นำธุรกิจนี้มาอ้างอิงทำเป็นเว็บไซต์ออนไลน์ที่จำหน่ายสินค้าหลากหลายประเภท

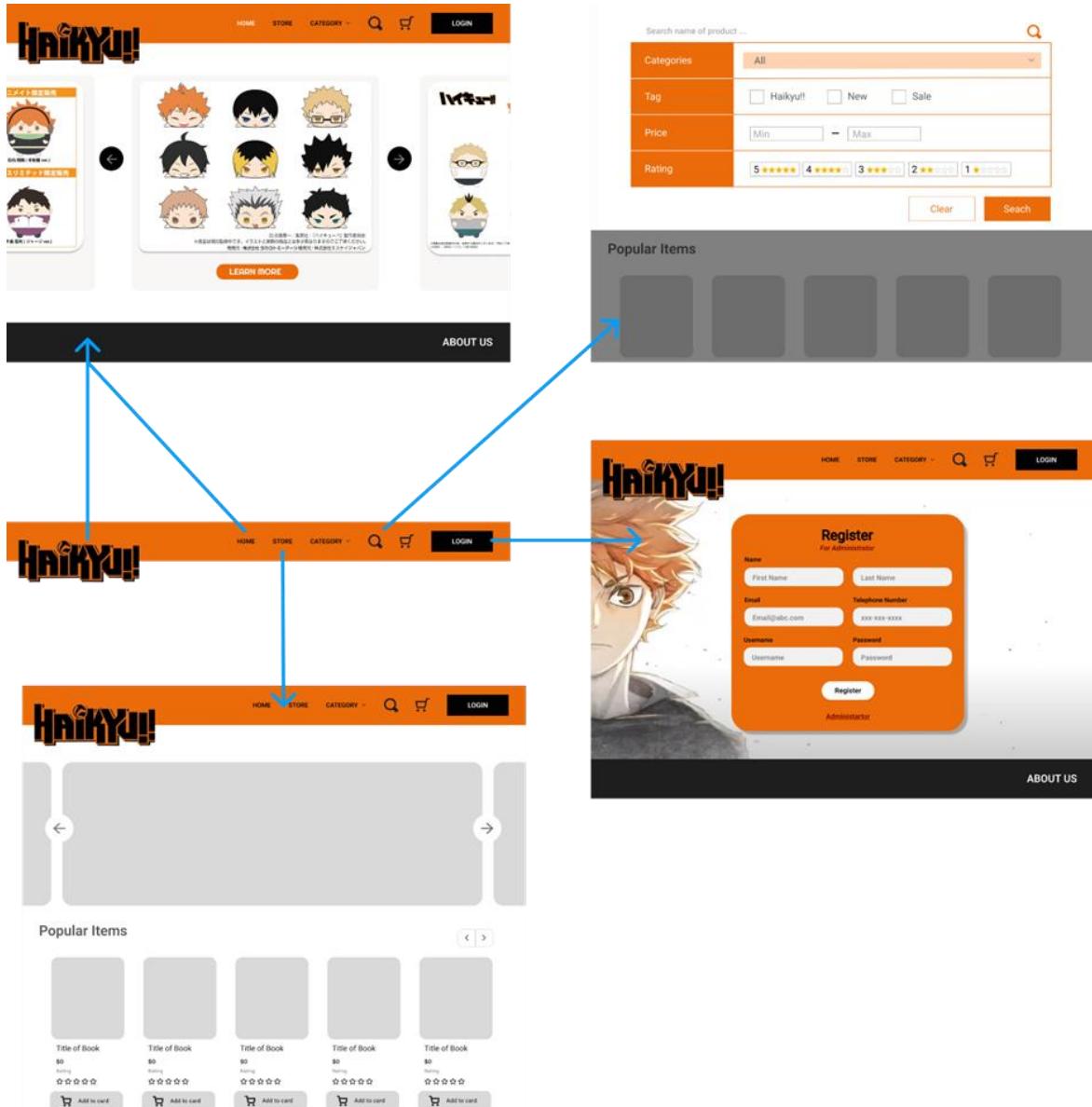
สินค้าที่มีจำหน่ายได้แก่ ฟิกเกอร์ โนรุ หนังสือการ์ตูน มังงะ และสินค้าลิขสิทธิ์อื่น ๆ

ซึ่งได้รับความนิยมอย่างมาก

รูปแบบและตัวอย่างของเว็บไซต์ของธุรกิจชี้อ้ายสินค้าไอยคิวนี้อ้างอิงมาจากเว็บไซต์ของร้านอะนิเมท

นอกจากนี้ขอบเขตของธุรกิจยังครอบคลุมการขายสินค้าทั่วประเทศผ่านช่องทางออนไลน์

## ແບນນຳທາງຂອງເວີ້ປີ່ເຊື່ອ (Navigation Bar)



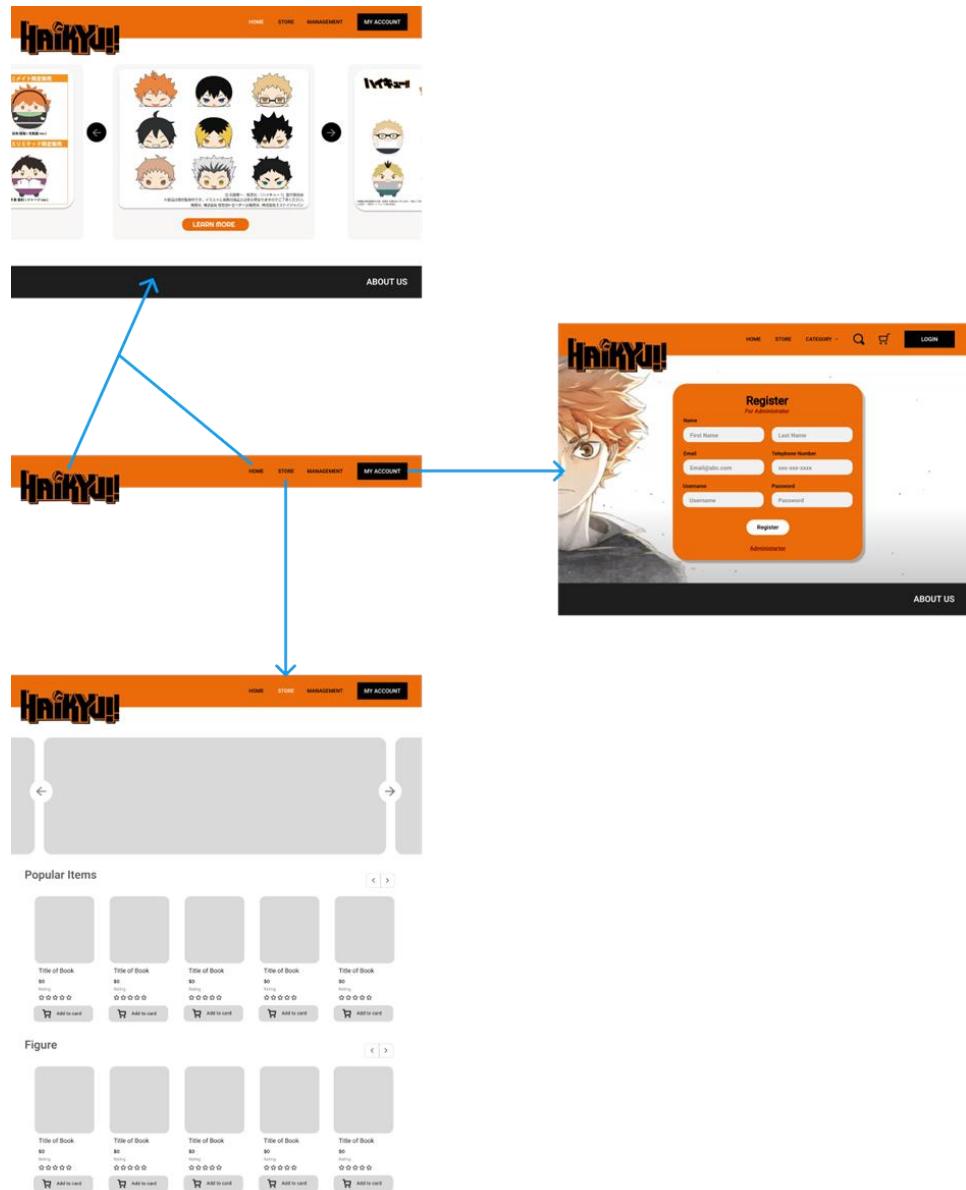
ສ່ວນແຮກຄື່ວ Navigation bar ແບນທີ1 ເປັນແບນສໍາຫຼັບຜູ້ໃຊ້ຫຼືລູກຄ້າ

ເພື່ອໃຫ້ລູກຄ້າສາມາດເລືອກອີກໄປຢັງໜ້າຕ່າງໆຂອງເວີ້ປີ່ເຊື່ອໄດ້ ໂດຍຄໍາວ່າ ປຸ່ມHome ຈະເຊື່ອມໄປຢັງໜ້າ Home Page, Store ໄປຢັງໜ້າຂອງສິນຄ້າ(Product), ປຸ່ມCategory

ສາມາດເລືອກໄດ້ວ່າຈະໄປຢັງປະເທດຂອງສິນຄ້ານີ້ໃດ, ປຸ່ມSearch ເປັນຮູບແວ່ນຂໍາຍາຍເຂື່ອມກັບໜ້າSearch,

ປຸ່ມShopping cart ເປັນໜ້າເຂື່ອມໄປສໍາຫຼັບໜ້າຕະກຳສິນຄ້າ, Login button

ເປັນປຸ່ມເຂື່ອມໄປຢັງໜ້າລົງຈຶ່ນເຂົ້າໃຈໆສໍາຫຼັບລູກຄ້າ



## Navigation bar แบบที่2 เป็นแบบสำหรับพนักงานหรือ Admin

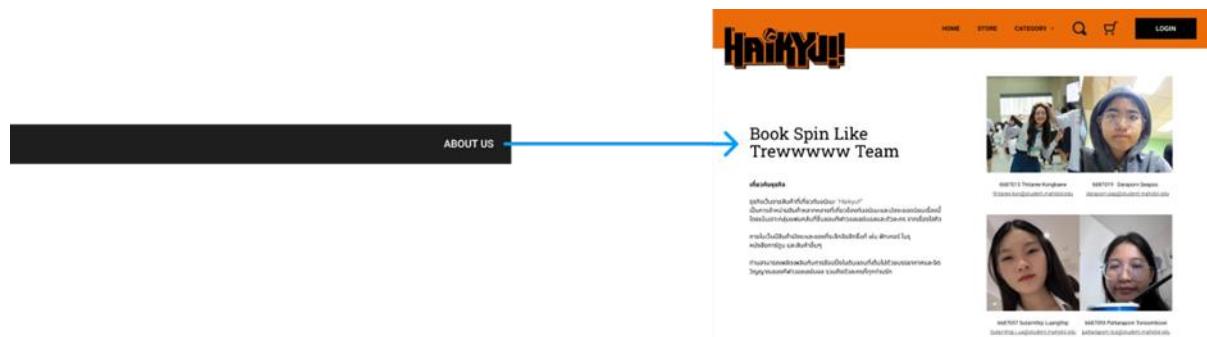
เพื่อให้พนักงานสามารถเข้าใช้เว็บไซต์เพื่อทำการแก้ไขหรือดูสินค้าภายในเว็บไซต์ โดยมี ปุ่ม Home

จะเชื่อมไปยังหน้า Home Page, Store ไปยังหน้าที่สามารถแก้ไขสินค้า(Product), ปุ่ม Management

เชื่อมไปยังหน้าที่จัดการข้อมูลของพนักงาน, และปุ่ม My account

เพื่อเชื่อมไปยังหน้าลงทะเบียนของพนักงานหรือAdmin

## ส่วนท้าย (Footer)

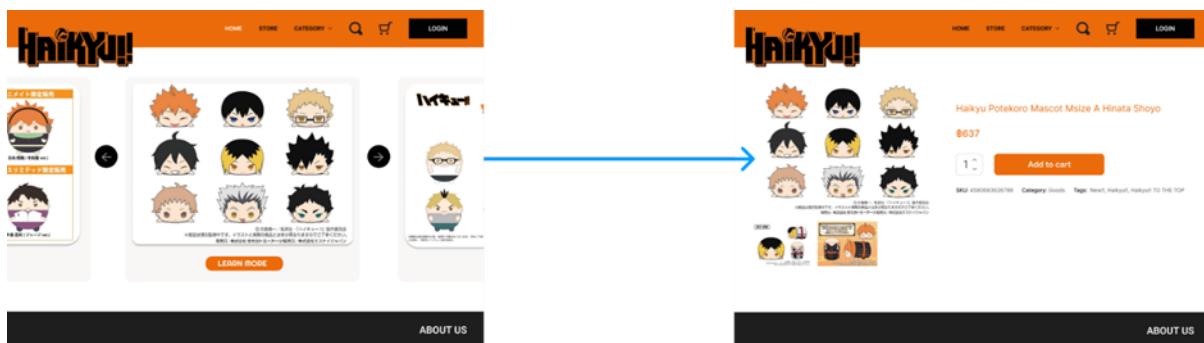


ส่วนล่างสุดของหน้าเว็บไซต์คือ footer โดยประกอบไปด้วยปุ่ม ABOUT US ที่จะเชื่อมไปยัง Team Page ซึ่งมีข้อมูลและโซเชียลมีเดียของผู้จัดทำ

## Wireframe

### 1. Home Page

ในส่วนของหน้า Home Page ส่วนข้างบนสุดทางขวาจะมี Logo ของสินค้าที่สามารถกดเพื่อกลับมาหน้า Home Page ได้และมี Navigation Bar เพื่อเป็น Tab สามารถกดไปหน้าอื่นๆได้ ส่วนตรงกลางเป็นตัวแสดงสินค้ามาใหม่ที่สามารถกดลูกราชายขาวเพื่อเลื่อนดูสินค้าได้ และสามารถกดเลือกได้ว่าจะดูสินค้าเพิ่มเติม หรือ ซื้อ เพื่อเข้าสู่หน้าเลือกซื้อได้เลย และส่วนสุดท้ายคือ footer มี about us ที่เป็นหน้าที่มีข้อมูลของผู้จัดทำ

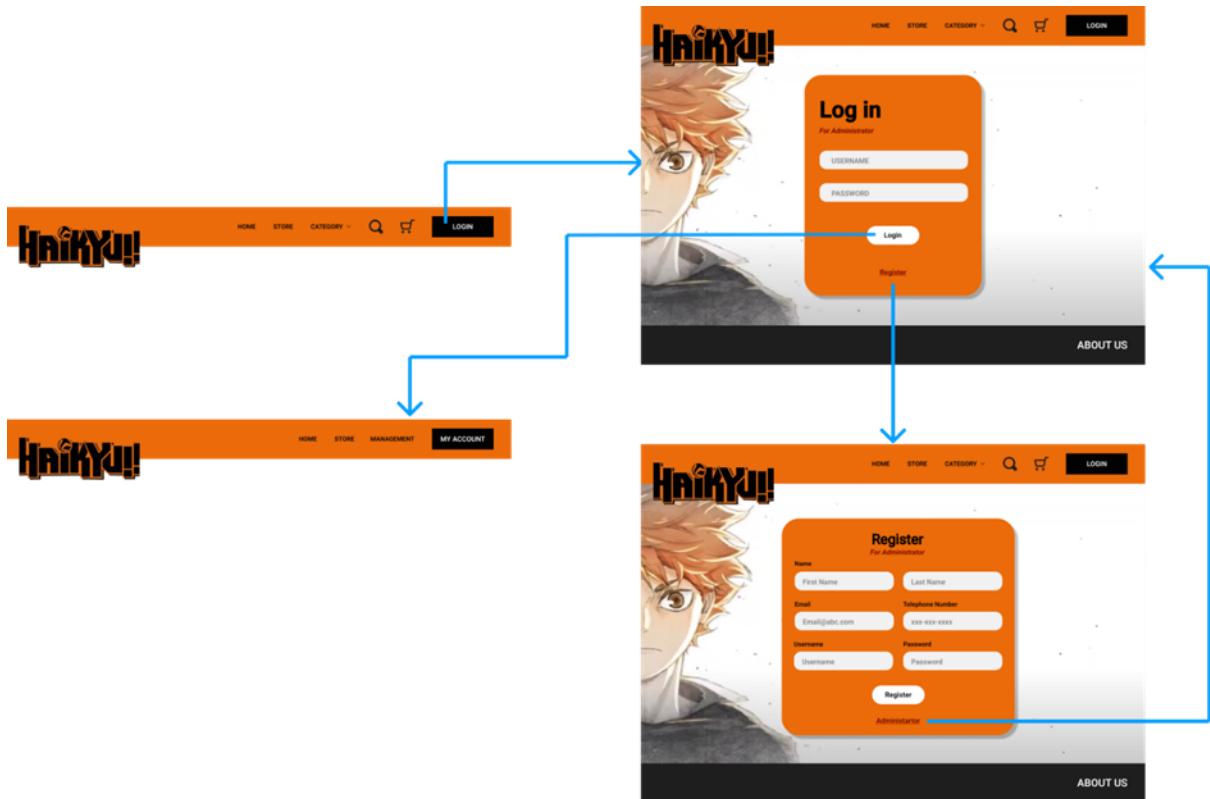


### 2. Login Page

ในส่วนของ Login Page จะเป็นหน้าซึ่งมีไว้เพื่อให้ Administrator ทำการล็อกอินเพื่อสามารถเข้าไปจัดการส่วนหลังบ้านบนเว็บไซต์ได้ ซึ่งหน้า Login จะมี 2 รูปแบบคือ สำหรับล็อกอิน และสำหรับลงทะเบียน ซึ่งในแต่ละหน้าจะประกอบไปด้วย nav bar, footer (ที่สามารถเชื่อมไปหน้าอื่นได้ตั้งที่กล่าวไว้ข้างต้น) และกล่องเพื่อกรอกแบบฟอร์ม

หน้าสำหรับการล็อกอิน ซึ่งจะเป็นหน้าหลักที่แสดงหลังจากผู้เข้าใช้กดปุ่ม Login ตรง nav ผู้ใช้จำเป็นต้องกรอกทั้ง username และ password ส่วนหน้าสำหรับการลงทะเบียนจะมีข้อมูลส่วนตัวของผู้ใช้ที่จำเป็นต้องกรอกทุกช่อง ซึ่งผู้ที่สามารถกรอกแบบฟอร์มดังกล่าวจะเป็นได้แค่ Administrator เท่านั้น

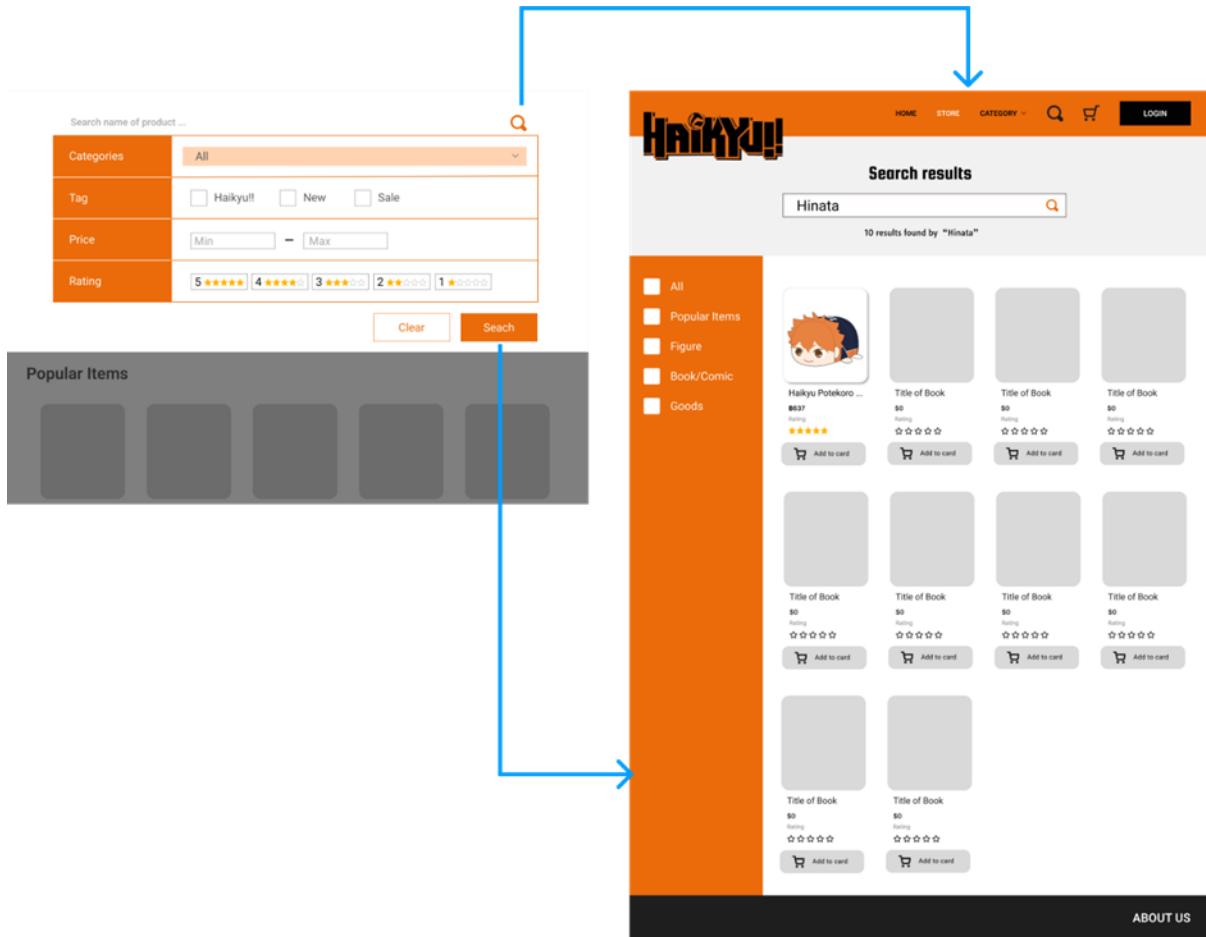
หลังจากการล็อกอินเสร็จสิ้น ผู้ใช้จะไปยังหน้า Management โดยที่ nav bar จะเปลี่ยนเป็นสำหรับ Administrator



### 3. Search Page

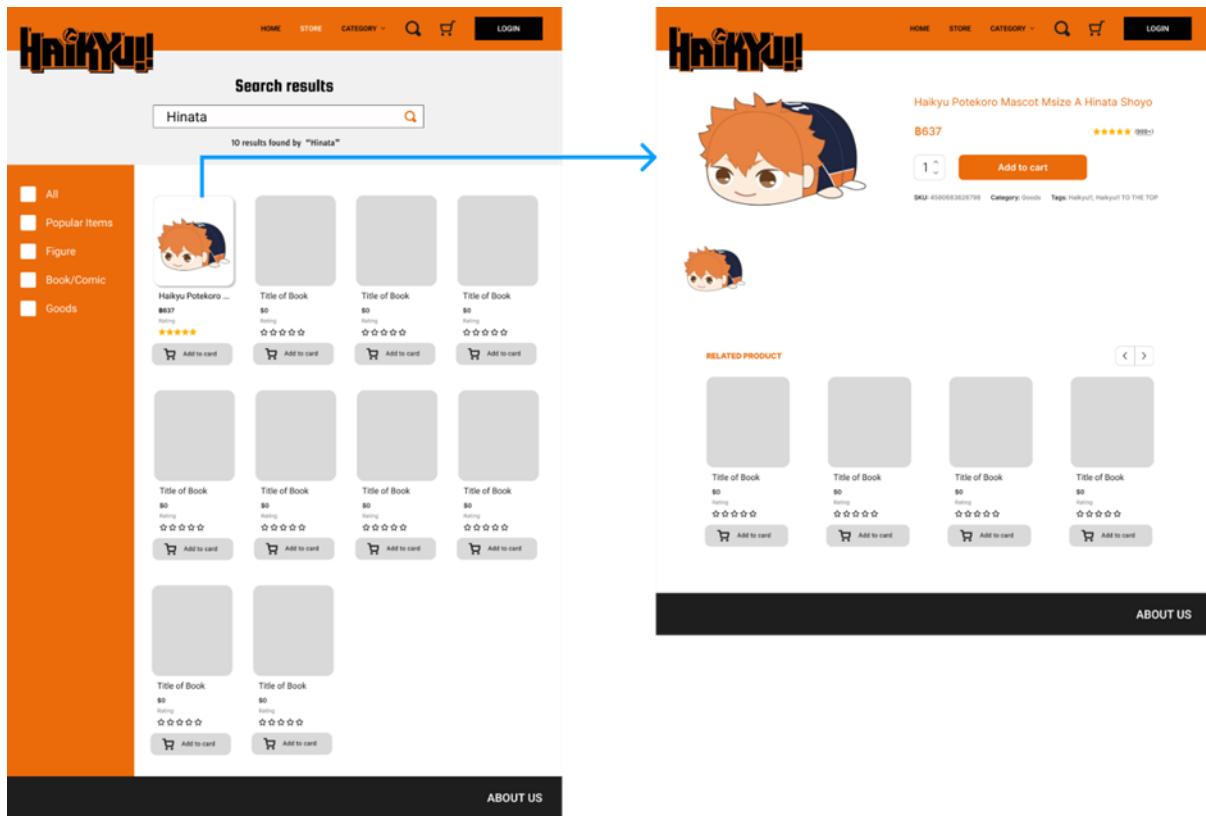
ในส่วนของหน้า Search Page

จะมีไว้ให้ผู้ใช้ได้สามารถค้นหาสินค้าที่สนใจโดยพิมพ์คำที่ต้องการจะค้นหาหรือสามารถเลือกตัวเลือกอื่น ๆ ที่ลูกค้าสนใจ โดยหน้า Page จะมี nav bar, footer (ที่สามารถเชื่อมไปหน้าอื่นได้ดังที่กล่าวไว้ข้างต้น), มีช่องค้นหา (Search name of product) ให้ผู้ใช้พิมพ์คำที่ต้องการจะค้นหา, มี Categories ซึ่งเป็น Form Element แบบ Drop-down list สามารถเลือกหมวดที่ต้องการจะค้นหาได้ เช่น Book/Comic, Goods และ Figure เป็นต้น, Tag ซึ่งเป็น Form Element แบบ Checkbox สามารถเลือก Tag ที่ต้องการจะค้นหาได้, Price ผู้ใช้สามารถใส่ช่วงราคาที่ลูกค้าต้องการได้ โดยใส่ราคาที่ต่ำสุดและสูงสุด, Rating ลูกค้าสามารถกดปุ่มเลือกคะแนนของสินค้านั้น ๆ ได้ และเมื่อเสร็จสิ้นการค้นหาสามารถกดปุ่ม Search ข้างล่างได้ หากอยากระบบให้กดปุ่ม Clear เพื่อล้างข้อมูลทั้งหมด



### 3.1 Result of search

ในส่วนของ Result of search จะแสดงผลลัพธ์ของการ Search โดยหน้า Page จะมี nav bar, footer (ที่สามารถเขื่อมไปหน้าอื่นได้ดังที่กล่าวไว้ข้างต้น), มีประวัติคำที่ผู้ใช้ค้นหา รวมถึงจำนวนผลลัพธ์ของสินค้าทั้งหมด, มีการแสดง Product ที่เกี่ยวข้องกับคำที่ค้นหา, มี Categories อยู่ทางด้านซ้าย ซึ่งเป็น Form Element แบบ Checkbox สามารถเลือกหมวดที่ต้องการจะค้นหาได้ เช่น Book/Comic, Goods และ Figure เป็นต้น ซึ่งจะทำให้แสดงผลการค้นหาที่ละเอียดมากขึ้น และแต่ละสินค้าที่ค้นหาได้จะสามารถคลิกเพื่อไปดูรายละเอียดของสินค้านั้นๆ ที่หน้า Detail page ได้



#### 4. Detail Page

หน้าสำหรับแสดงรายละเอียดของสินค้า ซึ่งหน้านี้จะประกอบไปด้วย nav bar และ footer สำหรับเชื่อมไปยังหน้าอื่นๆ

โดยรายละเอียดของสินค้าจะประกอบไปด้วย ชื่อสินค้า, ราคา, หมวดหมู่(category), Tags, Stock Keeping Unit(SKU), ปริมาณที่ต้องการซื้อ ซึ่งสามารถเพิ่มหรือลดจำนวนได้ และคะแนนเรตติ้ง หรือจำนวนดาวของสินค้าที่ได้รับจากผู้ที่เคยซื้อสินค้า

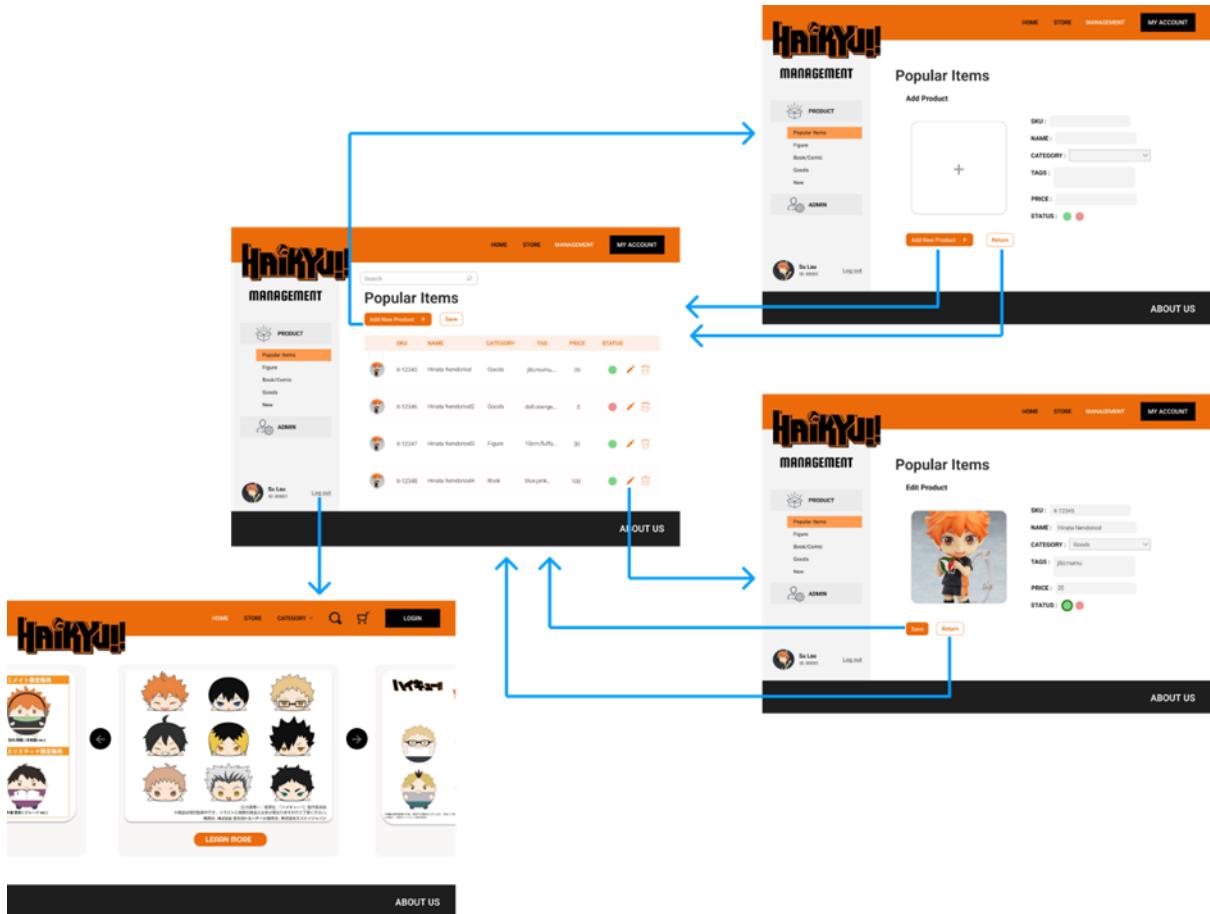
สุดท้ายคือ Related product หรือสินค้าที่เกี่ยวข้องที่จะดึงมาจากร้านค้า(Store) ที่ใช้แท็กหรือหมวดหมู่เดียวกันกับสินค้าปัจจุบัน มาโชว์ให้ลูกค้าได้เลือกดูเป็นแบบด้านล่างสินค้าปัจจุบัน

## 5. Product Management Page

ในส่วนของหน้า Product Management Page จะเป็นหน้าของการจัดการสินค้าในเว็บไซต์ ซึ่งสามารถเพิ่ม ลบ และแก้ไขข้อมูลสินค้าได้ สินค้าแต่ละตัวจะประกอบด้วย SKU(หมายเลขสินค้า), name(ชื่อสินค้า), category(ประเภทสินค้า), tag(คีย์เวิร์ดของสินค้า), price(ราคาสินค้า), status(สถานะของสินค้า ซึ่งจะแบ่งเป็น 2 อย่าง สีเขียว(available) และ สีแดง(not available))

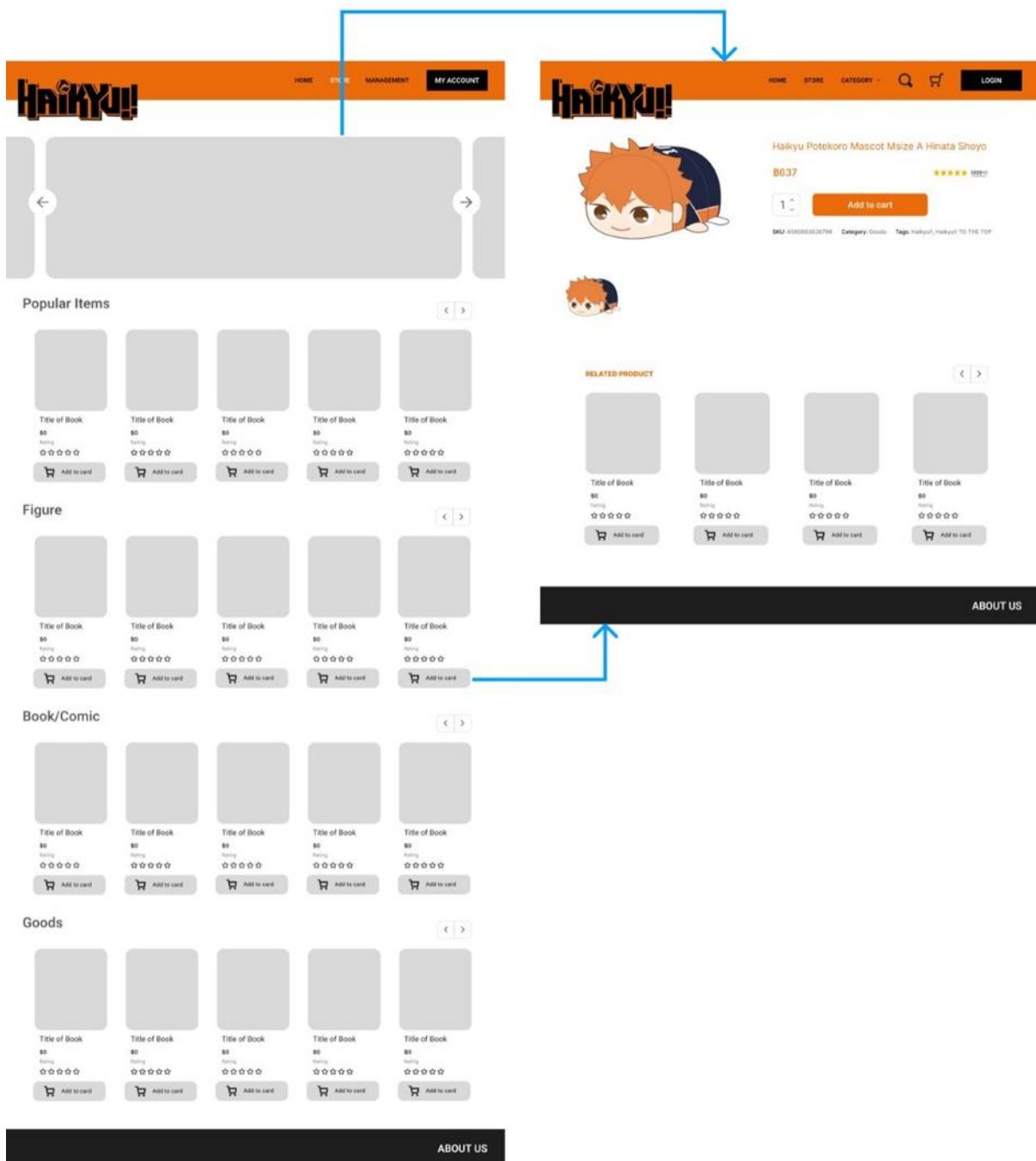
หน้า Product Management Page จะประกอบไปด้วย nav bar, footer, พาร์ทการจัดการสินค้า และ nav bar สำหรับ management ซึ่งแบบของ nav bar นี้จะสามารถกดเลือกสิ่งที่ต้องการแก้ได้อย่างเช่นในตัวอย่างคือแก้ Popular Items

เมื่อกด Add Product หรือแก้ไขข้อมูลโดยการกดไอคอนดินสอ  
เว็บจะเชื่อมไปยังหน้าที่สามารถกรอกฟอร์มข้อมูลของสินค้าได้ และเมื่อล็อกเอาท์จากการเป็นแอดมิน nav bar ข้างบนจะเปลี่ยนเป็นสำหรับ customer และเชื่อมไปยังหน้า homepage



### 5.1. Update product

ในส่วนหน้า Update product คือหน้าสำหรับองรับการอัปเดตข้อมูลของสินค้า จะเป็นการแสดงสินค้าทั้งหมด โดย Page นี้จะมี nav bar และ footer (ที่สามารถเชื่อมไปหน้าอื่นได้ดังที่กล่าวไว้ข้างต้น), ส่วนตรงกลางเป็นตัวแสดงสินค้าใหม่, สินค้าที่มี Promotion และสินค้าที่น่าสนใจและสามารถกดลูกศรซ้าย-ขวาเพื่อเลื่อนดูสินค้าได้ ข้างล่างจะเป็นสินค้าตามแต่ละประเภทสามารถกดลูกศรซ้าย-ขวาเพื่อเลื่อนดูสินค้าได้



## 6. User Account Management

ในส่วนของหน้า User Account Management เป็นหน้าเพจที่สามารถจัดการแอดมินได้โดยจะมี nav bar สำหรับ admin และ footer ซึ่งจะสามารถเข้ามายังหน้าอื่นๆได้ และเมื่อกดปุ่ม Log out จะกลับไปยัง Home page

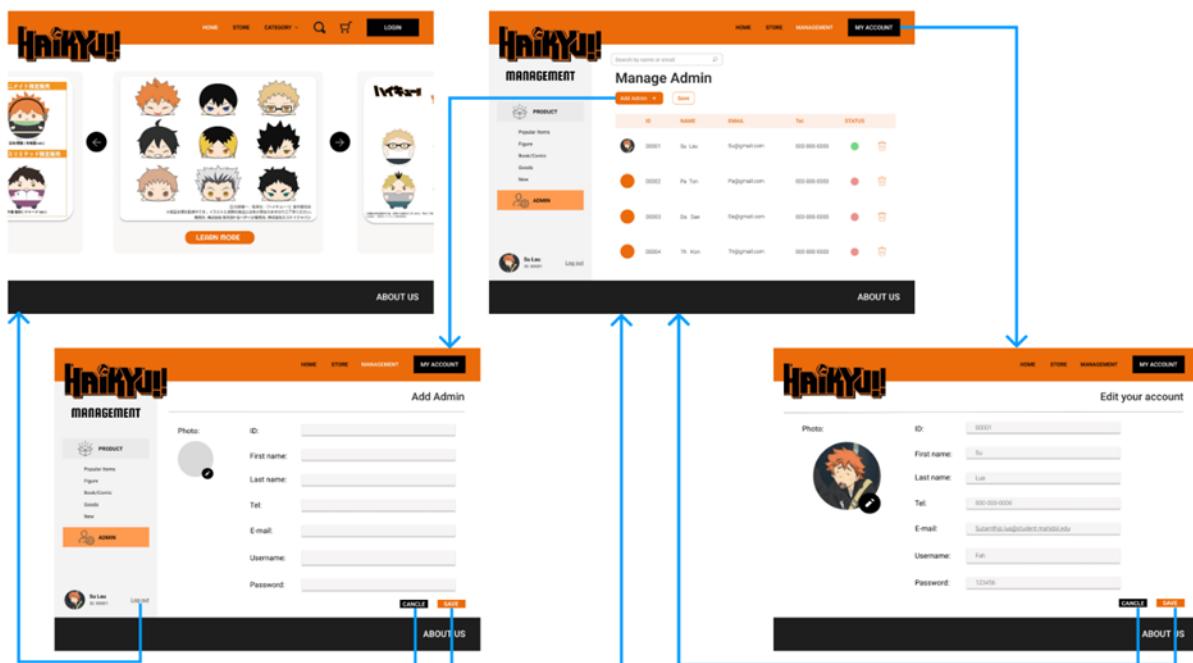
ปุ่ม Add admin จะสามารถเพิ่มแอดมินคนใหม่ได้ เมื่อกดแล้วจะไปยัง Add admin page โดยจะต้องกรอกข้อมูลเพื่อเชื่อมกับ account ของแอดมินคนใหม่ที่ทำการ register account เสร็จแล้ว กล่าวคือแอดมินคนใหม่จะต้องมีข้อมูล account อยู่ในระบบก่อน โดยข้อมูลที่กรอกจะประกอบไปด้วย ID,

Username, Password ข้อมูลไอเดียที่กรอกที่จะเป็นการตั้งไอเดียนี้ให้กับแอดมินคนใหม่ ส่วน username และ password จะเป็นการกรอกเพื่อลิ้งก์กับ account ที่มีชื่อผู้ใช้และรหัสผ่านตรงกับกันที่กรอก เมื่อกด save จะเพิ่มแอดมินคนใหม่โดยมี ชื่อจริง, นามสกุล, เบอร์โทร และอีเมล ของแอดมินคนดังกล่าวขึ้นไว้ในหน้า Manage Admin

การแก้ไขข้อมูลแอดมินจะสามารถแก้ไขได้ในหน้า Account ของแอดมินแต่ละคน โดยกดปุ่ม My account บน Nav bar จะเป็นการเข้าสู่หน้า Edit your account เจ้าของ account นั้นๆจะสามารถแก้ไขข้อมูลส่วนบุคคลได้ และเมื่อกด save ข้อมูลส่วนบุคคลที่ใช้ว้อยู่ในหน้า Manage Admin จะเปลี่ยนไปตามการแก้ไขของแอดมินคนนั้นๆ

ແກบ Status ในหน้า Mange Admin สีเขียวแสดงถึงการ online ของแอดมินคนนั้นๆ และสีแดงแสดงถึงการ offline ของแอดมินคนนั้นๆ

ปุ่มถังขยะ สามารถลบแอดมินคนนั้นๆได้



## 7. Team Page

ในส่วนของหน้าทีมด้านบนจะมี nav bar สำหรับ admin และ footer ซึ่งจะสามารถเชื่อมไปยังหน้าอื่นๆได้ มีชื่อทีมและคำอธิบายเกี่ยวกับธุรกิจทางด้านซ้าย ส่วนทางด้านขวาจะมีชื่อ และลิงก์ที่เชื่อมไปยังอีเมลของผู้จัดทำ



HOME STORE CATEGORY LOGIN

## Book Spin Like Trewwww Team

### เกี่ยวกับธุรกิจ

ธุรกิจเบ็บขายสินค้าที่เกี่ยวกับอินเม: "Haikyuu!!"  
เป็นการจำลองการขายสินค้าหลักหลายที่เกี่ยวข้องกับน้องมีเบะและบังงะยอดนิยมเรื่องนี้  
โดยเน้นเจาะกลุ่มแฟบคลับที่ชื่นชอบกีฬาวอลเลย์บอลและดูวอลเลย์บอล

ภายในเว็บมีสินค้าบังงะและของที่ระลึกลิขสิทธิ์แท้ เช่น พีกเกอร์ โนรุ  
หนังสือการ์ตูน และสินค้าอื่นๆ

ดำเนินการโดยเพลิดเพลินกับการซื้อปั๊บในอันเดนที่เต็มไปด้วยบรรยากาศและจิต  
วิญญาณของกีฬาวอลเลย์บอล รวมถึงตัวละครที่ทุกคนรัก



6687015 Thitaree Kongkaew  
[thitaree.kon@student.mahidol.edu](mailto:thitaree.kon@student.mahidol.edu)

6687019 Daraporn Seapoo  
[daraporn.sap@student.mahidol.edu](mailto:daraporn.sap@student.mahidol.edu)



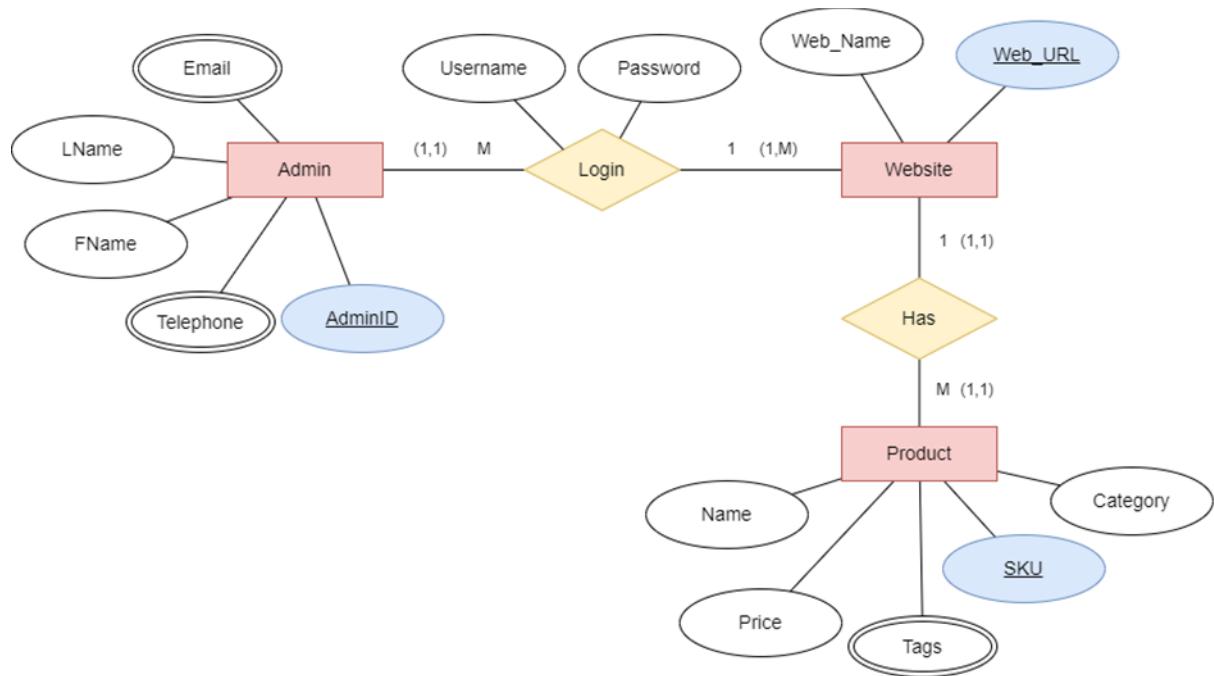
6687057 Sutarnthip Luangthip  
[Sutarnthip.Lua@student.mahidol.edu](mailto:Sutarnthip.Lua@student.mahidol.edu)

6687093 Pattraporn Tonsomboon

[pattraporn.tos@student.mahidol.edu](mailto:pattraporn.tos@student.mahidol.edu)

## Data Model

### 1.ERD (Chen)



## 2.Data Dictionary

Table name	Attribute Name	Contents	Data Type	Format	Nullable	Key	FK Referenced Table
Admin	AdminID	รหัสระบุตัวตนของพนักงาน	int	Xxxxxx		PK	
	Fname	ชื่อของพนักงาน	varchar(50)	Xxxxxxx			
	Lname	นามสกุลของพนักงาน	varchar(50)	Xxxxxx			
	Web_URL	ลิงค์เว็บไซต์	varchar(70)	Xxxxxxx		FK	Website
Website	Web_URL	ลิงค์เว็บไซต์	varchar(70)	Xxxxxxx		PK	
	Web_name	Promotion's name	varchar(50)	Xxxxxxx			
Product	SKU	รหัสหมายเลขบุสินค้า	varchar(20)	Xxxxxx		PK	
	Pname	ชื่อสินค้า	varchar(50)	Xxxxxxx			
	Price	ราคาสินค้า	varchar(50)	Xxxxxxx			
	Category	ประเภทสินค้า	varchar(50)	Xxxxxxx			

	Web_URL	ลิงค์เว็บไซต์	varchar(70)	Xxxxxxx		FK	Website
Email	Email	อีเมลของพนักงาน	Admin	Xxxxxxx		PK	
	AdminID	รหัสระบุตัวตนของพนักงาน	int	Xxxxxxx		FK,P K	Admin
Telephone	Tel	หมายเลขโทรศัพท์ของพนักงาน	int	xxx-xxx-xxxx		PK	
	AdminID	รหัสระบุตัวตนของพนักงาน	int	Xxxxxxx		FK,P K	Admin
Tags	Tag	ป้ายแท็กสินค้า	varchar(70)	Xxxxxxx		PK	
	SKU	รหัสหมายเลขระบุสินค้า	varchar(20)	Xxxxxxx		FK,P K	Product
Log in	AdminID	รหัสระบุตัวตนของพนักงาน	int	Xxxxxxx		FK,P K	Admin
	Web_URL	ลิงค์เว็บไซต์	varchar(70)	Xxxxxxxxx		FK,P K	Website
	Username	ชื่อที่เข้าใช้ล็อกอิน	varchar(20)	Xxxxxxxxx			
	Password	รหัสที่เข้าใช้ล็อกอิน	varchar(20)	Xxxxxxxxx			

## Phase2

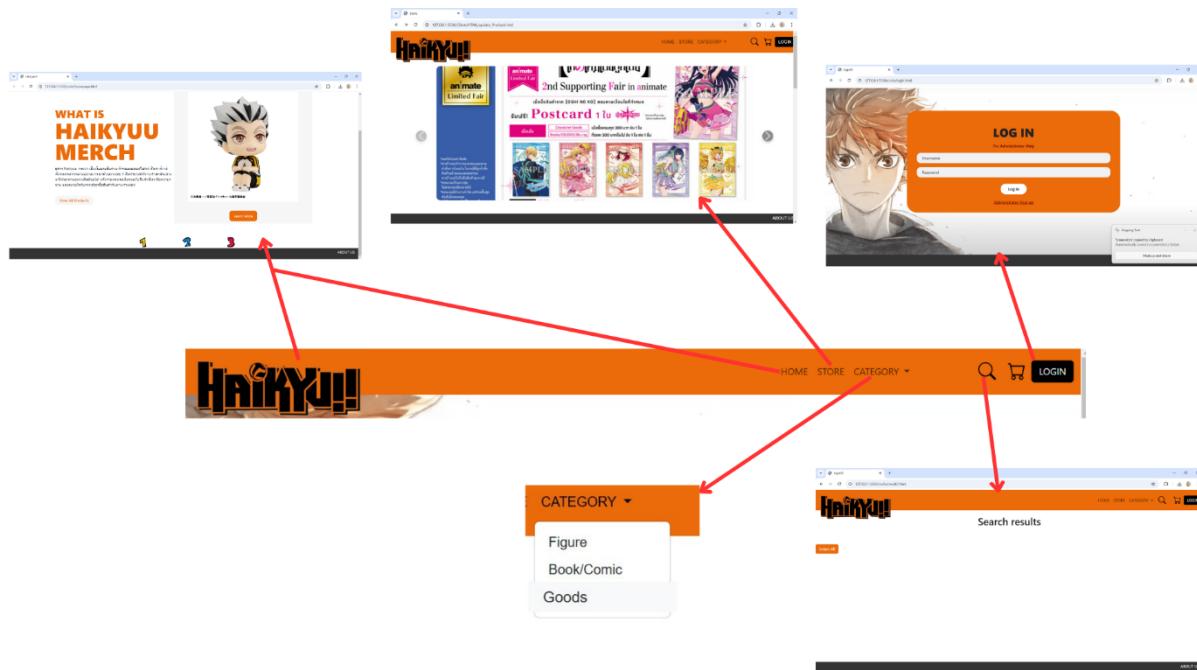
### ข้อมูลองค์รวมของโครงการ

#### คำอธิบายธุรกิจ

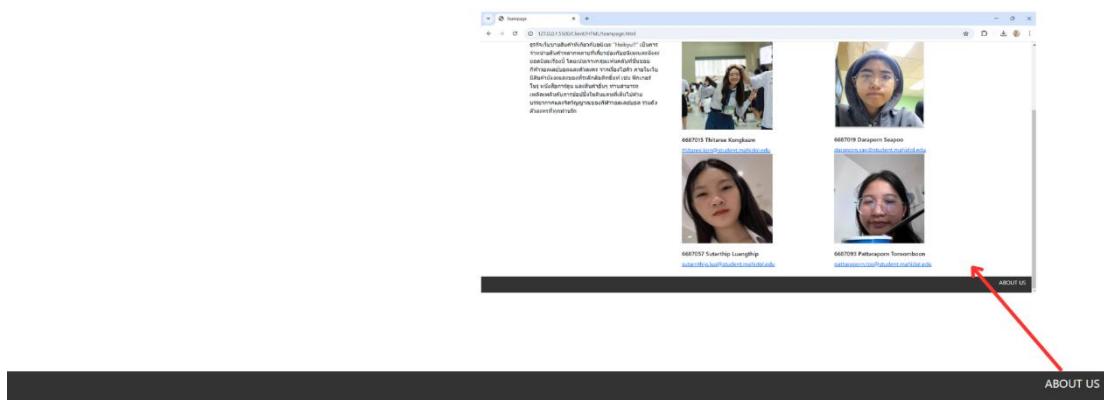
บริษัทที่จำหน่ายสินค้าเกี่ยวกับอนิเมะ "Haikyuu!!" ก่อตั้งขึ้นจากความนิยมของอนิเมะและมังงะเรื่องนี้ ในกลุ่มแฟนคลับทั่วโลก โดยเริ่มต้นจากการจำหน่ายสินค้าเล็ก ๆ น้อย ๆ เพื่อให้แฟนคลับสามารถสะสมของที่ระลึกที่เกี่ยวข้องกับตัวละครและกีฬาวอลเลย์บอลที่เป็นเอกลักษณ์ของเรื่อง ต่อมา ทางเราได้นำธุรกิจนี้มา ทางอิงทำเป็นเว็บไซต์ออนไลน์ที่จำหน่ายสินค้าหลากหลายประเภท สินค้าที่มีจำหน่ายได้แก่ พิกเกอร์ โนรุ หนังสือการ์ตูน มังงะ และสินค้าลิขสิทธิ์อื่น ๆ ซึ่งได้รับความนิยมอย่างมาก รูปแบบและตัวอย่างของเว็บไซต์ ของธุรกิจซื้อขายสินค้าให้คิวนี้อ้างอิงมาจากเว็บไซต์ของร้านออนไลน์ นอกจากนี้ขอบเขตของธุรกิจยังครอบคลุม การขายสินค้าทั่วประเทศผ่านช่องทางออนไลน์

## แผนผังหน้าต่างๆของเว็บแอปพลิเคชัน

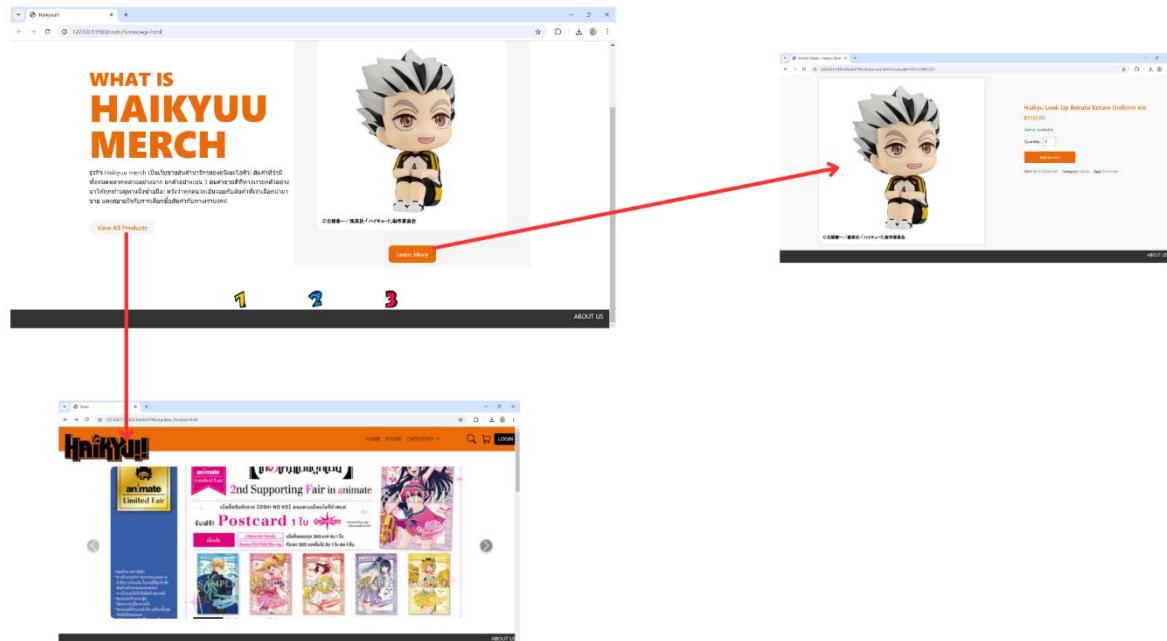
### 1. Navigation Bar



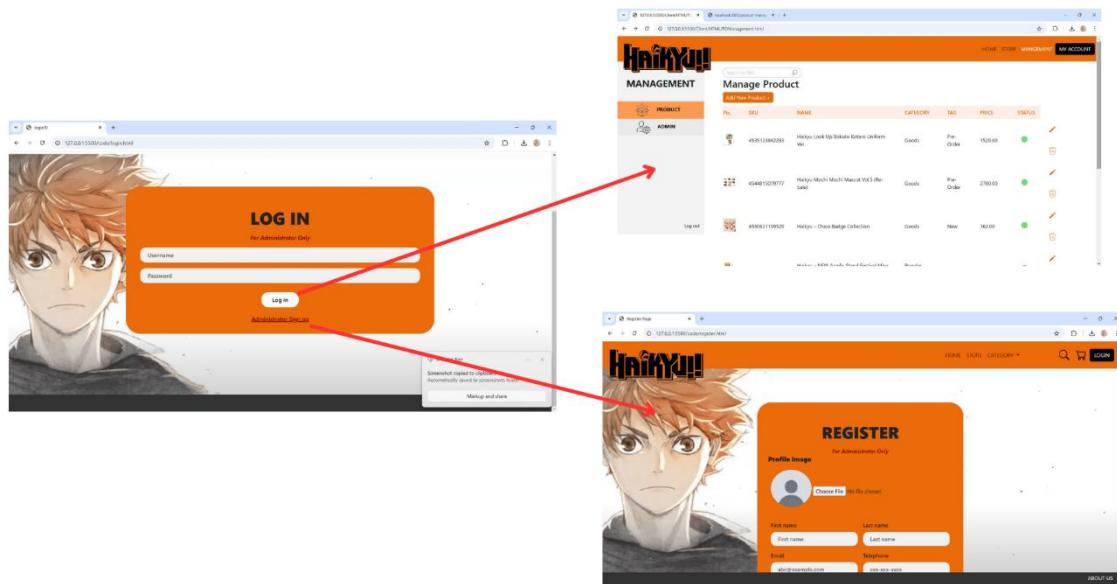
### 2. Footer



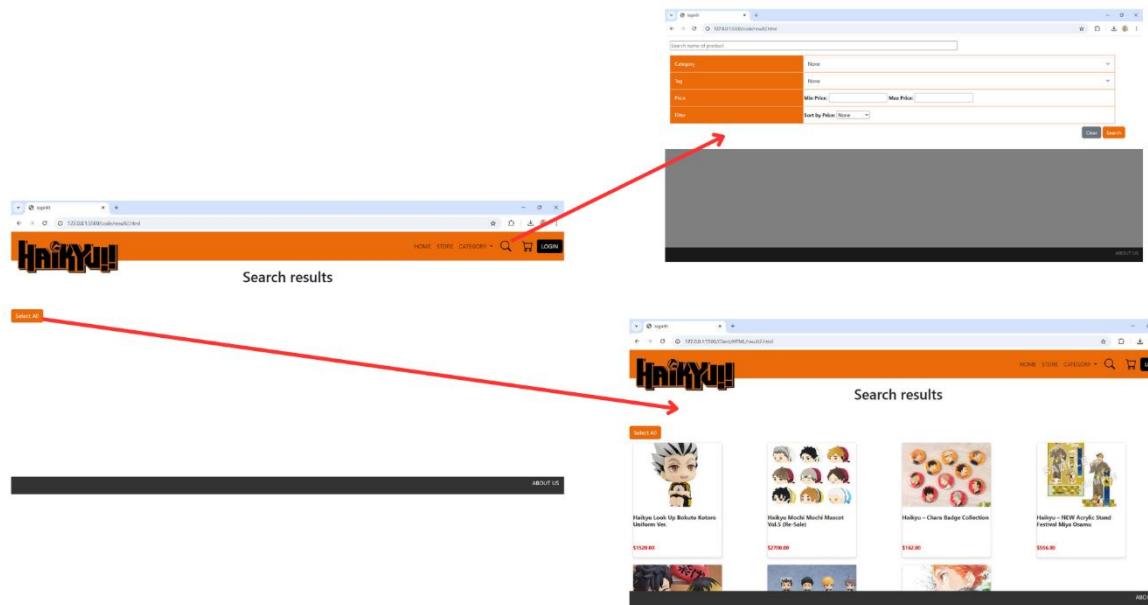
### 3.Home page



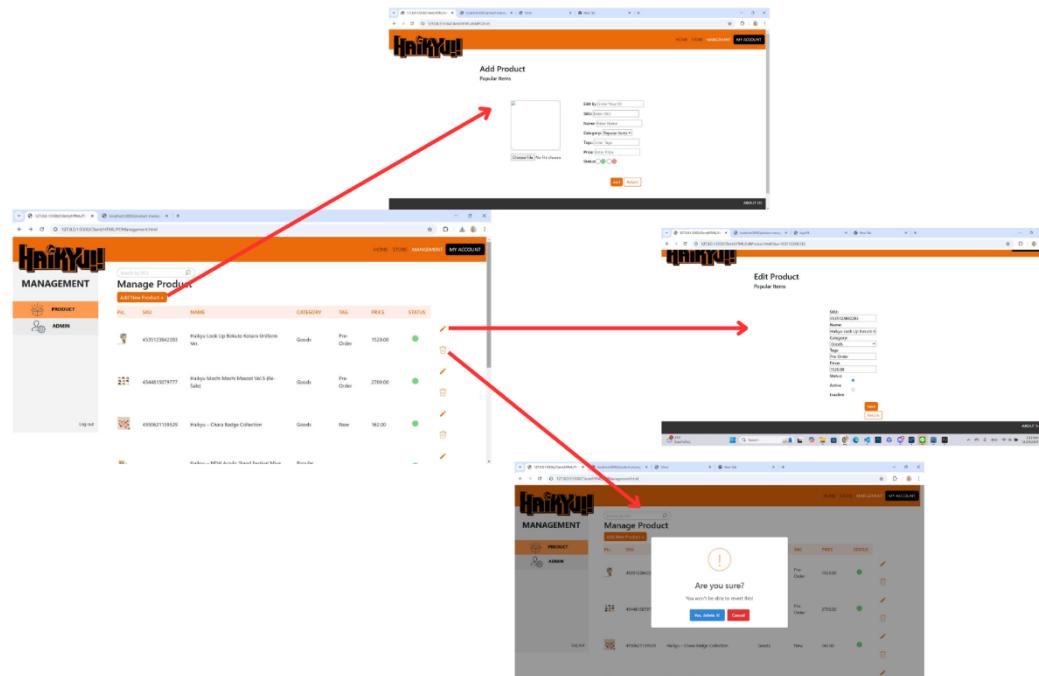
### 4.Log in



## 5.Search



## 6.Product Management



## 7.Admin Management

The image shows four screenshots of a web application interface for managing administrators:

- Add Admin:** A form with fields for First Name, Last Name, Tel, Email, Username, and Password. A "Save" button is at the bottom.
- Manage Admin:** A table listing two administrators:

Pic	ID	Name	Email	Tel
	34	sutarmthip.lua	sutarmthip@student.mahidol.edu	null
	35	Sutarmthip Mif	sutarmthip@gmail.com	092-458-0521

A red arrow points from this page down to the "Edit Account" page.
- Edit Account:** A form with fields for First Name, Last Name, Tel, Email, Username, and Password. A "Save" button is at the bottom.
- Confirmation Dialog:** A modal window asking "Are you sure? You won't be able to revert this." with "Yes, delete" and "Cancel" buttons.

## 8.Team Page

The image shows a team page with the following details:

- Section Header:** Book Spin Like Trewwwww Team  
เพ็บากนงรักฯ
- Text:** ดูทีมงานของคุณครับ/คุณครูที่เกี่ยวกับเว็บไซต์ "Haikyuu!!" นี่ครับ ข้างบนคือหน้าตาของพัฒนาที่เกี่ยวข้องกับเว็บไซต์และเว็บ แบบเชิงโครงสร้าง ใช้ออกแบบฐานข้อมูลที่ดีที่สุดของ ก็ตัวเราเองและมีความรวดเร็วในการอ่านเข้าสู่ภาษาในบันทึกนี้ มีลักษณะเชิงแบบและลึกซึ้ง รวมถึงความสามารถในการ อ่านและเขียนภาษาอังกฤษได้ดีมาก ทีมงานนี้ ทั้งหมดมีความรู้และทักษะที่ดี สามารถทำงานเป็นทีมได้อย่างมีประสิทธิภาพ และมีความตั้งใจในการพัฒนาเว็บไซต์ให้มีคุณภาพ สวยงามและใช้งานง่ายและรองรับการอ่านแบบมือถือ รวมถึง ผู้ชมที่หลากหลายทั่วโลก
- Members:**
  - Thitaree Kongkeaw**  
thitaree.kon@student.mahidol.edu
  - Daraporn Seapoo**  
daraporn.sap@student.mahidol.edu
  - Sutarmthip Luangthip**  
sutarmthip.lua@student.mahidol.edu
  - Pattaraporn Tonsomboon**  
pattaraporn.tos@student.mahidol.edu
- Navigation:** HOME, STORE, CATEGORY, LOGIN, ABOUT US

## รายละเอียดหน้าเว็บต่างๆของผู้ดูแล

### 1.Nav bar/Footer



#### 1.1 Navigation bar

ในส่วนของ Nav bar นั้นจะมีโลโก้ของแบรนด์ที่ และ ปุ่มHome ที่สามารถกลับไปยังหน้าHome Page ปุ่มstore ที่สามารถไปหน้าของสินค้าทั้งหมด มี แทปcategory ที่สามารถเลือกประเภทของสินค้า เพื่อจำแนกการค้นหาสินค้า ปุ่มรูปแฉ่งสามารถไปยังหน้าsearch เพื่อมีการค้นหาแบบขั้นตอนซึ่งได้ และสุดท้ายคือปุ่มlogin ที่สามารถไปหน้าลงทะเบียนได้



#### 1.2 Footer

ในส่วนของFooter จะมีปุ่มabout us เพื่อไปยังหน้าteam page ได้

### 2.Homepage

ตัว content แบ่งเป็น 3 พาร์ทหลักคือ

### 1.พาร์ท/oibayธุรกิจฝ่ายมือ

ประกอบด้วยเนื้อหาของเว็บธุรกิจและปุ่ม View all Product ซึ่งสามารถเชื่อมไปยังหน้า Product ได้

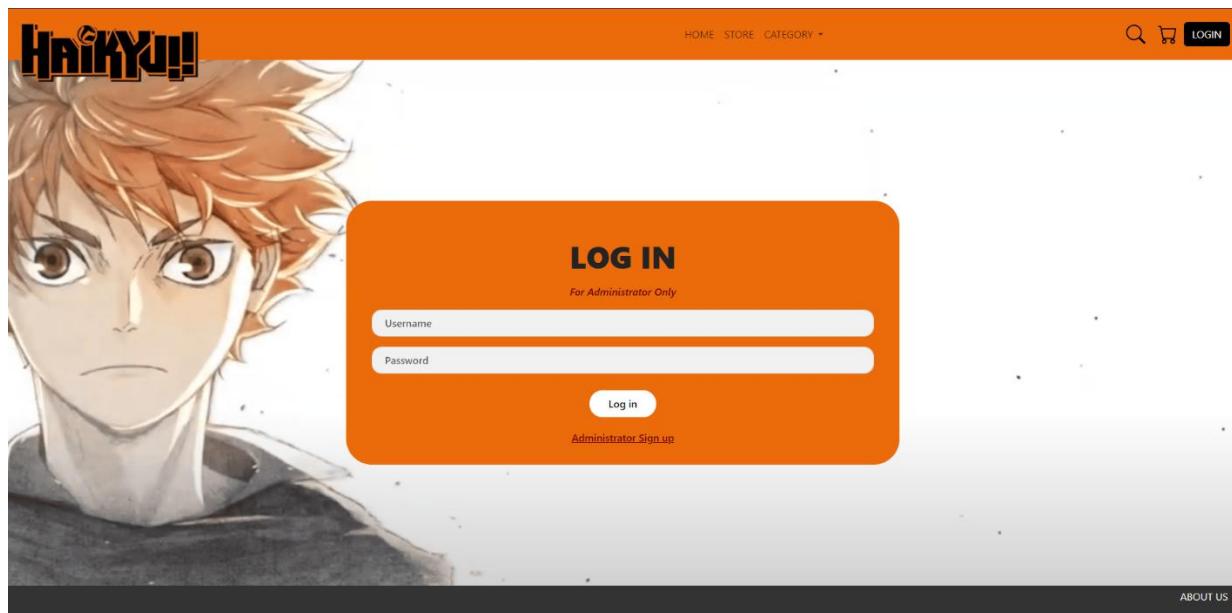
### 2.พาร์ทรูปสินค้าฝ่ายมือ

ประกอบด้วยการดูรูปภาพสินค้าและปุ่ม Learn more ซึ่งสามารถเชื่อมไปหน้ารายละเอียดสินค้านั้นๆได้

### 3.พาร์ทลำดับเลขข้างล่าง

เลขดังที่แสดงเมื่อกดเลขนั้นๆ สินค้าทางด้านฝ่ายขวาจะแสดงเปลี่ยนไปตามตัวเลข

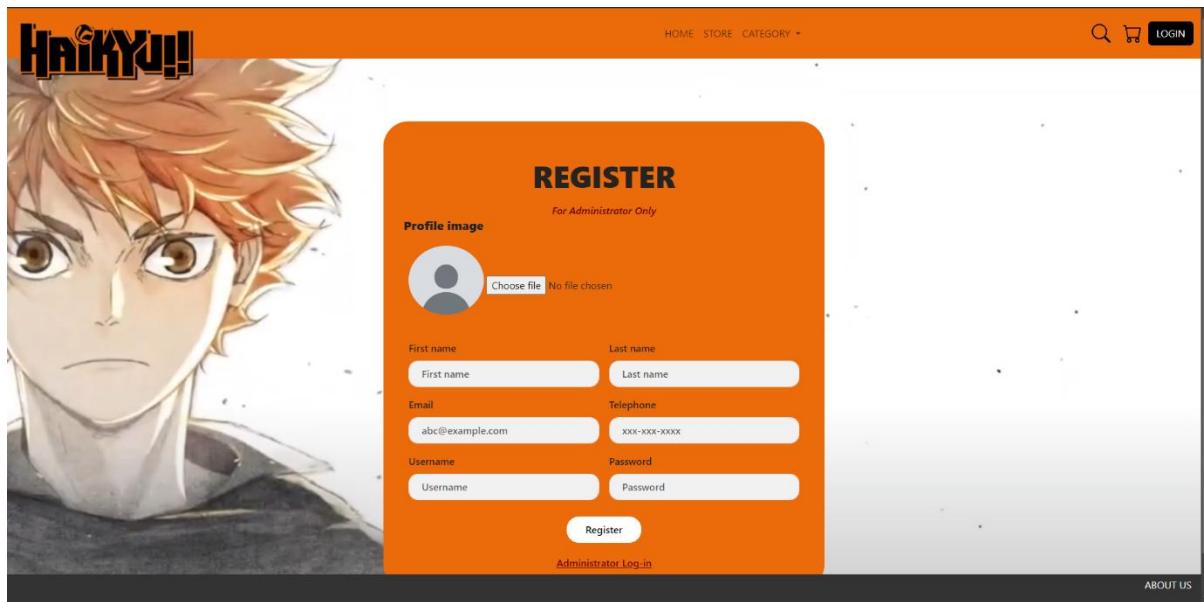
## 2.Login Page



ในส่วนของ content จะมีกล่องฟอร์ม 1 กล่อง ไว้สำหรับใส่รหัส Username และ Password ของแอดมิน  
เท่านั้นให้ถูกต้องถึงจะสามารถล็อกอินไปยังหน้า management ได้ หากทั้ง Username และ Password ไม่  
ถูกต้อง จะไม่สามารถเข้าไปที่หน้าแอดมินได้

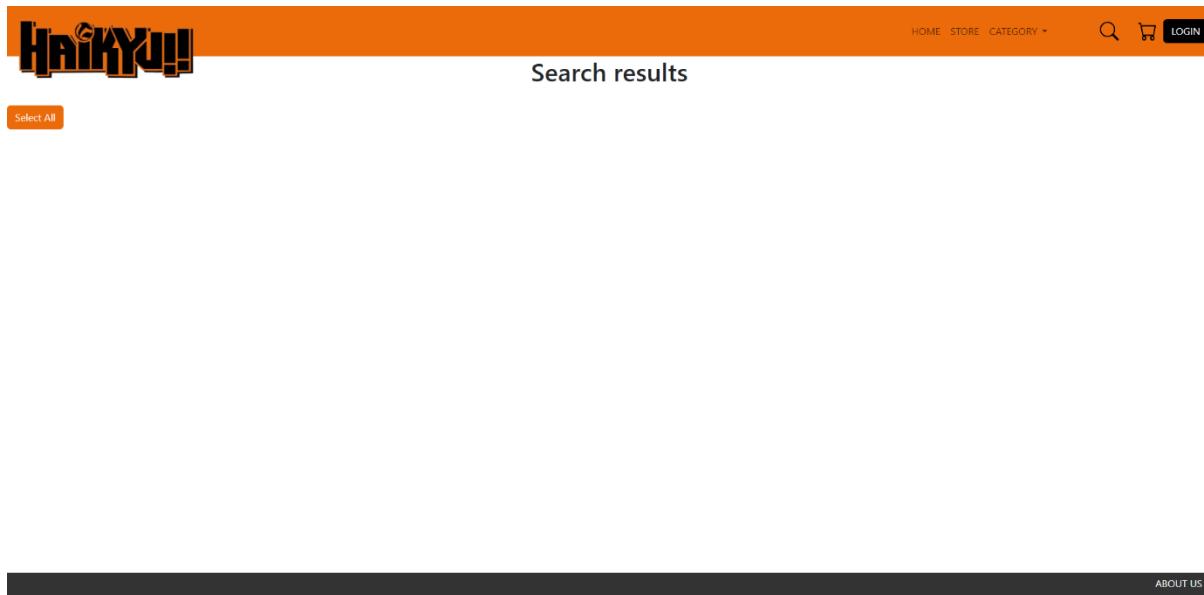
ในกล่องกรอกแบบฟอร์มนี้ลงข้างล่างสำหรับการเชื่อมโยงไปหน้า register admin

## 2.2 Register Page



ในส่วนของ content จะมีกล่องฟอร์ม 1 กล่อง ไว้สำหรับใส่ข้อมูลส่วนตัวของแอดมินเท่านั้น เช่น รูปภาพ, ชื่อ, นามสกุล เป็นต้น ในหน้านี้จำเป็นต้องกรอกทุกข้อมูลและแนบรูปไปด้วยจึงจะสามารถ register ได้ และเมื่อ register สำเร็จ จะกลับไปยังหน้า Homepage

## 3. Search Page



หน้า Search จะมีหัวข้อ Search result ไว้รอรับค่าผลลัพธ์ และมีปุ่ม Select all เพื่อทำการดูสินค้าทั้งหมด ในคลัง หากต้องการกรอกฟอร์มเพื่อค้นหาสินค้าโดยระบุรายละเอียด ให้ทำการกดสัญลักษณ์ค้นหาตรงแถบ nav bar อีกครั้ง ก็จะมีແບບฟอร์มเพื่อค้นหาเดิ่งขึ้นมาดังรูปข้างล่าง

Search name of product	
Category	None
Tag	None
Price	Min Price: <input type="text"/> Max Price: <input type="text"/>
Filter	Sort by Price: <input type="button" value="None"/>
<input type="button" value="Clear"/> <input type="button" value="Search"/>	



และเมื่อกรอกฟอร์มตามต้องการทั้งหมดเสร็จสิ้นเรียบร้อย ผลลัพธ์จะออกมารูปแบบดังรูปข้างล่าง

Haikyuu!!

[HOME](#) [STORE](#) [CATEGORY ▾](#)   [LOGIN](#)

Search results



Haikyuu Look Up Bokuto Kotaro Uniform Ver.

\$1520.00



Haikyuu!! Complete illustration book Owari to Hajimari

\$1647.00



Haikyuu Mochi Mochi Mascot Vol.5 (Re-Sale)

\$2700.00



Haikyuu Tenorinzu Collection 2

\$314.00



Haikyuu – Chara Badge Collection

\$162.00



Haikyuu!! – Qset - Shoyo Hinata & Tobio Kageyama

\$1986.00



Haikyuu!! Complete illustration book Owari to Hajimari

\$1691.00

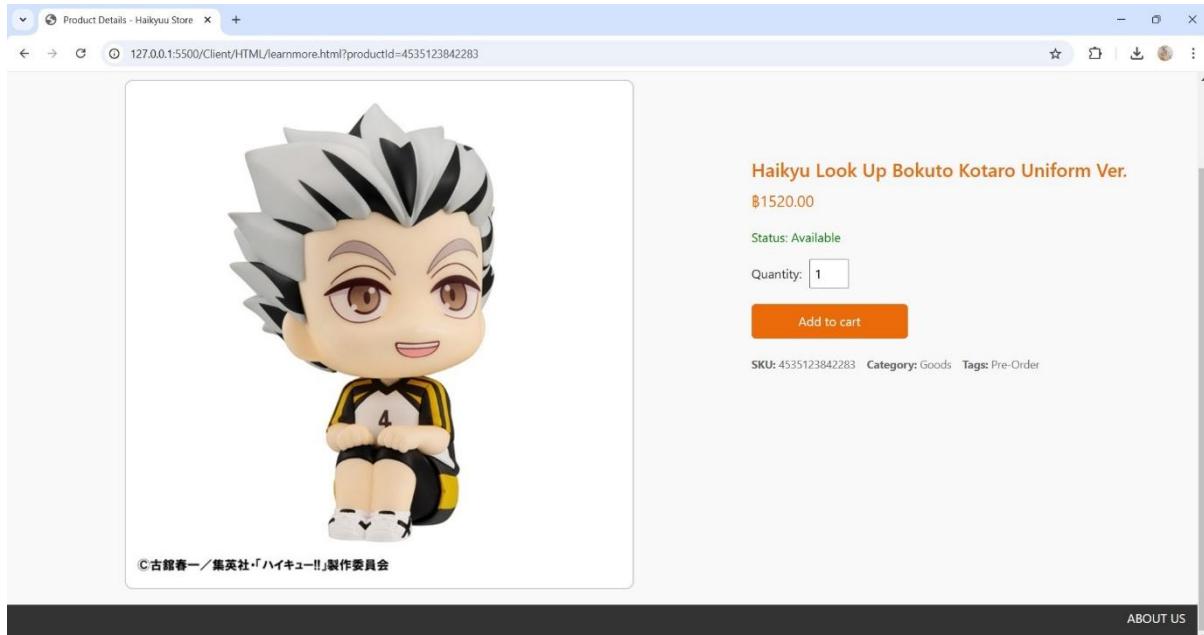


Haikyuu!! – Nendoroid Surprise Haikyuu!! Nationals Arc

\$3639.00

[ABOUT US](#)

#### 4.Detail Page



ในส่วนของหน้า detail page หรือหน้า learn more เป็นหน้าส่วนขยายของหน้า Home Page ที่จะแสดงรายละเอียดของสินค้าที่อยู่ในหน้า Home Page, Update Product และที่เป็นส่วนขยายมาจากหน้า result of search โดยในหน้า Detail Page จะมีชื่อสินค้า ราคาของสินค้า สเตตัสของสินค้า จำนวนสินค้าที่ลูกค้าต้องการ ปุ่มเพิ่มลงตะกร้า และด้านท้ายสุดจะเป็นรายละเอียดของสินค้า คือ SKU ประเภท และ Tag

## 5. Product Management Page

Pic.	SKU	NAME	CATEGORY	TAG	PRICE	STATUS
	4535123842283	Haikyu Look Up Bokuto Kotaro Uniform Ver.	Goods	Pre-Order	1520.00	
	4544815079777	Haikyu Mochi Mochi Mascot Vol.5 (Re-Sale)	Goods	Pre-Order	2700.00	
	4550621139529	Haikyu - Chara Badge Collection	Goods	New	162.00	
		Haikyu - NEW Anniversay Stand Festival Miu	Popular			

หน้า Product Management จะมีหน้าแรกเป็นหน้าดังรูป ในหน้านี้จะสามารถ เพิ่มสินค้าได้ โดยกดปุ่ม Add Product จะเข้าสู่หน้า Add Product

Add Product

Popular Items

Choose File No file chosen

Edit By:

SKU:

Name:

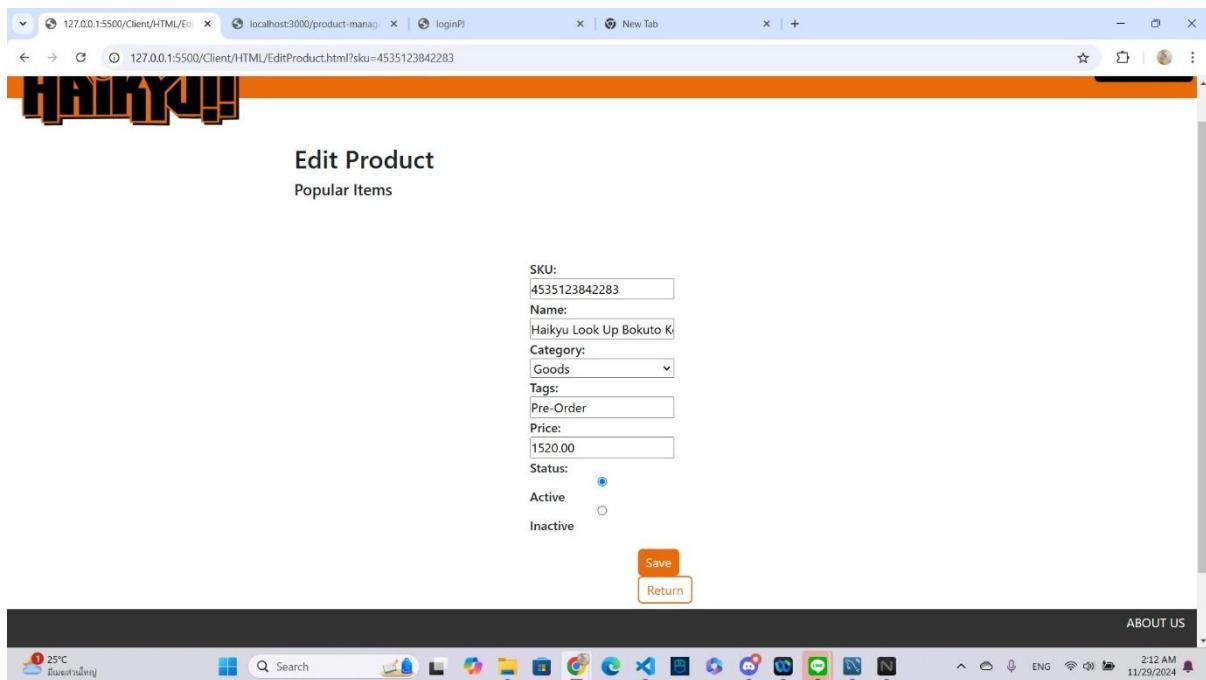
Category:

Tags:

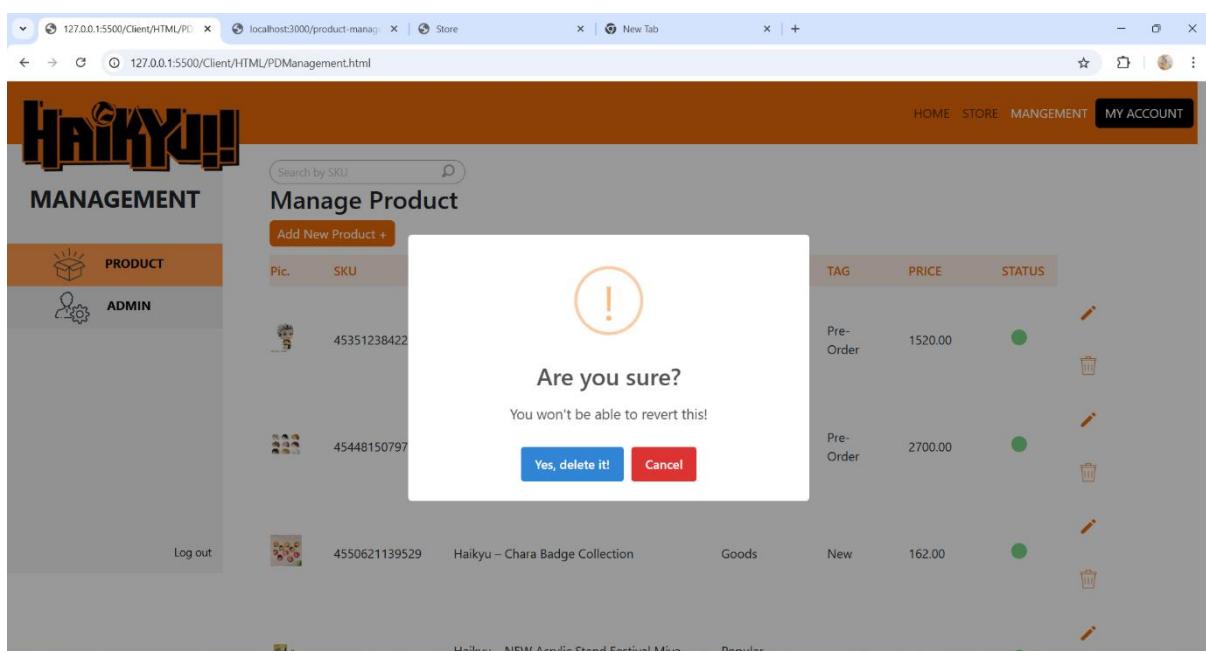
Price:

Status:

สามารถเปลี่ยนข้อมูลของสินค้าได้ โดยกดปุ่มรูปปิดนลสอนของสินค้านั้นๆ และจะเข้าสู่หน้า Edit Product



ສຸດທ້າຍຈະຄືກາລົບສິນຄ້າ ໂດຍຈະຕ້ອງກົດປຸ່ມຮູບຄັ້ງຂະໜາດຂອງສິນຄ້ານັ້ນໆ  
ແລະຈະເຂົ້າສູ່ຫ້າຕ່າງທີ່ເປັນການບອກວ່າຈະລົບສິນຄ້າໃໝ່ໂຮ້ອໍໄວ່



## 6. User Account Management

ในส่วนของหน้า User Account Management จะเป็นหน้าจัดการ Admin

โดยค่าเริ่มต้นของหน้าจะเป็นดังรูป

The screenshot shows a web browser window with the URL [127.0.0.1:5500/Client/HTML/AdminManage.html](http://127.0.0.1:5500/Client/HTML/AdminManage.html). The page has a header with the logo 'HaiYuu!! MANAGEMENT' and navigation links for HOME, STORE, MANGEMENT, and MY ACCOUNT. A search bar labeled 'Search by ID' is at the top. The main content area is titled 'Manage Admin' and contains a table with two rows of data:

Pic.	ID	NAME	EMAIL	Tel.	Action
	34	datimpa Luangtip	sutarnthip.lua@student.mahidol.edu	null	
	35	Sutarnthip fdlsf	sdsdsds@gmail.com	062-458-6521	

At the bottom left is a 'Log out' link, and at the bottom right is a 'ABOUT US' link.

โดยในหน้านี้จะสามารถเพิ่มแอดมินได้ โดยกดปุ่ม Add Admin เพิ่มข้อมูลของแอดมิน และเพิ่มรูปภาพได้ดังรูป

The screenshot shows a web browser window with the URL [127.0.0.1:5500/Client/HTML/AddAdmin.html](http://127.0.0.1:5500/Client/HTML/AddAdmin.html). The page has a header with the logo 'HaiYuu!! MANAGEMENT' and navigation links for HOME, STORE, MANGEMENT, and MY ACCOUNT. The main content area is titled 'Add Admin' and contains a form with fields for user information:

	<b>ID:</b> Enter ID	<b>First Name:</b> Enter First Name
<b>Last Name:</b> Enter Last Name	<b>Tel:</b> Enter Telephone Number	
<b>E-mail:</b> Enter E-mail	<b>Username:</b> Enter Username	
<b>Password:</b> Enter Password	<b>Edit By:</b> Enter Your ID	

Below the form are 'Cancel' and 'Save' buttons. At the bottom right is a 'ABOUT US' link.

สามารถแก้ไขข้อมูลของแอดมินได้โดยกดปุ่มดินสอ ของแอดมินแต่ละคนที่ต้องการ جانนี้จะเข้าสู่หน้าของ

Edit Admin ดังรูป

The screenshot shows a web browser window titled 'Edit Admin' with the URL '127.0.0.1:5500/Client/HTML/EditAdmin.html?id=34'. The page has an orange header with the logo 'Haikyu!!' and navigation links for HOME, STORE, MANAGEMENT, and MY ACCOUNT. The main content area is titled 'Edit Account' and contains fields for ID, First Name, Last Name, Tel, E-mail, Username, and Password. Below the form are 'Cancel' and 'Save' buttons, and a link to 'ABOUT US'.

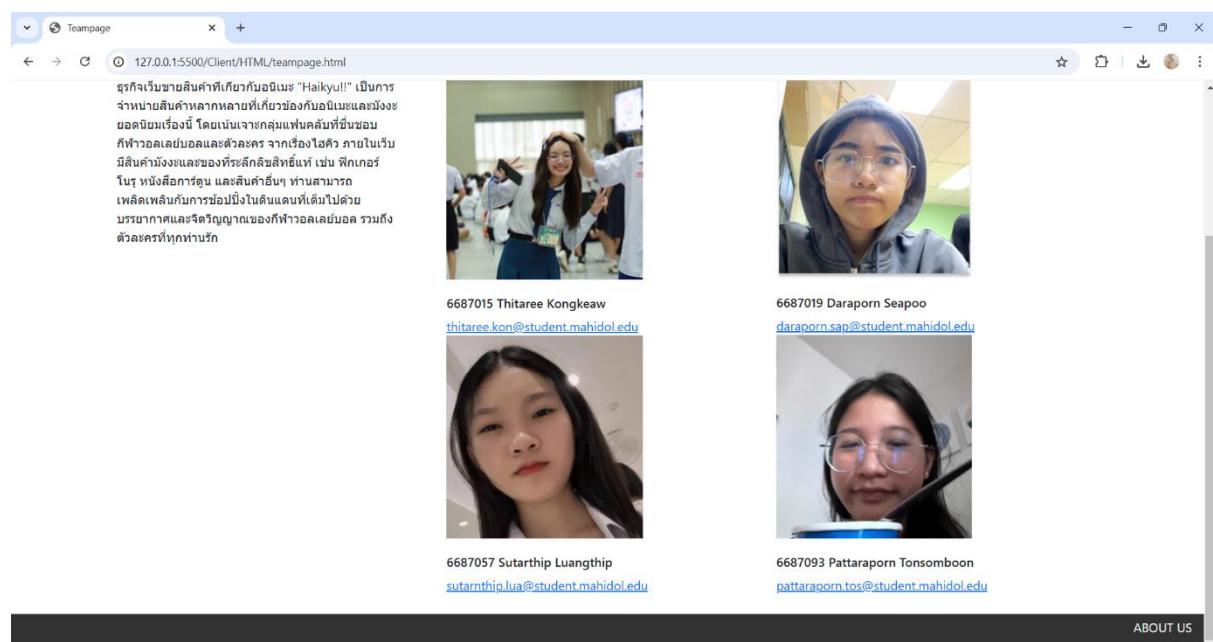
ID:	First Name:
Enter ID	Enter First Name
Last Name:	Tel:
Enter Last Name	Enter Telephone Number
E-mail:	Username:
Enter E-mail	Enter Username
Password:	
Enter Password	

Cancel Save

ABOUT US

## 7. Team Page

ในส่วนของหน้า team page จะเป็นหน้าคำอธิบายของธุรกิจ และเป็นข้อมูลของผู้จัดทำ และเป็นอีเมลของผู้จัดทำของแต่ละคน



## รายละเอียดของเว็บเซอร์วิส และโค้ด

### 1.External API

```
<script>
/*Use external api*/
const API_KEY = 'f23d0e1794e03e5225f1fbdb289c88513';
const city = 'Bangkok';
const url = `https://api.openweathermap.org/data/2.5/weather?q=${city}&appid=${API_KEY}&units=metric`;

fetch(url)
  .then(response => {
    if (!response.ok) {
      throw new Error(`HTTP error! status: ${response.status}`);
    }
    return response.json();
  })
  .then(data => {
    // Update the weather container with the fetched data
    document.getElementById('weather-city').textContent = data.name;
    document.getElementById('weather-temp').textContent = data.main.temp;
    document.getElementById('weather-desc').textContent = data.weather[0].description;
  })
  .catch(error => {
    console.error('Error fetching weather data:', error);
  });
});
```

ดึงข้อมูลสภาพอากาศของเมือง Bangkok จาก API ของ OpenWeatherMap และแสดงข้อมูลบนหน้าเว็บ โดยใช้ JavaScript เพื่อทำการร้องขอข้อมูลและอัปเดต DOM (Document Object Model) ของหน้าเว็บ

### 2.Product Management

#### 2.1 Show product data

```
422  /*ดึงข้อมูลจากproductsมาแสดงในหน้าproduct management*/
423  app.get('/product-management', (req, res) => [
424    const query = 'SELECT * FROM products';
425    connection.query(query, (err, results) => {
426      if (err) {
427        console.error('Error fetching data:', err);
428        res.status(500).send(err);
429        return;
430      }
431      res.json(results);
432    });
433  ]);
```

API นี้ทำหน้าที่ ดึงข้อมูลจากตาราง products ในฐานข้อมูล และส่งข้อมูลกลับไปยังผู้ใช้ในรูปแบบ JSON ผ่านเส้นทาง /product-management เพื่อแสดงข้อมูลสินค้าในหน้า Product Management

## 2.2 Add product

```
451 /*Add product*/
452 app.post('/add-product', upload.single('picture'), (req, res) => {
453   console.log("add product")
454   const { Editer, PDSKU, PDName, PDCategory, PDTag, PDPPrice, status } = req.body;
455   const file = req.file;
456 
457   if (!file) {
458     return res.status(400).send('File upload is required.');
459   }
460   const productQuery = `
461     INSERT INTO products (image, sku, name, category, tag, price, status)
462     VALUES (?, ?, ?, ?, ?, ?, ?);
463   `;
464   const productValues = [
465     `${file.filename}`, // Save relative file path
466     PDSKU,
467     PDName,
468     PDCategory,
469     PDTag,
470     PDPPrice,
471     status
472   ];
473 
474   connection.query(productQuery, productValues, (productErr, productResult) => {
475     if (productErr) {
476       console.error('Error inserting product:', productErr);
477       return res.status(500).json({ error: 'Error saving product.' });
478     }
479   }
480 
481   // Insert log into action_logs table
482   const actionQuery = `
483     INSERT INTO action_logs (admin_id, sku, action)
484     VALUES (?, ?, 'Add product');
485   `;
486   const actionValues = [Editer, PDSKU];
487 
488   connection.query(actionQuery, actionValues, (actionErr) => {
489     if (actionErr) {
490       console.error('Error inserting action log:', actionErr);
491       return res.status(500).json({ error: 'Error saving action log.' });
492     }
493   }
494 
495   // Insert file details into uploads table
496   const uploadSql = `
497     INSERT INTO uploadsPD (original_name, saved_name, path, mimetype, size, sku)
498     VALUES (?, ?, ?, ?, ?, ?);
499   `;
500   const uploadValues = [
501     file.originalname,           // Original file name
502     file.filename,              // Unique saved name
503     `${file.filename}`,         // File path
504     file.mimetype,              // File MIME type
505     file.size,                  // File size
506     PDSKU                      // Link to the product
507   ];
508 
509   connection.query(uploadSql, uploadValues, (uploadErr,uploadResult) => {
510     if (uploadErr) {
511       console.error('Error inserting upload:', uploadErr);
512       return res.status(500).json({ error: 'Error saving file upload.' });
513     }
514   }
515 });

res.redirect('/product-management'); // Redirect after successful addition
```

API รับคำขอ (Request) ผ่าน HTTP POST ที่ /add-product ใช้ upload.single('picture') สำหรับจัดการไฟล์ภาพที่อัปโหลด (รองรับการอัปโหลดไฟล์เดียวในฟิล์ดชื่อ picture)

ถ้าไม่มีไฟล์แนบมา กับคำขอ จะตอบกลับด้วย HTTP Status 400 (Bad Request) พร้อมข้อความแจ้งเตือน  
 จากนั้นสร้างคำสั่ง SQL เพื่อเพิ่มข้อมูลสินค้าลงในตาราง products หลังจากเพิ่มสินค้าแล้ว  
 จะเพิ่มข้อมูลการกระทำ (Action) ลงในตาราง action\_logs เพื่อเก็บประวัติการเพิ่มสินค้า  
 จากนั้นเพิ่มข้อมูลเกี่ยวกับไฟล์ภาพที่อัปโหลดลงในตาราง uploadsPD และส่งผู้ใช้กลับไปหน้า Product  
 Management หากทุกขั้นตอนสำเร็จ ระบบจะส่งผู้ใช้ไปที่หน้า /product-management

### 2.3 Edit product

```
/*show Product data*/
app.get('/product/:sku', (req, res) => {
  const PDSKU = req.params.sku;
  const query = 'SELECT * FROM products WHERE sku = ?';

  connection.query(query, [PDSKU], (err, results) => {
    if (err) {
      console.error('Error fetching product data:', err);
      return res.status(500).send('Error fetching product data');
    }

    if (results.length > 0) {
      res.json(results[0]);
    } else {
      res.status(404).send('Product not found');
    }
  });
});
```

รับ params sku จาก URL และดึงข้อมูล sku ที่ตรงกันจากตาราง products มาส่งเป็นรูปแบบ json  
 เพื่อไปแสดงในหน้า edit product

```
661 /*edit product */
662 app.put('/update_product/:sku', upload2.none(), (req, res) => {
663   const PDSKU = req.params.sku
664   console.log(PDSKU)
665   const { PDName, PDCategory, PDTag, PDPrice, status } = req.body;
666   console.log('Parsed Body:', req.body);
667
668   const query = `UPDATE products SET
669     name = ?,
670     category = ?,
671     tag = ?,
672     price = ?,
673     status = ?
674   WHERE sku = ?`;
675
676   connection.query(query, [PDName, PDCategory, PDTag, PDPrice, status, PDSKU], (err, results) => {
677     if (err) {
678       console.error('Error updating product data:', err);
679       return res.status(500).send('Error updating product data');
680     }
681
682     res.send('Product data updated successfully');
683   });
684});
```

รับ params sku จาก URL เพื่อรับสินค้าที่ต้องการแก้ไข ใช้ req.body เพื่อตีงข้อมูลใหม่ของสินค้า  
ใช้ค่าที่รับมาใน req.body และ sku ในการอัปเดตข้อมูลในตาราง products

## 2.4 Delete product

```
686  /*Delete product*/
687  app.delete('/delete-product/:sku', (req, res) => {
688      const sku = req.params.sku;
689      const query = "DELETE FROM products WHERE sku = ?";
690      connection.query(query, [sku], (err, result) => {
691          if (err) return res.status(500).json(err);
692          return res.status(200).json({ msg: "Deleted" });
693      })
694  }
695 })
```

รับ params sku จาก URL เพื่อรับสินค้าที่ต้องการลบ จากนั้นลบสินค้าที่มี sku ตรงกันในตาราง products

## 3.Admin management

### 3.1 Show admin data

```
435  /*ดึงข้อมูลจากadministratorมาแสดงในหน้าadmin management*/
436  app.get('/admin-management', (req, res) => {
437      const query = 'SELECT id, first_name, last_name, email, telephone, username, profile_image FROM administrator';
438      connection.query(query, (err, results) => {
439          if (err) {
440              console.error('Error fetching data:', err);
441              res.status(500).send(err);
442              return;
443          }
444          console.log('Fetched results:', results); // เช็คผลลัพธ์ที่ได้
445          res.json(results);
446      });
447  });
448 })
```

API นี้ทำหน้าที่ ดึงข้อมูลจากตาราง administrator ในฐานข้อมูล

แล้วส่งข้อมูลกลับไปยังผู้ใช้ในรูปแบบ JSON ผ่านเส้นทาง /admin-management

เพื่อแสดงข้อมูลแอดมินในหน้า Admin Management

### 3.2 Add admin

```
520  /*Add admin*/
521  app.post('/add-admin', upload.single('picture'), (req, res) => {
522    const { id, first_name, last_name, email, tel, username, password, editer } = req.body;
523    const file = req.file;
524
525    if (!file) {
526      return res.status(400).send('File upload is required.');
527    }
528
529    // Insert administrator details into the administrators table
530    const adminQuery =
531      `INSERT INTO administrator (id,first_name ,last_name , email, telephone, username, password , profile_image)
532      VALUES (?, ?, ?, ?, ?, ?, ?, ?);`;
533
534    const adminValues = [
535      id,
536      first_name,
537      last_name,
538      email,
539      tel,
540      username,
541      password,
542      `${file.filename}`
543    ];
544
545    connection.query(adminQuery, adminValues, (productErr, productResult) => {
546      if (productErr) {
547        console.error('Error Adding admin:', productErr);
548        return res.status(500).json({ error: 'Error.' });
549      }
550
551      // Insert log into action_logs table
552      const actionQuery =
553        `INSERT INTO action_admin (EditBy, admin_id, action)
554          VALUES (?, ?, 'Add Admin');`;
555
556      const actionValues = [editer, id];
557
558      connection.query(actionQuery, actionValues, (actionErr) => {
559        if (actionErr) {
560          console.error('Error inserting action log:', actionErr);
561          return res.status(500).json({ error: 'Error saving action log.' });
562        }
563
564        // Insert file details into uploads table
565        const uploadSql =
566          `INSERT INTO uploads (original_name, saved_name, path, mimetype, size, admin_id)
567            VALUES (?, ?, ?, ?, ?, ?);`;
568
569      const uploadValues = [
570        file.originalname,           // Original file name
571        file.filename,             // Unique saved name
572        `${file.filename}`,        // File path
573        file.mimetype,             // File MIME type
574        file.size,                 // File size
575        id,                      // Link to the product
576      ];
577
578      connection.query(uploadSql, uploadValues, (uploadErr, uploadResult) => {
579        if (uploadErr) {
580          console.error('Error inserting upload:', uploadErr);
581          return res.status(500).json({ error: 'Error saving file upload.' });
582        }
583
584        res.redirect('/admin-management'); // Redirect after successful addition
585      }
586    }
587  }
588}
```

API รับคำขอ (Request) ผ่าน HTTP POST ที่ /add-admin ใช้ upload.single('picture')

สำหรับจัดการไฟล์ภาพที่อัปโหลด (รองรับการอัปโหลดไฟล์เดียวในฟิล์ชื่อ picture)

ถ้าไม่มีไฟล์แนบมากับคำขอ จะตอบกลับด้วย HTTP Status 400 (Bad Request) พร้อมข้อความแจ้งเตือน

จากนั้นสร้างคำสั่ง SQL เพื่อเพิ่มข้อมูลสินค้าลงในตาราง administrator หลังจากเพิ่มแอدمินแล้ว

จะเพิ่มข้อมูลการกระทำ (Action) ลงในตาราง action\_admin เพื่อกีบประวัติการเพิ่มแอدمิน

จากนั้นเพิ่มข้อมูลเกี่ยวกับไฟล์ภาพที่อัปโหลดลงในตาราง upload และส่งผู้ใช้กลับไปหน้า Admin

Management หากทุกขั้นตอนสำเร็จ ระบบจะส่งผู้ใช้ไปที่หน้า /admin-management

### 3.3 Edit admin

```
591  /*showadmin data*/
592  app.get('/admin/:id', (req, res) => {
593      // if (!req.session.user) {
594      //     return res.status(401).send('Not logged in');
595      //
596
597      const adminId = req.params.id; // Assume user session contains admin's ID
598      const query = 'SELECT * FROM administrator WHERE id = ?';
599
600      connection.query(query, [adminId], (err, results) => {
601          if (err) {
602              console.error('Error fetching admin data:', err);
603              return res.status(500).send('Error fetching admin data');
604          }
605
606          if (results.length > 0) {
607              res.json(results[0]); // Send back the first admin result
608          } else {
609              res.status(404).send('Admin not found');
610          }
611      });
612  });


```

รับ params id จาก URL และดึงข้อมูล id ที่ตั้งกันจากตาราง administrator มาส่งเป็นรูปแบบ json เพื่อไปแสดงในหน้า edit admin

```
/*Update admin */
const upload2 = multer();
app.put('/update_admin/:id', upload2.none(), (req, res) => {
    // if (!req.session.user) {
    //     return res.status(401).send('Not logged in');
    // } //เช็คว่า user ซึ่งได้ล็อกอินมีอยู่

    const adminId = req.params.id
    const { first_name, last_name, tel, email, username, password } = req.body;
    console.log('Parsed Body:', req.body);

    const query = `UPDATE administrator SET
        first_name = ?,
        last_name = ?,
        telephone = ?,
        email = ?,
        username = ?,
        password = ?
    WHERE id = ?`;

    connection.query(query, [first_name, last_name, tel, email, username, password, adminId], (err, results) => {
        if (err) {
            console.error('Error updating admin data:', err);
            return res.status(500).send('Error updating admin data');
        }

        res.send('Admin data updated successfully');
    });
});


```

รับ params id จาก URL เพื่อรับและมินที่ต้องการแก้ไข ใช้ req.body

เพื่อดึงข้อมูลใหม่ของแอดมิน ใช้ค่าที่รับมาใน req.body และ id ในการอัปเดตข้อมูลในตาราง administrator

### 3.4 Delete admin

```
696  /*Delete Admin*/
697  app.delete('/delete-admin/:id', (req, res) => {
698      const id = req.params.id;
699      const query = "DELETE FROM administrator WHERE id = ?";
700      connection.query(query, [id], (err, result) => {
701          if (err) return res.status(500).json(err);
702          return res.status(200).json({ msg: "Deleted" });
703      })
704  }
705 )
```

รับ params id จาก URL เพื่อรับและมินที่ต้องการลบ จากนั้นลบแอดมินที่มี id ตรงกันในตาราง administrator

### 3.5 Search Admin by id

```
668  /*Search admin by id*/
669  app.get('/admin-management/:id', (req, res) => {
670      const id = req.params.id;
671
672      const query = 'SELECT id, first_name, last_name, email, telephone, profile_image FROM administrator WHERE id = ?';
673
674      connection.query(query, [id], (err, results) => {
675          if (err) {
676              console.error('Error fetching admin:', err);
677              return res.status(500).json({ error: 'Internal server error' });
678          }
679
680          if (results.length === 0) {
681              return res.status(404).json({ message: 'Admin not found' });
682          }
683
684          res.status(200).json(results[0]);
685      });
686  })
```

รับคำขอ GET ที่มี id ของผู้ดูแลระบบ จากนั้นตรวจสอบและดึงข้อมูลจากฐานข้อมูลในตาราง administrator ที่มี oid ตรงกับคำขอที่ส่งมา จัดการข้อมูลพลาดหรือตอบกลับข้อมูลผู้ดูแลระบบในรูปแบบ JSON ที่คำสั่ง if

## 4.Register

API รับคำขอ (Request) ผ่าน HTTP POST ที่ /register-submit ใช้ upload.single('AMFPic') สำหรับจัดการไฟล์ภาพที่อัปโหลด (รองรับการอัปโหลดไฟล์เดียวในฟิล์ดชื่อ AMFPic)  
ถ้าไม่มีไฟล์แนบมากับคำขอ จะตอบกลับด้วย HTTP Status 400 (Bad Request) พร้อมข้อความแจ้งเตือน  
จากนั้นสร้างคำสั่ง SQL เพื่อเพิ่มข้อมูลสินค้าลงในตาราง administrator หลังจากเพิ่มแออดมินแล้ว  
จะเพิ่มข้อมูลการกระทำ (Action) ลงในตาราง action\_admin เพื่อเก็บประวัติการเพิ่มแออดมิน  
จากนั้นเพิ่มข้อมูลเกี่ยวกับไฟล์ภาพที่อัปโหลดลงในตาราง upload และตอบกลับด้วย HTTP Status 200  
พร้อมคำว่า Registration successful

```
app.post('/register-submit', upload.single('AMFPic'), (req, res) => {
  const { AMFName, AMLName, AMEmail, AMTelephone, AMUsername, AMPassword } = req.body;
  const file = req.file;

  if (!file) {
    return res.status(400).send('File upload is required.');
  }

  // Insert administrator details into the administrators table
  const adminSql = `INSERT INTO administrator (first_name, last_name, email, telephone, username, password, profile_image)
    VALUES (?, ?, ?, ?, ?, ?, ?)`;

  const adminValues = [
    AMFName,
    AMLName,
    AMEmail,
    AMTelephone,
    AMUsername,
    AMPassword, // You may want to hash the password for security
    file.filename // Save the uploaded file's unique name
  ];

  connection.query(adminSql, adminValues, (adminErr, adminResult) => {
    if (adminErr) {
      console.error('Error inserting administrator:', adminErr);
      return res.status(500).send('Error saving administrator.');
    }

    const adminId = adminResult.insertId; // Get the inserted administrator's ID

    // Insert file details into the uploads table
    const uploadSql = `INSERT INTO uploads (original_name, saved_name, path, mimetype, size, admin_id)
      VALUES (?, ?, ?, ?, ?, ?)`;

    const uploadValues = [
      file.originalname,           // Original file name
      file.filename,              // Unique saved name
      `uploads/${file.filename}`, // File path
      file.mimetype,              // File MIME type
      file.size,                  // File size
      adminId                    // Link to the administrator
    ];

    connection.query(uploadSql, uploadValues, (uploadErr, uploadResult) => {
      if (uploadErr) {
        console.error('Error inserting upload:', uploadErr);
        return res.status(500).send('Error saving file upload.');
      }

      // Once everything is done, send a success message
      res.status(200).send('Registration successful');
    });
  });
});
```

## 5.Search

### 5.1 SelectAll

```
app.get("/products", (req, res) => {
  const query = `
    SELECT
      SKU,
      Name,
      Price,
      Tag,
      Category,
      image,
      status
    FROM products;
  `;

  connection.query(query, (err, results) => {
    if (err) {
      console.error("Error fetching products:", err);
      res.status(500).send("Error fetching products");
    } else {
      res.json(results);
    }
  });
});
```

API นี้ทำหน้าที่ ดึงข้อมูลจากตาราง products ในฐานข้อมูลทั้งหมดที่สินค้ายังคง active หรืออีกความหมาย  
หนึ่งคือสินค้าพร้อมขาย ซึ่งจะทำการ return result ในรูปแบบ json เพื่อให้ดึงข้อมูลที่ส่งกลับไปไว้ในหน้า  
result of search

## 5.2 Search with name and filters

```
router.post("/search", (req, res) => {
  const { PDName, tags, minPrice, maxPrice, category, sort } = req.body;
  let query = `SELECT p.SKU, p.Name, p.Price, p.Tag, p.Category, MAX(pi.image_path) AS image_path, MAX(pi.image_order) AS image_order
    FROM product AS p
    LEFT JOIN product_images AS pi ON p.SKU = pi.SKU
    WHERE pi.image_order = 1`;
  let queryParams = [];

  // Filter by Product Name
  if (PDName && PDName !== "") {
    query += ` AND p.Name LIKE ?`;
    queryParams.push(`%${PDName}%`);
  }

  // Filter by Category
  if (category && category !== "") {
    query += ` AND p.Category = ?`;
    queryParams.push(category);
  }

  // Filter by Tags
  if (tags && tags !== "") {
    query += ` AND p.Tag = ?`;
    queryParams.push(tags);
  }

  // Filter by Min Price
  if (minPrice && !isNaN(minPrice)) {
    query += ` AND p.Price >= ?`;
    queryParams.push(parseFloat(minPrice));
  }

  // Filter by Max Price
  if (maxPrice && !isNaN(maxPrice)) {
    query += ` AND p.Price <= ?`;
    queryParams.push(parseFloat(maxPrice));
  }

  query += ` GROUP BY p.SKU, p.Name, p.Price, p.Tag, p.Category`;

  if (sort && (sort === 'ASC' || sort === 'DESC') && sort !== "") {
    query += ` ORDER BY p.Price ${sort}`;
  }

  // Execute query
  connection.query(query, queryParams, (err, results) => {
    if (err) {
      console.error('Database query error:', err);
      return res.status(500).json({ success: false, message: 'Database query failed', error: err });
    }
    res.json(results);
  });
});
```

ในส่วนนี้ จะเป็น API ที่ใช้ search by name และมี filter โดยจะรับค่า name

ของสินค้ามาจาก input และเลือกตัวกรองที่ต้องการจากหน้า search เพื่อนำมาค้นหาใน data base

เพื่อหาสินค้าที่มีอยู่ใน stock เพื่อนำมาใช้ในหน้า result of search

## 6.Store Page

### 6.1 Show all of Product in Store Page

```
app.get("/products", (req, res) => {
  const query = `
    SELECT
      SKU,
      Name,
      Price,
      Tag,
      Category,
      image,
      status
    FROM products;
  `;

  connection.query(query, (err, results) => {
    if (err) {
      console.error("Error fetching products:", err);
      res.status(500).send("Error fetching products");
    } else {
      res.json(results);
    }
  });
});
```

ในส่วนนี้ จะเป็น API ที่ใช้สำหรับการดึงข้อมูลสินค้าทั้งหมด (fetch all products)

โดยไม่ต้องมีตัวกรองเพิ่มเติม API นี้จะส่งคำสั่ง SQL เพื่อดึงข้อมูลสินค้าจากฐานข้อมูล (database)

ทั้งหมดที่มีในตาราง products และส่งข้อมูลกลับมาในรูปแบบ JSON

หากระบบพบข้อผิดพลาดในการดึงข้อมูล ระบบจะแสดงข้อความแสดงข้อผิดพลาด (500 - Error fetching products) และหากไม่มีข้อผิดพลาด ข้อมูลสินค้าทั้งหมดที่ดึงมาได้จะถูกส่งกลับไปยังผู้ใช้งาน

### 6.2 Show Product of Category

```

app.get("/category/:Category", (req, res) => {
  const Category = decodeURIComponent(req.params.Category);

  const query = `
    SELECT
      SKU, Name, Price, Tag, Category, image, status
    FROM products
    WHERE Category = ? AND status = 'active';
  `;

  connection.query(query, [Category], (err, results) => {
    if (err) {
      console.error("Error fetching products by category:", err);
      return res.status(500).send("Error fetching products by category");
    }
    if (results.length === 0) {
      return res.status(404).send("No products found in this category");
    }
    res.json(results);
  });
});

```

ในส่วนนี้ จะเป็น API ที่ใช้สำหรับการค้นหาสินค้าตามหมวดหมู่ (search by category)

โดยรับค่าหมวดหมู่สินค้ามาจากการอ้างอิง (:Category) และนำค่าดังกล่าวไปตรวจสอบในฐานข้อมูล (database) เพื่อค้นหาสินค้าที่อยู่ในหมวดหมู่ดังกล่าว โดยจะเลือกเฉพาะสินค้าที่สถานะ (status) เป็น 'active' เท่านั้น หากพบสินค้าที่ตรงกับเงื่อนไข ระบบจะส่งข้อมูลสินค้าที่พับกลับเป็นรูปแบบ JSON แต่หากไม่มีสินค้าตรงกับหมวดหมู่ดังกล่าว ระบบจะตอบกลับด้วยข้อความแสดงข้อผิดพลาด (404 - No products found) และหากเกิดปัญหานในการเข้มต่อ กับฐานข้อมูล ระบบจะส่งข้อความแสดงข้อผิดพลาด (500 - Error fetching products by category)

### 6.3 Show Product of SKU (each Product)

```

app.get("/learnmore/:SKU", (req, res) => {
  const { SKU } = req.params;

  const query = `
    SELECT
      SKU,
      Name,
      Price,
      Tag,
      Category,
      image,
      status
    FROM products
    WHERE SKU = ?;
  `;

  connection.query(query, [SKU], (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");
    } else {
      res.json(results[0]); // ส่งเฉพาะสินค้าเดียว
    }
  });
});

```

ในส่วนนี้ จะเป็น API สำหรับค้นหาสินค้าตามรหัสสินค้า (search by SKU) โดยรับค่า SKU มาจากพารามิเตอร์ใน URL (:SKU) และนำค่า SKU ดังกล่าวไปใช้ในการค้นหาในฐานข้อมูล (database) เพื่อดึงข้อมูลสินค้าที่มีรหัสตรงกัน หากพบข้อมูลสินค้า ระบบจะส่งข้อมูลของสินค้านั้นกลับมาในรูปแบบ JSON แต่หากไม่มีข้อมูลสินค้าในฐานข้อมูลที่ตรงกับรหัสที่ระบุ ระบบจะตอบกลับด้วยข้อความแสดงข้อผิดพลาด (404 - Product not found) และหากเกิดข้อผิดพลาดในการเชื่อมต่อ กับฐานข้อมูล ระบบจะส่งข้อความแสดงข้อผิดพลาด (500 - Error fetching product) โดยจะแสดงสินค้าที่มีรายละเอียดเฉพาะสินค้าที่มีรหัสนั้น ๆ

## 7.Login Page

### 7.1 Show all of Admin

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

ในส่วนนี้จะเป็น API สำหรับดึงข้อมูลผู้ดูแลระบบทั้งหมด (select all administrators)

โดยเมื่อมีการเรียกใช้ endpoint /login ผ่าน HTTP GET ระบบจะส่งคำสั่ง SQL (SELECT \* FROM administrator) ไปยังฐานข้อมูลเพื่อตรวจสอบและดึงข้อมูลทั้งหมดจากตาราง administrator

หากข้อมูลถูกดึงสำเร็จ ระบบจะส่งข้อมูลทั้งหมดในรูปแบบ JSON กลับไปยัง client

ในกรณีที่เกิดข้อผิดพลาดระหว่างการดึงข้อมูล (เช่น การเชื่อมต่อฐานข้อมูลล้มเหลว)

ระบบจะทำการหยุดการทำงานทันทีด้วยการ throw error เพื่อแสดงข้อผิดพลาดดังกล่าว

### 7.1 Show all of Admin

```

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

  // ตรวจสอบข้อมูลในตาราง Admin
  const query = 'SELECT * FROM administrator WHERE username = ? AND password = ?';
  connection.execute(query, [username, password], (err, result) => {
    if (err) {
      return res.status(500).send('Database error');
    }

    if (result.length > 0) {
      // ถ้าพบข้อมูลใน Admin
      const insertQuery = 'INSERT INTO login_logs (Username, password) VALUES (?, ?)';
      connection.execute(insertQuery, [username, password], (err, insertResult) => {
        if (err) {
          return res.status(500).send('Error logging login attempt');
        }
        res.send('Login successful!');
      });
    } else {
      // ถ้าไม่พบข้อมูล
      res.send('Invalid username or password!');
    }
  });
});

```

ในส่วนนี้จะเป็น API สำหรับตรวจสอบข้อมูลการเข้าสู่ระบบของผู้ดูแลระบบ (Admin Login)

โดยรับข้อมูล username และ password จาก req.body ผ่าน HTTP POST ไปยัง endpoint /login และมีขั้นตอนการทำงานดังนี้:

- ตรวจสอบข้อมูลในฐานข้อมูล (Authentication)
- ระบบจะใช้คำสั่ง SQL SELECT เพื่อตรวจสอบว่าในตาราง administrator มีข้อมูลที่ตรงกับ username และ password ที่ผู้ใช้งานมาหรือไม่ หากเกิดข้อผิดพลาดในการเชื่อมต่อฐานข้อมูล ระบบจะตอบกลับด้วยสถานะ 500 (Database error)
- กรณีที่พบข้อมูลผู้ใช้งาน ระบบจะบันทึกข้อมูลการเข้าสู่ระบบลงในฐานข้อมูล login\_logs (ตารางเก็บข้อมูลการ Login) โดยเพิ่ม username และ password เพื่อเก็บเป็นประวัติ หากการบันทึกข้อมูลเกิดข้อผิดพลาด ระบบจะตอบกลับด้วยสถานะ 500 (Error logging login attempt) หากบันทึกสำเร็จ ระบบจะตอบกลับด้วยข้อความ Login successful!
- กรณีที่ไม่พบข้อมูลผู้ใช้งาน ระบบจะตอบกลับด้วยข้อความ Invalid username or password!

## ผลการทดสอบของเว็บเซอร์วิสทั้งหมด โดยใช้โปรแกรม Postman หรือโปรแกรมอื่น ๆ

1. เรียบเซอร์วิสสำหรับการพิสูจน์ตัวตน (Authentication) สำหรับการเข้าสู่ระบบของผู้ดูแลระบบ

- Login with valid Username and Password

Method : POST

URL: <http://localhost:3000/login>

Body: raw JSON

```
{  
  "username": "6687019",  
  "password": "000000000"  
}
```

The screenshot shows the Postman application interface. At the top, it displays a POST request to the URL <http://localhost:3000/login>. Below the URL, there are tabs for Params, Authorization, Headers (8), Body (selected), Scripts, and Settings. Under the Body tab, the content type is set to raw and JSON. The JSON payload is identical to the one shown above. In the bottom section, under the Body tab, the response is displayed as:

```
1  {  
2   "username": "6687019",  
3   "password": "000000000"  
4 }  
5
```

Below the response, there are tabs for Body, Cookies, Headers (8), Test Results, and a refresh icon. Under the Body tab, there are buttons for Pretty, Raw, Preview, Visualize, and HTML (selected). The response body is shown as:

```
1  Login successful!
```

- Login with invalid Username and Password

Method : POST

URL: <http://localhost:3000/login>

Body: raw JSON

```
{  
  "username": "6687019",  
  "password": "11"  
}
```

The screenshot shows the Postman application interface. At the top, there is a header bar with the URL <http://localhost:3000/login>. Below the header, the method is set to **POST**, and the URL is confirmed as <http://localhost:3000/login>. The **Body** tab is selected, showing the raw JSON payload:

```
1  {  
2   "username": "6687019",  
3   "password": "11"  
4 }  
5
```

Below the body, the response status is shown as **1 Invalid username or password!**.

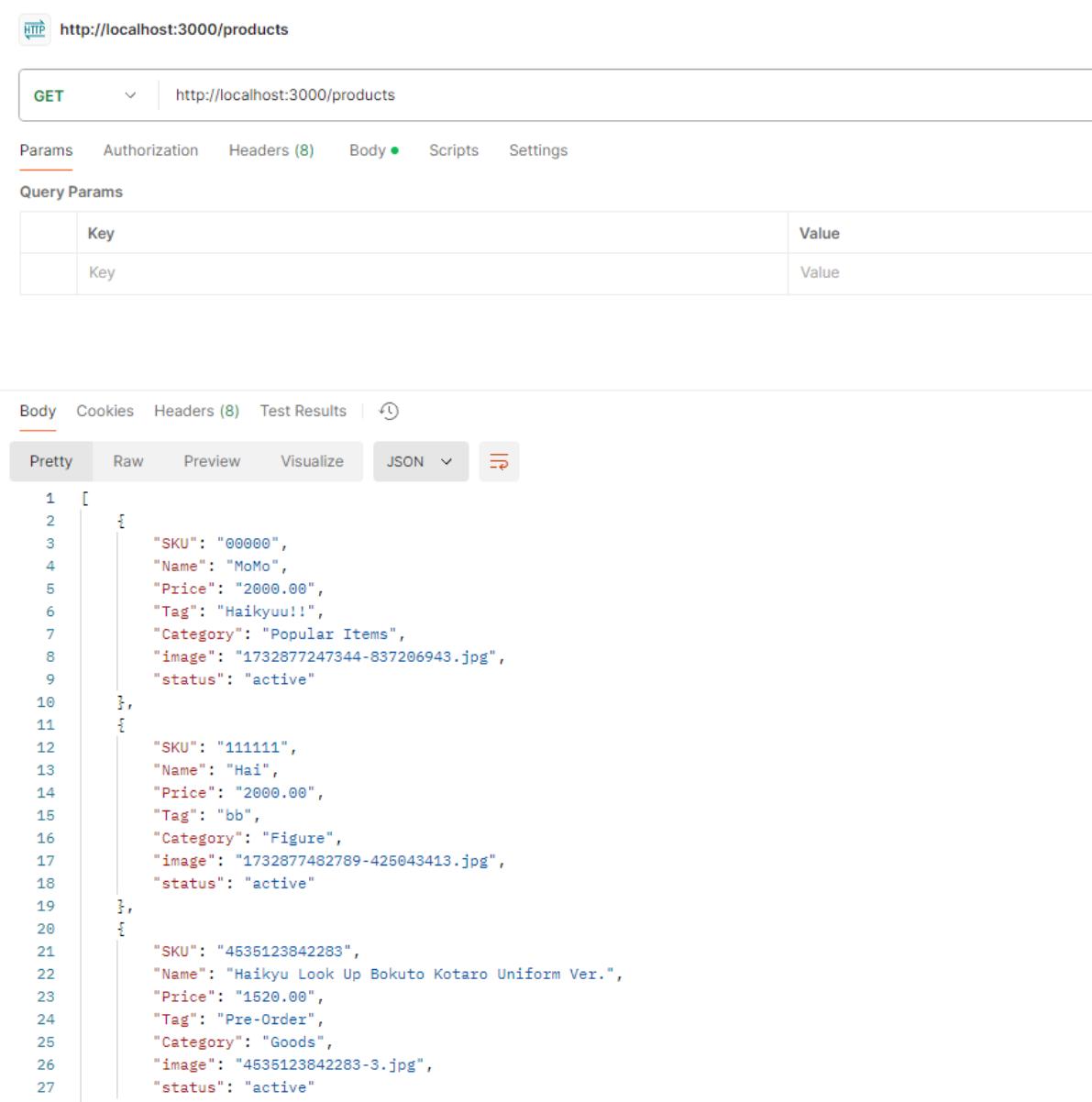
2. เว็บเซอร์วิสสำหรับการทำงานของผู้ดูแลระบบ การทำงานของผู้ดูแลระบบจะมีทั้งหมด 4 พังก์ชันหลัก

ก. การค้นหาและเข้าถึงข้อมูล (Search and View)

- Testing Select all product

Method : GET

URL: <http://localhost:3000/products>



The screenshot shows the Postman interface with a GET request to <http://localhost:3000/products>. The response body is a JSON array containing three products:

```
1  [
2    {
3      "SKU": "00000",
4      "Name": "MoMo",
5      "Price": "2000.00",
6      "Tag": "Haikyuu!!",
7      "Category": "Popular Items",
8      "image": "1732877247344-837206943.jpg",
9      "status": "active"
10    },
11    {
12      "SKU": "111111",
13      "Name": "Hai",
14      "Price": "2000.00",
15      "Tag": "bb",
16      "Category": "Figure",
17      "image": "1732877482789-426043413.jpg",
18      "status": "active"
19    },
20    {
21      "SKU": "4535123842283",
22      "Name": "Haikyu Look Up Bokuto Kotaro Uniform Ver.",
23      "Price": "1520.00",
24      "Tag": "Pre-Order",
25      "Category": "Goods",
26      "image": "4535123842283-3.jpg",
27      "status": "active"
28    }
]
```

- Testing Select product by name and category

Method : POST

URL: <http://localhost:3000/search>

Body: raw JSON

```
{
  "PDName": "Hai",
  "Tag": "Pre-Order",
  "MinPrice": null,
```

```

    "MaxPrice": 3000,
    "category": "Goods",
    "sort": "ASC"
}

}

```

http://localhost:3000/search

POST http://localhost:3000/search

Params Authorization Headers (8) Body Scripts Settings

Body (raw JSON)

```

1 {
2   "POName": "Hai",
3   "Tag": "Pre-Order",
4   "MinPrice": null,
5   "MaxPrice": 3000,
6   "category": "Goods",
7   "sort": "ASC"
8 }

```

200 OK 4 ms 793 B Save Response

Pretty Raw Preview Visualize JSON

```

1 [
2   {
3     "SKU": "4550621139529",
4     "Name": "Haikyu - Chira Badge Collection",
5     "Price": "162.00",
6     "Tag": "New",
7     "Category": "Goods",
8     "Image": "4550621139529.jpg",
9     "status": "active"
10   },
11   {
12     "SKU": "4636123842283",
13     "Name": "Haikyu Look Up Bokuto Kotaro Uniform Ver.",
14     "Price": "1620.00",
15     "Tag": "Pre-Order",
16     "Category": "Goods",
17     "Image": "4636123842283-3.jpg",
18     "status": "active"
19   },
20   {
21     "SKU": "4644815079777",
22     "Name": "Haikyu Mochi Mochi Mascot Vol.5 (Re-Sale)",
23     "Price": "2760.00",
24     "Tag": "Pre-Order",
25     "Category": "Goods",
26     "Image": "4644815079777-768x676.jpg",
27     "status": "active"
28 }

```

### ก. การใส่ข้อมูล (Insert)

- Testing register insert new admin with Picture

Method : POST

URL: <http://localhost:3000/register-submit>

Body: form-data

KEY	VALUES
AMFName	type:text JJ
AMLName	type:text JJ
AMEmail	type:text JJ@gmail.com
AMTelephone	type:text 083-000-0000

AMUsername	type:text	JJ
AMPassword	type:text	JJ
AMFPic	type:file	(use your local picture (.png only))

http://localhost:3000/register-submit

POST http://localhost:3000/register-submit

Body (8)

Key	Value	Description	Bulk Edit
AMFName	Text JJ		
AMLName	Text JJ		
AMEmail	Text JJ@gmail.com		
AMTelephone	Text 083-000-0000		
AMUsername	Text JJ		
AMPassword	Text JJ		
AMFPic	File cofj01gjld851.png		
Key	Text Value	Description	

Body Cookies Headers (8) Test Results

200 OK 37 ms 283 B Save Response

Pretty Raw Preview Visualize HTML

1 Registration successful

- Testing Add product

Method: POST

URL: <http://localhost:3000/add-product>

Body: form-data

Key	Value	Type
Editer	6687015	Text
PDSKU	00000000	Text
PDName	Mochi	Text
PDCategory	Goods	Text
PDTag	Haikyuu!!	Text
PDPrice	2500.00	Text
status	active	Text
picture	(use your local picture (.jpg only))	File

HTTP <http://localhost:3000/add-product>

POST <http://localhost:3000/add-product>

Params Authorization Headers (8) **Body** Scripts Tests Settings Cookies

None form-data x-www-form-urlencoded raw binary GraphQL

Key	Value	Description	Bulk Edit
<input checked="" type="checkbox"/> Editer	Text ▾ 6687015		
<input checked="" type="checkbox"/> PDSKU	Text ▾ 00000000		
<input checked="" type="checkbox"/> PDName	Text ▾ Mochi		
<input checked="" type="checkbox"/> PDCategory	Text ▾ Goods		
<input checked="" type="checkbox"/> PDTag	Text ▾ Haikyu!!		
<input checked="" type="checkbox"/> PDPrice	Text ▾ 2500.00		

Body Cookies Headers (8) Test Results 200 OK 5 ms 1.67 KB Save Response

Pretty Raw Preview Visualize JSON

```

1 [
2   {
3     "image": "1732881409208-349611180.jpg",
4     "SKU": "00000000",
5     "Name": "Mochi",
6     "Price": "2500.00",
7     "Tag": "Haikyu!!",
8     "Category": "Goods",
9     "status": "active"
10    },
11    {
12      "image": "4535123842283-3.jpg",
13      "SKU": "4535123842283-3",
14      "Name": "Hikyu Mochi",
15      "Price": "2500.00",
16      "Tag": "Hikyu!!",
17      "Category": "Popular Items",
18      "status": "active"
19    }
]

```

### ค. การปรับข้อมูลให้ทันสมัย (Update)

- Testing Edit product

Method: PUT

URL: [http://localhost:3000/update\\_product/4544815079777](http://localhost:3000/update_product/4544815079777)

Body: form-data

Key	Value	Type
Editer	6687015	Text
PDSKU	4544815079777	Text
PDName	Hikyu Mochi	Text
PDCategory	Popular Items	Text
PDTag	Hikyu!!	Text
PDPrice	2500.00	Text
status	active	Text

HTTP [http://localhost:3000/update\\_product/:4544815079777](http://localhost:3000/update_product/:4544815079777)

PUT [http://localhost:3000/update\\_product/:4544815079777](http://localhost:3000/update_product/:4544815079777)

**Body (8)**

Key	Value	Description
Editer	6687015	Text
PDSKU	4544815079777	Text
PDName	Hikyu Mochi	Text
PDCategory	Popular Items	Text
PDTag	Hikyuu!!	Text
PDPPrice	2500.00	Text
status	active	Text
picture	Haikuy-mochi.jpg	File

200 OK • 8 ms • 293 B

```
1 Product data updated successfully!
```

- Testing Edit admin

Method: PUT

URL: [http://localhost:3000/update\\_admin/1](http://localhost:3000/update_admin/1)

Body: form-data

Key	Value	Type
first_name	Sutarnthip	Text
last_name	Luangthip	Text
email	sutarnthip.lua@student.mahidol.edu	Text
tel	9876543210	Text
username	6687057	Text
password	123456	Text

The screenshot shows the Postman interface. At the top, it displays the URL `http://localhost:3000/update_admin/1`. Below the URL, the method is set to `PUT` and the target URL is `http://localhost:3000/update_admin/1`. The `Body` tab is selected, showing the following data:

Key	Type	Value	Description
first_name	Text	Sutarnthip	
last_name	Text	Luangthip	
email	Text	sutarnthip.lua@student.mahidol.edu	
tel	Text	9876543210	
username	Text	6687057	
password	Text	123456	
Key	Text	Value	Description

Below the body, the response section shows a `200 OK` status with a response message: `1 Admin data updated successfully`.

#### ๔. การลบข้อมูล (Delete)

- Testing Delete admin

Method: DELETE

URL: <http://localhost:3000/delete-admin/1>

HTTP <http://localhost:3000/delete-admin/1>

DELETE <http://localhost:3000/delete-admin/1>

Params Authorization Headers (6) Body Scripts Settings

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

This request does not have a body

Body Cookies Headers (8) Test Results | ⚡

Pretty Raw Preview Visualize JSON

```
1 {  
2   "msg": "Deleted"  
3 }
```

- Testing Delete product

Method: DELETE

URL: <http://localhost:3000/delete-product/4544815079777>

HTTP <http://localhost:3000/delete-product/:4544815079777>

DELETE <http://localhost:3000/delete-product/:4544815079777>

Params ● Authorization Headers (6) Body Scripts Tests Settings Cookies

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

This request does not have a body

HTTP <http://localhost:3000/delete-product/:4544815079777>

DELETE <http://localhost:3000/delete-product/:4544815079777>

Params ● Authorization Headers (6) Body Scripts Tests Settings Cookies

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

This request does not have a body

Body Cookies Headers (8) Test Results | ⚡

Pretty Raw Preview Visualize JSON

```
1 {  
2   "msg": "Deleted"  
3 }
```

## อ้างอิง

1.รูปภาพใน Home page phase1 เข้าถึงได้จาก: [Haikyu Mochi Mochi Mascot WA vol.1 – animate Bangkok Online Shop \(animatebkk-online.com\)](#)

2.รูปภาพใน Login page phase1 เข้าถึงได้จาก: [สื้นสุดการรอคอย! Haikyuu!! ประกาศสร้างฉบับภาพยนตร์อนิเมะจำนวน 2 พาร์ตในชี๊ \(thestandard.co\)](#)

3.Inspirational Design เข้าถึงได้จาก: [Product Management designs, themes, templates and downloadable graphic elements on Dribbble](#)

4.ตัวอย่างธุรกิจ เข้าถึงได้จาก: [animate Bangkok Online Shop – NO.1 MANGA AND ANIME STORE FROM JAPAN \(animatebkk-online.com\)](#), [GOODS | アニメ『ハイキュー!!』公式サイト \(haikyu.jp\)](#)

5.รูปภาพสินค้าต่างๆใน phase 2 เข้าถึงได้จาก: [animate Bangkok Online Shop – NO.1 MANGA AND ANIME STORE FROM JAPAN \(animatebkk-online.com\)](#)

6.Navigation Bar และ ICON ต่างๆ เข้าถึงได้จาก : <https://getbootstrap.com/docs/5.3/getting-started/introduction/> , <https://icons.getbootstrap.com/>

7. รูปภาพโลโก้ Haikyu เข้าถึงได้จาก :

<https://www.google.com/url?sa=i&url=https%3A%2F%2Flogos-world.net%2Fhaikyuu-logo%2F&psig=AOvVaw2GK7lgRRUcbc2UCH3HkyB&ust=1733038696918000&source=images&cd=vfe&opi=89978449&ved=0CBEOjRxqFwoTCIDknLHGg4oDFOAAAAAdAAAAABAE>