



PANNM OFFICIAL STORE

สมาชิก

6587002 แทนรัก ทองสมบูรณ์

6587019 นาถวัฒน์ เทพหัตดิน ณ อรุรยา

6587039 สิริวิษญ์ เหลืองไพฑูรย์

6587059 ปุณณพัฒน์ วลีสุขสันต์

6587102 ศุภณัฐ บรมสถิตย์

เสนอ

อาจารย์ ดร.จิตาภา ไกรสังข์

อาจารย์ ดร.วุฒิชชาติ แสงวงผล

รายงานเล่มนี้เป็นส่วนหนึ่งของวิชา ITDS241 Web Technologies and Applications

ภาคการเรียน 1 ปีการศึกษา 2566

มหาวิทยาลัยมหิดล

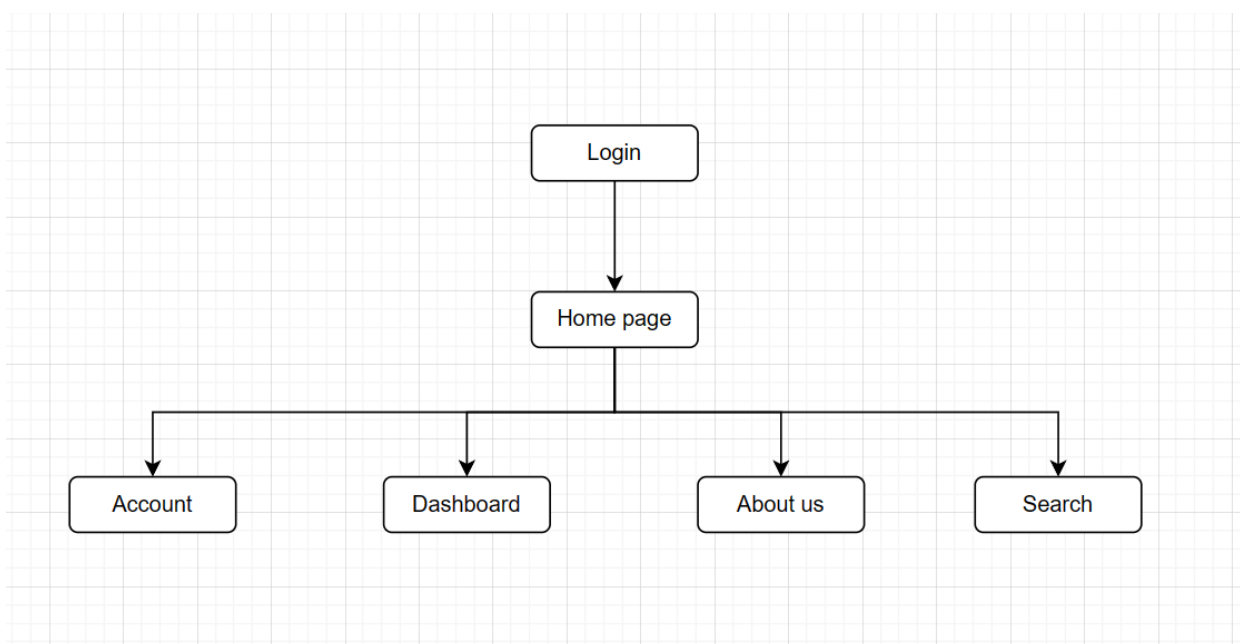
## สารบัญ

สารบัญ	2
ข้อมูลภาพรวมโครงการ	3
รายละเอียดหน้าเว็บต่าง ๆ ของผู้ดูแล	4-9
รายละเอียดเว็บเซอร์วิสและโค้ด	10
สรุปโครงสร้างเว็บไซต์	11-13
ผลการทดสอบ	14-20

## ข้อมูลภาพรวมโครงการ

- โครงการนี้ประกอบด้วยการพัฒนาเว็บแอปพลิเคชันซึ่งใช้งานบน Node.js และ MySQL สำหรับการจัดการฐานข้อมูล
- การเขียนโค้ดสำหรับเว็บแอปพลิเคชันนี้มีการใช้งาน Express.js ซึ่งเป็นเฟรมเวิร์กของ Node.js เพื่อจัดการกับการเรียกร้องจาก HTTP และการส่งคำตอบกลับไปยังผู้ใช้
- การออกแบบฐานข้อมูลและการสร้างคำสั่ง SQL เพื่อจัดการข้อมูล ใช้ MySQL ซึ่งเป็นระบบฐานข้อมูลที่มีความยืดหยุ่นและปลอดภัย

## แผนผังหน้าต่าง ๆ ของเว็บแอปพลิเคชัน



- Account: สำหรับการแสดงรายละเอียดบัญชีผู้ใช้
- Login : สำหรับเข้าสู่ระบบ
- Product: หน้าสำหรับการแสดงผลผลิตภัณฑ์
- Dashboard: หน้าแดชบอร์ดสำหรับการจัดการระบบของผู้ใช้
- About Us: หน้าเกี่ยวกับข้อมูลขององค์กรหรือโปรเจก
- Homepage: หน้าแรกของเว็บไซต์
- Search: หน้าสำหรับการค้นหาข้อมูลหรือผลิตภัณฑ์

## รายละเอียดหน้าเว็บต่าง ๆ ของผู้ดูแล

- หน้า Login: ให้ผู้ใช้สามารถเข้าสู่ระบบ

ข้อความจาก localhost:5050

Login successful

ตกลง

### PAN NM Login

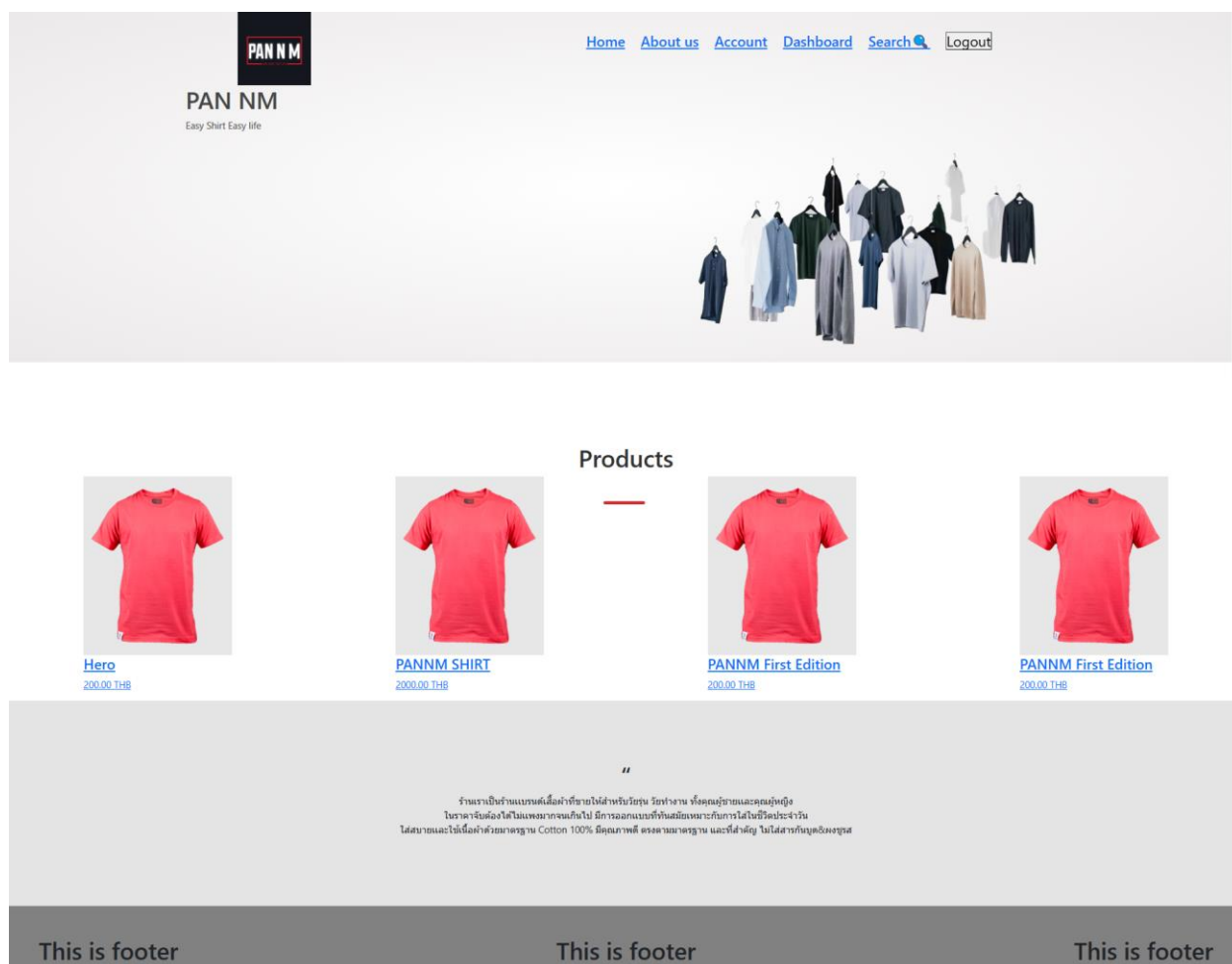
Username:

Password:

Login

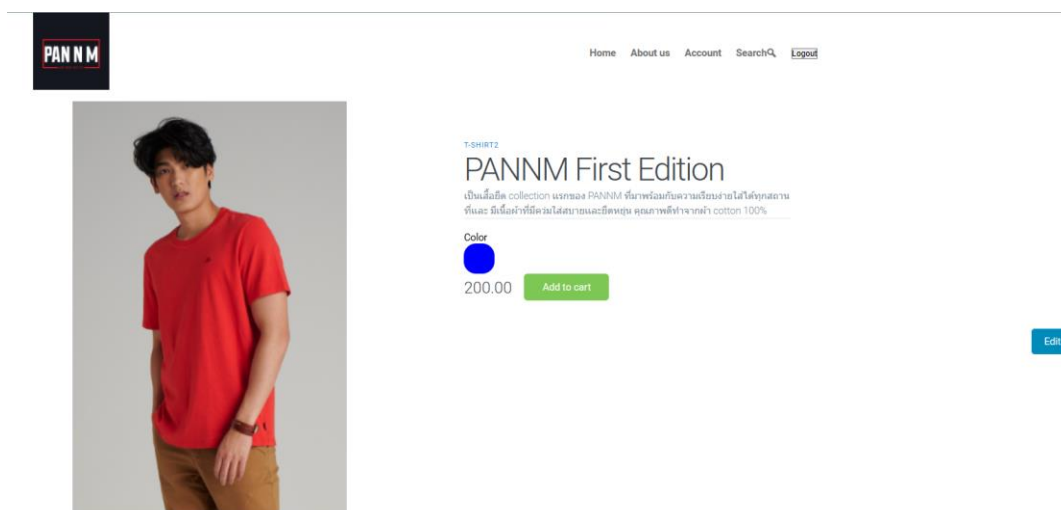
[ย้อนกลับ](#)

- หน้า Homepage: หน้าแรกที่เป็นจุดเริ่มต้นของผู้ใช้เมื่อเข้าสู่เว็บไซต์

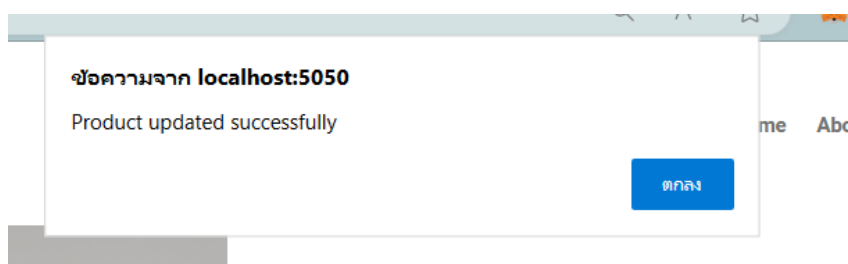


- ด้านบนของแต่ละหน้าจะมี Navigation bar สำหรับนำไปทางไปหน้าอื่นๆ
- จะมีสินค้าทั้งแสดงที่หน้านี้ และเมื่อกดก็สามารถเข้าไปดูสินค้าได้ที่หน้า Product

- หน้า Product: มีการแสดงรายการผลิตภัณฑ์ที่มีอยู่



- สามารถเลือกดู Product ได้และมีปุ่มให้กดสำหรับแก้ไข เมื่อกดจะแสดงผลดังนี้
- และเมื่อแก้ไขเสร็จกด edit อีกครั้งจะทำการบันทึกลงฐานข้อมูล



- หน้า Account: ให้ผู้ใช้สามารถดูข้อมูลส่วนตัวได้



Home About us Account Dashboard SearchQ Logout

### User Account

Name:

**AdminUser**

Email:

**admin@example.com**

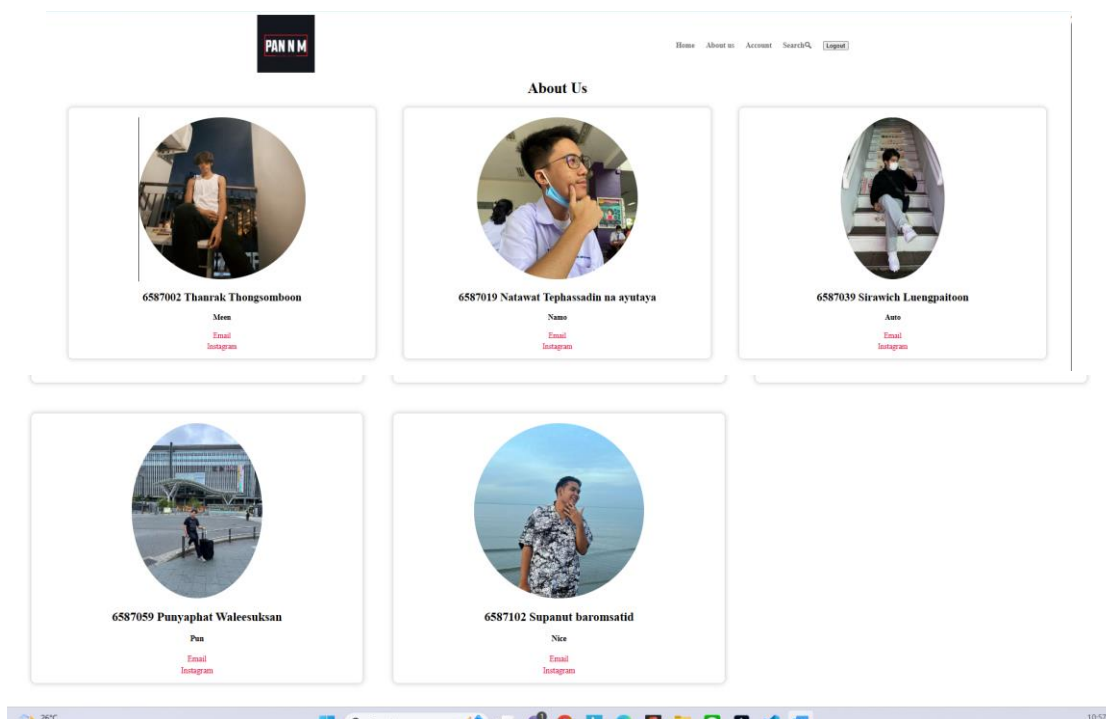
Phone:

**123456789**

Rank:

**Admin**

- หน้า About Us: ให้ข้อมูลเกี่ยวกับองค์กรหรือทีมพัฒนา



- หน้า Dashboard: สำหรับจัดการผู้ใช้

**Dashboard**

**Search**

**Add**

**Delete**

**Search**

**ผลการค้นหา**

**User ID: 1**  
**Username:** admin\_user  
**Email:** admin@example.com  
**Ranks:** Admin  
**Telephone:** 123456789  
**Firstname:** Admin  
**Lastname:** User

**User ID: 2**  
**Username:** john  
**Email:** asdsadad@gfasd  
**Ranks:** User  
**Telephone:** N/A  
**Firstname:** N/A  
**Lastname:** N/A

**User ID: 4**  
**Username:** Jonathan  
**Email:** jona@gmail.com  
**Ranks:** User  
**Telephone:** N/A  
**Firstname:** N/A  
**Lastname:** N/A

**Add**


- การค้นหาจะค้นหาด้วย Username ถ้าเว้นว่างจะเป็นการหาทั้งหมด
- การเพิ่มจะใช้ username password email
- การลบจะลบด้วย username

**Delete**




- หน้า Search: ให้ความสามารถในการค้นหาข้อมูลหรือผลิตภัณฑ์ตามคำค้นหาที่กำหนด

<input type="radio"/> เสื้อยืด <input type="radio"/> เสื้อแขนยาว <input type="radio"/> เสื้อแขนกุด	<input type="radio"/> \$0-\$2K <input type="radio"/> \$2K-\$4K <input type="radio"/> \$4K-\$6K	<input type="radio"/> Red <input type="radio"/> Black <input type="radio"/> White
--	--	---



**PANNM SHIRT**  
2000.00 THB

- โดยจะมีตัวกรอง คือ ชื่อสินค้า ชนิด ช่วง ราคา สี

 localhost:5050/result?name=&min=2000&max=4000&colorid=1&type=t-shirt

## รายละเอียดเว็บไซต์และโค้ด

### 0. โครงสร้างโปรเจต

Project/

- Backend/

- index.js

- package.json

- db.js

- .env

- database.sql

- query.sql

- public/

### 1. วิธีตั้งค่าฐานข้อมูล

- ไปที่โปรแกรมสำหรับ sql
- ทำการรันไฟล์ sql ตามลําดับเพื่อสร้างฐานข้อมูล
  - database.sql
  - query.sql

### 2. วิธีตั้งค่าโปรแกรม

- ลง nodeJs
- Unzip/clone project
- Cd เข้าไปที่ backend
- ถ้ายังไม่มี node\_module ให้ทำการพิมพ์ npm install
- ตั้งค่า .env ให้ตรงตามฐานข้อมูลตัวเอง

### 3. การรัน

- พิมพ์ node index.js

## สรุปโครงสร้างเว็บไซต์

Index.js จะเป็นไฟล์ backend ที่จะทำหน้าที่ต่างๆ โดยเริ่มต้นจะทำการ setup ค่าต่างๆ

```
const express = require('express')
const cookieParser = require('cookie-parser')
const cors = require("cors")
const http = require("http")
require("dotenv").config()
const bodyParser = require('body-parser');

const db = require('./db')
const app = express()
const PORT = process.env.PORT || 5000;

app.use(cors());
app.use(express.json());
app.use(bodyParser.urlencoded({ extended: true }));
app.use(cookieParser());
app.use(bodyParser.json());
app.use(express.static('public'));
```

- โดยหลักคือการตั้งค่า database จาก db.js
- ตั้ง port
- ตั้งตำแหน่งไฟล์ html

## Db.js

```
const mysql = require('mysql2');
require('dotenv').config();

const db = mysql.createConnection({
  host: process.env.MYSQL_HOST,
  user: process.env.MYSQL_USER,
  password: process.env.MYSQL_PASSWORD,
  database: process.env.MYSQL_DATABASE,
});

db.connect((err) => {
  if (err) {
    console.error('Error connecting to MySQL:', err);
  } else {
    console.log('Connected to MySQL');
  }
});

module.exports = db;
```

- ไฟล์นี้จะทำการเชื่อมต่อเข้ากับฐานข้อมูล mysql โดยจะใช้ข้อมูลจาก .env

## Index.js (ต่อ)

```

db.connect((err) => {
  if (err) {
    process.exit(1);
  } else {
    server.listen(PORT, () => {
      console.log("Server is listening on port localhost:${PORT}");
    });
  }
});

app.get('/api', (req, res) => {
  res.send("BACKEND RUNNING")
});

app.get('/', (req, res) => {
  res.sendFile(_dirname + '/public/homepage.html');
});

app.get('/:pagename', (req, res) => {
  const { pagename } = req.params;
  res.sendFile(`${_dirname}/public/${pagename}.html`, (err) => {
    if (err) {
      res.status(404).send('File not found');
    }
  });
});

```

- ฟังก์ชันแรกสำหรับทดสอบว่ารันถูกต้องไหมโดยไปทดสอบที่ localhost:5050/api
- / คือเมื่อเข้า url นี้จะทำการแสดง homepage.html

/:pagename คือเมื่อเข้า url เช่น /profile จะทำการแสดง /profile.html โดยจะต้องมีไฟล์ profile.html ใน public/profile.html เสมอ

```

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

  if (username || !password) {
    return res.status(400).json({ message: 'Please provide username and password' });
  }

  db.query('SELECT * FROM users WHERE username = ?', [username], (err, results) => {
    if (err) {
      return res.status(500).json({ message: 'Internal server error' });
    }

    if (results.length === 0) {
      return res.status(401).json({ message: 'Invalid username or password' });
    }

    const user = results[0];

    if (user.password !== password) {
      return res.status(401).json({ message: 'Invalid username or password' });
    }

    db.query(
      'INSERT INTO login_history (user_id, login_time) VALUES (?, ?)',
      [user.user_id, Datetime]
    );

    const userDetails = {
      user_id: user.user_id,
      username: user.username,
      email: user.email,
      ranks: user.ranks,
      telephone: user.telephone,
      firstnames: user.firstnames,
      lastname: user.lastname
    };

    res.status(200).json({ message: 'Login successful', user: userDetails });
  });
});

app.get('/api/users/search/:keyword?', (req, res) => {
  const { keyword } = req.params;

  // If no keyword is provided, retrieve all users
  if (!keyword) {
    db.query('SELECT * FROM users', (err, results) => {
      if (err) {
        return res.status(500).json({ message: 'Internal server error' });
      }

      res.status(200).json({ users: results });
    });
  } else {
    // If a keyword is provided, perform a search based on the keyword
    db.query('SELECT * FROM users WHERE username LIKE ?', [`%${keyword}%`], (err, results) => {
      if (err) {
        return res.status(500).json({ message: 'Internal server error' });
      }

      res.status(200).json({ users: results });
    });
  }
});

> app.post('/api/users', (req, res) => { ...
});

> app.delete('/api/users/:username', (req, res) => { ...
});

```

- Api สำหรับ login create search delete user โดยจะใช้การรับค่าและเรียกโดย sql

```
app.put('/api/products/:productId', (req, res) => {
  const { productId } = req.params;
  const { product_name, description, price, color_id, type_product } = req.body;

  db.query(
    'UPDATE products SET product_name = ?, description = ?, price = ?, color_id = ?, type_product = ? WHERE product_id = ?'
    [product_name, description, price, color_id, type_product, productId],
    (err, results) => {
      if (err) {
        return res.status(500).json({ message: 'Failed to update product' });
      }

      if (results.affectedRows === 0) {
        return res.status(404).json({ message: 'Product not found' });
      }

      res.status(200).json({ message: 'Product updated successfully' });
    }
  );
});
```

```
app.delete('/api/products/:productId', (req, res) => {
  const { productId } = req.params;

  db.query('DELETE FROM products WHERE product_id = ?', [productId], (err, results) => {
    if (err) {
      return res.status(500).json({ message: 'Failed to delete product' });
    }

    if (results.affectedRows === 0) {
      return res.status(404).json({ message: 'Product not found' });
    }

    res.status(200).json({ message: 'Product deleted successfully' });
  });
});
```

```
app.get('/api/products/:productId', (req, res) => {
  const { productId } = req.params;

  if (!productId) {
    db.query('select *, colors.color_name color from products join colors on products.color_id = color_id', (err, results) => {
      if (err) {
        return res.status(500).json({ message: 'Internal server error' });
      }

      res.status(200).json({ products: results });
    });
  } else {
    db.query('select *, colors.color_name color from products join colors on products.color_id = color_id where product_id = ?', [productId], (err, results) => {
      if (err) {
        return res.status(500).json({ message: 'Internal server error' });
      }

      if (results.length === 0) {
        return res.status(404).json({ message: 'Product not found' });
      }

      res.status(200).json({ product: results[0] });
    });
  }
});
```

```
// ...

app.get('/api/filter', (req, res) => {
  const { name, min, max, colorid, type } = req.query;

  // Build the SQL query dynamically based on the provided parameters
  let query = 'SELECT * FROM products WHERE 1 = 1';
  const params = [];

  if (name) {
    query += ' AND product_name LIKE ?';
    params.push(`%${name}%`);
  }

  if (min) {
    query += ' AND price >= ?';
    params.push(parseFloat(min));
  }

  if (max) {
    query += ' AND price <= ?';
    params.push(parseFloat(max));
  }

  if (colorid) {
    query += ' AND color_id = ?';
    params.push(parseInt(colorid));
  }

  if (type) {
    query += ' AND type_product = ?';
    params.push(type);
  }

  db.query(query, params, (err, results) => {
    if (err) {
      return res.status(500).json({ message: 'Internal server error' });
    }

    res.status(200).json({ products: results });
  });
});
```

- สามฟังก์ชันนี้สำหรับ แก้ไข เพิ่ม ลบ ข้อมูลสินค้าโดยการค้นหาจะมีทั้งค้นหาทั้งหมดและค้นหาด้วย การกรอง

## การทดสอบ

### 1. Login

Method : post

url : <http://localhost:5050/api/login>

request :

```
{
  "username": "admin_user",
  "password": "admin123"
}
```

Response 200:

```
{
  "message": "Login successful",
  "user": {
    "user_id": 1,
    "username": "admin_user",
    "email": "admin@example.com",
    "ranks": "Admin",
    "telephone": "123456789",
    "firstname": "Admin",
    "lastname": "User"
  }
}
```

Response 401

```
1 {
2   "message": "Invalid username or password"
3 }
```

## 2. Search user

Method : get

url : <http://localhost:5050/api/users/search/>

url2: [http://localhost:5050/api/users/search/admin\\_user](http://localhost:5050/api/users/search/admin_user) ค้นหาด้วย username

request :

Response 200:

```

{
  "users": [
    {
      "user_id": 1,
      "username": "admin_user",
      "password": "admin123",
      "email": "admin@example.com",
      "ranks": "Admin",
      "telephone": "123456789",
      "firstname": "Admin",
      "lastname": "User"
    },
    {
      "user_id": 2,
      "username": "john",
      "password": "1234",
      "email": "asdsadasd@gfasd",
      "ranks": "User",
      "telephone": null,
      "firstname": null,
      "lastname": null
    }
  ]
}

```

Response 200 not found

```

1 {
2   "users": []
3 }

```

## 3. Add user

Method : post

url : <http://localhost:5050/api/users>

request :

```
1  {
2    "username": "Hellomama",
3    "password": "1150",
4    "email": "a2sd34@gmail.com",
5    "telephone": "065424118754",
6    "firstname": "John",
7    "lastname": "Doeee"
8  }
```

Response 200:

```
{
  "message": "User created successfully"
}
```



#### 4. Delete user

Method : delete

url : <http://localhost:5050/api/users/Hellomama>

request :

Response 200:

```
{
  "message": "User deleted successfully"
}
```

Response 404

```
1 {
2   "message": "User not found"
3 }
```

## 5. Update user

Method : put

url : <http://localhost:5050/api/users/> [users id]

request :

```
1 { "username": "adbig",  
2   "password": "123",  
3   "email": "admin@example.com",  
4   "ranks": "Admin",  
5   "telephone": "123456789",  
6   "firstname": "adbig",  
7   "lastname": "Admin" }
```

Response 200:

```
1 {  
2   "message": "User updated successfully"  
3 }
```

Response 404

```
1 {  
2   "message": "User not found"  
3 }
```

Response 500

```
1 {  
2   "message": "Failed to update user"  
3 }
```

## 6. Search all Product

Method : get

url : http://localhost:5050/api/products

request :

Response 200:

```

1  {
2    "products": [
3      {
4        "product_id": 1,
5        "product_name": "Hero",
6        "description": "เป็นเสื้อยืด collection แรกของ PANNM ที่มาพร้อมกับความเรียบง่ายใส่ได้ทุกสถานที่และ
          มีเนื้อผ้าที่มีส่วนผสมของใยคอตตอนคุณภาพดีทำจากผ้า cotton 100%",
7        "price": "200.00",
8        "color_id": 1,
9        "type_product": "t-shirt",
10       "image": null,
11       "color_name": "Red",
12       "color": "Red"
13     },
14     {
15       "product_id": 4,
16       "product_name": "PANNM SHIRT",
17       "description": "เป็นเสื้อยืด collection แรกของ PANNM ที่มาพร้อมกับความเรียบง่ายใส่ได้ทุกสถานที่และ
          มีเนื้อผ้าที่มีส่วนผสมของใยคอตตอนคุณภาพดีทำจากผ้า cotton 100%",
18       "price": "2000.00",

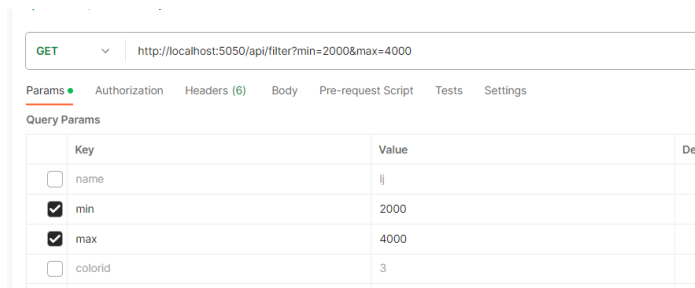
```

## 7. Search Product by filter

Method : get

url : http://localhost:5050/api/filter?min=2000&amp;max=4000&amp;colorid=3

request :



Key	Value	De
<input type="checkbox"/> name		
<input checked="" type="checkbox"/> min	2000	
<input checked="" type="checkbox"/> max	4000	
<input type="checkbox"/> colorid	3	

Response 200:

```

✓ 200
✓ {
  "products": [
    {
      "product_id": 4,
      "product_name": "PANNM SHIRT",
      "description": "เป็นเสื้อยืด collection แรกของ PANNM ที่มาพร้อมกับความเรียบง่ายใส่ได้
        มีเนื้อผ้าที่มีความใส่สบายและยืดหยุ่น คุณภาพดีทำจากผ้า cotton 100%",
      "price": "2000.00",
      "color_id": 1,
      "type_product": "t-shirt",
      "image": null
    }
  ]
}

```

Response 404

```

1 {
2   "message": "No products found with the provided filters"
3 }

```

## 8. Update product

Method : put

url : <http://localhost:5050/api/products/> [product id]

request :

```
1 {  
2   "product_name": "Cotton T-Shirt",  
3   "description": "Comfortable and breathable cotton shirt",  
4   "price": 19.99,  
5   "color_id": 1,  
6   "type_product": "T-Shirt"  
7 }  
8
```

Response 200:

```
Pretty Raw Preview Visualize JSON ↕  
1 {  
2   "message": "Product updated successfully"  
3 }
```

Response 404

```
1 {  
2   "message": "Product not found"  
3 }
```

Response 500

```
1 {  
2   "message": "Failed to update product"  
3 }
```