



## โครงการเขียนโปรแกรมเชิงอ็อบเจกต์และโครงสร้างข้อมูล

### เรื่อง แม่ศรีเรือน (ระบบช่วยในการจ่ายตลาด)

#### จัดทำโดย

- |                            |          |
|----------------------------|----------|
| 1. นายปัญญพนต์ นครชัย      | B6601492 |
| 2. นางสาวปริญานุช จิตรโคตร | B6601829 |
| 3. นายอพิเชษฐ์ อุตส่าห์    | B6602451 |
| 4. นายธนดิษฐ์ เกิดมะเรียง  | B6626587 |
| 5. นายเอกกวี จากโคกสูง     | B6642518 |

#### เสนอ

รองศาสตราจารย์ ดร.จิตินันต์ อังสกุล  
ผู้ช่วยศาสตราจารย์ ดร.ศุภกฤษณ์ นิวัฒนากุล  
อาจารย์ ดร.ธรรมศักดิ์ เขียวนิเวศน์

รายงานนี้เป็นส่วนหนึ่งของรายวิชา 1101093

โครงการเขียนโปรแกรมเชิงอ็อบเจกต์และโครงสร้างข้อมูล

ภาคการศึกษาที่ 2 ปีการศึกษา 2567

สาขาวิชาเทคโนโลยีดิจิทัล สำนักวิทยาศาสตร์และศิลปดิจิทัล  
มหาวิทยาลัยเทคโนโลยีสุรนารี

## รายละเอียดการแบ่งงานภายในกลุ่ม

| รหัสนักศึกษา | ชื่อ - นามสกุล          | หัวข้อที่รับผิดชอบ              | ร้อยละ |
|--------------|-------------------------|---------------------------------|--------|
| B6601492     | นายปัญญาพนธ์ นครชัย     | ช่วยทำ Class Diagram            | 5%     |
| B6601829     | นางสาวปริยานุช จิตรโคตร | ออกแบบ UX/UI + Class Diagram    | 30%    |
| B6602451     | นายอพิเชษฐ์ อุตส่าห์    | พัฒนาโปรแกรม                    | 35%    |
| B6626587     | นายธนดิษฐ์ เกิดมะเร็ง   | ออกแบบ UX/UI + ช่วยพัฒนาโปรแกรม | 25%    |
| B6642518     | นายเอกกวี จากโคกสูง     | ช่วยเขียน Code                  | 5%     |
| รวม 100%     |                         |                                 | 100%   |

## แม่ศรีเรือน (ระบบช่วยในการจ่ายตลาด)

### ที่มาและความสำคัญ

ในชีวิตประจำวันการจ่ายตลาดเป็นกิจกรรมที่ต้องทำเป็นประจำของคนทั่วไปและผู้ประกอบการ ไม่ว่าจะเป็นการซื้ออาหารสด ผลไม้ เนื้อสัตว์ เครื่องปรุง หรือสินค้าอื่น ๆ เพื่อใช้ในการประกอบอาหาร หลายครั้งการจัดการรายการสินค้าที่ต้องซื้อจำเป็นต้องมีการวางแผนการซื้อก่อนที่จะซื้อสินค้า โดยเฉพาะในกรณีที่เรเป็นผู้ประกอบการ การดูแลในเรื่องของการจัดซื้อและการจัดการรายการสินค้าถือเป็นสิ่งสำคัญมาก เพราะเป็นเรื่องของกำไรขาดทุนของกิจการ รวมไปถึงเรื่องการควบคุมงบประมาณที่ต้องมีการจัดการอย่างดี ไม่เพียงแต่ผู้ประกอบการเท่านั้น คนทั่วไปเองก็ต้องการการวางแผนการซื้อและการกำหนดงบประมาณในแต่ละครั้ง

โดยทั่วไป ผู้คนมักใช้วิธีการจดบันทึกลงบนกระดาษหรือในแอปพลิเคชันทั่วไปในการจดบันทึกรายการสินค้า แต่แอปพลิเคชันเหล่านั้นไม่ได้ออกแบบมาเพื่อการจัดการการจ่ายตลาดโดยเฉพาะ ทำให้ไม่สามารถติดตามค่าใช้จ่ายได้อย่างเต็มที่และแม่นยำ นอกจากนี้ การใช้กระดาษในการจดบันทึกอาจทำให้เกิดความยุ่งยากในการค้นหาและจัดเก็บข้อมูล อีกทั้งยังไม่สามารถให้การวิเคราะห์ข้อมูลการใช้จ่ายได้อย่างมีประสิทธิภาพ

คณะผู้จัดทำได้พัฒนาโปรแกรม"แม่ศรีเรือน" ขึ้นเพื่อตอบสนองความต้องการในการจัดการรายการสินค้าที่ต้องซื้อและการควบคุมงบประมาณอย่างมีประสิทธิภาพ โดยเน้นการใช้งานที่สะดวกสบายและเหมาะสมกับการจ่ายตลาดในชีวิตประจำวัน แอปนี้จะช่วยให้ผู้ใช้สามารถสร้างและจัดการรายการซื้อสินค้า ติดตามค่าใช้จ่าย และวิเคราะห์การใช้จ่ายได้อย่างเป็นระบบและมีประสิทธิภาพมากยิ่งขึ้น

### วัตถุประสงค์

1. เพื่อออกแบบและพัฒนาโปรแกรมการจัดการรายการซื้อสินค้า
2. เพื่อออกแบบการควบคุมและติดตามงบประมาณการจ่ายตลาด
3. เพื่อส่งเสริมทักษะการวางแผนการซื้อสินค้าให้แก่ผู้ใช้
4. เพื่อจัดทำระบบสรุปค่าใช้จ่ายและจัดหมวดหมู่การใช้จ่ายและติดตามการใช้จ่าย

## ขอบเขตของโครงการ

- ขอบเขตของโปรแกรมจะครอบคลุม การใช้จ่ายในการซื้อสินค้า เมนูต่างๆในระบบการดูการใช้จ่าย การสรุปยอดการใช้จ่ายของการซื้อสินค้า
- โปรแกรมนี้ต้องเข้าสู่โปรแกรมเพื่อใช้งานโดยจะแบ่งออกเป็น 3 ผู้ใช้ได้แก่ บุคคลทั่วไป ผู้ประกอบการ จะตรวจสอบสิทธิ์เหล่านี้ด้วยระบบ login เพื่อเข้าใช้งาน
- **ผู้ใช้งานทั่วไป** เป็นผู้ใช้ที่สามารถเรียกใช้งานได้ทุกระบบ ไม่ว่าจะเป็น การสร้างและจัดการรายการ การดูสรุปยอดการใช้จ่าย การดูประวัติสินค้าที่ซื้อ การตั้งงบประมาณในการซื้อสินค้า
- **ระบบสมาชิก** เป็นผู้ใช้ที่สามารถเรียกใช้งานได้ทุกระบบ ไม่ว่าจะเป็น การสรุปและวิเคราะห์การใช้จ่ายในแต่ละเดือน ประวัติการสั่งซื้อสินค้า และสามารถดูรายงานการใช้จ่ายแบบแยกตามหมวดหมู่สินค้า
- **เมนูหลักของระบบ**
  1. สร้างรายการซื้อสินค้า
  2. จัดการงบประมาณ
  3. ดูสรุปการใช้จ่าย
  4. ประวัติการซื้อ
  5. ตั้งค่าผู้ใช้งาน
  6. Input File
  7. เช็ครายการ

### 1. เมนูสร้างรายการซื้อสินค้า

**ฟังก์ชัน :** ผู้ใช้สามารถสร้างรายการซื้อสินค้าโดยระบุซื้อสินค้า จำนวน และราคาของแต่ละรายการที่จะซื้อ

**รายละเอียด :** ผู้ใช้สามารถเพิ่มหรือลบสินค้าที่ต้องการซื้อได้ตามต้องการ ในระบบจะมีการคำนวณยอดรวมของสินค้าที่ซื้อไปแล้วในขณะนั้น พร้อมทั้งสามารถแสดงรายการสินค้าที่ได้เพิ่มไว้เพื่อให้สะดวกต่อการตรวจสอบก่อนออกไปซื้อของ

## 2. เมนูจัดการงบประมาณ

**ฟังก์ชัน :** กำหนดวงเงินงบประมาณสำหรับการซื้อสินค้าในครั้งนั้น

**รายละเอียด :** ผู้ใช้สามารถตั้งงบประมาณการใช้จ่ายในแอปได้ เช่น งบประมาณสำหรับการซื้อของ 2,000 บาท ระบบจะหักลบตามที่ใช้ซื้อของ เมื่อยอดใช้จ่ายใกล้ถึงงบประมาณที่ตั้งไว้

## 3. เมนูดูสรุปการใช้จ่าย

**ฟังก์ชัน :** ให้ผู้ใช้ดูภาพรวมของการใช้จ่าย

**รายละเอียด :** สามารถดูการใช้จ่ายทั้งในระยะสั้น (รายวัน) หรือระยะยาว (รายสัปดาห์/รายเดือน) โดยสรุปการซื้อสินค้าต่างๆ พร้อมทั้งสามารถดูได้ว่าใช้จ่ายไปในหมวดหมู่ใดบ้าง เช่น ผัก ผลไม้ เนื้อสัตว์ เป็นต้น

## 4. ประวัติการซื้อ

**ฟังก์ชัน :** แสดงประวัติการซื้อสินค้าที่เคยซื้อไป

**รายละเอียด :** ผู้ใช้สามารถดูรายละเอียดของการซื้อสินค้าทั้งหมดที่ทำในแอป รวมถึงวันที่ซื้อ จำนวนสินค้า และราคาที่จ่ายไปในแต่ละครั้ง

## 5. เมนูตั้งค่าผู้ใช้งาน

**ฟังก์ชัน :** ตั้งค่าความสามารถของผู้ใช้งาน

**รายละเอียด :** ในฟังก์ชันนี้ผู้ใช้สามารถจัดการผู้ใช้งานให้เหมาะสมกับการใช้งานทั้ง 3 ระดับ และมีผู้ดูแลระบบสามารถแก้ไขปัญหา และรายละเอียดของผู้ใช้งานอื่นๆได้

## 6. ฟังก์ชันการ Input

**รายละเอียด :** ฟังก์ชันนี้จะทำหน้าที่ อ่านค่าแบบไฟล์ txt ที่เราได้มีการ input เข้ามา โดยมีเงื่อนไขในการระบุ “ราคา:” มันจะคำนวณยอดการใช้จ่ายทั้งหมดของรายการเราแบบ Input ไฟล์เข้าให้เลย

## 7. เมนูการเช็ครายการ

**รายละเอียด :** ฟังก์ชันนี้จะทำการค้นหารายการสินค้าที่เราต้องการเช็คที่เราได้มีการซื้อ รายการสินค้านั้นครบตามที่เรต้องการ ถ้ามีรายการไหนที่เราซื้อไม่ครบมันจะคอยแจ้งเตือนเรา และยังสามารถแก้ไขจำนวน ราคาได้

## ประโยชน์ที่คาดว่าจะได้รับ

1. สามารถพัฒนาโปรแกรมที่ช่วยในการจัดซื้อสินค้าที่ง่ายต่อการใช้งานแก่ผู้ใช้
2. สามารถสรุปค่าใช้จ่ายและจัดหมวดหมู่การใช้จ่ายและติดตามการใช้จ่ายได้อย่างมีประสิทธิภาพ
3. สร้างทักษะในการทำงานในด้านการพัฒนาโปรแกรม การแก้ไขปัญหา อย่างมีประสิทธิภาพ
4. สร้างทักษะทำให้สามารถนำองค์ความรู้ที่ได้รับจากการทำโครงการไปต่อยอดได้ในอนาคต

อนาคต

## ความคิดสร้างสรรค์

1. เพิ่มฟีเจอร์จัดการงบประมาณที่เหมาะสมให้ผู้ใช้ โดยคำนวณจากรายการสินค้าและหมวดหมู่ที่กำหนดไว้
2. เพิ่มความสามารถในการบันทึกรายการซื้อไว้เป็นรายการ เพื่อให้ผู้ใช้สามารถเรียกใช้ซ้ำได้สะดวกในครั้งถัดไป
3. ออกแบบระบบแจ้งเตือนเมื่อยอดใช้จ่ายใกล้ถึงงบประมาณที่ตั้งไว้ เพื่อช่วยผู้ใช้ควบคุมค่าใช้จ่าย

## ผลกระทบ

### ผลกระทบต่อผู้ใช้

- ช่วยลดความยุ่งยากในการวางแผนการจ่ายตลาด ทำให้การจัดการค่าใช้จ่ายมีประสิทธิภาพมากขึ้น
- ผู้ใช้สามารถติดตามพฤติกรรม การซื้อสินค้าและควบคุมงบประมาณได้ดีขึ้น

### ผลกระทบต่อการเรียนรู้

- ทีมผู้จัดทำได้พัฒนาทักษะการเขียนโปรแกรม การออกแบบระบบ และการแก้ปัญหาจริงจากการทำโครงการ
- เป็นโครงการที่สามารถต่อยอดไปสู่การพัฒนาแอปพลิเคชันอื่น ๆ ในอนาคตได้

### ผลกระทบต่อสิ่งแวดล้อม

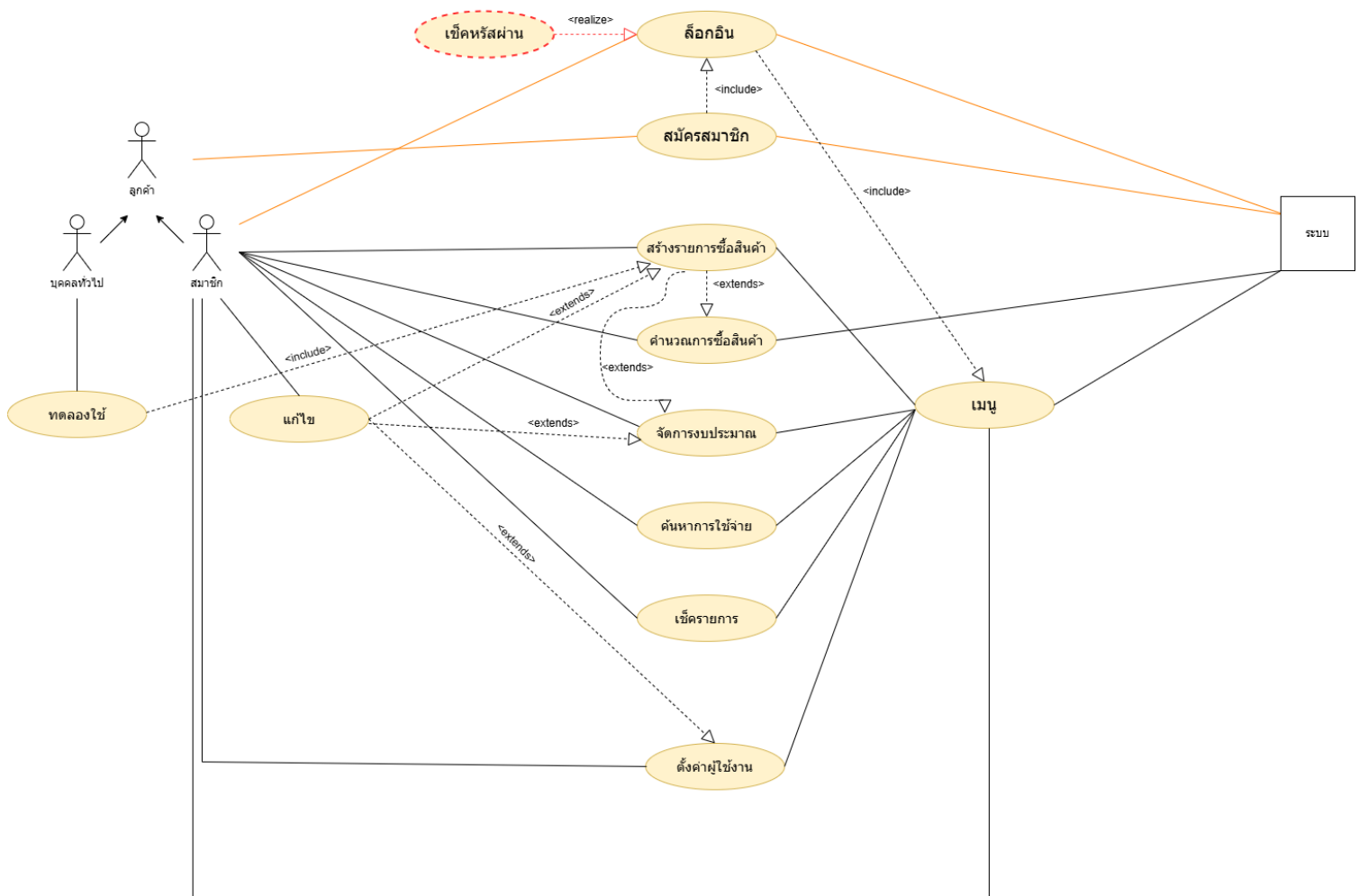
- ลดการใช้กระดาษในการจดบันทึก ทำให้เป็นมิตรต่อสิ่งแวดล้อม

## System Specification

- บริการหลักคือ ระบบที่ออกแบบมาเพื่อช่วยให้ผู้ใช้งานจัดการและติดตามการจ่ายตลาดได้ง่ายขึ้น โดยมีฟีเจอร์หลัก เช่น การจดบันทึกรายการ การกำหนดงบประมาณ และการสรุปค่าใช้จ่าย ระบบนี้รองรับทั้งผู้ใช้งานทั่วไปและสมาชิกที่ลงทะเบียน โดยมอบฟังก์ชันเพิ่มเติมสำหรับสมาชิก
- ผู้ใช้งานแบ่งเป็น 2 ประเภทได้แก่ บุคคลทั่วไป และผู้ใช้งานระบบสมัครสมาชิก
- บุคคลทั่วไป: ใช้งานฟังก์ชันพื้นฐาน เช่น การสร้างรายการ การคำนวณยอดสินค้าทั้งหมด สำหรับการจ่ายตลาดแต่ไม่สามารถดูสรุปยอดการใช้จ่ายในแต่ละวันเดือนปีได้
- ผู้ใช้งานที่เป็นสมาชิก จะใช้งานฟังก์ชันต่างๆ ในโปรแกรมได้ทั้งหมด และสามารถดูสรุปยอดการใช้จ่ายของตัวเองในแต่ละเดือนได้
- ฟังก์ชันการบันทึกรายการ ผู้ใช้งานสามารถใช้ได้ทั้งสอง โดยฟังก์ชันนี้จะเป็นการบันทึกรายการที่เราต้องการไปจ่ายตลาด ผู้ใช้สามารถ คำนวณราคาสินค้า ตอนที่เราไปจ่ายตลาดได้ และยังป้อนจำนวนสำหรับการคำนวณแบบภาษี
- ฟังก์ชันการกำหนดงบประมาณ ผู้ใช้สามารถกำหนดงบประมาณสำหรับการจ่ายตลาดแต่ละครั้ง ระบบจะแสดงค่าใช้จ่ายรวมที่เกิดขึ้นเทียบกับงบประมาณแบบเรียลไทม์ ไม่ให้ผู้ใช้เมื่อค่าใช้จ่ายรวมเกินงบประมาณและยังป้อนจำนวนสำหรับการคำนวณแบบภาษี
- ฟังก์ชันสรุปยอดการใช้จ่าย โดยผู้ใช้สามารถค้นหารายการที่ผู้ใช้บันทึกเสร็จแล้วได้ สมาชิกที่ลงทะเบียนสามารถดูสรุปยอดค่าใช้จ่ายที่แบ่งตาม รายวัน, รายสัปดาห์, และรายเดือน ในฟังก์ชันนี้จะมีการจัดเรียงรายการ เช่น จากจำนวนมากไปน้อย การเรียงรายการไหนล่าสุดหรือเก่าสุด และการจัดเรียงรายการตามตัวอักษร
- ฟังก์ชันการเช็คสินค้า โดยจะมีการค้นหารายการสินค้าที่เราได้มีการสร้างไว้แล้วใน ฟังก์ชันการจดบันทึกโดยมันจะเรียก จำนวนสินค้าต่างๆ มาแสดง เราก็เช็คว่ารายการของเราขาดอะไรหรือซื้อครบ? นอกจากนี้ยังสามารถแก้ไข จำนวน ราคาได้

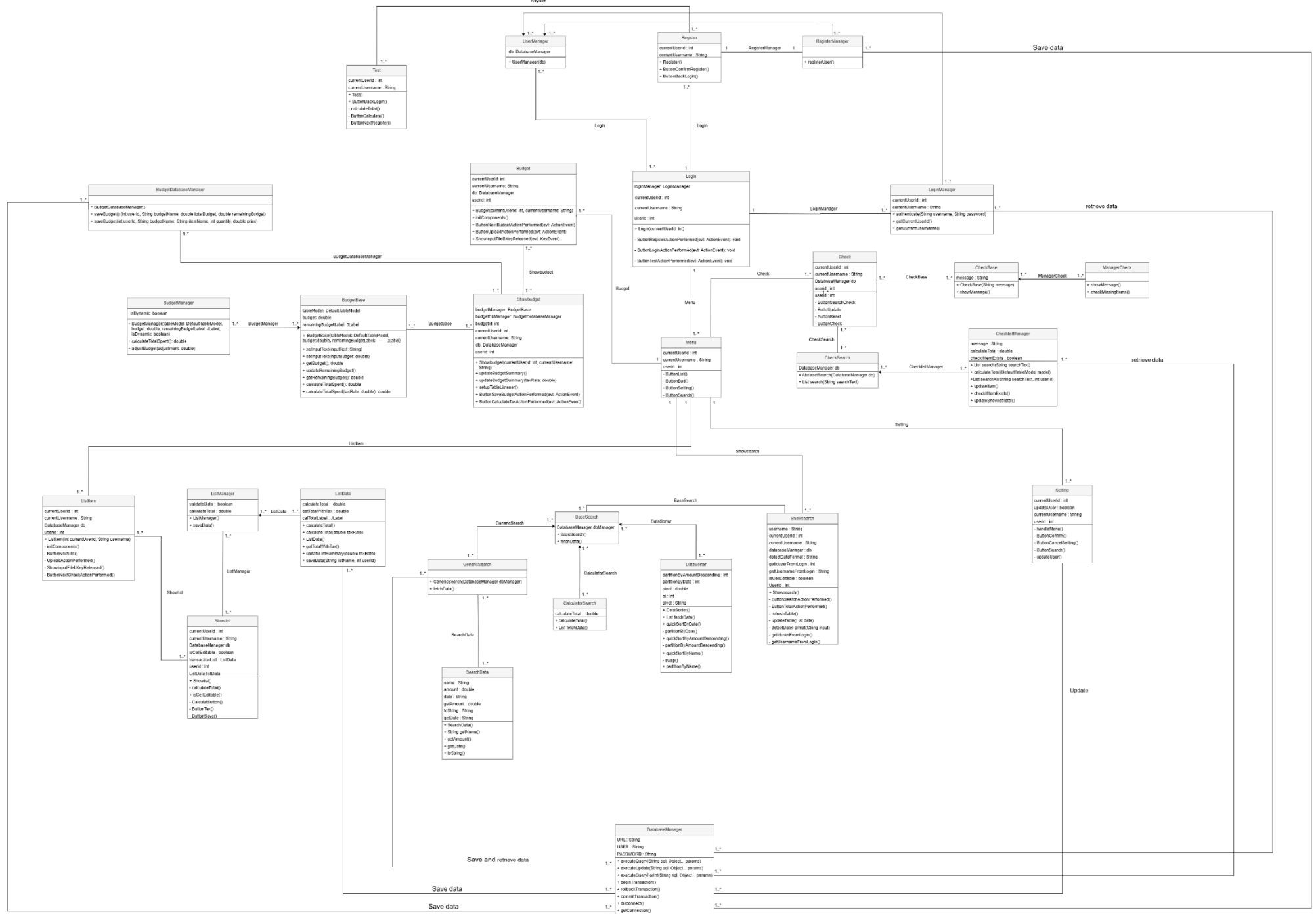
## Use Case

Use Case  
แม่ศรีเรือน (ระบบช่วยในการจ่ายตลาด)





## Class Diagram

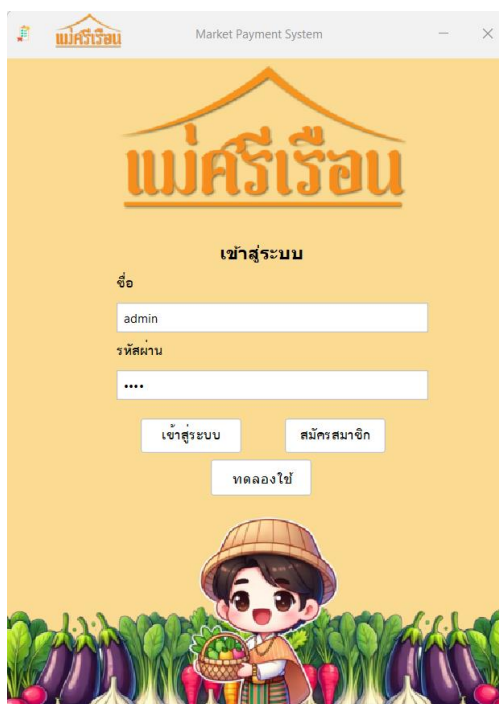


คู่มือการใช้งานโปรแกรม ระบบช่วยในการจ่ายตลาด (แม่ศรีเรือน) Market Payment System Program ในรายวิชา 1101093 การเขียนโปรแกรมเชิงอ็อบเจกต์และโครงสร้างข้อมูล

สาธิตการติดตั้งการใช้งาน : [Click Here](#)

ดาวน์โหลดโปรแกรม : [Click Here](#)

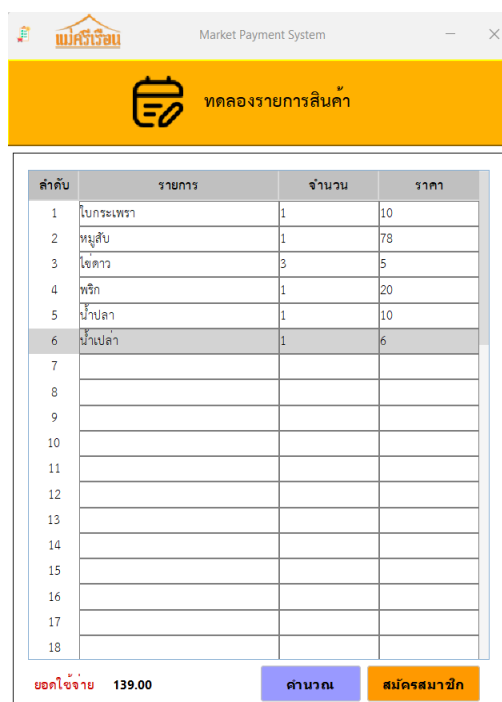
## 1. หน้าสมัครสมาชิกและระบบล็อกอิน



ภาพที่ 1 หน้า Homepage

คำอธิบาย : ผู้ที่ใช้ครั้งแรกผู้ใช้สามารถทดลองใช้งานได้ก่อนสมัครสมาชิก ต่อมา ผู้ใช้สามารถสมัครได้โดยกดไปที่สมัครสมาชิกเพื่อทำการสร้างชื่อผู้ใช้และรหัสผ่านถ้าผู้ใช้มีชื่อและรหัสอยู่แล้ว สามารถใส่ชื่อและรหัสผ่านแล้วก็กดเข้าสู่ระบบเพื่อเริ่มใช้งานได้

## 1.1 หน้าทดลองใช้งาน



The screenshot shows a web application titled "Market Payment System" with a logo for "แม่ศรีเรือน". The main heading is "ทดลองรายการสินค้า" (Test Item List). Below this is a table with 4 columns: ลำดับ (Sequence), รายการ (Item), จำนวน (Quantity), and ราคา (Price). The table contains 18 rows. The first 6 rows are pre-filled with data, and the remaining 12 rows are empty for input. At the bottom left, it shows "ยอดใช้จ่าย 139.00". At the bottom right, there are two buttons: "คำนวณ" (Calculate) and "สมัครสมาชิก" (Join Membership).

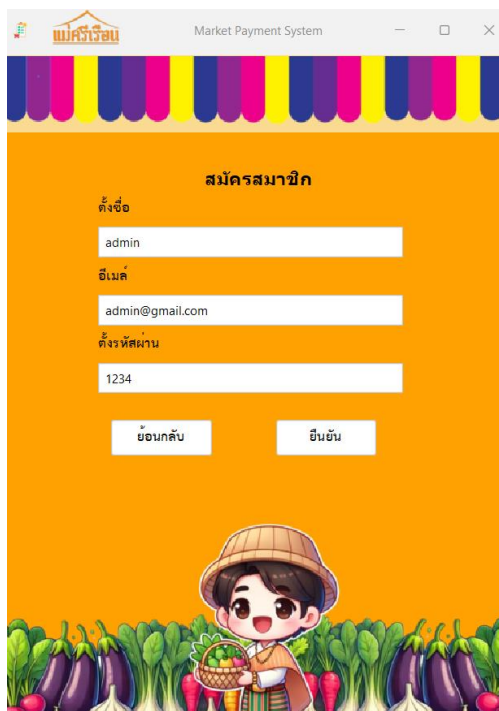
| ลำดับ | รายการ    | จำนวน | ราคา |
|-------|-----------|-------|------|
| 1     | ใบกระเพรา | 1     | 10   |
| 2     | หมูสับ    | 1     | 78   |
| 3     | ไข่ดาว    | 3     | 5    |
| 4     | พริก      | 1     | 20   |
| 5     | น้ำปลา    | 1     | 10   |
| 6     | น้ำเปล่า  | 1     | 6    |
| 7     |           |       |      |
| 8     |           |       |      |
| 9     |           |       |      |
| 10    |           |       |      |
| 11    |           |       |      |
| 12    |           |       |      |
| 13    |           |       |      |
| 14    |           |       |      |
| 15    |           |       |      |
| 16    |           |       |      |
| 17    |           |       |      |
| 18    |           |       |      |

ยอดใช้จ่าย 139.00    **คำนวณ**    **สมัครสมาชิก**

ภาพที่ 2 หน้าทดลองการใช้งาน

คำอธิบาย : เมื่อกดที่ทดลองใช้ ผู้ใช้สามารถสร้างรายการ จำนวนรายการและราคาได้คร่าวๆ และสามารถคำนวณราคาได้ด้วยการกดที่คำนวณว่าคำนวณ ถ้าทดลองใช้เสร็จแล้วสามารถใช้งานตัวเต็มโดยการกดสมัครสมาชิก

## 1.2 หน้าสมัครสมาชิก



Market Payment System

**สมัครสมาชิก**

ตั้งชื่อ  
admin

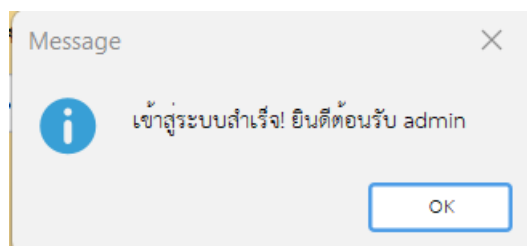
อีเมล  
admin@gmail.com

ตั้งรหัสผ่าน  
1234

ย้อนกลับ    ยืนยัน

ภาพที่ 3 หน้าสมัครสมาชิก

คำอธิบาย : เมื่อกดที่สมัครสมาชิก ให้กรอกชื่อผู้ใช้ ใส่อีเมล และสร้างรหัสผ่าน ถ้ากดย้อนกลับจะกลับไปหน้าเข้าสู่ระบบ และถ้ากดยืนยันจะสมัครสำเร็จ



ภาพที่ 4 หน้าจอแสดงผลเมื่อเข้าสู่ระบบสำเร็จ

คำอธิบาย : เมื่อผู้ใช้เข้าสู่ระบบสำเร็จจะแสดงหน้าจอว่า เข้าสู่ระบบสำเร็จ ยินดีต้อนรับ “ตามด้วยชื่อที่ผู้ใช้ตั้ง”

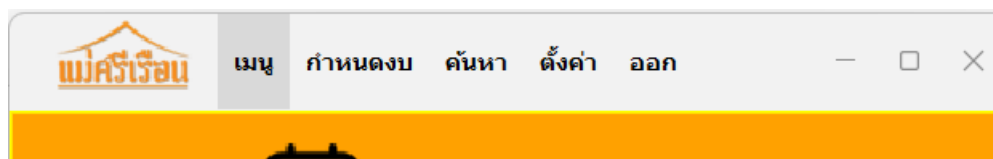
## 2. หน้า Menu List



ภาพที่ 5 หน้าเมนู

คำอธิบาย : ผู้ใช้สามารถ เลือกได้ว่าผู้จะไปยังหน้าต่างๆได้โดยการกดที่ไอคอน

### 3. Menu Bar



ภาพที่ 6 เมนูบาร์

คำอธิบาย : ส่วนนี้จะเป็นเมนูบาร์สามารถกดได้ ถ้ากดแล้วจะไปหน้าดังกล่าวมีเขียนไว้ คำว่าออก จะกำหนดให้กลับไปหน้าลือคอินหรือออกจากโปรแกรม

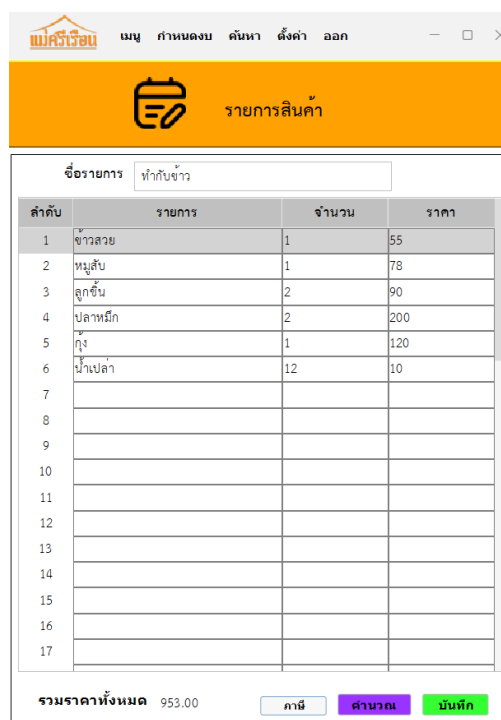
#### 4. หน้ารายการสินค้า

ภาพที่ 7 หน้ารายการสินค้าและอัปโหลด

คำอธิบาย : ผู้ใช้สามารถกดที่ไอคอนสร้างรายสินค้าเพื่อไปอีกหน้าสร้างรายการ และสามารถกดที่ไอคอนเช็ครายการสินค้าเพื่อไปหน้าเช็ครายการสินค้าได้ และกดอัปโหลดเพื่ออัปโหลดรายการสินค้าเข้ามาที่ผู้ใช้กดบันทึก



#### 4.1 หน้าสร้างรายการสินค้า



เมนู กำหนดแบบ ค้นหา สั่งค่า ออก

รายการสินค้า

ชื่อรายการ: ทำกับข้าว

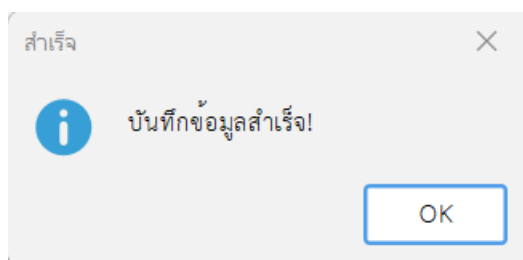
| ลำดับ | รายการ  | จำนวน | ราคา |
|-------|---------|-------|------|
| 1     | ข้าวสวย | 1     | 55   |
| 2     | หมูสับ  | 1     | 78   |
| 3     | ลูกชิ้น | 2     | 90   |
| 4     | ปลาหมึก | 2     | 200  |
| 5     | กุ้ง    | 1     | 120  |
| 6     | น้ำปลา  | 12    | 10   |
| 7     |         |       |      |
| 8     |         |       |      |
| 9     |         |       |      |
| 10    |         |       |      |
| 11    |         |       |      |
| 12    |         |       |      |
| 13    |         |       |      |
| 14    |         |       |      |
| 15    |         |       |      |
| 16    |         |       |      |
| 17    |         |       |      |

รวมราคาทั้งหมด 953.00

Cancel Save Print

ภาพที่ 8 หน้ารายการสินค้า

คำอธิบาย : ถ้ากดที่ไอคอนสร้างรายการสินค้า ผู้ใช้สามารถตั้งชื่อรายการได้ และกำหนดรายการที่จะซื้อ จำนวน และราคา ตามที่ต้องการจากนั้นกดคำนวณระบบจะคำนวณยอดรวมให้เป็นราคาทั้งหมด และกดบันทึก นอกจากนี้ยังมีปุ่มสำหรับคำนวณภาษีเพื่อใช้ในการคำนวณภาษีถ้าต้องการ



ภาพที่ 9 หน้าจอแสดงผลเมื่อบันทึกสำเร็จ

คำอธิบาย : เมื่อทำการกดบันทึก ระบบจะมีหน้าต่างแจ้งเตือนขึ้นมา

## 4.2 ใช้รายการสินค้า

0-0-0 ใช้รายการสินค้า

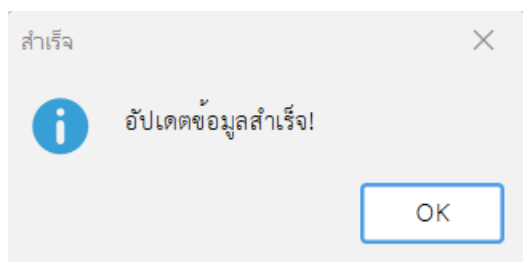
ทำกับข้าว ค้นหา

| เลือก                    | รายการ  | จำนวน | ราคา  |
|--------------------------|---------|-------|-------|
| <input type="checkbox"/> | ข้าวสวย | 1     | 55.0  |
| <input type="checkbox"/> | หนุสับ  | 1     | 78.0  |
| <input type="checkbox"/> | ลูกชิ้น | 1     | 90.0  |
| <input type="checkbox"/> | ปลาหมึก | 2     | 200.0 |
| <input type="checkbox"/> | กุ้ง    | 1     | 120.0 |
| <input type="checkbox"/> | น้ำปลา  | 12    | 10.0  |

ยอดรวม 953.00 ใส่ตะกร้า อัปเดต เช็ค

ภาพที่ 10 หน้าใช้รายการสินค้า

คำอธิบาย : หน้านี้จะสามารถใช้รายการสินค้าที่เรากำหนดจากหน้าสร้างรายการสินค้า สามารถพิมพ์ชื่อรายการและกดค้นหาก็จะขึ้นของที่เราจะซื้อ กด รีเซ็ตเพื่อล้างค่าทั้งหมด กดอัปเดตเพื่ออัปเดตจำนวนกับราคา และกดเช็คเพื่อแจ้งเตือนว่าผู้ใช้ซื้อหรือลืมซื้ออะไร



ภาพที่ 11 หน้าจอแสดงผลเมื่ออัปเดตข้อมูลสำเร็จ

คำอธิบาย : เมื่อทำการกดอัปเดต ระบบก็จะมีหน้าต่างแจ้งเตือนขึ้นมา

The screenshot shows a web-based application for a Thai restaurant. The top navigation bar includes a logo and menu items: 'เมนู', 'รายการ', 'กำหนดงบ', 'ค้นหา', and 'ออก'. The main header is orange with a clipboard icon and the text 'เช็ครายการสินค้า'. Below this, there is a search bar labeled 'ทำกับข้าว' and a 'ค้นหา' button. A table lists items with columns for selection, item name, quantity, and price. All items are checked. A notification dialog box is displayed in the center, indicating that all items have been successfully checked out. At the bottom, the total amount is 863.00, and there are buttons for 'รีเซ็ต', 'อัปเดต', and 'เช็ค'.

| เลือก                               | รายการ   | จำนวน | ราคา  |
|-------------------------------------|----------|-------|-------|
| <input checked="" type="checkbox"/> | ข้าวสวย  | 1     | 55.0  |
| <input checked="" type="checkbox"/> | หมูสับ   | 1     | 78.0  |
| <input checked="" type="checkbox"/> | ลูกชิ้น  | 1     | 90.0  |
| <input checked="" type="checkbox"/> | ปลาหมึก  | 2     | 200.0 |
| <input checked="" type="checkbox"/> | กุ้ง     | 1     | 120.0 |
| <input checked="" type="checkbox"/> | น้ำเปล่า | 12    | 10.0  |

Notification

แจ้งเตือน:  
คุณเช็คครบทุกอย่างแล้ว!

OK

ยอดรวม 863.00

รีเซ็ต อัปเดต เช็ค

ภาพที่ 12 หน้าจอแสดงผลเมื่อเช็คสำเร็จ

คำอธิบาย : เมื่อทำการกดเช็คแล้วจะมีหน้าต่างแจ้งเตือนขึ้นมาเมื่อผู้ใช้ชื่อของครบแล้ว

หน้าจอสื่อ

เมนู รายการ กำหนดแบบ ค้นหา ออก

เช็ครายการสินค้า

ทำกับข้าว ค้นหา

| เลือก                               | รายการ  | จำนวน | ราคา  |
|-------------------------------------|---------|-------|-------|
| <input checked="" type="checkbox"/> | ข้าวสวย | 1     | 55.0  |
| <input checked="" type="checkbox"/> | หมูสับ  | 1     | 78.0  |
| <input checked="" type="checkbox"/> | ลูกชิ้น | 1     | 90.0  |
| <input checked="" type="checkbox"/> | ปลาหมึก | 2     | 200.0 |
| <input type="checkbox"/>            | กุ้ง    | 1     | 120.0 |
| <input type="checkbox"/>            | น้ำปลา  | 10    | 10.0  |

Notification

แจ้งเตือน:

กรุณาตรวจสอบรายการซื้อสินค้าให้เรียบร้อย:

- กุ้ง
- น้ำปลา

OK

ยอดรวม 863.00 รีเซ็ต อัปเดต เช็ค

ภาพที่ 13 หน้าจอแสดงผลเมื่อเช็คสำเร็จ

คำอธิบาย : ถ้าทำการกดเช็คแล้วแล้วยังไม่ได้ทำการซื้อจะมีหน้าต่างแจ้งเตือนขึ้นมว่าสินค้าที่ยังไม่ได้ซื้อ

## 5. กำหนดงบประมาณ



ภาพที่ 14 หน้ากำหนดงบประมาณ

คำอธิบาย : สามารถกดไอคอนสร้างกำหนดงบประมาณเพื่อไปหน้าสร้างกำหนด และกดอัปโหลดเพื่ออัปโหลดรายการสินค้าเข้ามาที่ผู้ใช้กดบันทึก

## 5.1 สร้างกำหนดงบประมาณ

ชื่อรายการ

ใส่จำนวนเงิน  1282.00 จำนวนเงินคงเหลือ

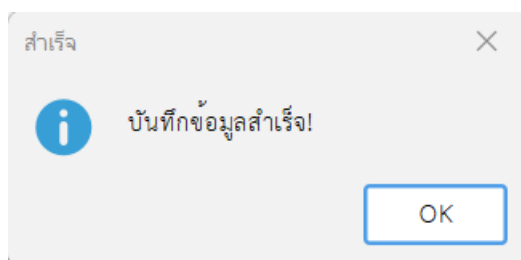
| ลำดับ | รายการ  | จำนวน | ราคา |
|-------|---------|-------|------|
| 1     | น้ำโคก  | 2     | 120  |
| 2     | เค้ก    | 1     | 350  |
| 3     | KFC     | 2     | 199  |
| 4     | ยำทะเล  | 3     | 120  |
| 5     | ไก่ย่าง | 4     | 40   |
| 6     | หมูกรอบ | 1     | 60   |
| 7     | ปลาเผา  | 1     | 150  |
| 8     |         |       |      |
| 9     |         |       |      |
| 10    |         |       |      |
| 11    |         |       |      |
| 12    |         |       |      |
| 13    |         |       |      |
| 14    |         |       |      |
| 15    |         |       |      |
| 16    |         |       |      |

ใช้ไปแล้ว 1718.00

ภาพที่ 15 หน้ากำหนดงบประมาณ

คำอธิบาย : สามารถใส่จำนวนเงินที่ช่องตามต้องการ ระบบจะทำการคำนวณเงินจากจำนวนเงินที่มีลบ(-)กับสินค้าที่ผู้ใช้จะซื้อหรือที่กำหนดไว้ และจะแสดงจำนวนเงินคงเหลือกับเงินที่ใช้ไปแล้ว นอกจากนี้ยังมีปุ่มสำหรับคำนวณภาษีเพื่อใช้ในการคำนวณภาษีถ้าต้องการ





ภาพที่ 16 หน้าจอแสดงผลเมื่อบันทึกข้อมูลสำเร็จ

คำอธิบาย : เมื่อทำการกดบันทึก ระบบจะมีหน้าต่างแจ้งเตือนขึ้นมา

## 6. หน้าการค้นหา

เมนู รายการ กำหนดงบ ตั้งค่า ออก

### การค้นหา

รายการ :


วัน/เดือน/ปี :

Ex: 2025-01-01

| ยอดการใช้จ่ายทั้งหมด |        | 0.00       |              |
|----------------------|--------|------------|--------------|
| ลำดับ                | รายการ | ยอดใช้จ่าย | วัน/เดือน/ปี |
|                      |        |            |              |

ภาพที่ 17 หน้าการค้นหา

คำอธิบาย : ผู้ใช้สามารถเรียกดูหรือค้นหารายการที่ต้องการค้นหาได้



เมนู

รายการ

กำหนดงบ

ตั้งค่า

ออก

การค้นหา

รายการ :

ค่าอาหาร

ค้นหา

วัน/เดือน/ปี :

ทั้งหมด

Ex: 2025-01-01


ยอดการใช้จ่ายทั้งหมด

110.00

| ลำดับ | รายการ   | ยอดใช้จ่าย | วัน/เดือน/ปี |
|-------|----------|------------|--------------|
| 1     | ค่าอาหาร | 110.0      | 2024-12-30   |

ภาพที่ 18 หน้าการค้นหาแบบการกรอกรายการ

คำอธิบาย : เมื่อผู้ใช้กรอกรายการที่อยากค้นหา แล้วกดคำว่าค้นหา ระบบก็จะแสดงรายการตามที่ผู้ใช้กรอกและแสดงยอดการใช้จ่ายทั้งหมดของรายการนั้น



เมนู

รายการ

กำหนดงบ

ตั้งค่า

ออก

การค้นหา

รายการ :

ค้นหา

วัน/เดือน/ปี :

2023-01-15

ทั้งหมด

Ex: 2025-01-01


ยอดการใช้จ่ายทั้งหมด

10000.00

| ลำดับ | รายการ         | ยอดใช้จ่าย | วัน/เดือน/ปี |
|-------|----------------|------------|--------------|
| 1     | ค่าเบี้ยประกัน | 10000.0    | 2023-01-15   |

ภาพที่ 19 หน้าการค้นหาแบบรายวัน

คำอธิบาย : เมื่อผู้ใช้กรอกวันที่ที่อยากค้นหา แล้วกดคำว่าค้นหา ระบบก็จะแสดงรายการตามที่ผู้ใช้กรอกและแสดงยอดการใช้จ่ายทั้งหมดของวันนั้น (Ex: 2025-01-01)


เมนู รายการ กำหนดค่า ค้นหา ออก

### การค้นหา

รายการ :

วัน/เดือน/ปี :


Ex: 2025-01-01

**ยอดการใช้จ่ายทั้งหมด 5540.00**

| ลำดับ | รายการ             | ยอดใช้จ่าย | วัน/เดือน/ปี |
|-------|--------------------|------------|--------------|
| 1     | ค่าอาหาร           | 110.0      | 2024-12-30   |
| 2     | ค่าเดินทาง         | 200.0      | 2024-12-30   |
| 3     | test admin         | 110.0      | 2024-12-31   |
| 4     | test               | 4400.0     | 2024-12-31   |
| 5     | งบค่าใช้จ่ายรายวัน | 320.0      | 2024-12-30   |
| 6     | งบรายวัน           | 400.0      | 2024-12-30   |

ภาพที่ 20 หน้าการค้นหาแบบรายเดือน

คำอธิบาย : เมื่อผู้ใช้อกรอกเดือนที่อยากค้นหา แล้วกดคำว่าค้นหา ระบบก็จะแสดงรายการตามที่ผู้ใช้อกรอกและแสดงยอดการใช้จ่ายทั้งหมดของเดือนนั้น (Ex: 2025-01)



เมนู

รายการ

กำหนดงบ

ตั้งค่า

ออก

การค้นหา

รายการ :

ค้นหา

วัน/เดือน/ปี :

2025

ทั้งหมด

Ex: 2025-01-01


ยอดการใช้จ่ายทั้งหมด

7937.00

| ลำดับ | รายการ     | ยอดใช้จ่าย | วัน/เดือน/ปี |
|-------|------------|------------|--------------|
| 1     | งานวันเกิด | 300.0      | 2025-01-01   |
| 2     | testnew    | 26.0       | 2025-01-02   |
| 3     | วันเกิด    | 1000.0     | 2025-01-07   |
| 4     | สินค้า     | 600.0      | 2025-01-10   |
| 5     | ทำกับข้าว  | 953.0      | 2025-01-15   |
| 6     | งานเลี้ยง  | 1900.0     | 2025-01-01   |
| 7     | testnewn   | 440.0      | 2025-01-02   |
| 8     | วันเกิด2   | 1000.0     | 2025-01-07   |
| 9     | งานวันเกิด | 1718.0     | 2025-01-15   |

ภาพที่ 21 หน้าการค้นหาแบบรายปี

คำอธิบาย : เมื่อผู้ใช้กรอกปีที่อยากค้นหา แล้วกดคำว่าค้นหา ระบบก็จะแสดงรายการตามที่ใช้กรอกและแสดงยอดการใช้จ่ายทั้งหมดของปีนั้น (Ex: 2025)


เมนู รายการ กำหนดงบ ตั้งค่า ออก

**การค้นหา**

รายการ :  ค้นหา

วัน/เดือน/ปี :  ทั้งหมด

Ex: 2025-01-01

**ยอดการใช้จ่ายทั้งหมด 23477.00**

| ลำดับ | รายการ             | ยอดใช้จ่าย | วัน/เดือน/ปี |
|-------|--------------------|------------|--------------|
| 1     | ค่าเบี้ยประกัน     | 10000.0    | 2023-01-15   |
| 2     | งบค่าใช้จ่ายรายวัน | 320.0      | 2024-12-30   |
| 3     | ค่าอาหาร           | 110.0      | 2024-12-30   |
| 4     | งบรายวัน           | 400.0      | 2024-12-30   |
| 5     | ค่าเดินทาง         | 200.0      | 2024-12-30   |
| 6     | test admin         | 110.0      | 2024-12-31   |
| 7     | test               | 4400.0     | 2024-12-31   |
| 8     | งานวันเกิด         | 300.0      | 2025-01-01   |
| 9     | งานเลี้ยง          | 1900.0     | 2025-01-01   |
| 10    | testnew            | 26.0       | 2025-01-02   |
| 11    | testnewn           | 440.0      | 2025-01-02   |
| 12    | วันเกิด            | 1000.0     | 2025-01-07   |
| 13    | วันเกิด2           | 1000.0     | 2025-01-07   |
| 14    | สินค้า             | 600.0      | 2025-01-10   |
| 15    | ฟ้ากับข้าว         | 953.0      | 2025-01-15   |
| 16    | รวมสิ้นปี          | 1710.0     | 2025-01-15   |

ภาพที่ 22 หน้าการค้นหาแบบทั้หมด

คำอธิบาย : เมื่อผู้ใ้กดคำว่าทั้หมด ระบบจะแสดงรายการและยอดการใ้จ่ายทั้หมด

## 6. หน้าตั้งค่าผู้ใช้



เมนู รายการ กำหนดจบ ค้นหา ออก

ตั้งค่าผู้ใช้

ตั้งชื่อใหม่  
newadmin

ตั้งอีเมลใหม่  
newadmin@gmail.com

ตั้งรหัสผ่านใหม่  
1234

ยืนยัน ยกเลิก

ภาพที่ 23 หน้าการตั้งค่าผู้ใช้งาน

คำอธิบาย : ผู้ใช้สามารถตั้งชื่อ อีเมล และรหัสผ่านใหม่ได้ เมื่อกดยืนยันระบบจะทำการบันทึกชื่อและรหัสผ่านใหม่ในการเข้าสู่ระบบ และกดยกเลิกเพื่อยกเลิกทั้งหมดของหน้านี้



## ภาคผนวก

### ปริมาณงานและขอบเขตของโปรแกรม

1. มีจำนวน class ทั้งหมด 28 class
2. มี method ที่เขียนขึ้นเอง 45 method
3. มีการประยุกต์ใช้การค้นหาข้อมูล
4. มีการประยุกต์ใช้การเรียงลำดับข้อมูล
5. มีการประยุกต์ใช้การนำไฟล์เข้า
6. มีการใช้แนวคิด Inheritance
7. มีการใช้แนวคิด Polymorphism

## การใช้แนวคิด Inheritance ในการพัฒนา

1. Inheritance คลาส LoginManager และ RegisterManager สืบทอด จาก UserManager ทำให้สามารถใช้ DatabaseManager db ได้โดยไม่ต้องกำหนดใหม่

\*ยกเว้น คลาส Login กับ Register ไม่สามารถ extends ได้ เนื่องจาก class extends javax.swing.JFrame

```
package project93;

import database.DatabaseManager;

public class UserManager {
    protected DatabaseManager db;

    public UserManager(DatabaseManager db) {
        this.db = db;
    }
}
```

ภาพที่ 24 คลาสแม่ UserManager

```
package project93;

import java.sql.ResultSet;
import database.DatabaseManager;

public class LoginManager extends UserManager {
    private static int currentUserId;
    private static String currentUserName;

    public LoginManager(DatabaseManager db) {
        super(db);
    }

    public boolean authenticate(String username, String password) {
        try {
            String query = "SELECT userid, username FROM user WHERE username = ? AND password = ?";
            ResultSet rs = db.executeQuery(query, username, password);
            if (rs.next()) {
                currentUserId = rs.getInt("userid");
                currentUserName = rs.getString("username");
                return true;
            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            db.disconnect();
        }
        return false;
    }
}
```

ภาพที่ 25 คลาสลูก LoginManager

```

package project93;

import database.DatabaseManager;

public class RegisterManager extends UserManager {
    public RegisterManager(DatabaseManager db) {
        super(db);
    }

    public boolean registerUser(String username, String email, String password) {
        if (username.isEmpty() || email.isEmpty() || password.isEmpty()) {
            throw new IllegalArgumentException("xxxxxxxxxxxxxxxxxxxxxxxx");
        }

        if (!email.matches("^[A-Za-z0-9+_.-]+@(.+)$")) {
            throw new IllegalArgumentException("xxxxxxxxxxxxxxxxxxxxxxxx");
        }

        try {
            String query = "INSERT INTO user (username, email, password) VALUES (?, ?, ?)";
            int rowsInserted = db.executeUpdate(query, username, email, password);
            return rowsInserted > 0;
        } catch (Exception e) {
            e.printStackTrace();
            throw new RuntimeException("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
        }
    }
}

```

ภาพที่ 26 คลาสลูก RegisterManager

2. ChecklistManager สืบทอดตัวแปร db และโครงสร้างของ search จาก CheckSearch  
ChecklistManager ต้อง Override เมทอด search เพราะ CheckSearch กำหนดให้เป็น abstract

*\*ยกเว้น คลาส Check ไม่สามารถ extends ได้ เนื่องจาก class extends javax.swing.JFrame*

```

package main;

import java.util.List;
import database.DatabaseManager;

public abstract class CheckSearch {
    protected DatabaseManager db;

    public CheckSearch(DatabaseManager db) {
        this.db = db;
    }

    public abstract List<Object[]> search(String searchText) throws Exception;
}

```

ภาพที่ 27 คลาสแม่ CheckSearch

```

package main;

import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import java.util.List;
import javax.swing.table.DefaultTableModel;
import database.DatabaseManager;

public class ChecklistManager extends AbstractSearch {
    public ChecklistManager(DatabaseManager db) {
        super(db);
    }

    @Override
    public List<Object[]> search(String searchText) throws Exception {
        List<Object[]> results = new ArrayList<>();
        String query = "SELECT listitem AS item_name, quantity, price FROM items WHERE listname LIKE ?";

        try (PreparedStatement ps = db.getConnection().prepareStatement(query)) {
            ps.setString(1, "%" + searchText + "%");
            ResultSet rs = ps.executeQuery();
            while (rs.next()) {
                results.add(new Object[]{
                    false,
                    rs.getString("item_name"),
                    rs.getInt("quantity"),
                    rs.getDouble("price")
                });
            }
        }
        return results;
    }
}

```

ภาพที่ 28 คลาสลูก ChecklistManager

3. ManagerCheck เป็นคลาสลูก (extends) ของ CheckBase เป็น คลาสแม่ (Abstract Class) ที่กำหนดโครงสร้างพื้นฐานสำหรับการแสดงข้อความ

```

package main;

abstract class CheckBase {
    protected String message;

    public CheckBase(String message) {
        this.message = message;
    }

    public abstract void showMessage();
}

```

ภาพที่ 29 คลาสแม่ CheckBase

```

package main;

import javax.swing.JOptionPane;
import javax.swing.JTable;

public class ManagerCheck extends CheckBase {
    private JTable jTable1;

    public ManagerCheck(JTable jTable) {
        super("");
        this.jTable1 = jTable;
    }

    @Override
    public void showMessage() {
        JOptionPane.showMessageDialog(null, message, "Notification", JOptionPane.INFORMATION_MESSAGE);
    }

    public void checkMissingItems() {
        StringBuilder missingItems = new StringBuilder();

        for (int i = 0; i < jTable1.getRowCount(); i++) {
            Boolean isChecked = (Boolean) jTable1.getValueAt(i, 0);
            String itemName = (String) jTable1.getValueAt(i, 1);

            if (isChecked == null || !isChecked) {
                missingItems.append("- ").append(itemName).append("\n");
            }
        }

        if (missingItems.length() > 0) {
            this.message = "❗\n" + missingItems.toString();
        } else {
            this.message = "✅\n";
        }
    }
}

```

ภาพที่ 30 คลาสลูก ManagerCheck

4. BudgetManager เป็นคลาสแม่ของ BudgetBase โดยสืบทอดคุณสมบัติและเมทอดทั้งหมด และสามารถ override เมทอดได้เอง ใน constructor ของ BudgetManager ใช้ super(...) เพื่อเรียก constructor ของ BudgetBase และกำหนดค่าเริ่มต้นให้กับ tableModel, budget, และ remainingBudgetLabel

\*ยกเว้น คลาส Showbudget ไม่สามารถ extends ได้ เนื่องจาก class extends javax.swing.JFrame

```

package budget;

import javax.swing.JLabel;
import javax.swing.table.DefaultTableModel;
import java.util.logging.Logger;
import java.util.logging.Level;

public abstract class BudgetBase {
    private static final Logger LOGGER = Logger.getLogger(BudgetBase.class.getName());

    protected DefaultTableModel tableModel;
    protected double budget;
    protected JLabel remainingBudgetLabel;

    public BudgetBase(DefaultTableModel tableModel, double budget, JLabel remainingBudgetLabel) {...8 lines}

    public void setInputText(String inputText) {...9 lines}

    public void setInputText(double inputBudget) {...7 lines}

    public double getBudget() {...3 lines}

    public void updateRemainingBudget() {...3 lines}

    public double getRemainingBudget() {...3 lines}

    public abstract double calculateTotalSpent();

    public double calculateTotalSpent(double taxRate) {...3 lines}
}

```

ภาพที่ 31 คลาสแม่ BudgetBase

```

package budget;

import javax.swing.JLabel;
import javax.swing.table.DefaultTableModel;
import java.util.logging.Logger;

public class BudgetManager extends BudgetBase {
    private static final Logger LOGGER = Logger.getLogger(BudgetManager.class.getName());

    private final boolean isDynamic;

    public BudgetManager(DefaultTableModel tableModel, double budget, JLabel remainingBudgetLabel, boolean isDynamic) {...4 lines }

    @Override
    public double calculateTotalSpent() {
        double total = 0.0;
        for (int i = 0; i < tableModel.getRowCount(); i++) {
            Object priceObj = tableModel.getValueAt(i, 3);
            Object quantityObj = tableModel.getValueAt(i, 2);

            try {
                if (priceObj != null && quantityObj != null) {
                    double price = Double.parseDouble(priceObj.toString().trim());
                    int quantity = Integer.parseInt(quantityObj.toString().trim());
                    total += price * quantity;
                }
            } catch (NumberFormatException e) {
                LOGGER.warning("Invalid data at row " + (i + 1));
            }
        }
        return total;
    }

    public void adjustBudget(double adjustment) {...7 lines }
}

```

ภาพที่ 32 คลาสลูก BudgetManager

5. ListData เป็นคลาสแม่ ของ ListManager ที่สืบทอดโครงสร้างพื้นฐานของการจัดการรายการ (tableModel) และต้อง override เมทอด calculateTotal() และ saveData() ใน constructor ของ ListData ใช้ super(...) เพื่อเรียก constructor ของ ListManager และกำหนดค่า calTotalLabel. นอกจากนี้ ListData ยังเพิ่มเมทอด calculateTotal(double taxRate), updateListSummary(), และการบันทึกข้อมูลลงในฐานข้อมูล (saveData)

\*ยกเว้น คลาส Showlist ไม่สามารถ extends ได้ เนื่องจาก class extends javax.swing.JFrame

```

package List;

import javax.swing.table.DefaultTableModel;

public abstract class ListManager {
    protected DefaultTableModel tableModel;

    public ListManager(DefaultTableModel tableModel) {
        this.tableModel = tableModel;
    }

    protected boolean validateData(Object priceObj, Object quantityObj) {...12 lines }

    public abstract double calculateTotal();
    public abstract double calculateTotal(double taxRate);

    public abstract void saveData(String listName, int userId);
}

```

ภาพที่ 33 คลาสแม่ ListManager

```

package List;

import database.DatabaseManager;
import java.sql.SQLException;
import java.util.logging.Level;
import javax.swing.table.DefaultTableModel;
import java.util.logging.Logger;
import javax.swing.JLabel;

public class ListData extends ListManager {
    private static final Logger LOGGER = Logger.getLogger(ListData.class.getName());
    private double totalWithTax = 0.0;
    private JLabel calTotalLabel;

    {...3 lines }

    public ListData(DefaultTableModel tableModel, JLabel calTotalLabel) {...4 lines }

    public double calculateTotal() {...14 lines }

    public double calculateTotal(double taxRate) {...5 lines }

    public double getTotalWithTax() {...3 lines }

    public void saveData(String listName, int userId) {...19 lines }

    public void saveData(int userId, String listName, String itemName, int quantity, double price) throws SQLException {...1}

    public void updateListSummary(double taxRate) {...9 lines }
}

```

ภาพที่ 34 คลาสลูก ListData

6. BaseSearch เป็นคลาสแม่ที่กำหนดฟังก์ชันการค้นหาข้อมูลจากฐานข้อมูล GenericSearch เป็นคลาสลูกที่สืบทอดจาก BaseSearch และทำการค้นหาข้อมูลที่เกี่ยวข้องจากฐานข้อมูลโดยใช้ SQL query CalculatorSearch เป็นคลาสที่คำนวณผลรวมของยอดค่าใช้จ่ายจากข้อมูลที่ได้ DataSorter เป็นคลาสที่จัดการการเรียงข้อมูลโดยใช้ Quick Sort หรือวิธีอื่นๆ

\*ยกเว้น คลาส Showsearch ไม่สามารถ extends ได้ เนื่องจาก class extends javax.swing.JFrame

```

package Search;

import database.DatabaseManager;
import java.util.List;

public abstract class BaseSearch {
    protected DatabaseManager dbManager;

    public BaseSearch(DatabaseManager dbManager) {
        this.dbManager = dbManager;
    }

    public abstract List<SearchData> fetchData(int userId, String date, String listName);
}

```

ภาพที่ 35 คลาสแม่ BaseSearch

```

package Search;

import database.DatabaseManager;
import java.util.List;

public class CalculatorSearch extends BaseSearch {

    public CalculatorSearch(DatabaseManager dbManager) {
        super(dbManager);
    }

    @Override
    public List<SearchData> fetchData(int userId, String date, String listName) {
        throw new UnsupportedOperationException("CalculatorSearch 000000000000 fetch 000000");
    }

    public double calculateTotal(List<SearchData> searchDataList) {
        double total = 0.0;
        for (SearchData data : searchDataList) {
            total += data.getAmount();
        }
        return total;
    }
}

```

ภาพที่ 36 คลาสลูก CalculatorSearch

```

package Search;

import database.DatabaseManager;
import java.util.List;

public class DataSorter extends BaseSearch {

    public DataSorter(DatabaseManager dbManager) {
        super(dbManager);
    }

    @Override
    public List<SearchData> fetchData(int userId, String date, String listName) {...3 lines }

    public static void quickSortByDate(List<SearchData> data, int low, int high) {...8 lines }

    private static int partitionByDate(List<SearchData> data, int low, int high) {...14 lines }

    public static void quickSortByAmountDescending(List<SearchData> data, int low, int high) {...8 lines }

    private static int partitionByAmountDescending(List<SearchData> data, int low, int high) {...14 lines }

    private static void swap(List<SearchData> data, int i, int j) {...5 lines }
}

```

ภาพที่ 37 คลาสลูก DataSorter

```

package Search;

import database.DatabaseManager;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import java.util.List;

public class GenericSearch extends BaseSearch {

    public GenericSearch(DatabaseManager dbManager) {
        super(dbManager);
    }

    public List<SearchData> fetchData(int userId, String date, String listName) {...78 lines }
}

```

ภาพที่ 38 คลาสลูก GenericSearch



## การใช้แนวคิด Polymorphism ในการพัฒนา

### 1. ในคลาส BudgetManager มีการใช้ Overloading

setInputText(String inputText); → แปลงค่าจาก String เป็น double

setInputText(double inputBudget); → กำหนดค่าเป็น double โดยตรง

```
public void setInputText(String inputText) {
    if (inputText != null) {
        try {
            setInputText(Double.parseDouble(inputText.trim()));
        } catch (NumberFormatException e) {
            LOGGER.log(Level.WARNING, "Invalid budget input: " + inputText, e);
        }
    }
}

public void setInputText(double inputBudget) {
    if (inputBudget >= 0) {
        this.budget = inputBudget;
    } else {
        LOGGER.warning("Attempted to set a negative budget: " + inputBudget);
    }
}
```

ภาพที่ 39 เมธอด setInputText ในคลาส BudgetManager

### 2. ในคลาส BudgetManager มีการ Override เมธอด calculateTotalSpent()

calculateTotalSpent() → คำนวณยอดรวมของรายการสินค้าใน tableModel

calculateTotalSpent(double taxRate) → คำนวณยอดรวมพร้อมภาษี

```
public double calculateTotalSpent(double taxRate) {
    return calculateTotalSpent() * (1 + taxRate);
}
```

ภาพที่ 40 เมธอด calculateTotalSpent() ในคลาส BudgetManager

```

@Override
public double calculateTotalSpent() {
    double total = 0.0;
    for (int i = 0; i < tableModel.getRowCount(); i++) {
        Object priceObj = tableModel.getValueAt(i, 3);
        Object quantityObj = tableModel.getValueAt(i, 2);

        try {
            if (priceObj != null && quantityObj != null) {
                double price = Double.parseDouble(priceObj.toString().trim());
                int quantity = Integer.parseInt(quantityObj.toString().trim());
                total += price * quantity;
            }
        } catch (NumberFormatException e) {
            LOGGER.warning("Invalid data at row " + (i + 1));
        }
    }
    return total;
}

```

ภาพที่ 41 เมธอด calculateTotalSpent() ในคลาส UnifiedBudgetManager

### 3. ในคลาส ListData มีการใช้ Overloading กับ calculateTotal()

calculateTotal() → คำนวณยอดรวมของสินค้าใน tableModel

calculateTotal(double taxRate) → คำนวณยอดรวมพร้อมภาษี

```

@Override
public double calculateTotal() {
    double total = 0.0;
    for (int i = 0; i < tableModel.getRowCount(); i++) {
        Object priceObj = tableModel.getValueAt(i, 3);
        Object quantityObj = tableModel.getValueAt(i, 2);

        if (validateData(priceObj, quantityObj)) {
            double price = Double.parseDouble(priceObj.toString().trim());
            int quantity = Integer.parseInt(quantityObj.toString().trim());
            total += price * quantity;
        }
    }
    LOGGER.fine("Total amount calculated: " + total);
    return total;
}

@Override
public double calculateTotal(double taxRate) {
    double total = calculateTotal();
    return total + (total * taxRate);
}

```

ภาพที่ 42 เมธอด calculateTotal() ในคลาส TransactionList

4. ใช้ Overriding ในคลาส ListData โดยที่ เมธอด ทั้ง 2 จะมีหน้าที่ต่างกันคือ saveData แรกจะบันทึกข้อมูลไปยังตาราง showlist ส่วน เมธอด saveData ที่สองจะบันทึกข้อมูลสินค้าไปยัง item

```
public void saveData(String listName, int userId) {
    DatabaseManager dbManager = new DatabaseManager();
    try {
        dbManager.beginTransaction();

        double totalAmount = calculateTotal();
        LOGGER.fine("Total amount calculated: " + totalAmount);

        String sqlShowlist = "INSERT INTO showlist (userid, list_name, total_amount, created_at) VALUES (?, ?, ?, NOW())";
        dbManager.executeUpdate(sqlShowlist, userId, listName, totalAmount);

        dbManager.commitTransaction();
        LOGGER.info("Showlist data added successfully!");
    } catch (SQLException ex) {
        Logger.getLogger(ListData.class.getName()).log(Level.SEVERE, null, ex);
    } finally {
        dbManager.disconnect();
    }
}
```

ภาพที่ 43 เมธอด saveData() ในคลาส ListData

```
public void saveData(int userId, String listName, String itemName, int quantity, double price) throws SQLException {
    DatabaseManager dbManager = new DatabaseManager();
    try {
        dbManager.beginTransaction();

        double total = quantity * price;

        String sqlItems = "INSERT INTO items (userid, listname, listitem, quantity, price, total, created_at) " +
            "VALUES (?, ?, ?, ?, ?, ?, NOW())";
        dbManager.executeUpdate(sqlItems, userId, listName, itemName, quantity, price, total);

        dbManager.commitTransaction();
        LOGGER.info("Item data added successfully!");
    } finally {
        dbManager.disconnect();
    }
}
```

ภาพที่ 44 เมธอด saveData() ในคลาส ListData

5. ในคลาส ChecklistManager มีการใช้ Method Overloading กับเมทอด search

search(String searchText); → ค้นหาข้อมูลโดยไม่ระบุ userId (ค้นหาทุกผู้ใช้)

search(String searchText, Integer userId); → ค้นหาข้อมูลโดยระบุ userId เพื่อให้แสดงเฉพาะของผู้ใช้คนนั้น

```
@Override
public List<Object[]> search(String searchText) throws Exception {
    return search(searchText, null);
}

public List<Object[]> search(String searchText, Integer userId) throws Exception {
    List<Object[]> results = new ArrayList<>();
    String query = "SELECT listitem AS item_name, quantity, price FROM items WHERE listname LIKE ?";

    if (userId != null) {
        query += " AND userid = ?";
    }

    try (PreparedStatement ps = db.getConnection().prepareStatement(query)) {
        ps.setString(1, "%" + searchText + "%");

        if (userId != null) {
            ps.setInt(2, userId);
        }

        ResultSet rs = ps.executeQuery();
        while (rs.next()) {
            results.add(new Object[]{
                false,
                rs.getString("item_name"),
                rs.getInt("quantity"),
                rs.getDouble("price")
            });
        }
    }
    return results;
}
```

ภาพที่ 45 เมทอด search() ในคลาส ChecklistManager

6. เมทอด updateBudgetSummary() คำนวณค่าใช้จ่ายรวม (totalSpent) และงบประมาณคงเหลือ (remaining) โดยใช้ข้อมูลจาก budgetManager และอัปเดตค่าใน jLabel5 และ UseTotal. เมทอด updateBudgetSummary(double taxRate) ทำงานแบบเดียวกัน แต่รวมภาษี (taxRate) เข้าไปในการคำนวณค่าใช้จ่ายรวม

```
private void updateBudgetSummary() {
    try {
        budgetManager.updateRemainingBudget();
        double totalSpent = budgetManager.calculateTotalSpent();
        double remaining = budgetManager.getBudget() - totalSpent;

        jLabel5.setText(String.format("%.2f", remaining));
        UseTotal.setText(String.format("%.2f", totalSpent));
    } catch (Exception e) {
        LOGGER.log(Level.SEVERE, "Error updating budget summary", e);
    }
}

private void updateBudgetSummary(double taxRate) {
    try {
        budgetManager.updateRemainingBudget();
        double totalSpent = budgetManager.calculateTotalSpent(taxRate);
        double remaining = budgetManager.getBudget() - totalSpent;

        jLabel5.setText(String.format("%.2f", remaining));
        UseTotal.setText(String.format("%.2f", totalSpent));
    } catch (Exception e) {
        LOGGER.log(Level.SEVERE, "Error updating budget summary", e);
    }
}
```

ภาพที่ 46 เมทอด updateBudgetSummary() ในคลาส showlist

7. เมทอด saveBudget ใช้บันทึกข้อมูลงบประมาณและรายการค่าใช้จ่าย ทั้ง 2 จะมีหน้าที่ต่างกันคือ saveBudget แรกจะบันทึกข้อมูลไปยังตาราง Budget ส่วน เมทอด saveBudget ที่สองจะบันทึกข้อมูลสินค้าไปยัง Budget\_item

```
public void saveBudget(int userId, String budgetName, double totalBudget, double remainingBudget) throws SQLException {
    try {
        dbManager.beginTransaction();

        String sqlBudget = "INSERT INTO budgets (userid, budget_name, total_budget, remaining_budget, created_at) " +
                           "VALUES (?, ?, ?, ?, NOW())";
        dbManager.executeUpdate(sqlBudget, userId, budgetName, totalBudget, remainingBudget);

        dbManager.commitTransaction();
    } finally {
        dbManager.disconnect();
    }
}

public void saveBudget(int userId, String budgetName, String itemName, int quantity, double price) throws SQLException {
    DatabaseManager dbManager = new DatabaseManager();
    try {
        dbManager.beginTransaction();

        String sqlItems = "INSERT INTO budget_items (userid, listname, itemname, quantity, price, created_at) " +
                           "VALUES (?, ?, ?, ?, ?, NOW())";
        dbManager.executeUpdate(sqlItems, userId, budgetName, itemName, quantity, price);

        dbManager.commitTransaction();
    } finally {
        dbManager.disconnect();
    }
}
```

ภาพที่ 46 เมทอด saveBudget () ในคลาส ChecklistManager

- ไม่มีการใช้แนวคิดนี้