N & N Group

SELLON ร้านค้าออนไลน์

ทีมพัฒนา

นาย ณัฏฐ์ จึงมาริศกุล รหัส 55010344 กลุ่มเรียน :1

(Mr. Nut Juangmarisakul)

นาย ธนภณ ซู รหัส 55010496 กลุ่มเรียน: 1

(Mr. Thanapon Soo)

Git Repository : online_shopping_system

เอกสารประกอบผลงานวิชา

Object-oriented Analysis and Design

ประจำ ภาคการศึกษาที่ 2557

บทคัดย่อ

SELLON : Online Shopping เป็นเว็บแอพลิชั่นสำหรับการขายสินค้าบนหน้าเว็บที่ใช้งานง่าย และมีหน้าตาเว็บที่สวยงาม ดึงดูดลูกค้าให้อยากที่จะซื้อสินค้าจากร้านค้าออนไลน์ พร้อมระบบติดตามสินค้าที่จะทำให้ลูกค้าสามารถทราบสถานะการส่งซื้อสินค้าได้ตลอดเวลา โดยเจ้าของร้านสามารถใส่รหัส EMS เพื่อแจ้งให้ลูกค้าทราบรหัสพัสดุที่จะใช้จัดส่งสินค้า เพื่อใช้ในการติดต่อขั้นตอนของพัสดุต่อไป ในมุมของเจ้าของร้านที่ต้องการบทสรุปวิเคราะห์ยอดขาย SELLON ยังมีระบบรายงานผลประกอบการแต่ละช่วงเวลาที่ผู้ขายต้องการ ที่ทำให้ผู้ขายสามารถวางแผนการประกอบการล่วงหน้าได้ จากการวิเคราะห์แนวโน้มที่เว็บแอพลิเคชั่นแสดงผลออกมา เพิ่มการปฏิสัมพันธ์ระหว่างเจ้าของร้าน และ ลูกค้า ด้วยระบบ live chat ที่สามารถทำให้ลูกค้าและ พ่อค้าสามารถติดต่อถึงกันได้ในทันที โดยมีระบบแสดงการ อยู่ในระบบของเจ้าของร้าน เพื่อให้ลูกค้าสามารถติดต่อในเวลาที่เข้าของร้านอยู่ในระบบได้ทันที SELLON ยังมีระบบ โปรโมชั่นที่เจ้าของร้านสามารถเลือกแผนการทำโปรโมชั่นได้เช่นกัน

บทนำ และ รายละเอียดการวิเคราะห์หัวข้อ

ในปัจจุบันผู้คนเริ่มเข้าถึงสื่อมัลติมีเดียผ่าน smart device หรือ คอมพิวเตอร์ได้มากขึ้น ซึ่งทำให้เกิดช่องทางการค้าขายใหม่ๆขึ้นมา ซึ่งกำลังเป็นที่นิยมเนื่องจากสะดวกสบาย ไม่ต้องไปเดินเลือกสินค้าเอง ร้านค้าต่างๆเมื่อมองเห็นช่องทางนี้จึงเลือกที่จะเปิดร้านค้าออนไลน์ เป็นช่องทางการขายอีกช่องทางหนึ่ง กันมากขึ้น เนื่องจากปัญหาที่พบในระบบร้านค้าออนไลส์ในปัจจุบันคือการปฏิสัมพันธ์โต้ตอบระหว่างลูกค้า และ พ่อค้ายังเป็นไปได้ยาก พ่อค้าแม่ค้าบางคนจึงใช้วิธีการแก้ปัญหาโดยการให้ลูกค้าติดต่อทาง Social media อื่นๆ ทีมผู้จัดทำได้มองเห็นว่า ในร้านค้าออนไลน์ควรต้องมีส่วนของระบบที่เข้ามาทำให้การติดต่อระหว่างเจ้าของร้าน และลูกค้าเป็นไปได้อย่างง่ายดายขึ้น เปรียบเสมือนการคุยกันที่หน้าร้านค้าจริงๆ ทีมผู้พัฒนาจึงเลือกพัฒนาห้องสนทนาระหว่างเจ้าของร้าน และ ลูกค้าขึ้นมาโดยพยามทำให้มีการตอบสนองแบบทันที (Real time) พร้อมทั้งแสดงสถานะการเข้าสู่ ระบบของเจ้าของร้าน เพื่อให้ลูกค้าสามารถติดต่อกับเจ้าของร้านในขณะที่เจ้าของร้านกำลังอยู่ในระบบ หรือลูกค้าจะฝากข้อความทิ้งไว้ก็สามารถทำได้เช่นกัน อย่างไรก็ตามทีมพัฒนาหวังเป็นอย่างยิ่งว่า SELLON จะสามารถแก้ปัญหาการปฏิสัมพันธ์ของเจ้าของร้านค้า และลูกค้าได้อย่างดี

งานที่เกี่ยวข้อง

- www.lnwshop.com/

  เป็นเว็บที่สร้างขึ้นมาเพื่อสร้างร้านค้าออนไลส์ ค่อนข้างครบวงจร แต่ข้อเสียคือเว็บค่อนข้างช้า

- www.shopup.com

  มีระบบโปรโมชั่น แต่ไม่มีระบบรายงานผล ไม่มีระบบแชท เว็บค่อนข้างเร็ว

  เรานำจุดที่เว็บไซต์ที่เกี่ยวข้องที่เราได้กล่าวขาดตกพร่องไปมาประกอบกันเป็นเว็บแอพพลิเคชั่นของเรา

ผลการวิเคราะห์ความต้องการของผู้ใช้ระบบ

Functional requirement

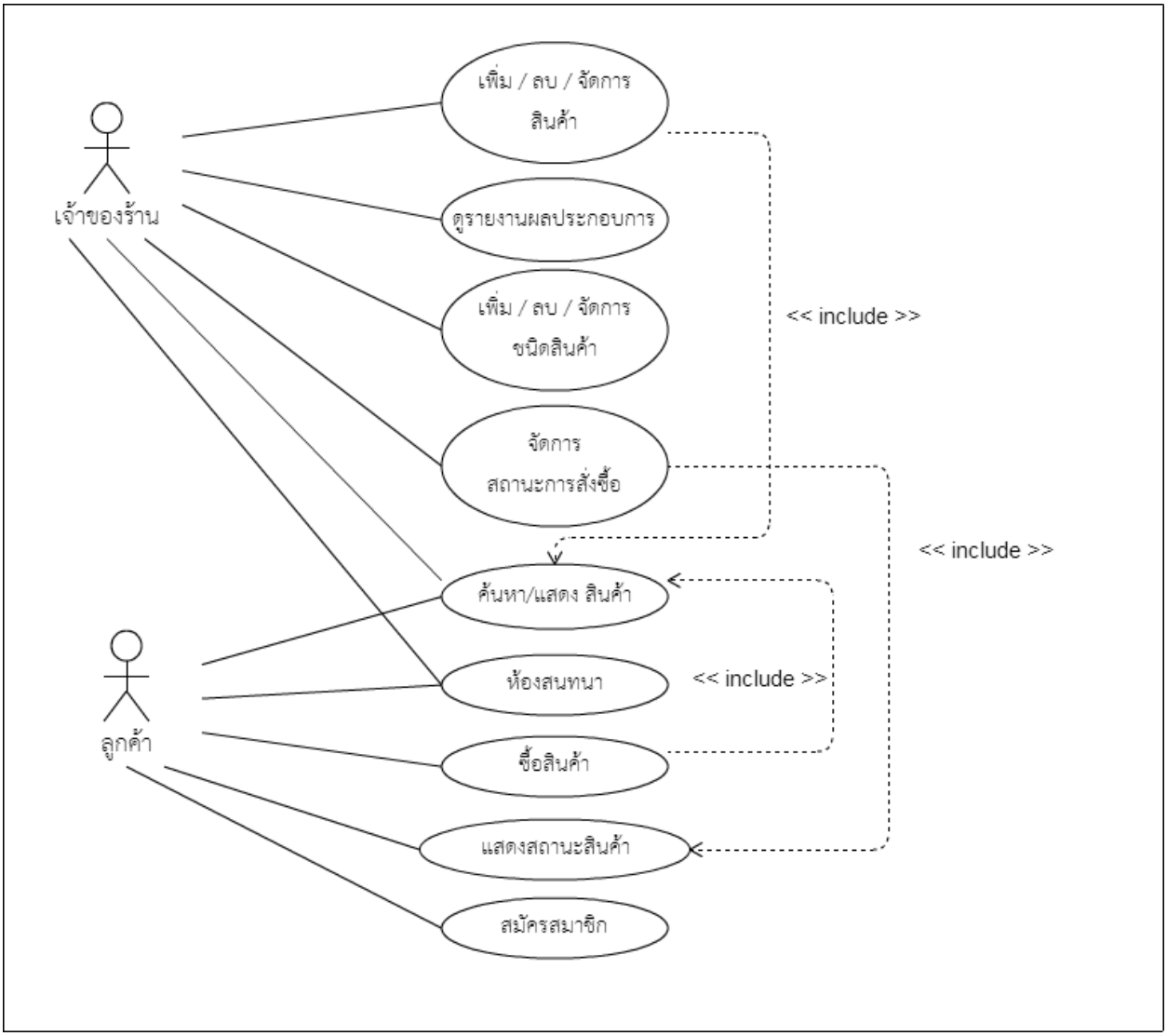
- ซื้อสินค้าผ่านหน้าเว็บ
- ตรวจสอบสถานะสินค้า
- ค้นหา และ แสดง สินค้าผ่านหน้าเว็บ
- แสดงผลประกอบการ โดยสามารถเลือกข้อมูลที่จะนำมาแสดงผลได้
- ระบบ เพิ่ม/ลบ/ จัดการ สินค้า , ชนิดสินค้า , คุณสมบัติสินค้า
- ระบบ สามารถเปลี่ยน สถานะสินค้า (order status) เพื่อให้ลูกค้าสามารถตรวจสอบได้เสมอว่า สินค้าอยู่ในขั้นตอนไหนแล้ว
- ระบบสามารถแนบรหัส EMS ลงในใบสั่งสินค้า (Order) เพื่อให้ลูกค้าสามารถติดตามพัสดุได้
- ระบบสนทนา(chat)ระหว่างเจ้าของร้าน และ ลูกค้า พร้อมแสดงสถานะของเจ้าของร้านว่า อยู่ในระบบ (online) หรือไม่
- ระบบส่วนลด และ โปรโมชั่น ที่สามารถแสดงสัดส่วน หรือเงื่อนไขการลด
- ระบบตระกร้าที่เก็บสินค้าที่ผู้ใช้เลือกแต่ยังไม่ซื้อ
- นักพัฒนาสามารถติดต่อกับฐานข้อมูลของร้านค้าผ่านทาง API ของระบบ

Non-functional requirement

- Accesibility
    - รองรับผู้ใช้พร้อมกันอย่างน้อย 10 คน
- Security
    - ส่งข้อมูลที่เป็นความลับโดยใช้ HTTPS
    - ป้องกัน Cross-site scripting
    - ป้องกัน Cross-site request forgery
    - ป้องกัน SQL Injeciton
- Usability
    - ผู้ใช้สามารถใช้งานได้จากอุปกรณ์ทุกอย่างที่มี Browser
    - หากลูกค้าเคยเข้าสู่ระบบแล้ว จะไม่ต้องเข้าสู่ระบบซ้ำอีก จนกว่าจะออกจากระบบ
- Privacy
    - มีระบบ Log in เพื่อรักษาข้อมูลของผู้ใช้งาน
- Reliability
    - หากลูกค้าทำการสั่งซื้อสินค้า ข้อมูลการสั่งซื้อจะไม่สูญหาย
- Integrity
    - ข้อมูลทุกอย่างที่ลูกค้าป้อนลงระบบ จะถูกนำไปแสดงผลอย่างแม่นยำ และถูกต้อง

แนวทางการใช้งาน

USE CASE DIAGRAM



Use Case Specification

1.Use case name: แสดงข้อมูลผลประกอบการ

- **Use case purpose:** เพื่อแสดงข้อมูลต่างๆที่สำคัญในรูปแบบของกราฟเพื่อให้เจ้าของร้านค้าสามารถนำไปประกอบการตัดสินใจในการวางแผนการค้า
- **Preconditions:** ผู้ใช้ที่จะดูข้อมูลผลประกอบการของร้านค้า จะต้องเป็นเจ้าของร้านเท่านั้น
- **Postconditions:** ผู้ใช้เห็นข้อมูลที่ต้องการในรูปแบบของกราฟ
- **Assumptions:**ผู้ใช้มีความเข้าใจในรูปแบบการแสดงผลของข้อมูล
- **Primary scenario:**
  A.ผู้ใช้เลือกข้อมูลที่ต้องการจะดู เช่น จำนวนสินค้าที่ขายได้ จำนวนผู้เข้าชม รายได้ และจำนวนรายการชำระเงิน เป็นต้น
  B.ผู้ใช้เลือกช่วงเวลาของยอดขายที่จะแสดง
  C.กราฟของยอดขาย ตามวันที่ ที่ระบุถูกแสดง
  D.จบ Use case
- **Alterntive scenario**:
  B1. ผู้ใช้เลือกช่วงเวลา และ เปลี่ยนให้กราฟแสดงผล ผลกำไร
  B2. กราฟของ ผลกำไร ตามวันที่ที่ระบุถูกแสดง
  B3. กลับสู่ขั้นตอน D

2.**Use case name**: แก้ไขสถานะการสั่งซื้อ

- **Use case purpose:** จุดประสงค์ของ use case นี้เพื่อ แก้ไขสถานะการสั่งซื้อของลูกค้า ในกรณีต่างๆ

- **Preconditions:**

  - ระบบต้องมีสินค้าที่ยืนยันการสั่งซื้อโดยลูกค้าเรียบร้อยแล้ว เจ้าของร้านจึงจะสามารถเข้ามาจัดการได้
  - ผู้ใช้ที่ต้องการใช้ use case นี้ต้องเข้าสู่ระบบในสิทธิ์ของเจ้าของร้าน
    นั้น

- **Postconditions:** สถานะการสั่งซื้อของลูกค้าจะถูกแก้ไข ตามที่ผู้ใช้ต้องการ
- **Limitation:** การแก้ไขสถานะการสั่งซื้อสินค้าจะไม่สามารถทำกับสินค้าเดียวกันในเวลาเดียวกันได้

- **Assumptions:** เจ้าของร้าน มีความเข้าใจในระบบ และ เปลี่ยนแปลงข้อมูลตามความจริง

- **Primary scenario:**

  A. ระบบแสดงสถานะการสั่งซื้อแสดงรายการของ รายการการสั่งซื้อทั้งหมด หรือ ผู้ใช้ค้นหารายการสั่งซื้อ

  B. เจ้าของร้านเลือกสถานะการสั่งซื้อ จากรายการที่ต้องการแก้ไข

  C. ระบบจะแสดงข้อมูลทั้งหมดของรายการสั่งซื้อนั้นๆ

  D. เจ้าของร้านทำการแก้ไข ข้อมูลของรายการการสั่งซื้อที่เจ้าของร้านเลือกและ กรอกข้อมูลอื่นๆเช่นรหัส EMS หากผู้ใช้ต้องการ

  E. ผู้ใช้ทำการยืนยันการแก้ไขข้อมูลรายการสั่งซื้อ

  F. จบ Use Case

- **Alternative scenario**:

  Condition : กรณีระบบตรวจสอบได้ว่าผู้ใช้ใส่รหัส EMS ผิดรูปแบบ (Format)

  D1. ระบบแจ้งเตือนผู้ใช้งานกรอกรูปแบบผิด

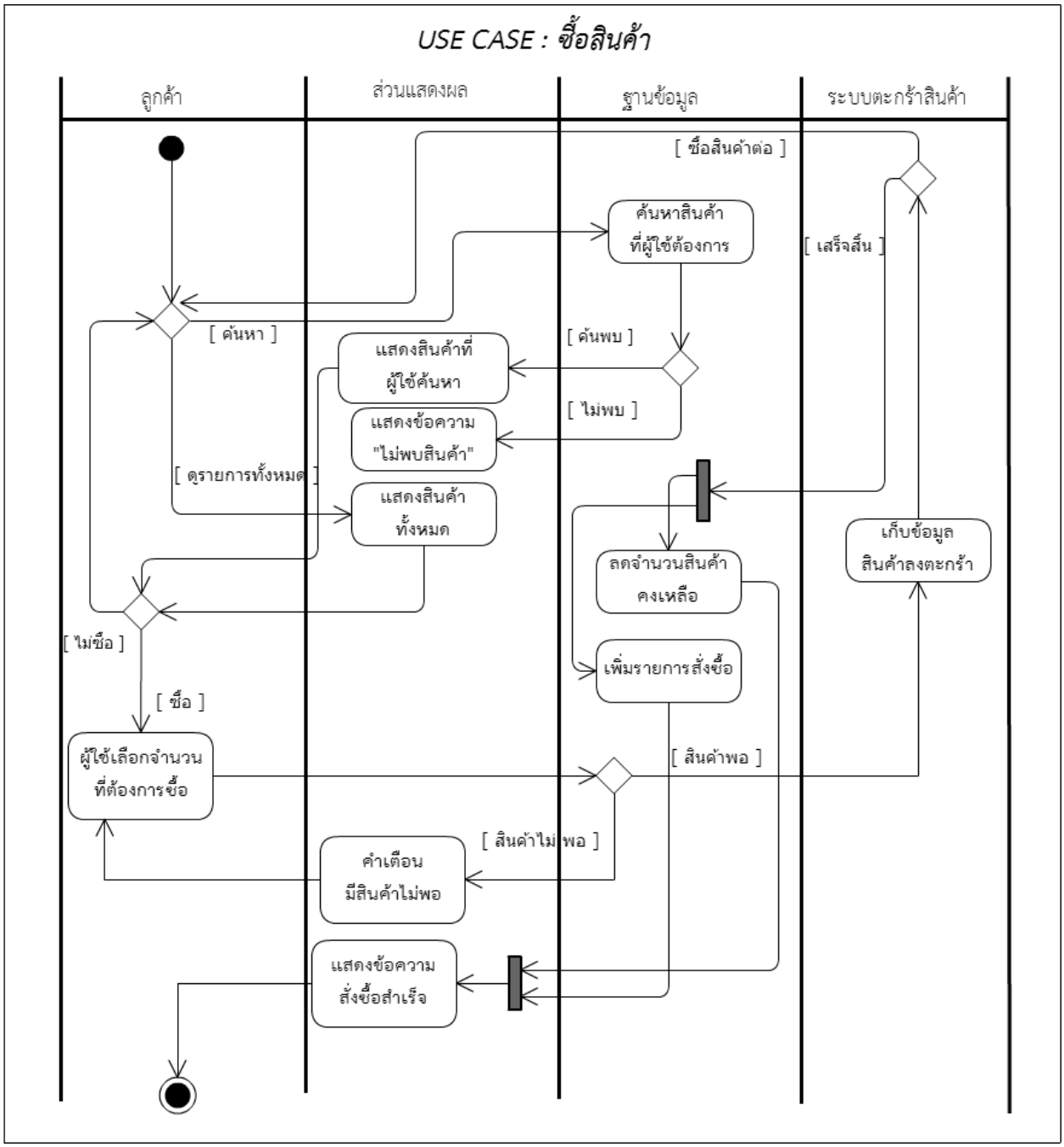
  D2. ระบบป้องกันไม่ให้ผู้ใช้ส่งข้อมูลจนกว่าจะกรอกข้อมูลถูกต้อง

  D3. ผู้ใช้กรอกข้อมูล EMS ในรูปแบบที่ถูกต้อง

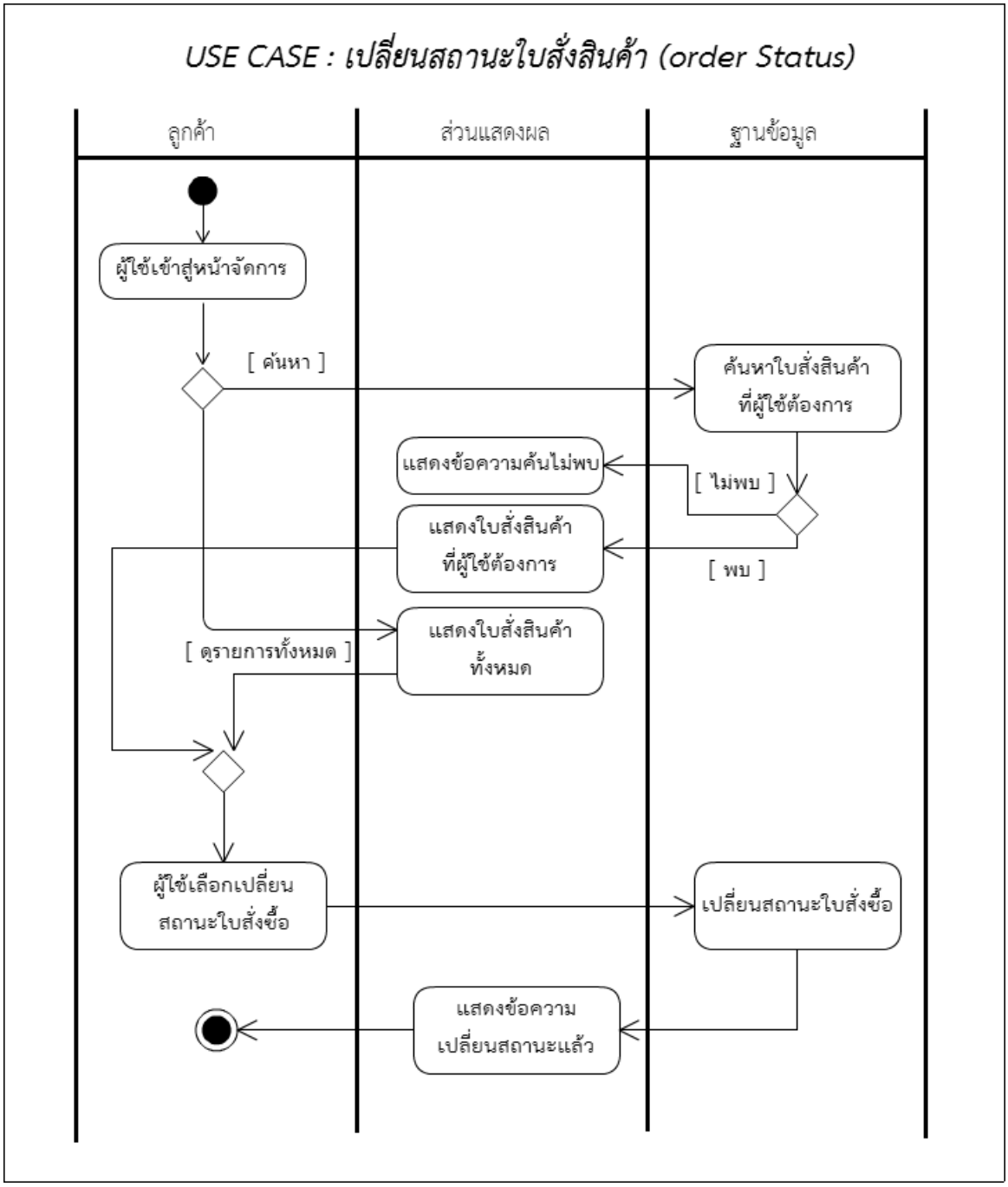
  D4. กลับสู่ขั้นตอน E

ACTIVITY DIAGRAM

(การซื้อสินค้า) Buying



การเปลี่ยนสถานะใบสั่งสินค้า (change Order Status)



สถาปัตยกรรมของระบบ

Abstraction :

เนื่องจากเป็นระบบร้านค้าออนไลส์ จึงทำการแบ่งส่วน Abstraction ตามหน้าที่การทำงาน แต่ละ Component ต่างมีหน้าที่การทำงานของตัวเอง ไม่ทับซ้อนกัน จึงสามารถแบ่ง Abstraction ต่างๆได้ดังนี้

- สินค้า (Product) ประกอบด้วย
    - ชื่อสินค้า
    - ชนิดสินค้า
    - ราคาสินค้า
    - รายละเอียดอื่นๆของสินค้า อาทิเช่น ขนาด (Size) และสี
- ลูกค้า (Customer) ประกอบด้วย
    - ชื่อลูกค้า
    - username ,password
    - ข้อมูลที่อยู่ลูกค้า
- การสั่งซื้อ (Order) ประกอบด้วย
    - เลขที่การสั่งซื้อ
    - รายการสินค้าที่ลูกค้าเลือกซื้อ
    - วันที่/เดือน/ปี
    - ลูกค้าที่สั่ง
    - สถานะการสั่งซื้อ

- ส่วนคำนวณ รายงานการขาย (Report) เพื่อวิเคราะห์ ค่าเฉลี่ยรายได้ , รายได้สูงสุด , สินค้าที่ทำรายได้สูงสุด

จากการวิเคราะห์ Abstraction จึงได้รายการ Component ดังนี้

- ส่วนย่อยเพิ่มสินค้า
- ส่วนย่อยแก้ไขสินค้า
- ส่วนย่อยลบสินค้า
- ส่วนย่อยแสดงสินค้า
- ส่วนย่อยสมัครสมาชิก
- ส่วนย่อยแก้ไขข้อมูล
- ส่วนย่อยเข้าสู่ระบบ
- ส่วนย่อยสร้างรายการการสั่งซื้อ
- ส่วนย่อยแก้ไขรายการสั่งซื้อ
- ส่วนย่อยลบรายการสั่งซื้อ
- ส่วนย่อยแสดงรายการสั่งซื้อ

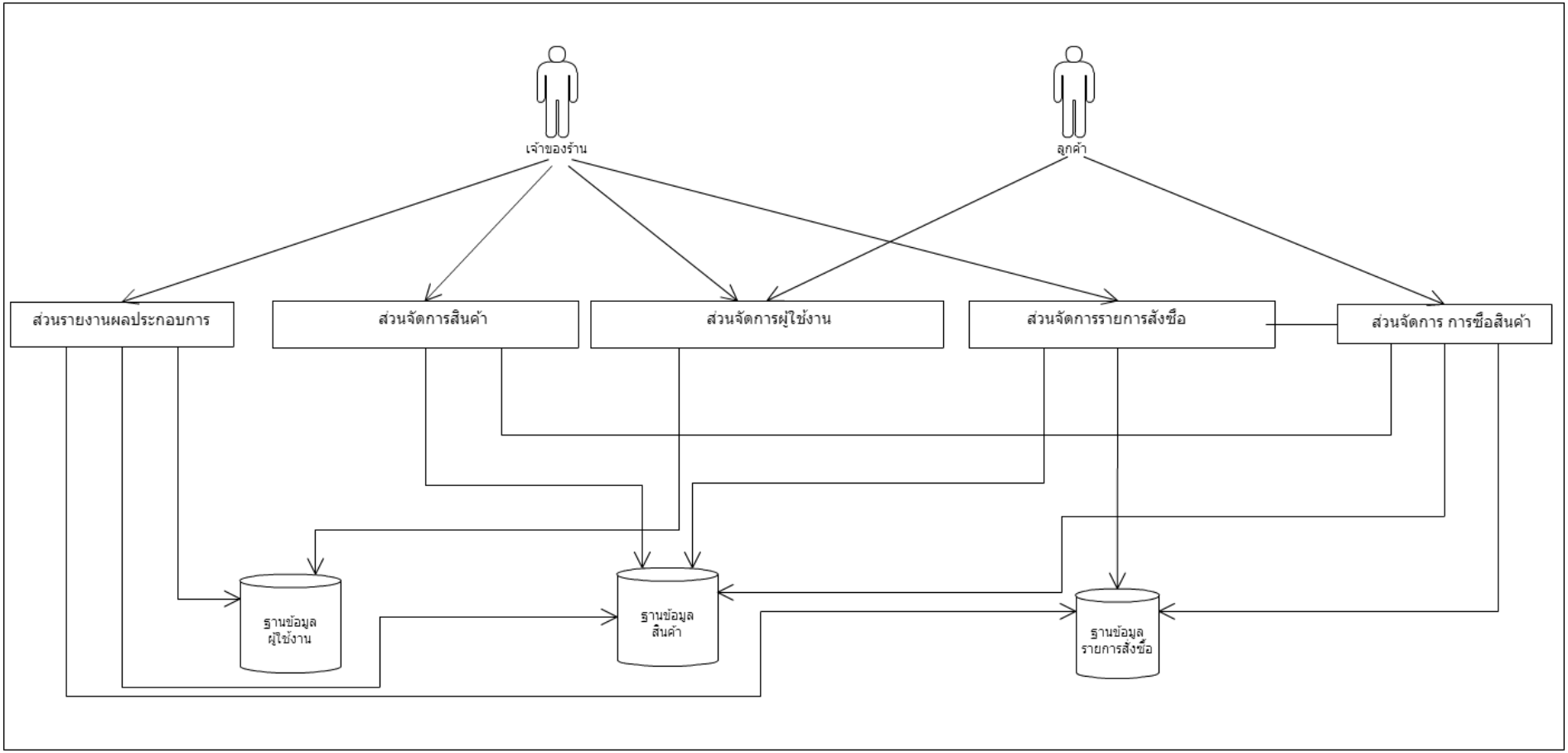
Application Architecture :

- ส่วนจัดการสินค้า
    - ส่วนจัดการสินค้ามีหน้าที่จัดการทุกอย่างที่มีส่วนเกี่ยวข้องกับสินค้า เป็นส่วนที่ติดต่อกับฐานข้อมูลสินค้า ซึ่งประกอบด้วยส่วนย่อยดังนี้
        - ส่วนย่อยเพิ่มสินค้า
        - ส่วนย่อยแก้ไขสินค้า
        - ส่วนย่อยลบสินค้า
        - ส่วนย่อยแสดงสินค้า
- ส่วนจัดการผู้ใช้งาน
    - ส่วนจัดการผู้ใช้งานมีหน้าที่จัดการทุกอย่างที่เกี่ยวข้องกับผู้ใช้งาน เป็นส่วนที่ติดต่อกับฐานข้อมูลผู้ใช้งาน ซึ่งประกอบด้วยส่วนย่อยดังนี้
        - ส่วนย่อยสมัครสมาชิก
        - ส่วนย่อยแก้ไขข้อมูล
        - ส่วนย่อยเข้าสู่ระบบ

- ส่วนจัดการรายการสั่งซื้อ
    - ส่วนจัดการรายการสั่งซื้อมีหน้าที่จัดการทุกอย่างที่เกี่ยวข้องกับผู้ใช้งาน เป็นส่วนที่ติดต่อกับฐานข้อมูลรายการสั่งซื้อ ซึ่งประกอบด้วยส่วนย่อยดังนี้
        - ส่วนย่อยสร้างรายการสั่งซื้อ
        - ส่วนย่อยแก้ไขรายการสั่งซื้อ
        - ส่วนย่อยแสดงรายการสั่งซื้อ
        - ส่วนย่อยลบรายการวั่งซื้อ

- ส่วนรายงานผลประกอบการ
    - เป็นส่วนที่รายงานผลประกอบการโดยดึงข้อมูลจาก ฐานข้อมูลรายการสั่งซื้อ ฐานข้อมูลผู้ใช้ ฐานข้อมูลสินค้า และนำมาประมวลผลเป็นผลประกอบการตามที่ผู้ใช้ต้องการ
- ส่วนจัดการ การซื้อ
    - เป็นส่วนที่มีหน้าที่เพื่อจัดการการซื้อของลูกค้า โดยต้องมีการติดต่อกับ ฐานข้อมูลสินค้า เพื่อดึงข้อมูลสินค้าที่ลูกค้าเลือก และ ฐานข้อมูลรายการสั่งซื้อเพื่อสร้างรายการสั่งซื้อหลังจากลูกค้าทำการยืนยันการสั่งซื้อ

Application Architecture Diagram



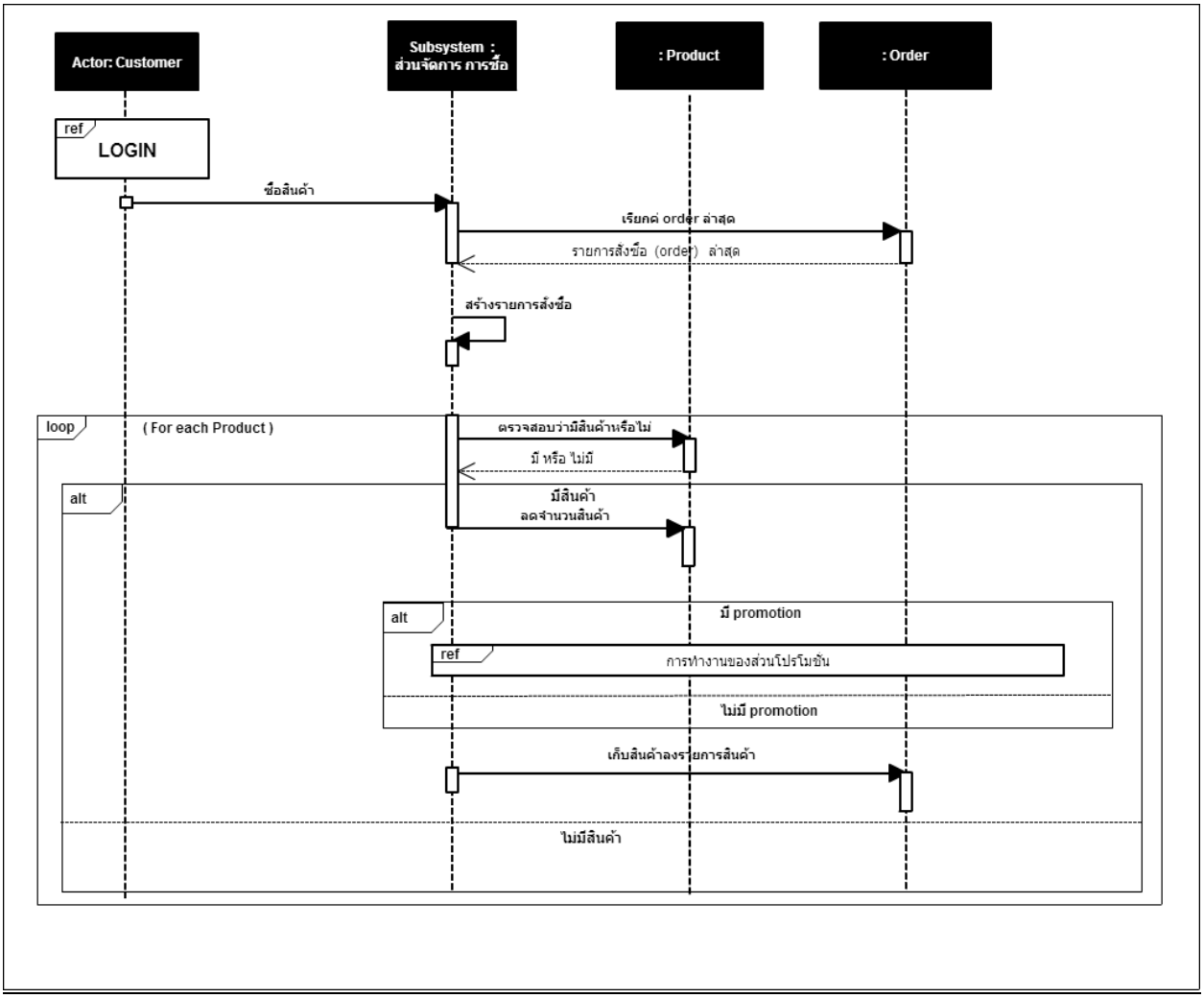
Subsystem / Component

- ส่วนจัดการสินค้า
    - ส่วนย่อยเพิ่มสินค้า : เพื่อเพิ่มสินค้าเข้าสู่ระบบ และเก็บลงฐานข้อมูลสินค้า โดยระบบจะตรวจจับห้ามเพิ่มสินค้าที่มีรายละเอียดที่เหมือนสินค้าที่มีอยู่แล้วทุกประการ
    - ส่วนย่อยแก้ไขสินค้า : เพื่อแก้ไขข้อมูลสินค้า อาทิเช่น ชื่อสินค้า ราคา และข้อมูลอื่นๆ และเก็บลงฐานข้อมูลสินค้า
      (การแก้ไขนี้จะไม่ส่งผลต่อ สินค้าที่ยืนยันการสั่งซื้อไปแล้ว ก่อนการเปลี่ยนแปลง)
    - ส่วนย่อยลบสินค้า : เพื่อลบสินค้าที่ต้องการออกจากฐานข้อมูลสินค้า
    - ส่วนย่อยแสดงสินค้า : เพื่อดึงรายการสินค้าทั้งหมดที่ต้องหาร หรือรายการสินค้าที่มีเงื่อนไขตรงตามที่กำหนด ออกมาแสดงในส่วนที่ต้องการ

- ส่วนจัดการผู้ใช้งาน
    - ส่วนย่อยสมัครสมาชิก : เพื่อสร้างผู้ใช้งานคนใหม่ และเก็บลงฐานข้อมูลผู้ใช้งาน
    - ส่วนย่อยแก้ไขข้อมูล : เพื่อแก้ไขข้อมูลผู้ใช้งาน หากผู้ใช้งานต้องการแก้ไขข้อมูลส่วนตัว อาทิเช่น ผู้ใช้ต้องการแก้ไขที่อยู่ในการส่งสินค้า
    - ส่วนย่อยเข้าสู่ระบบ : เพื่อให้ผู้ใช้ยืนยันตัวตน โดยส่วนย่อยนี้จะนำ username และ password ที่ผู้ใช้ป้อนเข้ามาไปตรวจสอบใน ฐานข้อมูลผู้ใช้งาน และมอบสิทธิ์ให้กับผู้ใช้ เพื่อใช้งานส่วนอื่น อาทิเช่น ส่วนจัดการ การซื้อ ที่จำเป็นต้องทำการยืนยันตะวตนเพื่อเข้าสู่ระบบก่อน
- ส่วนจัดการรายการสั่งซื้อ (Order)
    - ส่วนย่อยสร้างรายการการสั่งซื้อ : เพื่อสร้างรายการสั่งซื้อ โดยส่วนนี้จะถูกเรียกโดยส่วนจัดการ การซื้อสินค้า
    - ส่วนย่อยแก้ไขรายการสั่งซื้อ : เพื่อให้เจ้าของร้านสามารถเปลี่ยนแปลง รายการสั่งซื้อได้
        - หากเจ้าของร้านต้องการเปลี่ยนสถานะรายการสั่งซื้อ เพื่อแสดงถึงสถานะปัจจุบันของสินค้า เช่น เปลี่ยนสถานะจาก ยังไม่ชำระเงินเป็น ชำระเงินแล้ว
        - เจ้าของร้านต้องการเพิ่มรายละเอียดของ สถานะการสั่งซื้อ เช่น
          เจ้าของร้านต้องการ ใส่เลข EMS เพื่อให้ลูกค้าสามารถตามพัสดุสินค้าได้
    - ส่วนย่อยแสดงรายการสั่งซื้อ : เพื่อดึงข้อมูลรายการการสั่งซื้อของลูกค้าทั้งหมดที่ต้องการ หรือแสดงตามเงื่อนไขที่ต้องการ เพื่อไปแสดงผลในส่วนต่างๆ

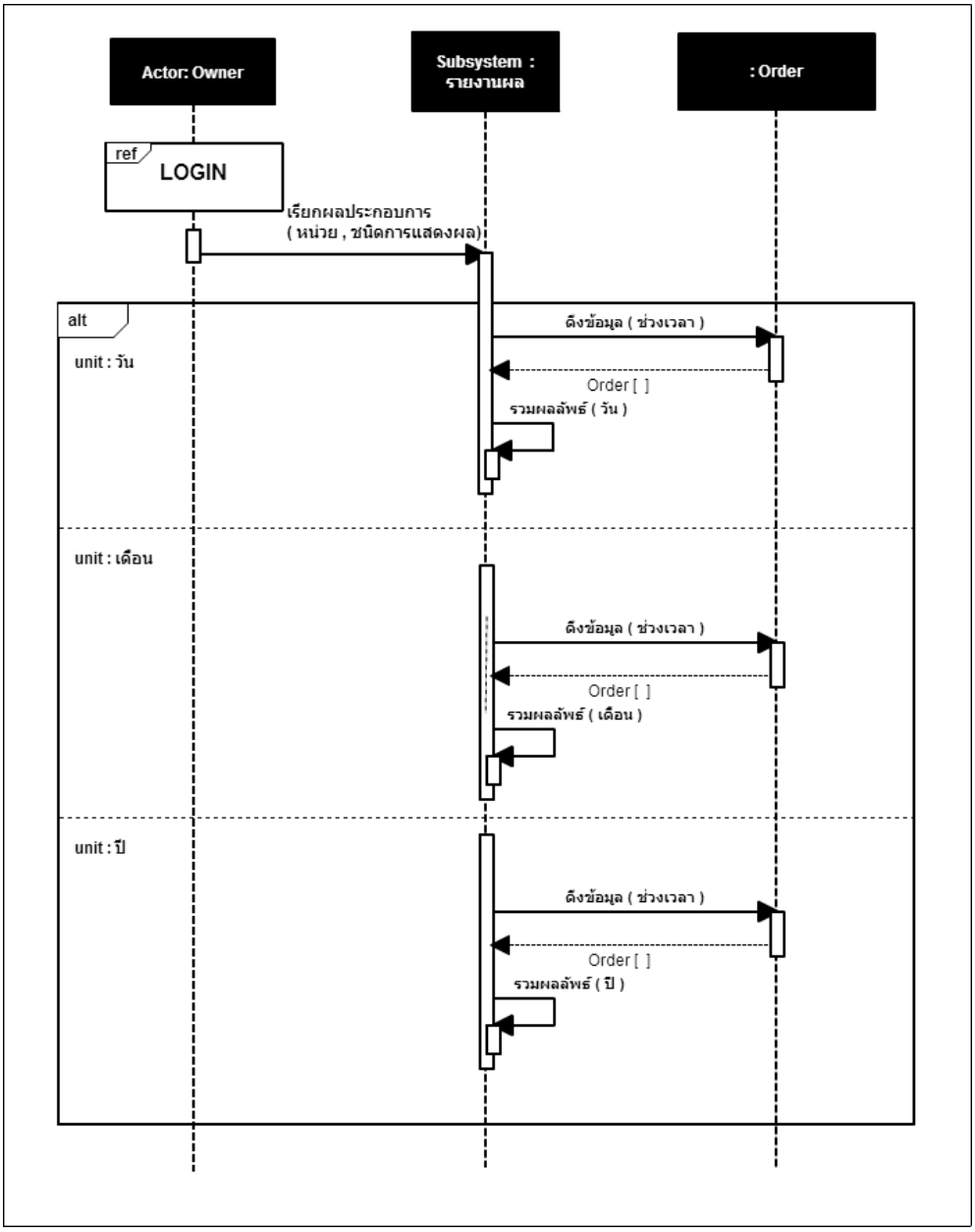
- ส่วนจัดการ การซื้อ
    - ส่วนนี้ทำหน้าที่จัดการ การซื้อของลูกค้าทั้งหมด โดยหากลูกค้าทำการซื้อสินค้า ระบบจะทำกระบวนการดังนี้
        - ระบบจะเข้าไปสร้างรายการสั่งซื้อ ในฐานข้อมูล รายการสั่งซื้อ (Order) โดยจะเก็บ
            - id สินค้าทั้งหมดที่ลูกค้าเลือกซื้อ เพื่อใช้โยงข้อมูลในภายหลัง
            - เก็บ id ของลูกค้า เพื่อใช้แสดงผลในกรณีลูกค้า ต้องการทราบสินค้าที่ผู้ใช้ซื้อไปทั้งหมด หรือ ใช้เป็นเงื่อนไขในการได้โปรโมชั่นของลูกค้า (ระบบโปรโมชั่นไม่ใช่ระบบหลักจึงขอละไว้ ไม่ได้กล่าวถึงในเอกสารนี้)
            - เก็บวันเวลาที่ผู้ใช้ยืนยันการสั่งซื้อ เพื่อใช้ในระบบ รายงานผลประกอบการ
        - ระบบจะเข้าไปลดจำนวนสินค้าใน ฐานข้อมูลสินค้า (Product) ตัวเลขของจำนวนสินค้านี้ที่เปลี่ยนแปลงไปนี้ จะไปแสดงผลในหน้าเลือกสินค้า ทำให้ลูกค้าทราบถึงจำนวนสินค้าคงเหลือที่แท้จริง
- ส่วนรายงานผลประกอบการ
    - เป็นส่วนที่รายงานผลประกอบการโดยดึงข้อมูลจาก ฐานข้อมูลรายการสั่งซื้อ(Order) ฐานข้อมูลผู้ใช้ (User) ฐานข้อมูลสินค้า (Product) และนำมาประมวลผลเป็นผลประกอบการตามที่ผู้ใช้ต้องการ ซึ่งรายงานผลประกอบการจะประกอบด้วย
        - จำนวนสินค้าที่ขายได้ในหน่วยวัน/สัปดาห์/ปี ต่อระยะเวลาที่ต้องการ
          เช่น จำนวนสินค้าที่ขายได้ในแต่ละวัน ต่อ 1 เดือน
        - รายได้ที่ทำได้ในหน่วยวัน/สัปดาห์/ปี ต่อระยะเวลาที่ต้องการ
        - กำไรที่ทำได้ในหน่วยวัน/สัปดาห์/ปี ต่อระยะเวลาที่ต้องการ

Sequence Diagram



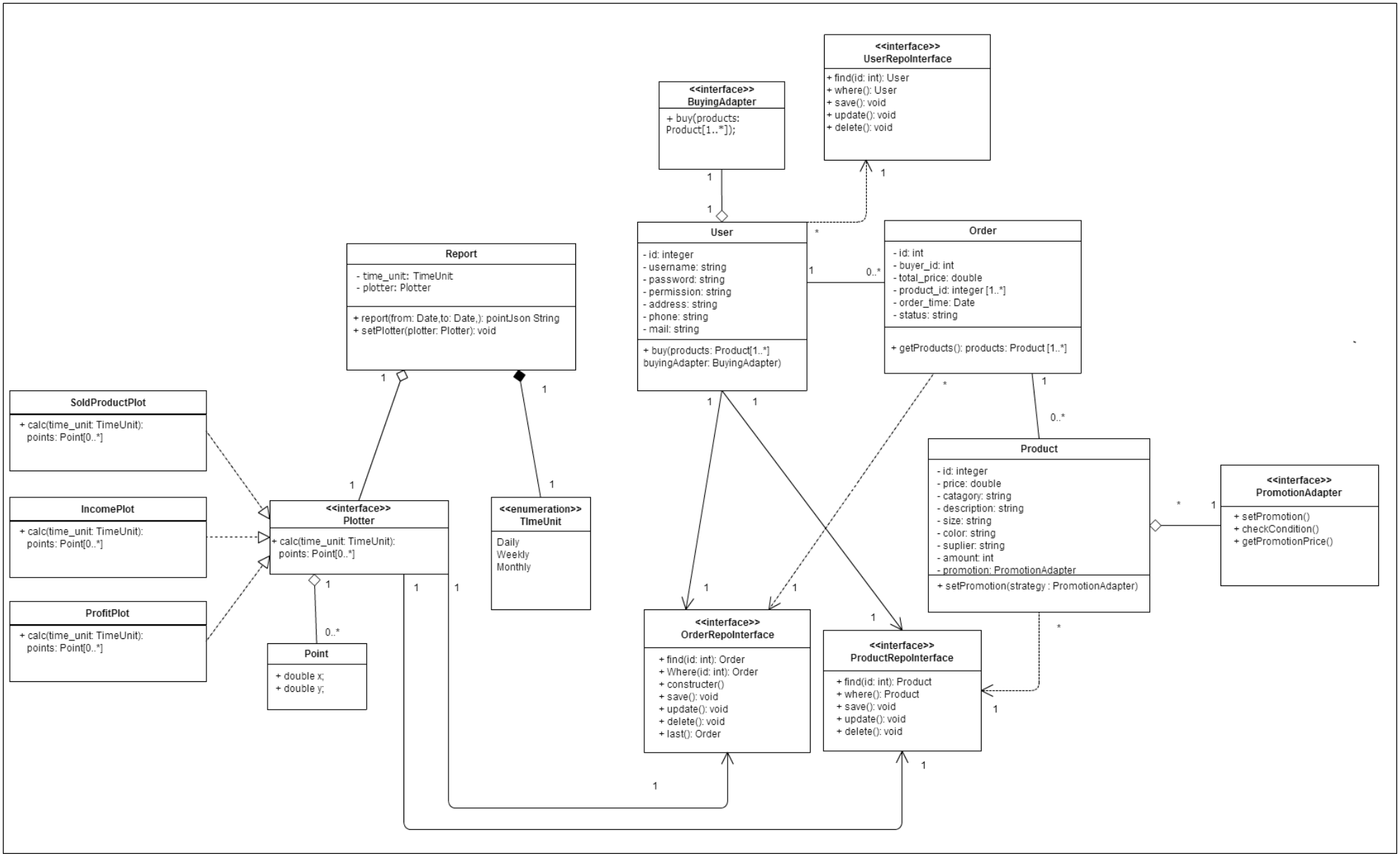
ภาพ sequence diagram แสดงถึงกระบวนการทำงานของการสั่งซื้อสินค้า โดยผู้ใช้งานจะเรียกส่วนจัดการการซื้อ หลังจากนั้นส่วนจัดการการซื้อจะทำการดึง Order สุดท้าย หรือ Order ล่าสุดเพื่อนำมา + 1 และสร้างเป็น id ของ Order ใหม่ เมื่อได้ id แล้วจึงสร้าง Order ใหม่หลังจากนั้นจะเข้าลูป เพื่อกระทำการซื้อกับสินค้าทุกชิ้นในตระกร้า(Cart)ที่ลูกค้ายืนยันการสั่งซื้อมาแล้ว ซึ่งมีขั้นตอนการตรวจสอบว่าสินค้าชนิดที่ลูกค้าต้องการมีหรือไม่ หากมีจะเริ่มทำกระบวนการซื้อ โดยมีการกระทำ 2 อย่างคือ

- ไปลดจำนวนสุทธิ (amount of product) ของสินค้าชนิดนั้นๆ
- หากมีโปรโมชั่นจะเข้าไปทำในส่วนโปรโมชั่นเพื่อเปลี่ยนแปลงราคาสินค้านั้นก่อนนำสินค้าเก็บลง Order
- ไปสร้างรายการสั่งซื้อจากสินค้าที่ลูกค้าซื้อ โดยจะเก็บ id ของ Order ใหม่ที่สร้างตอนต้นกระบวนการ กับ id สินค้าชนิดนั้น



ภาพ sequence diagram แสดงถึงกระบวนการเรียกผลประกอบการ ผู้ใช้จะส่งไปที่ส่วนรายงานผลโดยส่งค่าไปด้วยอีก 2 ตัวประกอบด้วย หน่วย ที่จะบ่งบอกว่าต้องการแสดงผลในหน่วย วันหรือ เดือนหรือปี ตัวต่อมาคือ ชนิดการแสดงผล เพื่อบ่งบอกว่าต้องการให้แสดงผลอะไร อาทิเช่น จำนวนสินค้าที่ขายได้ในหน่อยเวลานั้นๆ ,รายได้ และอื่นๆ หลังจากนั้น ส่วน Order ซึ่งเป็นส่วน data-mapper กับฐานข้อมูล ระบบจะส่งชุดข้อมูลของ Order กลับมา หลังจากที่ส่วนรายงานผลได้รับ ชุดข้อมูลของ Order แล้วก็จะไปคำนวณเป็นผลรวมของสิ่งที่ผู้ใช้ต้องการ เช่น ผลรวมของรายได้ที่เก็บใน Order ที่มีวันที่เดียวกัน หลังจากนั้นจะนำไปพล็อตกราฟเพื่อแสดงผล

CLASS DIAGRAM



รายละเอียดของการออกแบบ Class เพิ่มเติม

- การออกแบบ Class ที่เป็นส่วนของการเก็บข้อมูล ทีมพัฒนาใช้ **Repository Pattern** ในการออกแบบ และออกแบบให้มีการส่ง object ของคลาสที่จัดการรับข้อมูลไปติดต่อกับ database ในที่นี้ใช้ Eloquent ของ Laravel ส่งเข้ามาเพื่อกระทำการใดๆ โดยมี Class ที่สร้างขึ้นโดยทีมพัฒนาครอบไว้อีกที สังเกตุว่าจะมี Interface ของ Class ที่ทำหน้าที่ติดต่อฐานข้อมูลทุก Class เพื่อให้มีการ implement adapter ของ Eloquent หรือ data mapper class อื่นๆหากต้องการเปลี่ยนแปลงอาทิเช่น Doctrine
    - คลาสที่ใช้ Pattern ในการออกแบบรูปแบบนี้คือ
        - Product
        - Order
        - User
- การออกแบบในส่วนของ Promotion ได้ใช้หลักการของ **Strategy Pattern** ในการออกแบบเพื่อให้ผู้พัฒนาสามารถโปรโมชั่นรูปแบบใหม่ๆได้เรื่อยๆโดยไม่ต้องเข้าไปยุ่งเกี่ยวกับส่วนปฏิบัติการต่างๆ เช่นส่วนซื้อสินค้าที่ต้องมาตรวจสอบ และ ดึงราคาสินค้าโปรโมชั่น โดยส่วน Class ของโปรโมชั่นต่างๆจะ implement PromotionAdapter ซึ่งเป็นตัวจัดการตรรกะของโปรโมชั่น และ ตรวจสอบเงื่อนไขการได้โปรโมชั่นต่างๆ ซึ่งสุดท้ายส่วนปฏิบัติการจะมาเรียกใช้ Adapter เหล่านี้อีกทีผ่าน method ที่ถูกกำหนดไว้ใน Interface PromotionAdapter แล้ว

- การออกแบบในส่วนของ Live chat ใช้หลักการของ **Observer Pattern** เพื่อรอรับ Event ที่ Socket Server ตรวจเจอซึ่งก็คือการเข้าสู่ห้องแชทของ User ไปจนถึงการส่งข้อความหากันระหว่างผู้ใช้งาน และเจ้าของร้าน

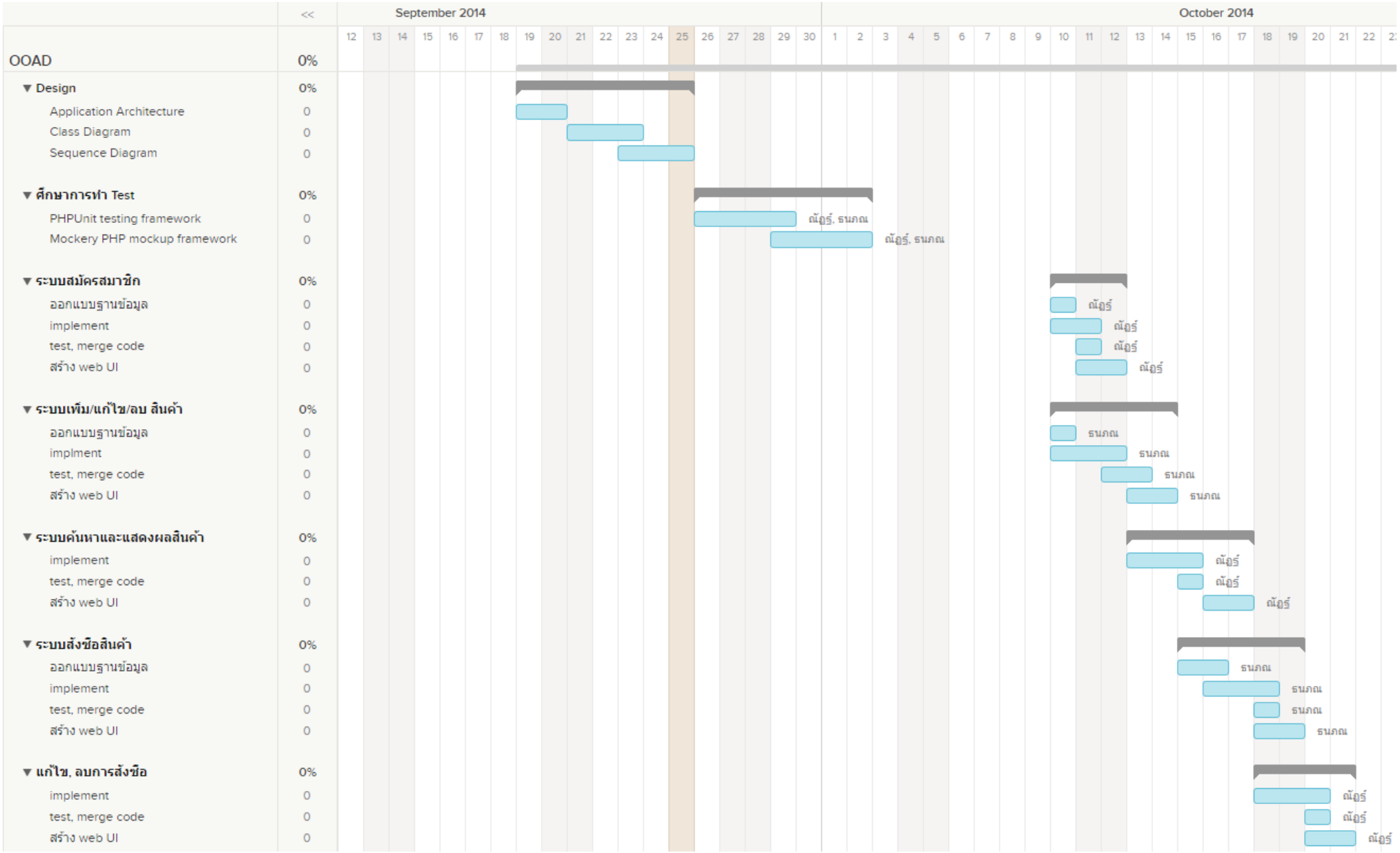
- class Report จะทำหน้าที่รับคำสั่งการ plot กราฟสำหรับรายงาน โดยจะรับค่าช่วงวันที่ที่ต้องการดูกราฟและรับค่าความถี่ของข้อมูล (รายวัน, รายสัปดาห์, รายเดือน) และ return JSON สำหรับให้ client-side นำไปพล็อตกราฟ ภายใน Report จะมี class Plotter แบบต่างๆอยู่ เพื่อให้การพล็อตกราฟแต่ละแบบสามารถถูกเรียกใช้ผ่าน interface เดียวกันคือ class Report

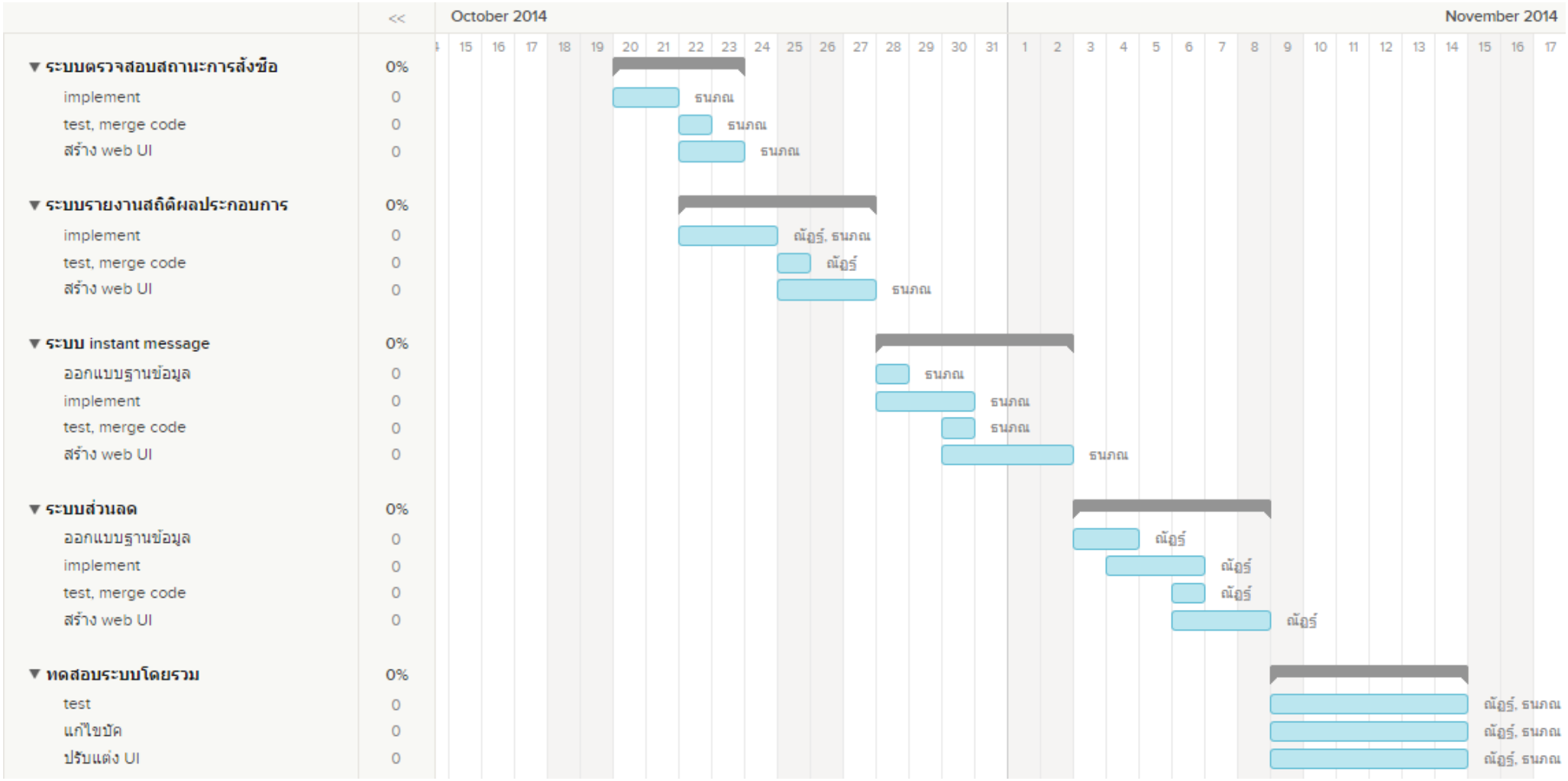
- การสั่งซื้อ จะเรียกผ่านฟังก์ชัน buy ของ object user ซึ่งรับค่า object product ที่ต้องการจะซื้อ และ BuyingAdapter โดย BuyingAdapter จะมีฟังก์ชัน buy เป็นส่วนที่ทำงานใน logic ของการซื้อ เช่น คำนวณราคา, ลดจำนวนสินค้าในคงเหลือลง เป็นต้น

Deployment

- **system**
    - VM ติดตั้งระบบปฏิบัติการ Ubuntu 14.04 LTS
    - Apache web server สำหรับการประมวลผลหน้าเว็บ
    - PHP5.5 เราจะใช้ PHP ในการ implement logic ต่างๆของระบบ
    - MySQL5.5 เป็นฐานข้อมูลที่ทำการเก็บข้อมูลต่างๆเอาไว้
- **framework**
    - Laravel PHP web application framework
        - ทำหน้าที่รับ request จาก browser และเรียกใช้ core ของ application โดยเราเลือกใช้
        - โดยเลือกใช้ packages ต่างเพื่อช่วยในการพัฒนาดังนี้
            - cboden/ratchet และ brainboxlabs/brain-socket
              สำหรับการทำ real time และ เปิด socket server
    - Bootstrap framework สำหรับการพัฒนาส่วน front end
    - CSS3 , HTML5 Technology
- **Testing**
    - PHPUnit testing framework สำหรับการทำ unit test
        - Mockery ใช้ในการสร้าง Mock เพื่อทดสอบระบบ

ตารางเวลาการทำงาน(Implementation Plan)




<<
October 2014
November 2014
14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
▼ ระบบตรวจสอบสถานะการสั่งซื้อ 0%
implement 0 ธนภณ
test, merge code 0 ธนภณ
สร้าง web UI 0 ธนภณ
▼ ระบบรายงานสถิติผลประกอบการ 0%
implement 0 ณัฏฐ์, ธนภณ
test, merge code 0 ณัฏฐ์
สร้าง web UI 0 ธนภณ
▼ ระบบ instant message 0%
ออกแบบฐานข้อมูล 0 ธนภณ
implement 0 ธนภณ
test, merge code 0 ธนภณ
สร้าง web UI 0 ธนภณ
▼ ระบบส่วนลด 0%
ออกแบบฐานข้อมูล 0 ณัฏฐ์
implement 0 ณัฏฐ์
test, merge code 0 ณัฏฐ์
สร้าง web UI 0 ณัฏฐ์
▼ ทดสอบระบบโดยรวม 0%
test 0 ณัฏฐ์, ธนภณ
แก้ไขบัค 0 ณัฏฐ์, ธนภณ
ปรับแต่ง UI 0 ณัฏฐ์, ธนภณ

UNIT TEST

- Product Repository Test

การทดสอบระบบในส่วนของ Product ซึ่งเป็นการทดสอบ function ต่างๆใน \core\Product (class product ที่สร้างเอง) และลอง Mock \Product (class product ที่ extend Eloquent) แล้วส่งเข้าไปทำการเซฟ object \Product

- directory อยู่ที่ online_shopping_system/app/tests/
- โค้ด

```php
<?php
	use \core\Product as Product;
	use \core\EloProductRepo as EloProductRepo;

	Class ProductRepoTest extends TestCase {
		protected $product_repo;
		protected $product;
		protected $mock_product_elo;

		public function setUp() {
			parent::setUp();

			Artisan::call('migrate'); // Call migration and seed database
			$this->seed();
		}
		public function teardown() {
			Mockery::close(); // clear all Mock.
		}
		public function testCreateProduct() {

			$this->product = Mockery::mock('\core\Product'); //Create mocking Product Object
			//all getter method test
			$this->product->shouldReceive('getPrice')->once();
			$this->product->shouldReceive('getCategory')->once();
			$this->product->shouldReceive('getDescription')->once();
			$this->product->shouldReceive('getSize')->once();
			$this->product->shouldReceive('getColor')->once();
			$this->product->shouldReceive('getSuplier')->once();
			$this->product->shouldReceive('getAmount')->once();
			$this->product->shouldReceive('getProductName')->once();
			$this->product->shouldReceive('getImgPath')->once();
			//Mocking Model Product only call save method.
			$this->mock_product_elo = Mockery::mock('\Product[save]');
			//create product_repo and pass adapter that really call save method.
			$this->product_repo = new EloProductRepo($this->mock_product_elo);
			// Tell Laravel to use mock_product_elo instead of \Product
			$this->app->instance('\Product',$this->mock_product_elo);
			$this->mock_product_elo->shouldReceive('save')->once();
			//pass mocking product object
			$this->product_repo->save($this->product);
		}
	}
```

- ผลการทดสอบ

```
n1ghtingale@sagittarius:/var/www/html/sellon$ phpunit app/tests/ProductRepoTest.php
PHPUnit 3.7.37 by Sebastian Bergmann.

Configuration read from /var/www/html/sellon/phpunit.xml

.

Time: 284 ms, Memory: 15.00Mb

OK (1 test, 0 assertions)
```

- **Buying Test**

  ทดสอบระบบซื้อสินค้า โดยลองสร้าง object ของ class \core\Product แล้วทำการซื้อ จากนั้นดูว่า Order ที่เกิดขึ้นถูกต้องหรือไม่

  - directory อยู่ที่ online_shopping_system/app/tests/
  - โค้ด

```php
<?php

class BuyingTest extends TestCase

 {
   public function setUp()
   {
     parent::setUp();
     Artisan::call('migrate');
     $this->seed();
   }
   public function testBuying()
   {
     $userRepo = new core\EloUserRepo(new \User());
     $productRepo = new core\EloProductRepo(new \Product());
     $orderRepo = new core\EloOrderRepo(new \Order);
     $user = $userRepo->first();
     $buyingAdapter = new core\DefaultBuyingAdapter($orderRepo, $productRepo);
     $products = array();
     array_push($products, $productRepo->find(1)->setAmount(1), $productRepo->find(2)->setAmount(1),
 $productRepo->find(3)->setAmount(1));
     $user->buy($products, $buyingAdapter);
     $total_price = 0.0;
     foreach($products as $product)
     {
       $total_price = $total_price + $product->getPrice();
     }
     $order = $orderRepo->find(1);
     $this->assertEquals($order->getTotal_price(), $total_price);
   }
 }
```

- ผลการทดสอบ

```
n1ghtingale@sagittarius:/var/www/html/sellon$ phpunit app/tests/BuyingTest.php
PHPUnit 3.7.37 by Sebastian Bergmann.

Configuration read from /var/www/html/sellon/phpunit.xml

.

Time: 255 ms, Memory: 14.25Mb

OK (1 test, 1 assertion)
```

- All Test

ระบบยังมี test ระบบอื่นๆอีกอาทิรวมเป็น 6 tests ซึ่งได้ผลดังต่อไปนี้

```
n1ghtingale@sagittarius:/var/www/html/sellon$ phpunit
PHPUnit 3.7.37 by Sebastian Bergmann.

Configuration read from /var/www/html/sellon/phpunit.xml

......

Time: 927 ms, Memory: 22.00Mb

OK (6 tests, 7 assertions)
```

EVALUATION

การทดลองระบบจัดการสินค้า

จุดประสงค์ในการทดลอง : เพื่อทดสอบระบบสินค้าว่าสามารถใช้งานได้จริง

สิ่งที่จะวัด : หากระบบสามารถทำการเพิ่มสินค้าได้ เมื่อเพิ่มรายการเสร็จ จะมีสินค้าชนิดนั้นๆ อยู่ที่หน้าแสดงรายการสินค้าทั้งหมด

วิธีทำการทดลอง : การทดลองสามารถทำได้โดยการทดลองกับระบบจริงๆ โดยมีสิ่งที่ต้องใช้ในการทดลองระบบดังนี้

- Production server environment
- ข้อมูลรูปแบบต่างที่เป็นไปได้ในการใช้งานจริง พร้อมรูปภาพขนาดต่างๆ
- Browser สำหรับการทดลอง

โดยทำการทดลองดังนี้

1 ทดลองใส่ข้อมูลจำลอง ที่เตรียมเอาไว้

2 อัพโหลดไฟล์ภาพของสินค้าชนิดนั้นๆ

3 ทำการยืนยัน เพื่อบันทึกสินค้าลงบนฐานข้อมูล

4 ทดสอบกลับไปที่หน้าแสดงผลสินค้า แล้วตรวจสอบว่ามีสินค้าเพิ่งเพิ่มเข้าไปในระบบหรือไม่

5 ทดสอบการแก้ไข สินค้า โดยเลือกสัญลักษณ์การแก้ไขสินค้าในรายการสินค้า

6 ทดสอบแก้ไขสินค้าใน ฟิลด์ ต่างๆ แล้วทำการยืนยัน

7 ที่หน้าแสดงผลสินค้า ลองตรวจสอบดูว่าข้อมูลแก้ไขตามที่ต้องการหรือไม่

8 ทดสอบการลบสินค้า โดยการคลิกที่ปุ่มลบสินค้า

9 หลังจากลบสินค้ารูปของสินค้านั้นก็ควรจะต้องหายไปด้วย ลองไปเช็คว่ายังมีรูปสินค้านั้นในไดเรกทอรี่รูปภาพ หรือไม่

10. ทดสอบเพิ่มโปรโมชั่นให้สินค้า โดยเข้าไปกรอกข้อมูลโปรโมชั่นทั้งสองรูปแบบ

11. ตรวจสอบโดยการกลับไปดูที่หน้าแสดงสินค้า มีการแสดงโปรโมชั่นหรือไม่

12 ทำการทดสอบขั้นตอน 1 - 11 กับข้อมูลในรูปแบบต่างๆกันให้ครบทุกรูปแบบ

ผลการทดลอง

สามารถทำการ เพิ่ม ลบ แก้ไข สินค้าได้โดยข้อมูลเปลี่ยนไปจริง รวมไปถึงสามารถเพิ่มโปรโมชั่นให้กับสินค้าได้ โดยข้อมูลที่แสดงผลหน้าแสดงสินค้าเปลี่ยนตามเช่นกัน

สรุปสิ่งที่ได้จากการทดลอง

สามารถมั่นใจได้ว่าระบบ การเพิ่ม/ลบ/แก้ไข สินค้าสามารถทำได้จริง รวมไปถึงการเพิ่มโปรโมชั่นให้สินค้าก็สามารถทำได้จริง

การทดลองระบบสั่งซื้อสินค้า

จุดประสงค์ในการทดลอง เพื่อทดสอบความถูกต้องในการทำงาน ของระบบการสั่งซื้อ

สิ่งที่จะวัด : หากผู้ใช้สามารถทำการซื้อสินค้าได้ จะมีรายการสั่งซื้อสินค้าเกิดขึ้นในระบบ โดยที่ราคารวมของสินค้าและวันเวลาการสั่งซื้อ จะต้องตรงตามข้อมูลจริง

วิธีทำการทดลอง : การทดลองสามารถทำโดย ให้ผู้ใช้ทดลองใช้งานระบบจริงๆ โดยมีสิ่งที่ต้องใช้ในการทดลองระบบดังนี้

1.Production server environment

2.ข้อมูลสินค้าในระบบสำหรับทำการทดลองซื้อ

3.Browser สำหรับการทดลอง

โดยทำการทดลองดังนี้

1.ให้ผู้ใช้ที่มีสิทธิ์เป็นลูกค้ากดสั่งซื้อสินค้า

2.ระบุจำนวนที่ต้องการสั่งซื้อ

3.ทดลองกลับออกไปเลือกสินค้าอื่นๆ เช่น สินค้าที่มีโปรโมชั่น

4.ตรวจสอบราคาเป็นราคาโปรโมชั่นหรือไม่

4.ยืนยันการสั่งซื้อสินค้าทั้งหมด

5.ผู้ใช้ที่มีสิทธิ์เป็นเจ้าของร้านเข้าสู่ระบบ

6.ตรวจสอบว่ามีการสั่งซื้อใหม่เกิดขึ้นจริง

7.ตรวจสอบว่าจำนวนสินค้าคงเหลือในระบบลดลงตามจำนวนที่ลูกค้าสั่งซื้อ

ผลการทดลอง

สามารถซื้อสินค้าจากหน้าร้าน สามารถกลับไปเลือกสินค้าต่อจนกว่าจะพอใจ แล้วกลับมายืนยันการสั่งซื้อ เมื่อลองสังเกตุจำนวนสินค้าที่เลือกซื้อลดลงจริง แม้ลองซื้อสินค้าโปรโมชั่นระบบแสดงราคารวมเป็นราคาโปรโมชั่นจริง

สรุปสิ่งที่ได้จากการทดลอง

การทดลองทำให้มั่นใจว่าระบบการซื้อสินค้าสามารถทำได้จริง แม้ว่าจะเป็นสินค้าโปรโมชั่นแบบต่างๆก็สามารถคิดราคาได้ถูกต้อง

บทสรุป

ด้วยวัตถุประสงค์แรกของการสร้าง SELLON ขึ้นมาของทีมผู้พัฒนาคือความพยายามในการแก้ปัญหาการติดต่อสื่อสารของลูกค้า และ เจ้าของร้านที่เปิดร้านค้าออนไลน์ ที่เราพบว่าในบางครั้งการปฏิสัมพันธ์โต้ตอบของ ลูกค้าและเจ้าของร้านยังไม่ดีเท่าที่ควร เสน่ห์อย่างหนึ่งที่ร้านค้าที่มีหน้าร้านจริงๆเหนือกว่าร้านค้าออนไลน์คือการมีปฏิสัมพันธ์กันจริงๆ ลูกค้าสามารถสอบถามรายละเอียดของสินค้า รวมไปถึงรายละเอียดของโปรโมชั่นต่างๆ พวกเราจึงเล็งเห็นว่าควรมี ระบบที่สามารถสร้างปฏิสัมพันธ์โต้ตอบระหว่างลูกค้า และ เจ้าของร้านขึ้นมาได้เปรียบเสมือนการได้คุยกันที่ร้านจริงๆบนหน้าร้านค้าออนไลน์ จึงเป็นจุดประสงค์เริ่มต้นของการริเริ่มพัฒนาเว็บแอพลิเคชั่นชิ้นนี้ขึ้นมา อย่างไรก็ตามเว็บไซต์สำหรับการขายของจำเป็นต้องมีระบบการซื้อขาย และระบบพื้นฐานอื่นๆที่เว็บไซต์ร้านขายของต้องมี ซึ่งปัญหาที่เจอในเว็บขายของออนไลน์บางเว็บไซต์คือมี ระบบมากเกินไปจนระบบจำเป็นต้องใช้เวลาในการทำความเข้าใจก่อนใช้มาก N&N group เล็งเห็นถึงปัญหานี้และได้พยายามออกแบบระบบ ไปจนถึงส่วนโต้ตอบผู้ใช้งานให้สามารถใช้งานได้ง่ายที่สุด เพื่อให้ผู้ใช้สามารถเข้าใจระบบได้ในทันที โดยไม่ต้องเรียนรู้การใช้งานมากมาย N&N Group หวังเป็นอย่างยิ่งว่า SELLON จะสามารถทำลายกำแพงของการติดต่อสื่อสารระหว่างลูกค้า และ เจ้าของร้าน และเพิ่มปฏิสัมพันธ์ระหว่างลูกค้า และเจ้าของร้านค้าออนไลน์ได้ประสบความสำเร็จ

ในช่วงแรกของการพัฒนา ทีมพัฒนายังมีแนวทางการพัฒนาที่ไม่ชัดเจน รวมถึงการมองไม่เห็นภาพรวมของระบบ ทีมพัฒนาจึงทำได้เพียงการเก็บเกี่ยวข้อมูลของร้านค้าออนไลน์อื่นๆที่มีอยู่ในท้องตลาด และนำมาวิเคราะห์ออกมาเป็นเอกสารความต้องการของระบบในรูปแบบต่างๆ ซึ่งสุดท้ายหลังทำการออกแบบ และ ลองทำระบบจริงๆแล้วก็พบว่าเกิดการเปลี่ยนแปลงการออกแบบอยู่บ่อยครั้ง โดยส่วนตัวทีมพัฒนาไม่เคยออกแบบระบบโดยเริ่มจากการทำเอกสาร และ วิเคราะห์ระบบก่อนเลย ซึ่งประสบการณ์ที่ได้รับในครั้งนี้มีประโยชน์ต่อการทำงานในบริษัทใหญ่ๆที่ต้องมีการออกแบบระบบอย่างดีก่อนเริ่มทำงานในอนาคตอย่างมาก ซึ่งการออกแบบในวิชานี้เป็นการออกแบบโดยใช้หลักการของ Object Oriented Design ประโยชน์ของการออกแบบในรูปแบบนี้คือ โค้ดที่ออกมาเป็นระเบียบ แยกการทำงานออกจากกันอย่างชัดเจน รวมไปถึงการนำ Pattern ต่างๆที่ได้เรียนรู้มาจากวิชานี้ใช้ในการออกแบบระบบเพื่อรองรับเปลี่ยนแปลงความต้องการของผู้ใช้งาน ซึ่งมักจะเกิดขึ้นในการพัฒนาแอปพลิเคชั่นจริงๆ จนไปถึงการออกแบบระบบเพื่อรองรับการขยายตัวของแอพลิเคชั่น ทีมผู้พัฒนาได้รับแนวคิด และ ได้ลองปฏิบัติจริง จนเห็นผลประโยชน์ และ ผลกระทบ ของการออกแบบในรูปแบบต่างๆ ซึ่งถือเป็นประสบการณ์ที่มีประโยชน์อย่างมากต่อการออกไปประกอบอาชีพ นักพัฒนาแอพลิเคชั่นในอนาคต สุดท้ายนี้วิชานี้ได้มอบโอกาสให้เห็นเทคโนโลยีที่ใช้งานในโลกจริง ไปจนถึงเทคโนโลยีใหม่ๆที่เป็นข้อมูลที่มีประโยชน์ต่อการทำงานในอนาคตหลายอย่าง ถึงแม้การพัฒนาแอพลิชั่นนี้จะกินเวลายาวนาน และใช้เวลาในการทำการพัฒนาเยอะ อาจเนื่องด้วยเป็นการเขียนเว็บแอพลิเคชั่นในรูปแบบที่ไม่คุ้นเคย แต่สิ่งตอบแทนที่ได้รับกลับคุ้มค่ากับเวลาที่เสียไปในการพัฒนาอย่างมาก

บรรณานุกรม

Books

- Dayle Rees. Laravel:Code Bright. Leanpub,2014-06-01
- JeffreyWay. Laravel Testing Decoded. Leanpub,2013-05-28

Online Tutorials

- Phillop Brown. 2013. “Creating flexible Controllers in Laravel 4 using Repositories.”

  [link].http://culttt.com/2013/07/08/creating-flexible-controllers-in-laravel-4-using-repositories/

- Ri Xu.2014.“Laravel4 Real Time Chat.”

  [link]. https://xuri.me/2014/09/08/laravel-4-real-time-chat.html