MY PRECIOUS

ระบบจองที่นั่งร้านอาหารแบบออนไลน์

(Restaurant Booking)

Git hub repository: Restaurant_booking

ผู้จัดทำ

นางสาว รินทร์ลภัส เลิศธนกุลพงษ์ รหัสนักศึกษา 55011049 Sec 2

นางสาว วชิรญาณ์ ตันติวัฒนารมย์ รหัสนักศึกษา 55011067 Sec 2

Object-oriented Analysis and Design

ภาคการศึกษาที่ 1 ปีการศึกษา 2557

สารบัญ

บทคัดย่อ

เป็นเว็บศูนย์กลางสำหรับลูกค้าและเจ้าของร้านอาหาร โดยเจ้าของร้านสามารถมาสมัครใช้บริการระบบสำรองที่นั่งเพื่อใช้กับร้านของตนเองได้ และ ลูกค้าสามารถเข้ามาใช้บริการในเว็บเพื่อสำรองที่นั่งในร้านอาหารที่ต้องการได้ผ่านระบบออนไลน์ตลอด 24 ชั่วโมง นอกจากนี้ยังมีระบบ-ตรวจสอบและจัดการการจองต่างๆสำหรับเจ้าของร้านและลูกค้าอีกด้วย ดังนั้นจึงทำให้ลูกค้าไม่ต้องโทรศัพท์ไปที่ร้านเพื่อจองที่นั่งอีกต่อไป ทำให้การจองที่นั่งสามารถดำเนินการได้อย่างสะดวกสบายมากยิ่งขึ้น สอดคล้องกับการใช้ชีวิตของคนในปัจจุบันที่นิยมทำธุระกรรมต่างๆ ผ่านทางอุปกรณ์เทคโนโลยี

บทนำและรายละเอียดการวิเคราะห์หัวข้อ

เนื่องจากในปัจจุบัน เวลาที่เราต้องการจะจ้องที่นั่งในร้านอาหาร เราจะต้องหาเบอร์โทรศัพท์ของร้านอาหารนั้น และโทรศัพท์ไปที่ร้านเพื่อที่จะจองที่นั่ง โดยจ้องบอกรายละเอียดต่างๆ เช่น วัน เวลา จำนวนที่นั่ง ผ่านทางโทรศัพท์ ซึ่งอาจทำให้เกิดข้อผิดพลาดทางการสื่อสารทางเสียงผ่านโทรศัพท์ได้ง่าย และต้องยุ่งยากในการหาเบอร์โทรศัพท์เพื่อติดต่อร้านอาหารเหล่านั้นอีกด้วย เราจึงมีความคิดที่จะทำระบบดังกล่าวเพื่อตอบสนองต่อปัญหานี้ โดยการทำเป็นระบบออนไลน์ที่เป็นศูนย์กลางการติดต่อระหว่างร้านอาหารและลูกค้า โดยเจ้าของร้านสามารถกรอกข้อมูลร้านเพื่อขอสมัครใช้บริการระบบจองที่นั่งออนไลน์โดยเป็นการเปิดร้านอาหารออนไลน์เพื่อการจองที่นั่ง ส่วนลูกค้าก็สามารถเข้ามาเลือกดูร้านอาหารและจองที่นั่งในร้านอาหารที่ต้องการได้ ดังนั้นผู้ใช้งานจึงสามารถจองที่นั่งในร้านอาหารที่ต้องการได้ง่ายขึ้น และสามารถกรอกข้อมูลการจองผ่านทางเว็บไซต์ ซึ่งเป็นการลดข้อผิลพลาดทางการสื่อสารที่เกิดขึ้นทางโทรศัพท์ และยังสามารถทำการจองได้ตลอด 24 ชั่วโมงอีกด้วย

งานที่เกี่ยวข้อง

ลักษณะการทำงานของเว็บจะใกล้เคียงกับ https://www.eatigo.com/ แต่เว็บของเราจะไม่เน้นเรื่องส่วนลดของร้านอาหารต่างๆ แต่จะเน้นเรื่องของการจองที่นั่งร้านอาหารล่วงหน้าเป็นหลัก โดยเว็บของเรานั้นลูกค้าจะสามารถระบุรายละเอียดการจองได้อย่างละเอียด แม้กระทั่งมุมในร้านที่ต้องการจะนั่ง (เพื่อลดการเสียโอกาสจากการจองล่วงหน้าที่ไม่สามารถเลือกที่นั่งที่ต้องการได้เอง) และอยากจะเจาะกลุ่มร้านอาหารที่มีระดับของราคาต่ำลงมากว่า eatigo ด้วย เพื่อให้ผู้ใช้งานสามารถเข้าถึงได้ทุกระดับ

ผลการวิเคราะห์ความต้องการของผู้ใช้ระบบ

Functional requirements

1. ระบบสมัครสมาชิก: เพื่อเก็บข้อมูลพื้นฐานของสมาชิก

2. การแก้ไขข้อมูลของผู้ใช้งาน: เพื่อแก้ไขข้อมูลที่ผู้ใช้ต้องการ

3. การสมัครเปิดร้านอาหาร: สำหรับเจ้าของร้านที่ต้องการใช้บริการระบบการจองที่นั่งร้านอาหาร

4. การแก้ไขข้อมูลร้านอาหารโดยเจ้าของร้าน: เพื่อแก้ไขและอัพเดทข้อมูลตามที่ร้านต้องการ

5. การค้นหาร้านอาหาร: เพื่อความสะดวกในการหาร้านอาหารที่ต้องการ

6. ดูรายละเอียดร้านอาหาร: เพื่อบอกข้อมูลของร้านอาหารที่ต้องการเพื่อการตัดสินใจ

7. การจองที่นั่งในร้านอาหาร: เพื่อทำการจองที่นั่งในร้านอาหารที่ต้องการ

8. การตรวจสอบรายการที่จอง: ในกรณีที่ต้องการดูรายการและรายละเอียดที่ได้จองไว้สามารถใช้ฟังก์ชั่นนี้ได้

9. การยกเลิกการจอง: เมื่อไม่ต้องการ / ไม่สามารถไปตามที่จองไว้ได้ก็จะสามารถยกเลิกการจองนั้นได้

10. การเพิ่มรูปภาพของร้านสำหรับเจ้าของร้าน: เพื่อเพิ่มความน่าสนใจและรายละเอียดของร้าน

Non-Functional requirements

1. ส่งรายละเอียดการจองให้ลูกค้าทาง Email ทันที เมื่อลูกค้าทำรายการจอง

2. สุ่มร้านอาหารพร้อมสุ่มรูปภาพของร้านมาแสดงหน้าแรกจำนวน 9 ร้าน

3. หน้าเว็บรองรับทุกขนาดหน้าจอ

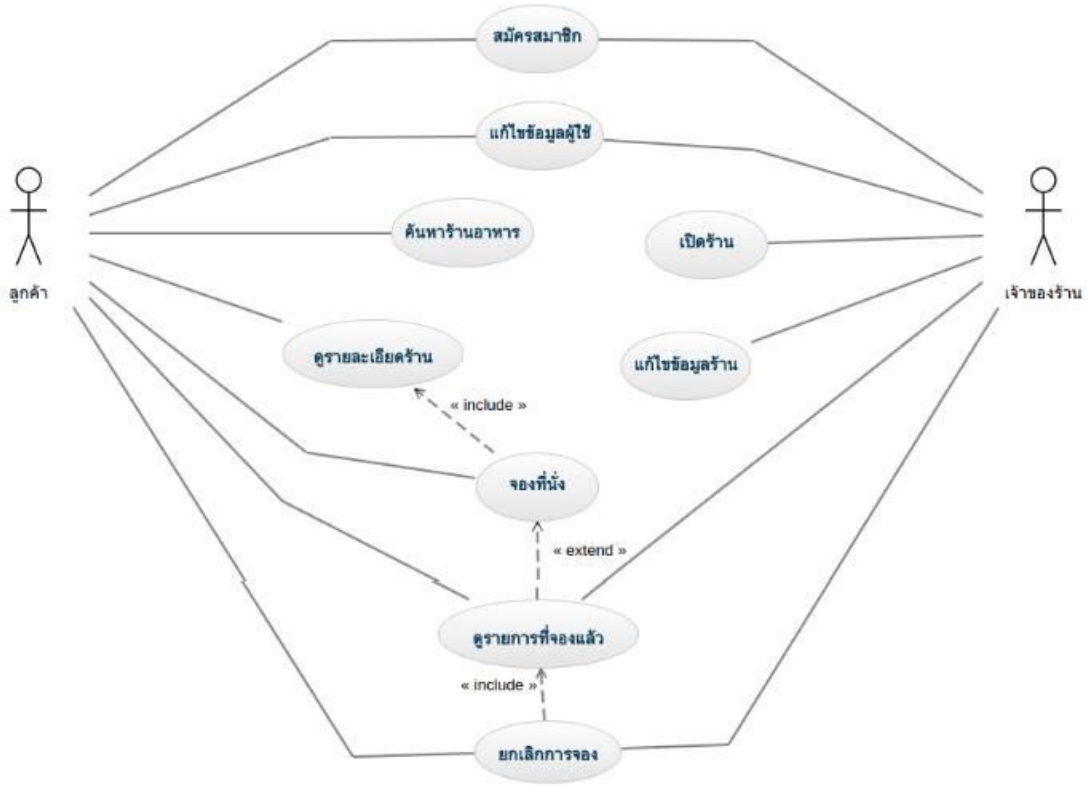
4. รองรับการใช้งานตลอด 24 ชั่วโมง

5. เว็บสามารถใช้งานได้ง่าย, เข้าใจง่าย

6. รองรับการใช้งานโดยผู้ใช้อย่างน้อย 10 คนพร้อมกัน

แนวทางการใช้งาน

Use case diagram



ระบบจองที่นั่งร้านอาหารแบบออนไลน์

Use case specifications

Use case name : จองที่นั่ง

Use case purpose : Use case นี้ สามารถจองที่นั่งภายในร้านที่ระบุได้ โดยสามารถกรอกรายละเอียด ต่างๆ ได้แก่ วันที่, เวลาเข้า, เวลาออก, จำนวนที่นั่งที่ต้องการจอง และ มุมในร้านที่ต้องการจอง

Preconditions : ต้องอยู่ในสถานะสมาชิกที่เข้าสู่ระบบแล้ว

Postconditions : จำนวนที่นั่งว่างในมุมนั้นๆ ของร้านนั้นๆ จะต้องลดน้อยลงตามจำนวนที่จองไป

Limitations : ไม่สามารถจองเกินจำนวนที่นั่งที่เจ้าของร้านระบุไว้ได้

Assumptions : ผู้ใช้เข้าสู่ระบบแล้ว, ผู้ใช้จองที่นั่งตามความต้องการจริง และ ยังมีที่นั่งว่างในร้าน

Primary scenario

A. เข้าสู่ระบบ

B. ค้นหาร้านอาหารที่ต้องการ

C. เลือก "จองที่นั่ง"

D. กรอกรายละเอียดการจอง

E. กด “submit” เพื่อบันทึกการจอง

G. รายการที่จองไปปรากฏอยู่ในรายการจองของบัญชีผู้ใช้นั้น

Alternate scenarios:

Condition triggering an alternate scenario:

Condition 1: บัญชีผู้ใช้หรือรหัสผ่าน ไม่ตรงกับบัญชีในฐานข้อมูล

A1. ผู้ใช้กรอกบัญชีหรือรหัสผ่าน ไม่ตรงกับบัญชีในฐานข้อมูล

A2. ระบบตรวจสอบได้ว่าไม่มีบัญชีผู้ใช้ในฐานข้อมูล และแสดงข้อผิดพลาดให้กับผู้ใช้

A3. ระบบแสดงหน้า "เข้าสู่ระบบ" อีกครั้ง

A4. กลับสู่ Primary scenario ขั้นตอน A

Condition triggering an alternate scenario:

Condition 2: กรอกรายละเอียดการจองไม่ครบตามที่ระบบต้องการ

E1. ผู้ใช้กรอกรายละเอียดการจองไม่ครบตามที่ระบบต้องการ

E2. ระบบตรวจสอบได้ว่าผู้ใช้กรอกข้อมูลไม่ครบตามที่ระบบต้องการ และแจ้งเตือนในช่องนั้นๆ

E3. ระบบแสดงหน้ากรอกรายละเอียดการจอง อีกครั้ง

E4. กลับสู่ Primary scenario ขั้นตอน D

Condition triggering an alternate scenario:

Condition 3: ที่นั่งในที่ร้านในลักษณะที่ผู้ใช้ต้องการไม่ว่าง (เช่น มุมที่ผู้ใช้ต้องการจองไม่ว่าง หรือ เวลาที่ผู้ใช้ต้องการจองไม่ว่าง) เนื่องจากผู้ใช้อาจจะเปิดหน้าเว็บค้างไว้นานเกินไป และในระหว่างนั้นจึงอาจมีผู้ใช้อื่นจองที่นั่งที่ผู้ใช้เข้าใจว่าว่าง

E1. ผู้ใช้เลือกที่นั่งที่ไม่มีว่างในขณะนั้น

E2. ระบบตรวจสอบได้ว่าที่นั่งในลักษณะนั้นไม่ว่าง และแสดงข้อความแจ้งเตือนแก่ผู้ใช้

E3. ระบบกลับสู่หน้าจองที่นั่งในร้านนั้นอีกครั้ง

E4. กลับสู่ Primary scenario ขั้นตอน D

Use case name : เปิดร้าน

Use case purpose : Use case นี้ สามารถใช้ในการลงทะเบียนเพื่อนำร้านเข้าสู่ระบบ ทำให้ลูกค้าของร้านสามารถใช้บริการจองที่แบบนั่งออนไลน์ได้

Preconditions : ต้องอยู่ในสถานะสมาชิกที่เข้าสู่ระบบแล้ว

Postconditions : เจ้าของร้านจะได้ระบบจัดการร้านที่ประกอบด้วย การจัดการรายการจองและการ แก้ไขข้อมูลของร้าน

Limitations : 1 บัญชีผู้ใช้ ไม่สามารถเปิดร้านได้เกิน 5 ร้าน

Assumptions : เจ้าของร้านเข้าสู่ระบบแล้ว และ ร้านที่ถูกเพิ่มเข้ามาในระบบเป็นร้านที่มีอยู่จริง

Primary scenario

A. เข้าสู่ระบบด้วยบัญชีผู้ใช้ที่มี

B. เลือกสมัครเปิดร้าน

C. กรอกรายละเอียดร้าน

D. กด”submit” เพื่อบันทึกร้านเข้าสู่ระบบ

E. ร้านที่เปิด เข้าไปอยู่ในฐานข้อมูลของระบบ

Alternate scenarios:

Condition triggering an alternate scenario:

Condition 1: บัญชีผู้ใช้/รหัสผ่าน ไม่ตรงกับบัญชีในฐานข้อมูล

A1. ผู้ใช้กรอกบัญชี/รหัสผ่าน ไม่ตรงกับบัญชีในฐานข้อมูล

A2. ระบบตรวจสอบได้ว่าไม่มีบัญชีผู้ใช้ในฐานข้อมูล และแสดงข้อผิดพลาดให้กับผู้ใช้

A3. ระบบแสดงหน้า"เข้าสู่ระบบ"อีกครั้ง

A4. กลับสู่ Primary scenario ขั้นตอน A

Condition triggering an alternate scenario:

Condition 2: กรอกรายละเอียดร้านไม่ครบตามที่ระบบต้องการ

D1. เจ้าของร้านกรอกรายละเอียดของร้านไม่ครบตามที่ระบบต้องการ

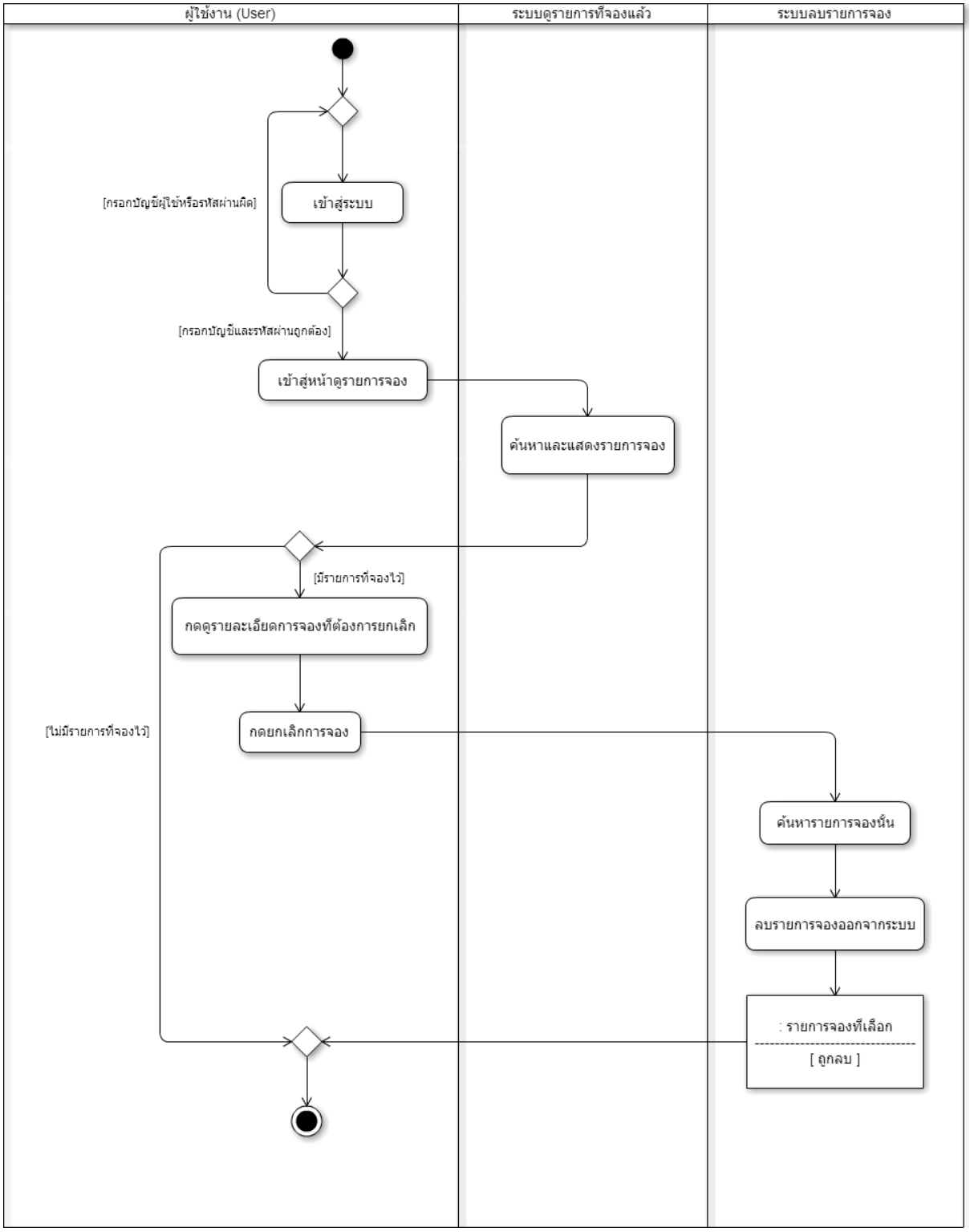
D2. ระบบตรวจสอบได้ว่าแบบฟอร์มสมัครเปิดร้านที่ส่งมายังกรอกข้อมูลไม่ครบตามที่ระบบต้องการ และแจ้งเตือนในช่องที่ระบบต้องการ แต่ยังไม่มีข้อมูลในช่องนั้น

D3. ระบบแสดงหน้ากรอกข้อมูลเพื่อเปิดร้าน อีกครั้ง เพื่อให้เจ้าของร้านกรอกข้อมูลให้ครบ

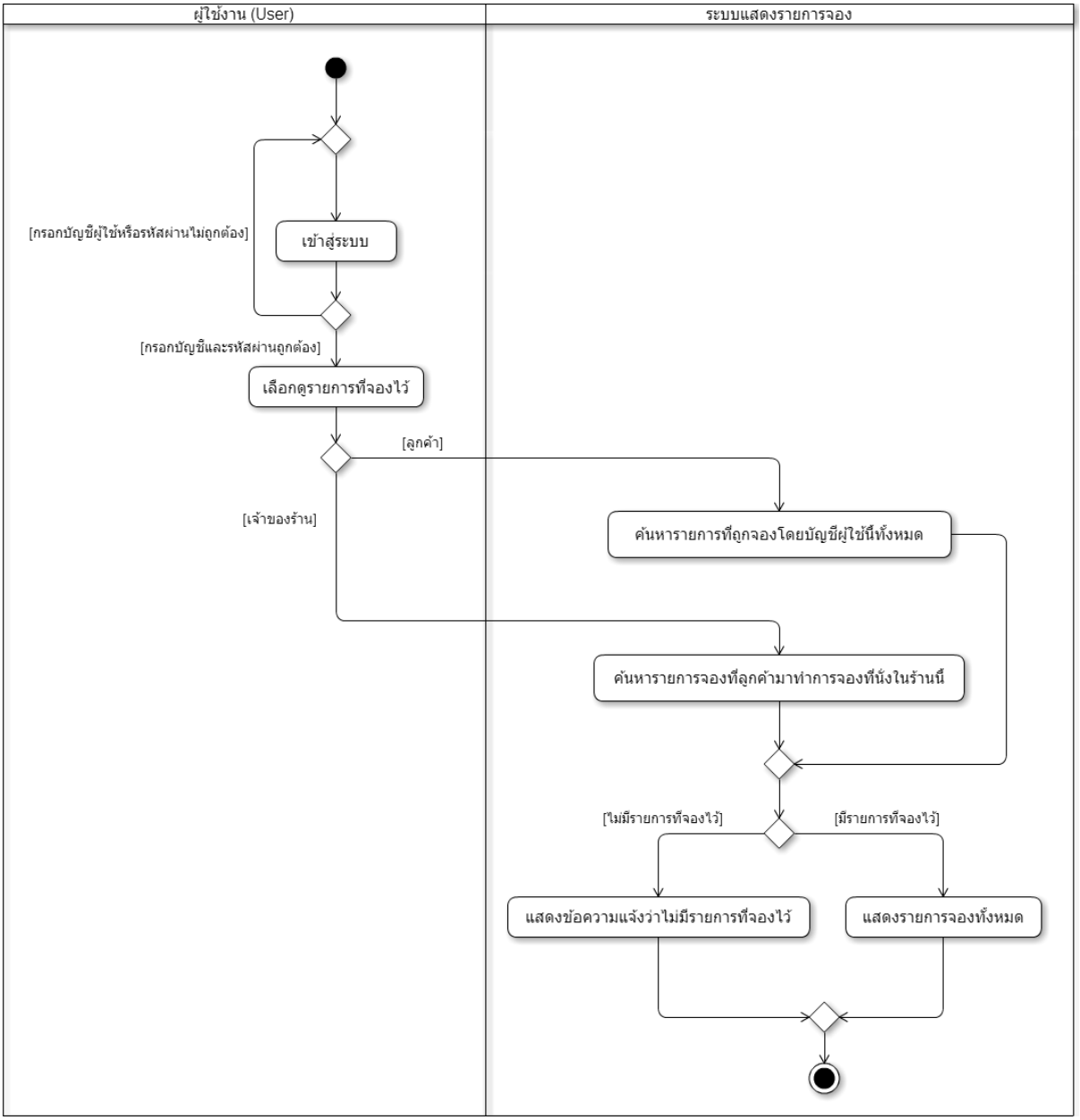
D4. กลับสู่ Primary scenario ขั้นตอน C

Activity diagram

Use case name : ยกเลิกการจอง



Use case name : ดูรายการที่จองแล้ว



สถาปัตยกรรมของระบบ

Problem Analysis

เหตุผล : เนื่องจากคนในปัจจุบันอยู่กับการใช้งาน Internet เป็นเวลานาน และการจองร้านอาหารต้องมีการโทรไปเพื่อจองกับที่ร้าน หรือต้องไปจองที่ร้านด้วยตนเอง จึงได้มีการแก้ปัญหานี้โดยการจำลองสิ่งต่างๆที่ใช้ในการจองมาเป็น abstraction ดังนี้

Abstraction

- ปฏิทิน ซึ่งประกอบด้วย วัน/เวลา ที่มีคนจอง และ วัน/เวลา ที่ที่นั่งว่าง

- ร้านอาหาร ซึ่งประกอบด้วย ชื่อร้าน, ที่อยู่ร้าน, จำนวนที่นั่งในร้าน, เวลาเปิด/ปิดร้าน และ เบอร์โทรศัพท์ติดต่อร้าน

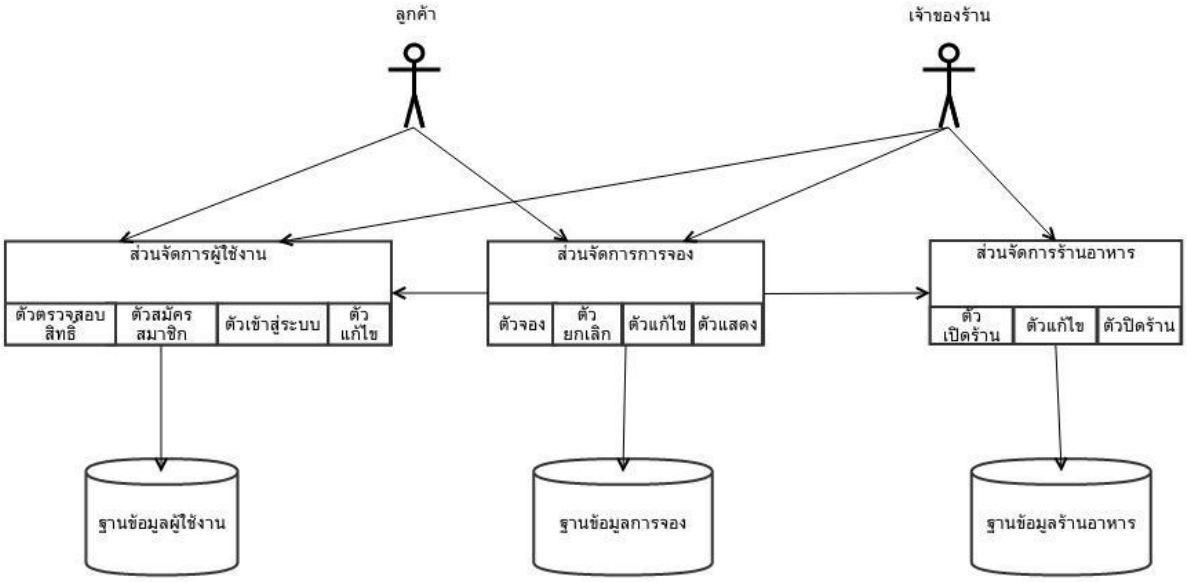
- จัดการการจอง ประกอบไปด้วยรายละเอียดการจองต่างๆ โดยสามารถจองที่นั่ง, แก้ไขการจอง, ลบการจอง และแสดงรายการจองได้

- ลูกค้า ซึ่งประกอบด้วย ชื่อ, E-mail และรายการจองของลูกค้า

Component

- ตัวจัดการการสมัครสมาชิก
- ตัวจัดการการเข้าสู่ระบบ
- ตัวตรวจสอบสิทธิ์ผู้ใช้
- ตัวจัดการการจอง
- ตัวยกเลิกการจอง
- ตัวดูรายการจอง
- ตัวแก้ไขการจอง
- ตัวจัดการการแก้ไขข้อมูลส่วนตัว
- ตัวจัดการการแก้ไขข้อมูลร้าน
- ตัวจัดการการเปิดร้าน
- ตัวจัดการการปิดร้าน

Application Architecture



แผนภาพอธิบายภาพรวมของระบบ

- ส่วนจัดการผู้ใช้งาน

  ในส่วนนี้จะประกอบด้วย ตัวสมัครสมาชิก ตัวจัดการการเข้าสู่ระบบ ตัวตรวจสอบสิทธิ์ผู้ใช้ และ ตัวจัดการการแก้ไขข้อมูลของผู้ใช้งาน โดยส่วนจัดการผู้ใช้งานนี้จะเป็นตัวติดต่อฐานข้อมูลของผู้ใช้งาน และเชื่อมต่อกับส่วนจัดการการจอง (ต้องใช้ข้อมูลจากส่วนนี้เพื่อทำการจอง) โดยที่ทั้งลูกค้าและเจ้าของร้านสามารถจัดการส่วนนี้ได้

- ส่วนจัดการการจอง

  ในส่วนนี้จะประกอบด้วย ตัวจอง ตัวจัดการการยกเลิกการจอง ตัวจัดการการแก้ไขการจอง และตัวแสดงรายการจองที่ยังไม่ถึงเวลา ซึ่งส่วนนี้จะติดต่อกับฐานข้อมูลการจอง และติดต่อกับส่วนจัดการผู้ใช้งาน และส่วนจัดการร้านอาหาร เพราะในการจัดการการจอง จำเป็นต้องใช้ข้อมูลจากทั้ง 2 ส่วนที่ได้กล่าวไป โดยส่วนนี้สามารถถูกใช้บริการได้จากทั้งลูกค้าและเจ้าของร้าน

- ส่วนจัดการร้านอาหาร

  ในส่วนนี้จะประกอบด้วยตัวจัดการการเปิดร้าน ตัวจัดการการแก้ไขข้อมูลร้าน และตัวจัดการการปิดร้าน(ลบร้านออกจากระบบ) ซึ่งส่วนนี้จะติดต่อกับฐานข้อมูลร้านอาหารและส่วนจัดการการจอง (ต้องใช้ข้อมูลจากส่วนนี้เพื่อทำการจอง) และสามารถถูกใช้บริการได้จากเจ้าของร้านคนเดียว

- ฐานข้อมูลผู้ใช้งาน

  ประกอบด้วยชื่อผู้ใช้ รหัสผ่าน E-mail และเบอร์โทรศัพท์ของผู้ใช้งาน โดยเชื่อมต่อกับส่วนจัดการผู้ใช้งาน

- ฐานข้อมูลการจอง

  ประกอบด้วยชื่อผู้จอง และข้อมูลการจองของผู้ใช้นั้นๆ โดยเชื่อมต่อกับส่วนจัดการการจอง

- ฐานข้อมูลร้านอาหาร

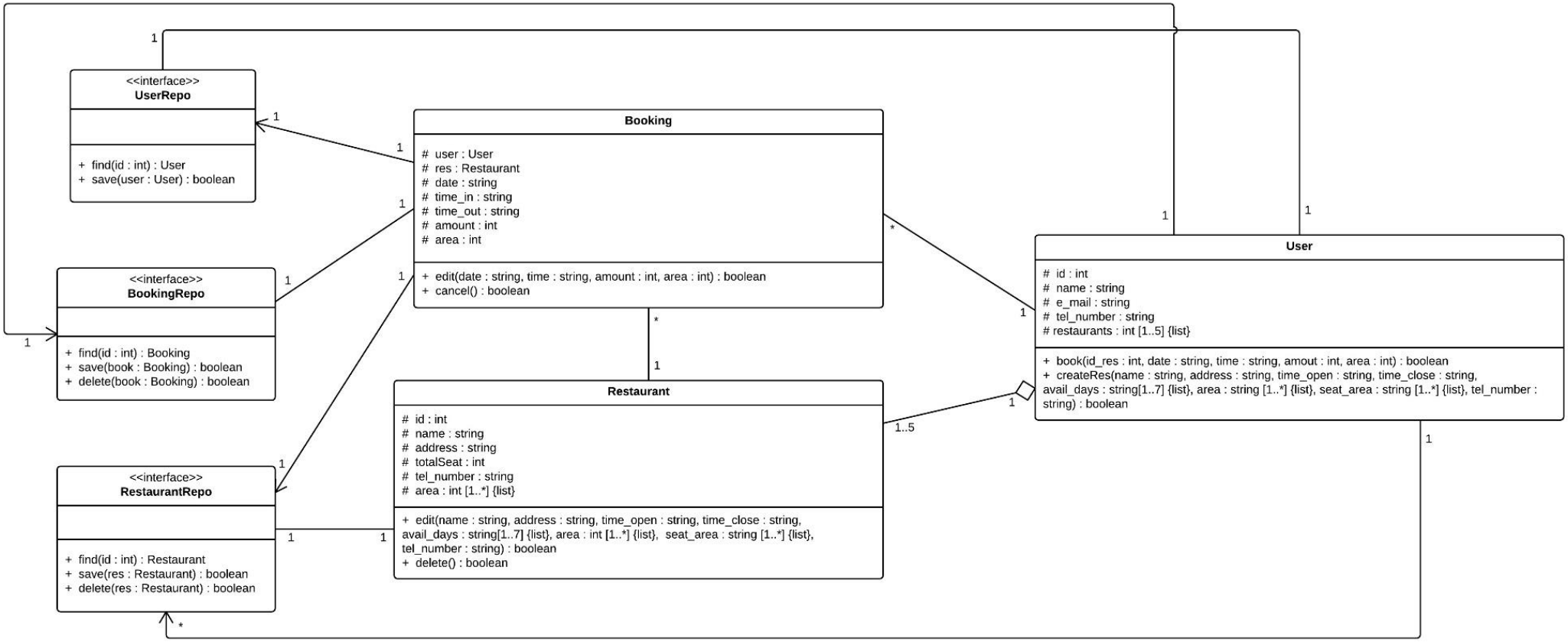
  ประกอบด้วยชื่อร้านอาหาร จำนวนที่นั่งในร้านอาหาร มุมในร้านอาหาร ชื่อเจ้าของร้าน ที่อยู่ร้าน เวลาเปิด/ปิดร้าน และเบอร์โทรศัพท์ติดต่อร้าน โดยเชื่อมต่อกับส่วนจัดการร้านอาหาร

Subsystem / Component

- **ส่วนจัดการผู้ใช้งาน**
  - **ตัวจัดการการสมัครสมาชิก** : มีหน้าที่จัดการเก็บข้อมูลต่างๆของการสมัครสมาชิกของผู้ใช้งาน เข้าสู่ฐานข้อมูลผู้ใช้งาน
  - **ตัวจัดการการเข้าสู่ระบบ** : มีหน้าที่เปรียบเทียบบัญชีผู้ใช้กับระบบฐานข้อมูลผู้ใช้งาน เพื่อให้ผู้ใช้เข้าถึงข้อมูลส่วนของตัวเองอย่างถูกต้อง
  - **ตัวจัดการการแก้ไขข้อมูลผู้ใช้** : มีหน้าที่รับข้อมูลที่เปลี่ยนแปลงหรือข้อมูลเพิ่มเติมของผู้ใช้ส่งไปแก้ไขในฐานข้อมูลของผู้ใช้นั้น
  - **ตัวตรวจสอบสิทธิ์ผู้ใช้** : มีหน้าที่ตรวจสอบสิทธิ์ของผู้ใช้ว่าอยู่ในฐานะลูกค้าหรือเจ้าของร้าน โดยตรวจสอบจากจำนวนร้านในฐานข้อมูลผู้ใช้ ถ้าหากเป็นผู้ใช้ที่จำนวนร้านเป็น 0 จะได้สิทธิ์เป็นลูกค้าซึ่งไม่สามารถจัดการร้านใดๆได้ แต่ถ้าหากมีจำนวนร้านมากกว่า 1 จะได้สิทธิ์เจ้าของร้าน มีสิทธิ์ในการจัดการร้านของตัวเองได้
- **ส่วนจัดการการจอง**
  - **ตัวจอง** : มีหน้าที่เก็บข้อมูลการจองร้านอาหารของผู้ใช้งานนั้นๆเข้าสู่ระบบฐานข้อมูลการจอง และเปรียบเทียบเวลาของการจอง หากเวลาที่จองมีการตรงกันกับข้อมูลที่มีอยู่แล้วในระบบ หรือจำนวนที่นั่งไม่พอในช่วงเวลานั้น จะไม่อนุญาติให้การจองล่าสุดสำเร็จได้
  - **ตัวจัดการการยกเลิกการจอง** : มีหน้าที่ลบข้อมูลการจองที่มีในระบบที่ผู้ใช้ไม่ต้องการออกจากระบบฐานข้อมูลการจอง ในส่วนนี้มีผลกระทบกับการแสดงรายการจองของผู้ใช้งาน ทั้งลูกค้าและเจ้าของร้าน
  - **ตัวจัดการการแก้ไขการจอง** : มีหน้าที่รับข้อมูลที่เปลี่ยนแปลงจากผู้ใช้งาน เช่น วัน เวลา จำนวนโต๊ะ และมุมที่ต้องการ ส่งไปแก้ไขยังฐานข้อมูลการจองของผู้ใช้ ซึ่งทำให้มีผลกระทบกับการแสดงรายการการจองของผู้ใช้งาน ทั้งลูกค้าและเจ้าของร้าน
  - **ตัวดูรายการจอง** : มีหน้าที่แสดงข้อมูลจากฐานข้อมูลการจอง โดยในฝั่งของลูกค้าจะแสดงรายการที่ลูกค้าได้จองไว้เท่านั้น (ไม่รวมการจองที่ผ่านมา) ส่วนในฝั่งของเจ้าของร้านจะแสดงรายการที่มีลูกค้าจองในแต่ละวันและแต่ละช่วงเวลาพร้อมทั้งรายละเอียดต่างๆที่ลูกค้าระบุ
- **ส่วนจัดการร้านอาหาร**
  - **ตัวจัดการการเปิดร้าน** : มีหน้าที่จัดการเก็บข้อมูลต่างๆของร้านอาหารที่ต้องการเปิดในระบบ เข้าสู่ฐานข้อมูลร้านอาหาร
  - **ตัวจัดการการแก้ไขข้อมูลร้าน** : มีหน้าที่รับข้อมูลที่เปลี่ยนแปลงหรือข้อมูลเพิ่มเติมของร้านอาหาร เช่น เวลาเปิด/ปิดร้าน วันทำการ หรือที่อยู่ร้าน(โดยที่ไม่สามารถแก้ไขชื่อร้านได้) ส่งไปแก้ไขในฐานข้อมูลร้านอาหาร
  - **ตัวจัดการการปิดร้าน** : มีหน้าที่ลบร้านอาหารการออกจากฐานข้อมูลร้านอาหาร
- **ฐานข้อมูลผู้ใช้งาน**
  - **ตารางข้อมูลผู้ใช้งาน** : ในตารางนี้จะประกอบไปด้วย ชื่อผู้ใช้ รหัสผ่าน(ที่เข้ารหัสแล้ว) E-mail และเบอร์โทรศัพท์ของผู้ใช้งาน ซึ่งใช้เป็น ฐานข้อมูลในการเข้าถึงข้อมูลของผู้ใช้นั้นๆ

- **ฐานข้อมูลการจอง**
  - **ตารางข้อมูลการจอง** : ในตารางนี้จะประกอบด้วย id ผู้จอง, id ร้านอาหาร และข้อมูลการจองของผู้ใช้นั้นๆ ได้แก่ วัน เวลาเข้า จำนวนที่นั่ง และมุมที่ต้องการ ซึ่งใช้เป็นฐานข้อมูลของการจองของลูกค้า แต่ตารางนี้สามารถเข้าถึงได้จากทั้งสองฝั่งของผู้ใช้งาน คือ ลูกค้า และ ร้านอาหาร โดยลูกค้าจะเข้าถึงข้อมูลที่เป็นของลูกค้าเอง ส่วนร้านอาหารจะเข้าถึงข้อมูลที่ตรงกับร้านอาหารของตนเองได้
- **ฐานข้อมูลร้านอาหาร**
  - เก็บข้อมูลของร้านอาหาร ได้แก่ id ของร้านอาหารนั้นๆ, id ของเจ้าของร้านอาหาร, ชื่อรูปภาพของร้านอาหาร (จัดเก็บรูปภาพที่อัพโหลดมาเป็นในรูปแบบชื่อเดียวกัน), ชื่อร้านอาหาร, ที่อยู่ร้านอาหาร, วันที่เปิดให้บริการ, เวลาเปิดร้าน, เวลาปิดร้าน, มุมในร้าน, จำนวนที่นั่งในแต่ละมุม, เบอร์โทรศัพท์ติดต่อ ซึ่งเจ้าของร้านมีสิทธิ์แก้ไขข้อมูลหรือลบข้อมูลออกจากระบบได้ ส่วนลูกค้ามีสิทธิ์เรียกดูข้อมูลได้

แผนภาพของคลาสหลัก



จาก class diagram นี้อธิบายได้ว่า

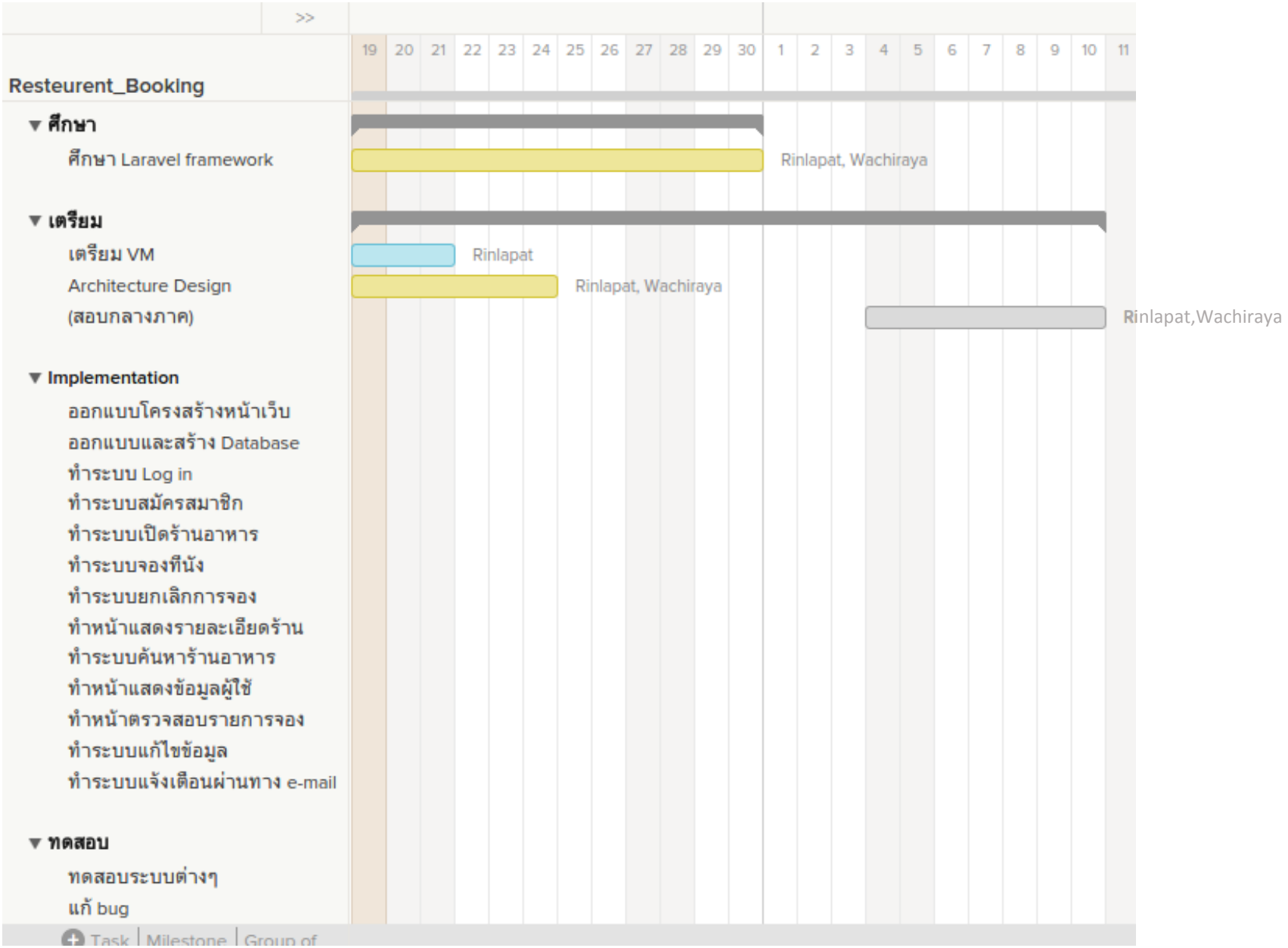
- ส่วน BookingRepo, RestaurantRepo และ UserRepo เป็นตัวเชื่อมต่อกับส่วน Database
- ส่วน User, Books และ Restaurant เป็น Model ของข้อมูลที่เราสนใจใน problem domain ซึ่งสามารถเอาข้อมูลข้างใน ออกมาได้ผ่าน method get (ซึ่งในที่นี้ไม่ได้เขียนลงไป) และเขียนข้อมูลผ่าน method set (ซึ่งในที่นี้ไม่ได้เขียนลงไป)
- ระหว่าง Model และ Repo จะมีเส้นเชื่อมถึงกันทุกคู่ เพื่อให้ Repo รู้จัก Object ชนิดเดียวกับ Model
- Owner(เจ้าของร้าน) inherit User(ผู้ใช้) เนื่องจาก เจ้าของร้านเป็นผู้ใช้ที่มีการเปิดร้านอาหารอยู่ในบัญชีผู้ใช้
- Restaurant เป็น part of Owner โดยที่ เจ้าของร้าน(Owner) 1 คน มีร้านอาหารได้ไม่เกิน 5 ร้าน
- ลูกค้า(User) 1 คน สามารถมีได้หลายการจอง(Books)

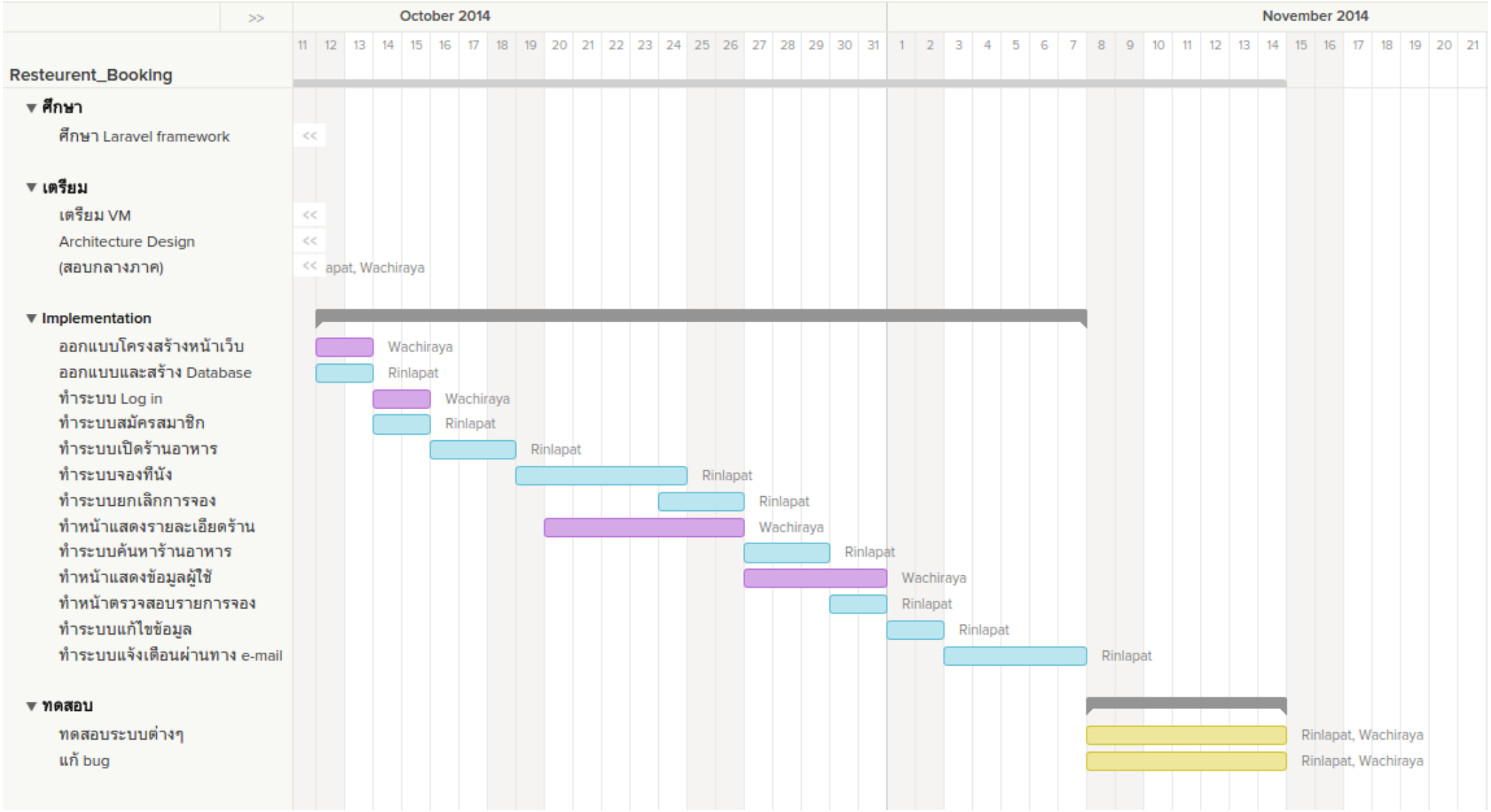
รายละเอียดการพัฒนาซอฟต์แวร์

deployment

- ใช้ VM 1 ตัว
- ลง LAMP stack server บน VM (ลง Ubuntu server บน VM, ใช้ apache2 เป็น Application server, ใช้ MySQL เป็นตัว จัดการฐานข้อมูล และลง php5 (libapache2-mod-php5 php5-mcrypt)
- ใช้ phpmyadmin เป็นตัวช่วยจัดการฐานข้อมูล
- ใช้ Laravel เป็น Framework ที่ใช้พัฒนา (PHP5)
- ใช้ Composer เป็น PHP Dependency manager
- ใช้ Bootstrap เป็น Frontend Framework (HTML, CSS, และ JavaScript)
- ระบบทุกระบบท างานอยู่บน VM Domain name : eatwme.cloudapp.net

Implementation plan




October 2014
November 2014
11 12 13 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 18 19 20 21
Resteurent_Booking
ศึกษา
ศึกษา Laravel framework
เตรียม
เตรียม VM
Architecture Design
(สอบกลางภาค)
apat, Wachiraya
Implementation
ออกแบบโครงสร้างหน้าเว็บ
Wachiraya
ออกแบบและสร้าง Database
Rinlapat
ทำระบบ Log in
Wachiraya
ทำระบบสมัครสมาชิก
Rinlapat
ทำระบบเปิดร้านอาหาร
Rinlapat
ทำระบบจองที่นั่ง
Rinlapat
ทำระบบยกเลิกการจอง
Rinlapat
ทำหน้าแสดงรายละเอียดร้าน
Wachiraya
ทำระบบค้นหาร้านอาหาร
Rinlapat
ทำหน้าแสดงข้อมูลผู้ใช้
Wachiraya
ทำหน้าตรวจสอบรายการจอง
Rinlapat
ทำระบบแก้ไขข้อมูล
Rinlapat
ทำระบบแจ้งเตือนผ่านทาง e-mail
Rinlapat
ทดสอบ
ทดสอบระบบต่างๆ
Rinlapat, Wachiraya
แก้ bug
Rinlapat, Wachiraya

ผลการทดสอบซอฟต์แวร์

Unit test

Test 1: Test เพื่อตรวจสอบเวลาสำหรับจอง ไม่ให้เป็นเวลาก่อนเวลาปัจจุบัน

(Restaurant_booking / app / tests / CheckTimeTest.php)

```php
<?php

class CheckTimeTest extends TestCase {


        public function testLateTime()
        {
                // Arrange
                $onTime = date("H:i", strtotime("+5 minute", strtotime(date("H:i"))));
                $lateTime = date("H:i", strtotime("-30 minute", strtotime(date("H:i"))));

                //Act
                $checker = new CheckTime;
                $result_True = $checker->checkLateTime($onTime);
                $result_notTrue = $checker->checkLateTime($lateTime);

                //Assert
                $this->assertEquals(true,$result_True);
                $this->assertNotEquals(true,$result_notTrue);
        }
}
```

Test 2 :

(Restaurant_booking / app / tests / BookTest.php)

```php
<?php

class BookTest extends TestCase {

        public function mockBook ($id_res, $id_user, $date, $amout, $time, $area)
        {
                $book = App::make('CoreBook');
                $book->setIdRes($id_res);
                $book->setIdUser($id_user);
                $book->setDate($date);
                $book->setAmout($amout);
                $book->setTime($time);
                $book->setArea($area);

                return $book;
        }

        public function testBook()
        {

                $mock = BookTest::mockBook ('99','55','2014-10-30','2','12:00','A');

                $this->assertEquals('99',$mock->getIdRes());
                $this->assertEquals('55',$mock->getIdUser());
                $this->assertEquals('2014-10-30',$mock->getDate());
                $this->assertEquals('2',$mock->getAmout());
                $this->assertEquals('12:00',$mock->getTime());
                $this->assertEquals('A',$mock->getArea());
        }
}
```

ผลการ Test เป็นดังนี้

```
silverice@ubuntu:/var/www/html/ResBook$ vendor/bin/phpunit
PHPUnit 3.7.38 by Sebastian Bergmann.

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

.....

Time: 109 ms, Memory: 12.75Mb

OK (5 tests, 10 assertions)
```

Evaluation

1. ระบบจองที่นั่ง

- จุดประสงค์ของการทดลอง (Goal of the experiment)
  - เพื่อทดสอบว่าระบบสามารถรับข้อมูลที่ผู้ใช้กรอกได้ครบถ้วนและถูกต้อง
  - เพื่อทดสอบว่าระบบสามารถจัดเก็บข้อมูลการจองลงฐานข้อมูลได้ถูกต้อง
  - เพื่อทดสอบการตรวจสอบต่างๆก่อนทำการจอง (เช่น ตรวจสอบวันที่ร้านเปิด, ตรวจสอบว่ายังมีที่ว่าง)
  - สิ่งที่จะวัด (Measurement metrics)
  - ความถูกต้องครบถ้วนของข้อมูลการจอง
  - การจัดเก็บข้อมูลการจอง
- วิธีทำการทดลองและสิ่งที่ต้องใช้ในการทดลอง (Setup and methods of your experiment)
  - วิธีทำการทดลอง
    - เข้าสู่ระบบ
    - เลือกร้านอาหารที่ต้องการทำการจอง
    - กดจองร้านอาหาร (book)
    - กรอกรายละเอียดการจองต่างๆให้ครบถ้วน
    - กด “submit”
  - สิ่งที่ต้องใช้ในการทดลอง
    - E-mail และ รหัสผ่าน ที่เป็นสมาชิกของเว็บ เพื่อใช้เข้าสู่ระบบ
    - ชื่อ หรือ ID ร้านอาหารที่ต้องการจอง
- ผลที่ได้จากการทดลอง (Experimental results)
  - ระบบสามารถจัดเก็บข้อมูลการจองลงฐานข้อมูลได้อย่างถูกต้อง
  - ระบบสามารถตรวจสอบการเลือกวันและเวลาจองที่ผ่านมาแล้ว(เป็นอดีต) และแจ้งเตือนให้ผู้ใช้กรอกใหม่อีกครั้งได้
  - ระบบสามารถตรวจสอบความครบถ้วนของข้อมูลการจองได้ และแจ้งให้ผู้ใช้กรอกให้ครบในกรณีที่ผู้ใช้กรอกข้อมูลไม่ครบ
- สรุปและสิ่งที่คุณได้จากการทดลองนี้ (Conclusion)
  ระบบการจองที่นั่งสามารถทำงานได้สมบูรณ์และถูกต้องตามที่ได้ระบุไว้ใน Requirement

2. ระบบยกเลิกการจอง

- จุดประสงค์ของการทดลอง (Goal of the experiment)
  - เพื่อทดสอบว่าปุ่มยกเลิกการจองสามารถตอบสนองได้อย่างถูกต้อง
  - เพื่อทดสอบว่าข้อมูลการจองถูกลบออกจากระบบได้
  - เพื่อทดสอบว่าหลังจากยกเลิกการจองแล้ว ที่นั่งว่างในระบบของร้านนั้นๆจะเพิ่มขึ้นตามจำนวนที่ยกเลิกไป
- สิ่งที่จะวัด (Measurement metrics)
  - การตอบสนองของปุ่มยกเลิกการจอง
  - การลบข้อมูลการจองออกจากระบบ
  - จำนวนที่นั่งว่างในร้าน
- วิธีทำการทดลองและสิ่งที่ต้องใช้ในการทดลอง (Setup and methods of your experiment)
  - วิธีทำการทดลอง
    - เข้าสู่ระบบ
    - เลือกหน้าตรวจสอบรายการจอง
    - กดปุ่มยกเลิกการจองหลังรายการที่ต้องการยกเลิกการจอง
    - กดปุ่มยืนยัน
  - สิ่งที่ต้องใช้ในการทดลอง
    - E-mail และ รหัสผ่าน ที่เป็นสมาชิกของเว็บ เพื่อใช้เข้าสู่ระบบ
    - ข้อมูลของรายการที่ต้องการยกเลิกการจอง เช่น ชื่อร้าน, วัน, เวลา ที่จองไว้
- ผลที่ได้จากการทดลอง (Experimental results)
  - เมื่อกดปุ่มยกเลิกการจองบนหน้าเว็บ รายการจองนั้นหายไปจากหน้ารายการจองของผู้ใช้จริง
  - ข้อมูลรายการจองนั้นหายไปจากฐานข้อมูล หลังจากกดปุ่มยกเลิกการจองแล้ว
  - หลังจากยกเลิกการจอง ที่นั่งร้านในร้านอาหารนั้น ณ วัน, เวลา และ พื้นที่นั้นๆ มีจำนวนว่างมากขึ้นเท่ากับจำนวนที่นั่งที่ทำการยกเลิกไป
- สรุปและสิ่งที่คุณได้จากการทดลองนี้ (Conclusion)
  ระบบยกเลิกการจองสามารถทำงานได้ถูกต้องและสมบูรณ์

บทสรุป

สิ่งที่กลุ่มผู้พัฒนาคาดหวังจากผู้ใช้งานแอพพลิเคชั่นนี้คือ ความสะดวกสบายที่ได้รับหลักจากใช้งานแอพลิเคชั่น และมีความพึงพอใจกับลักษณะของการจัดการและการใช้งานที่สะดวก เนื่องจากปัญหาการรอคิวและการจัดการที่อาจจะไม่ได้มาตรฐานของร้านอาหาร อาจจะทำให้ลูกค้าหมดอารมณ์ที่ตั้งใจจะเข้าไปใช้บริการร้านอาหารนั้น และเกิดเป็นความเสียหายหรือสูญรายได้ที่จะได้จากกลุ่มลูกค้าเหล่านั้น จนอาจขนาดที่ว่าลูกค้าเหล่านั้นไปมองหาร้านอาหารประจำอื่นแทนและอาจจะไม่เข้าใช้บริการร้านเดิมอีกต่อไป ซึ่งเป็นเรื่องที่เกิดขึ้นเป็นประจำสำหรับร้านอาหารที่มีลูกค้ามากมายแต่การจัดการคิวอาจจะไม่ดีพอ หรือลูกค้าไม่สะดวกที่จะใช้รูปแบบบริการการจองของร้าน ซึ่งทางผู้พัฒนาได้เล็งเห็นปัญหาเหล่านี้ จึงได้มีความตั้งใจพัฒนาระบบให้สอดคล้องกับความต้องการและเป็นไปในรูปแบบที่ใช้งานได้สะดวกสบายทั้งในกลุ่มผู้ใช้ที่เป็นลูกค้าของร้านอาหาร และร้านอาหารเองก็สามารถเข้ามาใช้บริการการจัดการของแอพพลิเคชั่นของเราได้ ดังนั้นแล้วกลุ่มผู้พัฒนาแอพพลิเคชั่นจึงหวังว่าผู้ใช้จะพึงพอใจและใช้บริการแอพลิเคชั่นของเราต่อไป

สำหรับสิ่งที่ผู้พัฒนาได้รับจากการจัดทำเว็บแอพพลิเคชั่นนี้คือ การพัฒนาเว็บแอพพลิเคชั่นจะเกิดขึ้นได้นั้นจำเป็นต้องมีการวางแผนในหลายส่วน โดยเริ่มตั้งแต่การคิด requirement ให้ตรงกับความต้องการของการใช้งาน พร้อมทั้งมองหางานต้นแบบที่คล้ายคลึงกับแอพพลิเคชั่นของเรา เพื่อนำมาพัฒนาให้ดีขึ้นจากการมองหาจุดบอดของงานต้นแบบ และต่อมาคือการวางแผนการออกแบบการเขียนโปรแกรมให้ออกมาใช้งานได้และสามารถนำไปพัฒนาต่อได้ โดยการเลือกใช้การออกแบบแบบ object oriented นั้นทำให้เราได้ฝึกฝนและเรียนรู้ในสิ่งที่ไม่เคยทำ และทำให้รู้ว่า object oriented design นั้นมีข้อดีข้อเสียอย่างไร พร้อมทั้งยังได้เข้าใจการออกแบบแบบ object oriented ที่ถูกต้องควรเป็นอย่างไร ต่อจากนั้นยังได้ฝึกการเขียนเว็บแอพพลิเคชั่นเพื่อการใช้งานโดยใช้ PHP และ Laravel Framework ซึ่งเป็นสิ่งที่ผู้พัฒนาไม่เคยได้เรียนมาก่อนจึงทำให้ได้ประสบการณ์ใหม่ๆเพิ่มมากขึ้น และสุดท้ายคือการได้เรียนรู้การใช้งานของ Git ในรูปแบบ Git workflows ซึ่งถือเป็น version control ตัวแรกที่ผู้พัฒนาเคยได้ใช้ร่วมกัน ทำให้เกิดการเรียนรู้ในการใช้ version control สำหรับการทำงานร่วมกันเป็นกลุ่ม ดังนั้นแล้วสิ่งที่ได้เรียนรู้ทั้งหมดนี้จึงถือเป็นประสบการณ์ที่ดีในการเริ่มทำสิ่งใหม่ๆ ที่ไม่เคยได้ลองและเรียนรู้มันมาก่อน

บรรณานุกรม

Dayle Rees. Laravel: Code Bright. พิมพ์ครั้งที่ 4. : Leanpub, 2014

Jeffrey Way. Laravel Testing Decoded. : Leanpub, 2013

http://stackoverflow.com

http://laravel.com

https://getcomposer.org

http://www.alessioatzeni.com/blog/brushed-template/

http://laravel.io/forum

http://culttt.com/2014/04/07/working-configuration-laravel-4/