

การสอบ บทที่ 1-3
เรื่อง แอปพลิเคชันช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ด
Application Find The Way home with QR Scan

นายชัยุต สถิตยธรรม (ปร)

64123709

โทร. 0858259064

นาย ชัยวุฒิ เตชะดำรงธรรม (ปร)

64123710

โทร. 0966797272

อาจารย์ที่ปรึกษา

ผศ.ดร.รัศลิน เพตະกร

กรรมการ

อ.ดร.พิรุษ พาก้วฟุ่งรังษี

ผศ.ภาณุวัฒน์ สุวรรณกุล

บทที่1 บทนำ

1.1 ความสำคัญและที่มาของปัญหา

ผู้สูงอายุในสังคมไทยเพิ่มขึ้นอย่างรวดเร็ว คาดว่ามีผู้ป่วยด้วยโรคสมองเสื่อมอยู่ราวร้อยละ 2-4 ของประชากรไทยในวัย 60 ปีขึ้นไป หรือคิดเป็นสัดส่วนร้อยละ 60-70 ของผู้ตรวจพบว่ามีอาการ สมองเสื่อมในปัจจุบัน ด้วยจำนวนประชากรที่เพิ่มขึ้นและช่วงชีวิตที่ยืนยาวขึ้น จำนวนผู้ป่วยโรคอัลไซ เมอร์มีแนวโน้มเพิ่มสูงขึ้นเรื่อยๆ ในอนาคต จากยอดผู้ป่วยไทยที่มีภาวะสมองเสื่อมจำนวน 229,000 รายที่เคยสำรวจพบในปี พ.ศ. 2548 เป็นที่คาดการณ์ว่าตัวเลขจะเพิ่มสูงขึ้นเป็น 450,000 คนในปี พ.ศ. 2568 และอาจเพิ่มขึ้นเป็นจำนวนมากกว่า 1 ล้านคนในอีกราว 40 ปีข้างหน้า ผู้ป่วยอัลไซเมอร์มี ความบกพร่องทางความจำที่ทำให้เกิดการหลงทางได้ง่าย การหลงทางเป็นปัญหาที่ร้ายแรงสำหรับ ผู้ป่วยอัลไซเมอร์ เนื่องจากผู้ป่วยอาจเสียต่อการเกิดอุบัติเหตุ เช่น การถูกรถชน หรือการตกลงในที่ไม่ ปลอดภัย อีกทั้งผู้คนในสังคมยังไม่ทราบถึงปัญหานี้อย่างแพร่หลาย การขาดเครื่องมือที่สามารถช่วย ให้ผู้ป่วยหากทางกลับบ้านได้อย่างมีประสิทธิภาพเป็นอีกหนึ่งปัญหาสำคัญที่ต้องได้รับการแก้ไข (Hitap, 2556)

จากนั้นจึงได้ค้นหาและศึกษาระบบแอปพลิเคชันที่มีวัตถุประสงค์หรือแนวทางคล้ายคลึงกับ แอปพลิเคชันที่กำลังจะพัฒนาขึ้นเพื่อเป็นแนวทาง จึงได้พบรหัสแอปพลิเคชันแนะนำสถานที่ท่องเที่ยว และที่พักในอำเภอจอมทอง จังหวัด เชียงใหม่บนระบบปฏิบัติการแอนดรอยด์ (เที่ยงจันตา, 2565) ที่ โดยแอปพลิเคชันนี้มีการใช้ระบบปฏิบัติการกูเกิล แมพ ที่เป็นระบบแบบสามาตรที่ทุกคนสามารถเข้าถึง ได้ และมีฐานข้อมูล มหาเอกสารวิทยาศาสตร์ เป็นตัวช่วยเก็บข้อมูลที่บันทึกไว้ได้หลากหลาย และยังดูแลการ จัดการระบบได้อย่างสะดวก จึงทำให้นำรูปแบบของแอปพลิเคชันดังกล่าวมามีเป็นมีระบบวัตถุประสงค์ หรือแนวทางคล้ายคลึงกับที่ต้องการ

จากปัญหาดังกล่าวมาข้างต้นจึงได้พัฒนาแอปพลิเคชันช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์ โค้ด แอปพลิเคชันมีระบบสแกนจดจำใบหน้า สามารถช่วยเหลือผู้ป่วยอัลไซเมอร์โดยค้นหาเส้นทาง กลับบ้านได้ แอปพลิเคชันมีระบบนำทางที่สามารถเข้าสู่ระบบแล้วกรอกข้อมูลตั้งกล่าวและถ่ายรูป ภาพของผู้ป่วยลงไว้ในแอปพลิเคชันที่ระบบต้องการไว้ จากนั้นระบบผู้ดูแลจะทำการตรวจสอบ ข้อมูลตั้งกล่าวและกระทำการส่งข้อมูลตำแหน่งที่อยู่ของผู้ป่วยผ่านระบบ Google map ให้ผู้ใช้ได้รับรู้ เพื่อดำเนินการช่วยเหลือผู้ป่วยในการนำทางกลับบ้านอย่างมีประสิทธิภาพ ซึ่งในปัจจุบันยังไม่มีเครื่อง เครื่องมือหรือแอปพลิเคชันที่สามารถตอบโจทย์นี้ได้อย่างมีประสิทธิภาพ

1.2 วัตถุประสงค์ของการโครงการ

1.2.1 พัฒนาแอปพลิเคชันเพื่อช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ด สำหรับผู้ป่วยหรือผู้สูงอายุ

1.2.2 เพื่อเป็นการทดสอบแอปพลิเคชันเพื่อช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ด สำหรับผู้ป่วยหรือผู้สูงอายุ

1.3 ประโยชน์ที่คาดว่าจะได้รับ

1.3.1 ได้แอปพลิเคชันที่สามารถช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ดลดภาระและความกังวลของผู้ดูแล

1.3.2 ได้ทดสอบระบบแอปพลิเคชันเพื่อช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ด สำหรับผู้ป่วยหรือผู้สูงอายุ

1.4 ขั้นตอนและวิธีการดำเนินงาน

1.4.1 ศึกษาและสืบค้นข้อมูลเกี่ยวกับผู้สูงอายุ และโปรแกรมที่ใช้ในการพัฒนา

- 1) รวบรวมข้อมูลเกี่ยวกับผู้สูงอายุ
- 2) ศึกษาระบบปฏิบัติการแอนดรอยด์ (Andriod)
- 3) ศึกษา ฟลัตเตอร์ เฟรมเวิร์ค (Flutter Framework)
- 4) ศึกษาข้อมูลเกี่ยวกับโปรแกรมประยุกต์บนอุปกรณ์พกพาหรือโทรศัพท์เคลื่อนที่มีระบบปฏิบัติการแอนดรอยด์ (Andriod Operating System) ศึกษาโดยการใช้โปรแกรม Visual Studio Code

1.4.2 วิเคราะห์ความต้องการ (Requirement Analysis) โดยการพัฒนา สำหรับค้นหา เป้าหมาย บนแผนที่ Google Map ของ แอปพลิเคชันบนระบบปฏิบัติการแอนดรอยด์ ส่วนของผู้ใช้งาน เข้าสู่ระบบ เพิ่ม ลบ แก้ไข ข้อมูลรายละเอียดข้อมูลของตนเองได้ส่วนของผู้ดูแลระบบ สามารถเข้าสู่ระบบได้ สามารถกำหนดสิทธิการเข้าถึงของผู้เยี่ยมชมได้ สามารถดูข้อมูลการใช้งานของผู้เยี่ยมชมได้ และยังสามารถ เพิ่ม ลบ แก้ไข ผู้ใช้งานได้ สำหรับ แอปพลิเคชันช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ด

1.4.3 ขั้นตอนการออกแบบระบบ (Design) ออกแบบระบบการทำงาน ข้อมูลนำเข้า การแสดงผล ข้อมูลลำดับตามขั้นตอนให้สอดคล้องกับความต้องการของผู้ใช้ วิเคราะห์ข้อมูลการออกแบบระบบโดยใช้ UML (Unified Modeling Language) ได้แก่

1.4.3.1 ยูริเคสไดอะแกรม (Use Case Diagram) เพื่อแสดงแผนภาพแสดงการทำงาน ของระบบผู้ใช้ระบบ (User) และความสัมพันธ์กับระบบย่อย (Sub systems) ได้แก่ เอเยี่ยสเคส ไดอะแกรมมาใช้อธิบายดังนี้ เมื่อผู้ใช้งานเข้าแอปพลิเคชัน ระบบจะแสดงภาพรวมของแอปพลิเคชัน รวมถึงรายละเอียดต่าง ๆ ทั้งหมดในแอปพลิเคชัน

1.4.3.2 ชีเควนซ์ไดอะแกรม (Sequence Diagram) เพื่อแสดงส่วนต่างๆ ของแอปพลิเคชันช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ดแสดงลำดับกิจกรรมการใช้งานผู้ใช้งาน โดยที่สามารถเห็นถึงการทำงานทั้งระบบตั้งแต่ต้นจนจบการทำงาน

1.4.3.3 แอ็คทิวิตี้ไดอะแกรม (Activity Diagram) ใช้ในส่วนของการแสดง Workflow การทำงานของกิจกรรมของระบบจากจุดหนึ่งไปสู่อีกจุดหนึ่งตั้งแต่ต้นจนจบโปรแกรม

1.4.3.4 คลาส ไดอะแกรม (Class Diagram) ของแอปพลิเคชันช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ด จะแสดงถึงการทำงานของระบบจะแบ่งออกเป็นคลาสโดยแต่ละคลาสนั้นจะแสดงส่วนของตัวข้อมูลระบบที่จะถูกใช้งาน เพื่อให้มองเห็นภาพรวมระบบ

1.4.4 ขั้นตอนการเขียนโปรแกรม (Programming) เขียนโปรแกรมที่ออกแบบไว้เพื่อให้สอดคล้องกับ วัตถุประสงค์ สร้างระบบงานโดยโปรแกรม Visual Studio Code ที่ใช้ฟลัตเตอร์ เฟรมเวิร์ก (Flutter Framework) ในการเขียนโปรแกรม ประกอบการใช้งานแอปพลิเคชัน เอกสารประกอบการใช้งานเพื่อนำเสนอเป็นรูปเล่ม

1.4.5 ขั้นตอนการทดสอบระบบ (Testing) มีการทดลองใช้ระบบและรับฟีดแบ็คเพื่อนำมาแก้ไขและบำรุงดูแลแอปพลิเคชันให้ดียิ่งขึ้น

1.4.6 มีการแก้ไขเพิ่มเติมตามที่ได้ทดสอบเพื่อให้ประสิทธิภาพดีที่สุด

1.4.7 ขั้นตอนในการจัดทำเอกสารเกี่ยวกับระบบ (Document) คู่มือการติดตั้งและการใช้งานประกอบการใช้งานแอปพลิเคชัน เอกสารประกอบการใช้งานเพื่อนำเสนอในรูปแบบรูปเล่ม

1.5 ขอบเขตการจัดโครงการ

ขอบเขตการพัฒนาแอปพลิเคชันช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ดตั้งต่อไปนี้
ด้านผู้ดูแลระบบ

1.5.1 ผู้ดูแลระบบสามารถจัดเก็บข้อมูลผู้ป่วย อนุญาตให้ผู้เยี่ยมชมสามารถเข้าถึงข้อมูลผู้ป่วยได้โดยการขออนุญาต

1.5.2 ผู้ดูแลระบบสามารถตรวจสอบข้อมูลผู้เยี่ยมชมและทำการอนุญาตให้ผู้เยี่ยมชมเข้าถึงข้อมูลได้

1.5.3 ผู้ดูแลระบบสามารถดูแลระบบ เพิ่ม ลบ และแก้ไขข้อมูลของผู้ป่วยได้ทั้งหมด

1.5.4 ผู้ดูแลระบบเมื่อได้รับข้อมูลจากผู้เยี่ยมชมจะทำการตรวจสอบและส่งข้อมูลที่อยู่ด้วยการสแกนให้ผู้เยี่ยมชม

ด้านผู้เยี่ยมชม

1.5.5 ผู้เยี่ยมชม ต้องกรอกเบอร์ติดต่อปัจจุบัน และ ต้องถ่ายรูปภาพผู้ป่วย

1.5.6 ผู้เยี่ยมชม เมื่อกรอกส่งข้อมูลแล้วรอซักครู่เพื่อรับข้อมูลของผู้ป่วย

1.5.7 ผู้เยี่ยมชม สามารถใช้ Map เพื่อนำทางผู้ป่วยกลับถึงบ้านได้

ด้านผู้ป่วย

1.5.8 ผู้ใช้จำเป็นต้องสมัครสมาชิกทางแอปมินโดยตรงเพื่อใช้งาน

1.6 รายละเอียดเครื่องมือที่ใช้ในการจัดการ

- 1.6.1. คอมพิวเตอร์ส่วนบุคคล (PC) - รุ่น: ASUS TUF Gaming A15 FA506QM_FA506QM ฟอร์มแฟกเตอร์: แล็ปท็อป Windows 11 (64 บิต)
1.6.2. ข้อมูลโปรเซสเซอร์: ผู้จำหน่าย CPU: AuthenticAMD ยี่ห้อ CPU: AMD Ryzen 7 5800H with Radeon Graphics ตระกูล CPU: 0x19 รุ่น CPU: 0x50 จำนวนการปรับปรุง CPU: 0x0 ชนิด CPU: 0x0 ความเร็ว: 3194 Mhz 16 หน่วยประมวลผลทางตรรกะ 8 หน่วยประมวลผลทางกายภาพ HyperThreading:

1.6.3. โปรแกรมที่ใช้พัฒนา:

- โปรแกรม Microsoft Word 2019 ใช้ในการทำเอกสาร
- โปรแกรมวิชาลสตูดิโอโคด (Visual Studio Code) ใช้ในการเขียนแอปพลิเคชัน
- โปรแกรมแอนดรอยด์สตูดิโอ (Android Studio) ใช้ในการพัฒนาแอปพลิเคชัน
- ภาษา Dart ใช้ในการพัฒนาแอปพลิเคชัน

1.7 สถานที่ใช้ในการทำวิจัยและเก็บข้อมูล

ห้องเรียนคณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยราชภัฏเชียงใหม่ (سلح)

บทที่2

ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในการพัฒนาแอปพลิเคชันช่วยเหลือสันทางกลับบ้านด้วยคิวอาร์โค้ด ได้ทำการศึกษาได้มาแบ่งทฤษฎีและงานวิจัยที่ เกี่ยวข้องออกเป็น 6 ส่วนใหญ่ โดยมีรายละเอียดดังนี้

- 2.1 ความรู้ทั่วไปเกี่ยวกับผู้ป่วยอัลไซเมอร์
- 2.2 คิวอาร์โค้ด
- 2.3 ภูเก็ตแมปและเอปีโอล
- 2.4 การวิเคราะห์และออกแบบระบบเชิงวัตถุ
- 2.5 โปรแกรมและภาษาที่ใช้พัฒนาโปรแกรม
- 2.6 ระบบการจัดการฐานข้อมูล
- 2.7 การทดสอบซอฟต์แวร์
- 2.8 งานวิจัยที่เกี่ยวข้อง

2.1 ความรู้ทั่วไปเกี่ยวกับผู้ป่วยอัลไซเมอร์

2.1.1 โรคอัลไซเมอร์คือความบกพร่องในการทำหน้าที่ของสมองในกลุ่มโรคสมองเสื่อม อาการที่เด่นชัดคือ ความบกพร่องในกระบวนการทางความคิด ความจำ หลงวัน เวลา สถานที่ และแม้กระทั่งลืมคน ใกล้ชิดหรือผู้ดูแล ความสามารถทางสติปัญญาลดลง เรียนรู้ได้ช้าลง ในด้านการรับรู้ก็จะเปลี่ยนไป ความสนใจและสามารถลดลง การต้อตอบและความสามารถในการอยู่ร่วมกับผู้อื่นในสังคมก็ลดลง ผู้ป่วยอัลไซเมอร์ จำเป็นต้องได้รับการดูแลเอาใจใส่อย่างถูกวิธี เนื่องจากผู้ป่วยจะมีความสามารถในการทำกิจกรรมลดลงเรื่อยๆ การสังเกตอย่างເອາໄສ และการเข้าใจการดำเนินโรคจะเป็นสิ่งที่จะช่วยประคับประคองให้ผู้ป่วยอยู่ได้อย่างมีความสุข และตัวผู้ดูแลเองก็จะไม่เกิดความวิตกกังวลและรู้สึกเหนื่อยมาก จากภาวะและปัญหาดังที่กล่าวมา เมื่อผู้ดูแลต้องดูแลผู้ป่วยเป็นนานๆ มักจะเกิดความอ่อนล้า เกิดความเครียด เปื่อยหน่าย กังวล ซึมเศร้า และบางครั้งอาจรู้สึกผิดหวังที่ผู้ป่วยอาการไม่ดีขึ้น ทั้งยังแสดงออกถึงการไม่พอใจในการดูแลของตนและไม่ให้ความร่วมมือในการดูแลตนเอง ถึงจุดนี้ผู้ทำหน้าที่ดูแลผู้ป่วยก็มักจะมีคำถามว่า จะดูแลผู้ป่วย อัลไซเมอร์อย่างไรดี เพื่อให้ทั้งตนและผู้ป่วยมีความสุข

2.1.2 ทำความเข้าใจโรคอัลไซเมอร์ การทำความเข้าใจกับโรคสมองเสื่อมและอัลไซเมอร์ให้ดี โดยหาความรู้จากสื่อต่างๆ เช่น อ่านหนังสือ พงวิทยุ ดูโทรทัศน์ อ่านบทความทางอินเตอร์เน็ต หรืออาจสอบถามจากแพทย์ที่ดูแลผู้ป่วย เมื่อมีความรู้ความเข้าใจ ไม่ว่าจะเป็นลักษณะอาการ ระยะเวลา วิธีการรักษาการพยากรณ์ โรค ตลอดจนวิธีการช่วยเหลือดูแลผู้ป่วยแล้ว ก็จะสามารถรับมือและแก้ปัญหาต่างๆ ที่จะเกิดขึ้นได้อย่างเหมาะสมยอมรับผู้ป่วย และอาการของโรคของผู้ป่วยว่าโรคนี้รักษาไม่หาย การดูแลที่เหมาะสมจะทำให้ผลกระทบด้านต่างๆ ลดลงได้

2.1.3 ผู้ดูแลเข้าใจผู้ป่วยอัลไซเมอร์ ควรแก้ไขอารมณ์และพฤติกรรมที่เป็นปัญหามากที่สุดของผู้ป่วย ก่อน เพราะจะช่วยทำให้การดูแลผู้ป่วยง่ายขึ้น สิ่งใดก่อให้เกิดอารมณ์หรือความไม่พอใจแก่ผู้ป่วย ควรหาสาเหตุ

แก้ไขหรือหลีกเลี่ยง ช่วยลดความเครียดแก้ผู้ป่วย ถ้าผู้ดูแลเข้าใจถึงจุดนี้ ก็จะไม่รู้สึกว่าตนเองดูแลผู้ป่วยได้ไม่ดี พอ ความเครียดก็จะไม่เกิดขึ้นบางครั้งผู้ป่วยอาจแสดงอาการณที่ทำให้ผู้ดูแลรู้สึกผิดหวัง ผู้ดูแลต้องเข้าใจว่าเป็นผลมาจากการของโรค ไม่ใช่ผู้ป่วยไม่พึงพอใจ โกรธ หรือตั้งใจจะต่อว่าผู้ดูแล เนื่องจากก่อนป่วยผู้ป่วยมีเด้มีบุคลิกภาพเช่นนั้น

2.1.4 ให้ผู้ป่วยอัลไซเมอร์เข้าใจตนเองว่าทราบอะไรที่เปลี่ยนไปของผู้ป่วยในขณะที่ยังสามารถรับรู้และเข้าใจได้ตั้งแต่อาการยังไม่รุนแรงนัก เพื่อให้ผู้ป่วยเตรียมพร้อมและยินดีให้ความร่วมมือในการดูแลตนเอง คอยให้กำลังใจและสนับสนุนให้ผู้ป่วยเข้าใจว่า มีกิจวัตรประจำวันหลายอย่างที่ผู้ป่วยสามารถทำเองได้ เพื่อให้ผู้ป่วยไม่รู้สึกด้อยค่าหรือเป็นภาระ จะทำให้ผู้ป่วยเกิดความภาคภูมิใจ รู้สึกมีคุณค่า และมีความมั่นใจมากขึ้น

2.1.5 หลีกเลี่ยงและป้องกันก่อนเกิดปัญหาผู้ป่วยจำนำวนมากเมื่อมีอาการ อาจทำให้การพูดคุยสื่อสารได้ลดลง อาจจะพูดไม่ออก หรือคุนิ่งเงียบ แต่โดยปกติจะฟังเข้าใจ ฉะนั้นการพูดแต่ไม่เห็นผู้ป่วยตอบสนองอาจจะไม่ใช่ไม่ได้ยินหรือไม่เข้าใจหลีกเลี่ยงคำพูดหรือพฤติกรรมใดๆ ที่ส่งผลกระทบทางอารมณ์กับผู้ป่วย อะไรที่ผู้ป่วยไม่ชอบ โกรธ เสียใจ ผิดหวังก็และเกิดความเครียด และเพิ่มการเก็บตัว ซึ่งควรได้ไม่หลอกล่อให้ผู้ป่วยทำสิ่งใดสิ่งหนึ่ง แล้วไม่รักษาสัญญาที่ให้ไว้กับผู้ป่วย หากรับปากผู้ป่วยไว้อย่างไรก็ควรทำตามนั้น เพราะผู้ป่วยจะหมดความเชื่อถือและเกิดการต่อต้านได้ ไม่วิจารณ์ ต่อว่า โต้เลียง หรือต่อตะครอผู้ป่วยต่อหน้าผู้อื่น โดยเฉพาะเวลาที่ผู้ป่วยจำอะไรไม่ได้ เพราะจะทำให้ผู้ป่วยอับอาย และอย่าทำโทษผู้ป่วย เพราะจะทำให้ผู้ป่วยเกิดความรู้สึกผิดนำไปสู่อาการทางจิตได้ทำสร้อยหรือกำไลข้อมือให้ผู้ป่วยสวมข้อมือไว้ โดยมีข้อความระบุว่า ผู้ที่สวมใส่มีปัญหาด้านความจำ พร้อมใส่หมายเลขโทรศัพท์ของญาติหรือผู้ดูแล เพราะหากผู้ป่วยพลัดหลงหรือออกจากบ้านไปโดยไม่มีใครรู้ เมื่อมีผู้พบเห็นจะได้ติดต่อผู้ดูแลได้ ทำให้ตามหาตัวผู้ป่วยได้ง่ายขึ้น

2.1.6 ปฏิบัติตามคำแนะนำของแพทย์

1) เพื่อให้การรักษาได้ผลดียิ่งขึ้น ผู้ดูแลควรสังเกตอาการที่ผิดปกติของผู้ป่วย บันทึกพฤติกรรมและแจ้งให้แพทย์ทราบเมื่อถึงเวลาดัดตรวจโรค

2) การรักษาผู้ป่วย อาจมีความจำเป็นที่ต้องใช้ยาที่ออกฤทธิ์ต่อจิตประสาท เช่น นอนไม่หลับ วิตกกังวล ซึมเศร้า ก้าวร้าว ผู้ดูแลต้องให้ผู้ป่วยรับประทานยาอย่างสม่ำเสมอและสังเกตอาการหลังจากใช้ยาเพื่อแจ้งแก่แพทย์ได้อย่างถูกต้อง

3) หากผู้ป่วยมีอาการผิดปกติควรรีบปรึกษาแพทย์ เช่น อาการนอนไม่หลับ วิตกกังวล หรือซึมเศร้ามากเกินไป พฤติกรรมเปลี่ยนแปลง ก้าวร้าว หลงผิด หูแว่ว หรืออาการทางจิตอื่นๆ

2.1.8 การดูแลตนเองของผู้ดูแลผู้ป่วยโรคอัลไซเมอร์ ผู้ดูแลผู้ป่วยควรจัดสรรเวลาพักผ่อนให้เพียงพอ โดยเปลี่ยนให้ผู้อื่นดูแลผู้ป่วยแทนบ้าง เพราะการดูแลผู้ป่วยอย่างต่อเนื่องนานๆ อาจทำให้เกิดความอ่อนล้า ความเครียด หงุดหงิด ซึ่งไม่เป็นผลดีต่อทั้งผู้ดูแลและผู้ป่วยในระยะยาว ผู้ดูแลผู้ป่วยควรมีเวลาทำกิจกรรมที่ตนเองชอบ เพื่อความผ่อนคลาย หากผู้ป่วยสามารถร่วมกิจกรรมต่างๆ ไปพร้อมกันได้ ก็จะเป็นการสร้างความสัมพันธ์และส่งผลดีต่อทั้งผู้ดูแลและผู้ป่วยไปพร้อมๆ กัน

2.2 คิวอาร์โค้ด

2.2.1 คิวอาร์โค้ด (QR Code) คือ สัญลักษณ์รูปสี่เหลี่ยมที่ประกอบไปด้วยโมดูลสีดำ (จุดสีเหลี่ยม) ที่จัดอยู่ตารางสี่เหลี่ยมพื้นผ้าสี่ขา เป็นสัญลักษณ์แทนข้อมูล ซึ่งอุปกรณ์ที่ใช้ในการสแกน QR Code นั้นก็จะใช้กล้องที่ติดกับโทรศัพท์มือถือในการสแกน นอกจากนี้ QR Code ยังเป็นสัญลักษณ์ที่ได้รับความนิยมเป็นอย่างมากเนื่องจากช่วยความสะดวกสบาย

2.2.2 ประเภทของ QR Code ในปัจจุบันนี้จะมีทั้งหมด 5 ประเภท ดังนี้

1) QR Code Model เป็น QR Code แบบดั้งเดิมที่มีขนาดใหญ่ที่สุด โดยจะมีขนาด 73×73 โมดูล สามารถบรรจุข้อมูลได้มากถึง 1,167 ตัว และอีกหนึ่งแบบ คือ Model 2 เป็นเวอร์ชันที่ถูกพัฒนามาจาก Model 1 สามารถบรรจุข้อมูลได้มากถึง 7,089 ตัว ซึ่งในปัจจุบันนี้ Model 2 เป็นที่นิยมใช้กันอย่างมากเลยที่เดียว

2) Micro QR Code เป็น QR Code ที่มีขนาดเล็กกว่าแบบแรกมากพอสมควร เนื่องจากจะแสดงผลบางจุดตรวจสอบได้แต่ละแผ่นเพียงตัวเดียว ซึ่งขนาดใหญ่ที่สุดของ Micro QR Code 17×17 โมดูล ที่สามารถบรรจุข้อมูลได้ทั้งหมด 35 ตัว

3) IQR Code เป็น QR Code ที่มีขนาดเล็กและพิมพ์ออกมานานวนอน ที่สามารถทำการเก็บข้อมูลได้มากกว่า 80% แต่หากมีการจัดเก็บข้อมูลในปริมาณที่เท่ากัน ก็จะประหยัดพื้นที่ในการแสดงผลได้มากถึง 30% ซึ่งสามารถเก็บข้อมูล 40,000 ตัวอักษร

4) SQRC เป็น QR Code ที่จะมีคุณลักษณะเหมือนกับ QR Code Model 1 และ QR Code Model 2 ทุกประการ แต่จะมีข้อแตกต่างกันเล็กน้อย คือ สามารถทำการเก็บข้อมูลลับได้นั่นเอง

5) Frame QR สามารถนำรูปภาพ กราฟิกมาติดบริเวณกลางของ QR Code ได้ ส่วนใหญ่ก็จะนิยมใช้ QR Code ประเภท Frame QR ในงานประชาสัมพันธ์ Event นิทรรศการ เพื่อทำให้เกิดความสดุดตามากที่สุดนั่นเอง

2.2.3 ระบบ QR Code นำมาใช้กับระบบสแกน QR Code ในปัจจุบันนี้ เป็นสิ่งที่ได้รับความนิยมเป็นอย่างมาก เนื่องจากสามารถนำมาประยุกต์ใช้งานได้หลากหลายรูปแบบมาก ซึ่งสามารถนำมาประยุกต์ใช้ได้ดังนี้

1) ใช้ในการชำระเงินได้ทันที ซึ่งในยุคนี้จะนิยมนำระบบคิวอาร์โค้ดมาใช้ในการชำระเงิน เนื่องจากสามารถทำการชำระเงินได้อย่างรวดเร็ว ทั้งยังตอบโจทย์การใช้เงินในสังคมໄร์เงินสดอีกด้วย

2) ใช้โปรโมทร้านค้า เนื่องจากในยุคนี้คนจำนวนมากไม่ชอบอ่านหนังสือที่มีจำนวนมาก ดังนั้นการนำ QR Code ไปแปะไว้บนโปสเตอร์ก็จะทำให้กลุ่มลูกค้าสามารถเข้าถึงร้านได้ง่ายขึ้นและรวดเร็วมากยิ่งขึ้นนั่นเอง

3) โปรโมชั่นเสริมการขาย โดยอาจจะใช้ในการสแกนเพื่อรับสะสมแต้ม รับโปรโมชั่นพิเศษ ต่างๆ รวมถึงการทำแบบสอบถามเพื่อรับส่วนลด

4) อีเวนท์ สำหรับการจัดงาน Event ที่มีการสแกนเพื่อละเบียนเข้างานการใช้คิวอาร์โค้ด ถือเป็นสิ่งที่ช่วยเพิ่มความรวดเร็วในการลงทะเบียนได้เป็นอย่างดี ทำให้ไม่ต้องต่อแถลงลงทะเบียน เป็นการสร้างความประทับใจให้กับผู้ร่วมงานได้เป็นอย่างดี

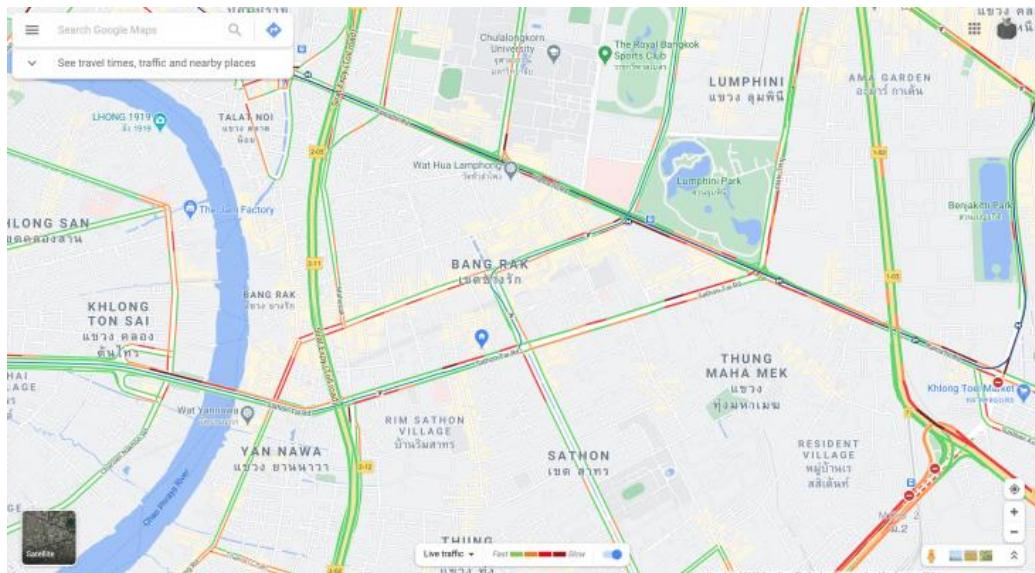
2.2.4 QR Code เป็นบาร์โค้ดที่มีทั้งหมด 2 มิติ ที่มีต้นกำเนิดมาจากประเทศญี่ปุ่น โดยบริษัท เดโนโซ เวฟ ตั้งแต่ปี ค.ศ 1994

2.2.5 ความแตกต่างระหว่างบาร์โค้ดกับคิวอาร์โค้ด คือ QR Code นั้นจะมีลักษณะเป็นบาร์โค้ด 2 มิติ ในขณะที่ Barcode นั้นมี 1 มิติ และ Barcode จะมีลักษณะเป็นเส้นสีดำยาวอยู่บนพื้นสีขาว ส่วน QR Code จะมีลักษณะเป็นสีเหลือง ซึ่งทั้งบาร์โค้ดและคิวอาร์โค้ดนั้นจะสามารถเห็นได้ง่ายทั่ว ๆ ไป

2.3 ภูเก็ตแมปເອີ້ໂ

ภูเก็ตแมปເອີ້ໂ (Google Map API) เป็นชุดເອີ້ໂ (API) ของ Google สำหรับพัฒนา เว็บແອປ ພລິເຄັນ ແລະ ໂມບາຍແອປພລິເຄັນ (Android, iOS) ໄວສໍາຮັບເຮັກໃຫ້ແຜນທີ່ແລະຊຸດ ເຊວງວິສ ຕ່າງ ๆ ຂອງภູກຶກ ເພື່ອພັນນາ ແອປພລິເຄັນ ໄດ້ເໝືອນກັບທີ່ ຜູກຶກ ໂດຍມີແຜນທີ່ຢັງໜ້າຕ່າງ ມາກມາຍໃຫ້ເຮັກໃຫ້ປະກອບດ້ວຍ

- การปรับแต่งແຜນທີ່ (Styled Map)
- ຊຸດຄວບຄຸມແຜນທີ່ (Map Control)
- ຊຸດເຄື່ອງມືວັດພາບນັບແຜນທີ່ (Drawing)
- ການນໍາທາງຈາກຈຸດໜຶ່ງໄປຢັງອັກຈຸດໜຶ່ງ (Directions Service)
- ກາຣນຸວັນຄວາມສູງຂອງຈຸດພິກັດ (Elevation Service)
- ກາຣແປລງທີ່ຍູ້ເປັນພິກັດ Latitude ແລະ Longitude (GeoCoding Service)
- ກາຣດຶງຂໍ້ມູນ POI (Point of Interest) ອີ້່ຂໍ້ມູນສະຖານທີ່ຕ່າງ ๆ ທີ່ Google ລວບຮົມໄວ້ໃຫ້ ເຊັ່ນ ໂຮງແຮມ ຫ້າງສຽບສິນຄ້າ ໂຮງເຮັນ -ສະຖານທີ່ຮາຊການຕ່າງໆ ແລະອື່ນໆ ອົກມາກມາຍ (Places API) ມາໃຊ້ງານໃນ Application
- Street View ເປັນໂປຣແກຣມເສັງໝົດທີ່ໃຫ້ມູນຄອງກາພແບບເສົ່າມືອນຈິງ ຈາກຕໍາແໜ່ງຕ່າງໆ ຕາມຄົນໜ່າຍແທ່ງບົນໂລກ ໂປຣແກຣມນີ້ໄດ້ມີກາເປີດຕ້ວຍຢ່າງເປັນທາງກາຣວັນທີ 25 ພຸດັງການຄ.ສ. 2007 (ພ.ສ. 2550) ໃນຫລາຍເນື້ອຂອງປະເທດສຫະລຸອມເຣິກາ ແລະຂໍາຍາດຕ້ວສໍາຮັບເມື່ອງຕ່າງໆ ແລະໜັບທຳວ່າໂລກ



ກາພທີ່ 2.1 ໜ້າຕ່າງ Google Map
ທີ່ມາ: <https://tips.thaiware.com/1433.html>

2.4 การวิเคราะห์และออกแบบระบบเชิงวัตถุ

การวิเคราะห์และออกแบบระบบเชิงวัตถุ เป็นวิธีการที่ได้รับความนิยม โดยการดูระบบจากมุมมองของตัว Object เอง เพราะ Object หน้าที่ปฏิบัติงานและเป็นตัวโต้ตอบหรือปฏิสัมพันธ์กับระบบ โดยผลลัพธ์สุดท้ายของการวิเคราะห์เชิงวัตถุ คือ การจำลองแบบเชิงวัตถุ (Object Model) ซึ่งจะเป็นตัวแทนของระบบสารสนเทศใน ความหมายของ Object และแนวความคิดเชิงวัตถุ ซึ่งเมื่อถึงระยะของการทำให้เกิดผลใน วงจรการ พัฒนาระบบ นักวิเคราะห์ระบบและนักเขียนโปรแกรมจะทำการแปลง Object ให้เป็น ส่วน จำเพาะ ของรหัสชุดคำสั่ง ซึ่งการใช้วิธีการแยกเป็นส่วนจำเพาะหรือ Modular จะช่วยประหยัดเงินและเวลา เนื่องจากสามารถถูกใช้อย่างเต็มที่ สามารถถูกตรวจสอบ และสามารถนำเอกสารลับมาใช้ใหม่ได้อีก

2.4.1 ข้อดีของ Object-Oriented (OO)

- 1) ลดความซับซ้อนของการพัฒนาระบบ และยังทำให้การสร้างและการดูแลเป็นไปได้ง่าย และรวดเร็ว
- 2) พัฒนาความสามารถในการสร้าง และคุณภาพของโปรแกรมเมอร์ เนื่องจากมีการ วางแผนร่างการนำมาใช้งาน และมีการทดสอบ สามารถที่จะนำระบบนี้ไปใช้กับระบบอื่นๆได้อีก
- 3) ระบบที่มีการพัฒนาด้วย Object-Oriented (OO) จะมีความยืดหยุ่น สามารถแก้ไข และเพิ่มเติมได้
- 4) Object-Oriented (OO) จะถูกนักวิเคราะห์ระบบมองในแง่ของระบบในโลกของ ความ เป็นจริง ไม่ใช่แค่เพียงระดับของโปรแกรมทางภาษา (Programming Language) คือสามารถหาทาง แก้ไข ปัญหาที่เกิดขึ้นได้อย่างทันที

2.4.2 ยูเอ็มแอล (UML) ย่อมาจาก Unified Modeling Language เป็นภาษาที่ใช้อธิบายแบบจำลอง ต่างๆ หรือเป็นภาษาสัญลักษณ์รูปภาพมาตรฐาน สำหรับใช้ในการสร้าง แบบจำลองเชิงวัตถุโดย ยูเอ็มแอล เป็นภาษามาตรฐานสำหรับสร้างแบบพิมพ์เขียวให้แก่ ระบบงาน สามารถใช้ยูเอ็มแอลในการสร้างมุมมอง กำหนดรายละเอียด สร้างระบบงานและ จัดทำเอกสารอ้างอิงให้แก่ระบบงานได้ เนื่องจากยูเอ็มแอล เป็นภาษา ที่มีการใช้สัญลักษณ์ รูปภาพ จึงอาจมีสับสนว่า ยูเอ็มแอล เป็นการสร้างแผนภาพหรือเป็นเพียงการใช้ สัญลักษณ์ เพื่ออธิบายระบบงานเท่านั้น แต่แท้จริงแล้ว ยูเอ็มแอลมีลักษณะของแบบจำลองข้อมูล คือ เป็น แบบจำลองที่เอาไว้อธิบายแบบจำลองอื่น ๆ อีกที การใช้งานภาษา yUML นอกจากจะต้องเข้าใจใน แนวความคิดเชิงวัตถุแล้ว ยัง จำเป็นต้องมีพื้นฐานความเข้าใจเกี่ยวกับแบบจำลองภาพด้วยเช่นกัน แบบจำลอง (Modeling) เป็นวิธีการวิเคราะห์ออกแบบ (Analysis and Design) อย่างหนึ่งที่เน้นการใช้ งานแบบจำลอง เป็นหลัก ซึ่งแบบจำลองที่สร้างขึ้นมาจะสามารถช่วยให้เข้าใจในปัญหาได้ง่าย ขึ้น อีกทั้งยังสามารถนำ แบบจำลองมาเป็นเครื่องมือในการสื่อสารถ่ายทอดความคิดกับบุคคลอื่นๆ ที่เกี่ยวข้องในโครงการได้ เช่น ลูกค้า นักวิเคราะห์ระบบ นักออกแบบระบบ เป็นต้น ส่วนแบบจำลองภาพ คือการใช้สัญลักษณ์รูปภาพในการสร้าง แบบจำลองของระบบที่จะ พัฒนาเพื่อประโยชน์ที่คล้ายคลึงกันในการทำความเข้าใจกับความต้องการของ ลูกค้า การออกแบบระบบที่เป็นไปได้อย่างชัดเจนขึ้นและการบำรุงรักษาที่ง่ายยิ่งขึ้นแบบจำลอง เกิดขึ้นโดยการ นำเสนอส่วนต่างๆ ของระบบแต่เพียงส่วนที่สำคัญโดยไม่คำนึงถึงรายละเอียดปลีกย่อยต่างๆ ใน การพัฒนาระบบซอฟต์แวร์ที่ซับซ้อน นักพัฒนาจำเป็นต้องทำความเข้าใจ กับมุมมองด้านต่างๆ ของระบบก่อนทำการ พัฒนาจริง โดยการสร้างแบบจำลองอัน เปรียบเสมือนพิมพ์เขียวที่แสดงถึงภาพรวมทั้งหมดของระบบ แบบจำลองที่สร้างขึ้นจะต้องมี ความสอดคล้องกับความต้องการของผู้ใช้งานระบบเป็นสำคัญ ในส่วนของ รายละเอียดต่าง ๆ จะค่อย ๆ ถูกเพิ่มเติมลงไปในตัวแบบจำลองและในที่สุดแบบจำลองจะถูกนำไปพัฒนาขึ้น

เป็นระบบจริง สรุปเป้าหมายของ UML นั้นสามารถกำหนดให้เป็นกลไกการสร้างแบบจำลองอย่าง ง่ายเพื่อ จำลองระบบการปฏิบัติที่เป็นไปได้ทั้งหมดในสภาพแวดล้อมที่ซับซ้อน 2.2.2 แผนผังกรณี (Use Case Diagram) Use Case Diagram คือ แผนภาพที่แสดงการทำงานของผู้ใช้ระบบ (User) และ ความสัมพันธ์กับ ระบบย่อย (Sub systems) ภายในระบบใหญ่ ในการเขียน Use Case Diagram ผู้ใช้ระบบ (User) จะถูก กำหนดว่าให้เป็น Actor และ ระบบย่อย (Sub systems) คือ Use Case จุดประสงค์หลักของการเขียน Use Case Diagram ก็เพื่อเล่าเรื่องราวทั้งหมด ของระบบว่ามีการทำงานอะไรบ้าง เป็นการตึง Requirement หรือ เรื่องราวต่างๆ ของระบบจากผู้ใช้งาน ซึ่งถือว่าเป็นจุดเริ่มต้นในการวิเคราะห์และออกแบบระบบ สัญลักษณ์ที่ ใช้ใน Use Case Diagram จะใช้สัญลักษณ์รูปคนแทน Actor ใช้สัญลักษณ์วงรีแทน Use Case และใช้เส้นตรง ในการเชื่อม Actor กับ Use Case เพื่อแสดงการใช้งานของ Use Case ของ Actor นอกจากนั้น Use Case ทุกๆ ตัวจะต้องอยู่ภายใต้เหลี่ยมเดียวกันซึ่งมีชื่อของระบบระบุอยู่ด้วย

2.4.3 Use case Diagram คือแผนภาพที่ใช้แสดงปฏิสัมพันธ์ระหว่างระบบงานและสิ่งที่อยู่นอก ระบบงาน และแสดงให้เห็นถึงส่วนประกอบทั้งหมด หรือ ภาพรวมของระบบ เป็นฐานในการเริ่มต้นการ วิเคราะห์ระบบ โดยค้นหาคำว่าระบบทำอะไร โดยไม่สนใจกลไกการทำงานหรือเทคนิคการทำงาน เปรียบเสมือน "กล่องดำ" โดย Use Case Diagram จะช่วยให้ผู้พัฒนาระบบสามารถแยกแยะกิจกรรมที่ อาจจะเกิดขึ้นในระบบ เป็น Diagram พื้นฐาน ที่สามารถอธิบายสิ่งต่าง ๆ ได้โดยใช้รูปภาพที่ไม่ซับซ้อน ประโยชน์ของ Use Case Diagram

- ช่วยให้ผู้พัฒนาระบบสามารถแยกแยะกิจกรรมที่อาจจะเกิดขึ้นในระบบ
- เป็น Diagram พื้นฐาน ที่สามารถอธิบายสิ่งต่าง ๆ ได้โดยใช้รูปภาพที่ไม่ซับซ้อน
- Use Case Diagram จะมีประสิทธิภาพหากผู้เขียนมีความเข้าใจใน Problem Domain อย่างแท้จริง

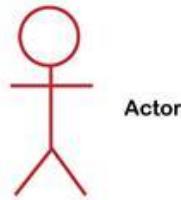
ส่วนประกอบของ Use Case Diagram ประกอบด้วย

- 1) Actor คือผู้ที่กระทำการบนระบบ อาจเป็นผู้ที่ทำการส่งข้อมูล, รับข้อมูล หรือ แลกเปลี่ยน ข้อมูลกับระบบนั้น ๆ เช่นลูกค้ากับระบบสั่งซื้อสินค้าทาง โทรศัพท์
 - Use Case คือ หน้าที่หรืองานต่าง ๆ ในระบบ เช่น การเข้าสู่ระบบ การสั่งซื้อ สินค้า เป็นต้น

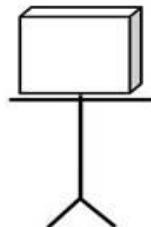
- Relationship คือความสัมพันธ์ระหว่าง Use Case กับ Actor
- System & Use Case Diagram ในระบบใหญ่มักแบ่งระบบออกเป็นระบบย่อย เรียกว่า Subsystem
- ใน Use Case Diagram จะใช้ Use Case แทน Subsystem ผู้ใช้งานระบบ ย่อยจะเรียกว่า User
- ใช้ Use case Diagram จะใช้ Actor แทน User สัญลักษณ์ที่ใช้ดังภาพที่ 2.2



ภาพที่ 2.2 สัญลักษณ์ของ Use Case Diagram
ที่มา: <https://devjourneys.com/2020/08/24/use-case-model>



ภาพที่ 2.3 สัญลักษณ์ของ Actor
ที่มา: <https://devjourneys.com/2020/08/24/use-case-model>



ภาพที่ 2.4 สัญลักษณ์ของ Actor System
ที่มา: <https://devjourneys.com/2020/08/24/use-case-model>

2) ความสัมพันธ์ของ Use Case มี 2 แบบคือ

2.1) ความสัมพันธ์แบบ include คือ ความสัมพันธ์ที่ Use Case หนึ่งต้องพึ่งพาความสามารถหรือฟังก์ชันอื่นๆ ของ Use Case อื่น กล่าวได้ว่าการทำงานของ Use Case หนึ่งเป็นส่วนหนึ่งของการทำงานของอีก Use Case หนึ่ง โดยเรียก Use Case ที่เป็นส่วนหนึ่งของ Use Case อื่นว่า Base Use Case และ Use Case ที่ถูกดึงมาใช้ว่า Include Use Case การวัดความสัมพันธ์ คือ ลากเส้นประจาก Base Use Case ทันทุกครั้งไปที่ Include Use Case และเขียนชื่อความสัมพันธ์ไว้ตรงกลาง



ภาพที่ 2.5 สัญลักษณ์ <<include>>
ที่มา: <https://devjourneys.com/2020/08/24/use-case-model>

2.2) ความสัมพันธ์แบบ Extends หมายถึงการที่ Use case หนึ่งไปมีผลต่อการทำงานตามปกติของอีก Use case หนึ่ง Use Case ที่มา Extends นั้นจะมีผลให้การดำเนินงานของ Use Case ถูกรบกวนหรือมีการสะกดดูด หรือมีการเปลี่ยนแปลงกิจกรรมไป สัญลักษณ์แทน Extends ดังภาพที่ 2.6 เส้นประพร้อมหัวลูกศร ซึ่งไปยัง Use Case ที่ถูก Extends มีคำว่า <>Extends>> กำกับอยู่บนเส้นหา Use Case และ actor ของระบบ



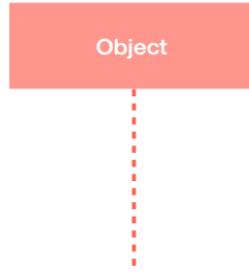
ภาพที่ 2.6 สัญลักษณ์แทน Extends

ที่มา: <https://devjourneys.com/2020/08/24/use-case-model>

2.4.4 Sequence Diagram เป็นแผนผังการทำงานแบบลำดับปฏิสัมพันธ์ คือ แผนภาพที่แสดงรายละเอียดความสัมพันธ์ของการทำงาน หรือการทำงานของระบบโดย Sequence Diagram จะ포กสืบที่เวลาและลำดับการโต้ตอบกันระหว่าง Object (วัตถุ) โดยแผนภาพจะอธิบายว่าลำดับเหตุการณ์นั้นเกิดขึ้นได้อย่างไรและเกิดขึ้นเมื่อไหร่ มีลำดับการทำงานอย่างไร ความสัมพันธ์ระหว่าง Use Case Diagram และ Sequence Diagram โดย Use Case Diagram จะมี Actor ที่มีปฏิสัมพันธ์กับแต่ละ Use Case โดยในแต่ละ Use Case ก็จะเป็นตัวแทนของการกิจกรรม task ในระบบที่ Actor ต้องทำให้สำเร็จจะสร้าง Sequence Diagram สำหรับแต่ละ Use Case โดยแต่ละ Sequence Diagram จะระบุขั้นตอนหลักสำหรับโต้ตอบเพื่อทำการกิจกรรมนั้นให้สำเร็จ

1) Notations ใน Sequence Diagram

- Lifeline Notation คือเส้นชีวิตของวัตถุหรือ class เป็นตัวแทนของวัตถุหรือส่วนประกอบต่างๆ ที่มีปฏิสัมพันธ์ซึ่งกันและกันในระบบในลำดับต่างๆ Format การเขียนชื่อเส้นชีวิตคือ Instance Name: Class Name ดังภาพที่ 2.6



ภาพที่ 2.7 สัญลักษณ์แทน Object

ที่มา: <https://devjourneys.com/2020/09/13/sequence-diagram/>

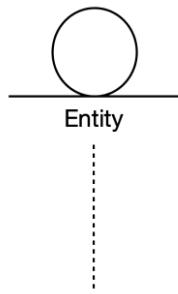
- เส้นชีวิตกับ Actor เส้นชีวิตกับสัญลักษณ์ Actor จะใช้เมื่อลำดับได้ลำดับหนึ่ง โดยเฉพาะนั้นเป็นส่วนหนึ่งของ Use Case ดังภาพที่ 2.7



ภาพที่ 2.8 สัญลักษณ์แทน Actor

ที่มา: <https://devjourneys.com/2020/09/13/sequence-diagram/>

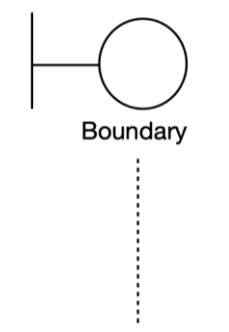
- เส้นชีวิตกับ Entity เส้นชีวิตที่มีองค์ประกอบเป็น Entity แสดงถึงข้อมูลระบบตัวอย่างเช่น แอพพลิเคชันสำหรับ customer service เอนทิตี้ลูกค้าจะจัดการข้อมูลทั้งหมดที่เกี่ยวข้องกับลูกค้า ดังภาพที่ 2.7



ภาพที่ 2.9 สัญลักษณ์แทน Entity

ที่มา: <https://devjourneys.com/2020/09/13/sequence-diagram/>

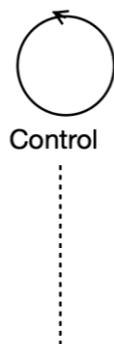
- เส้นชีวิตกับ Boundary element เส้นชีวิตที่มีองค์ประกอบเป็น Boundary element แสดงขอบเขตของระบบหรือ software element ในระบบ เช่น หน้าจอสำหรับผู้ใช้งาน, data gateways หรือ เมนูที่ผู้ใช้งานสามารถตอบโต้ได้



ภาพที่ 2.10 สัญลักษณ์ Boundary element

ที่มา: <https://devjourneys.com/2020/09/13/sequence-diagram/>

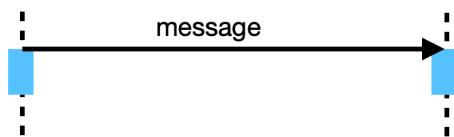
- เส้นชีวิตกับ control element เส้นชีวิตที่มีองค์ประกอบเป็น control element แสดงเออนทิตี้การควบคุมหรือผู้จัดการ มันจะจัดระเบียบและกำหนดเวลาการตอบโต้ และทำหน้าที่เป็นเป็นสื่อกลางระหว่างขอบเขตและเออนทิตี้



ภาพที่ 2.11 สัญลักษณ์แทน เส้นชีวิตกับ control element

ที่มา: <https://devjourneys.com/2020/09/13/sequence-diagram/>

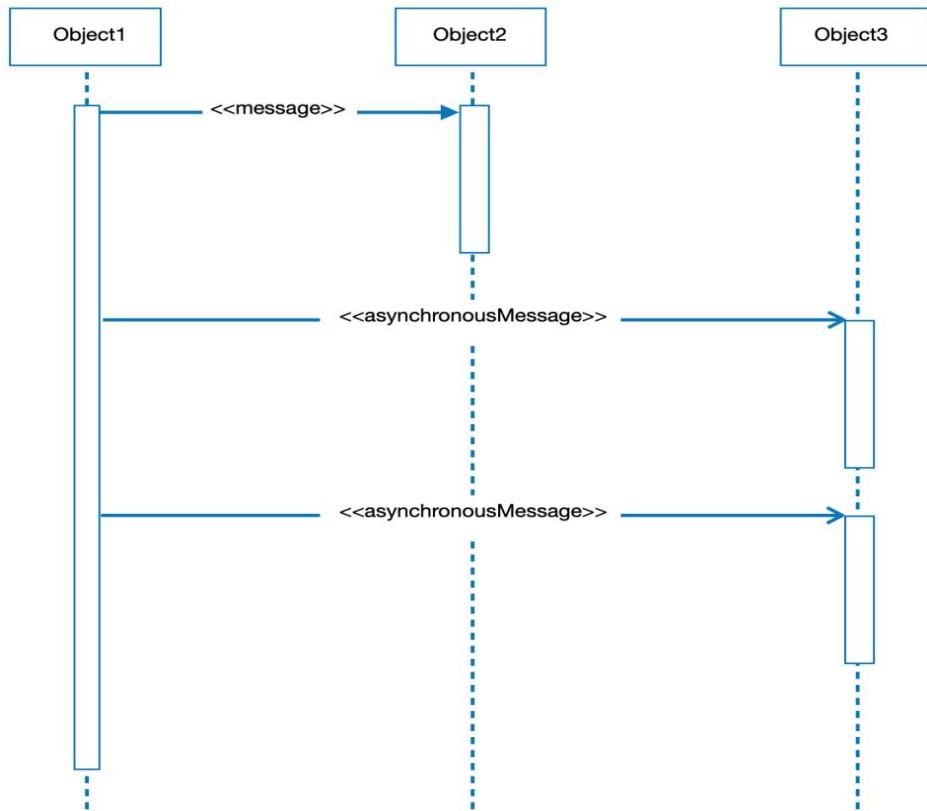
- Activation Bars จะถูกวางอยู่บนเส้นชีวิตเพื่อแสดงการโต้ตอบระหว่างวัตถุ ฟังก์ชัน หรือ module ความยาวของสี่เหลี่ยมจะแสดงระยะเวลาการโต้ตอบของวัตถุ หรือ จุดเริ่มต้นและจุดสิ้นสุดของแต่ละกิจกรรมของวัตถุนั้น
 - การโต้ตอบระหว่าง 2 วัตถุ เกิดเมื่อวัตถุหนึ่งส่งข้อความไปให้วัตถุหนึ่ง วัตถุที่ส่งข้อความเรียกว่า Message Caller ในขณะที่วัตถุที่รับข้อความเรียกว่า Message Receiver เมื่อมีแบบ Activation บนเส้นชีวิตของวัตถุ นั่นหมายความวัตถุนั้นมีการทำงานในขณะที่ส่งข้อความโต้ตอบกัน
 -



ภาพที่ 2.12 สัญลักษณ์แทน Message

ที่มา: <https://devjourneys.com/2020/09/13/sequence-diagram/>

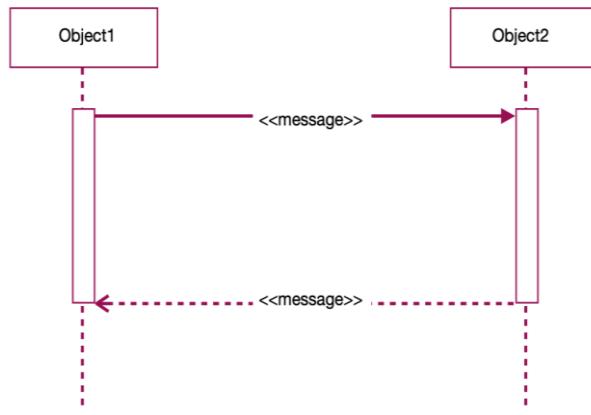
- Message Arrows ลูกศรจาก Message Caller ชี้ไปที่ Message Receiver ระบุข้อความที่เกิดขึ้นใน Sequence Model ข้อความสามารถไหลไปในทิศทางใดก็ได้ จากซ้ายไปขวา ขวาไปซ้าย หรือส่งข้อความกลับไปที่ตัวมันเองก็ได้ การอธิบายทิศทางการไหลของข้อความได้ด้วยหัวลูกศรที่ชี้ไปในทิศทางที่ต้องการ และพิจารณาว่าวัตถุใดเป็นตัวส่ง วัตถุใดเป็นตัวรับข้อความ
 - รูปแบบของ Message
 - Synchronous message จะถูกใช้เมื่อวัตถุที่ส่งข้อความรอให้วัตถุที่รับปั๊บ ความรวมผลและส่ง return กลับมาก่อนที่จะส่งข้อความอันต่อไป หัวลูกศรที่ใช้จะเป็นลูกศรแบบทึบ
 - Asynchronous message จะถูกใช้เมื่อวัตถุที่ส่งข้อความไม่รอให้วัตถุรับ ข้อความประมวลผลข้อความและส่งค่า return กลับมา แต่จะส่งข้อความต่อไปให้แก่วัตถุอื่นในระบบโดยหัวลูกศรที่แสดงในข้อความประเภทนี้เป็นหัวลูกศรสีน้ำเงิน



ภาพที่ 2.14 Asynchronous message

ที่มา: <https://devjourneys.com/2020/09/13/sequence-diagram/>

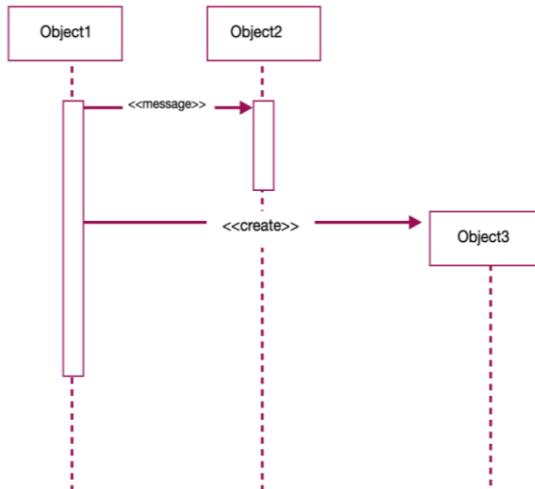
- return message ใช้เพื่อบ่งบอกว่าตัวถุรับข้อความประมวลผลข้อความเสร็จสิ้นแล้ว และกำลังส่งคืนการควบคุมไปยังวัตถุที่ทำหน้าที่ส่งข้อความ return message เป็นตัวเลือกที่จะเลือกให้มีหรือไม่มีก็ได้ สำหรับการส่งข้อความบนแบบ Activation ด้วย Synchronous message จะให้ความหมายโดยนัยว่ามี return message ด้วยแม้จะไม่ได้มีสែន return message แสดงก็ตาม สามารถหลีกเลี่ยงการทำให้แผนภาพดูยุ่งเหยิงโดยการไม่ใช้ return message เมื่อไม่จำเป็น



ภาพที่ 2.15 return message

ที่มา: <https://devjourneys.com/2020/09/13/sequence-diagram/>

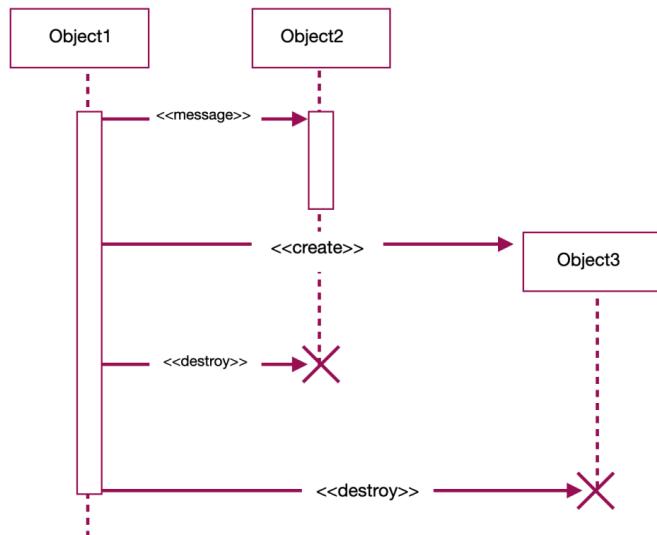
- Participant creation message วัตถุไม่จำเป็นต้องปรากฏอยู่ทุกช่วงเวลาในแผนภาพ วัตถุหรือผู้มีส่วนร่วมสามารถถูกสร้างขึ้นได้เมื่อข้อความถูกส่งออกไปหาวัตถุนั้น เครื่องหมายสำหรับวัตถุใหม่ที่ถูกสร้างคือกล่องสี่เหลี่ยมผืนผ้าที่มีชื่อของวัตถุอยู่ด้านใน เมื่อวัตถุใหม่ถูกสร้างขึ้น จะแสดงให้เห็นว่าวัตถุนั้นไม่ได้มีตัวตนมาก่อนจนกระทั่งมีการส่งข้อความจากวัตถุอื่นที่ปรากฏอยู่แล้วในแผนภาพ เมื่อสร้างวัตถุใหม่ขึ้นมาแล้ว หากวัตถุนั้นมีการทำงานใดหนึ่งที่ถูกสร้าง ก็จะจะใส่แบบ Activation ด้านล่างวัตถุนั้นด้วย ดังภาพที่ 2.16



ภาพที่ 2.16 Participant creation message

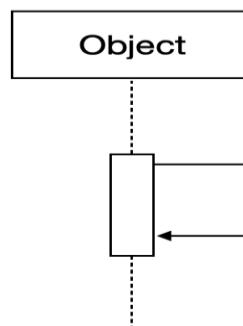
ที่มา: <https://devjourneys.com/2020/09/13/sequence-diagram/>

- Participant destruction message เมื่อวัตถุนั้นดำเนินงานมาจนถึงลำดับที่ไม่จำเป็นอีกต่อไป สามารถลบวัตถุนั้นออกจากแผนภาพได้ วิธีทำคือ การเพิ่มเครื่องหมาย X ที่ปลายเส้นชีวิตของวัตถุนั้น ดังภาพที่ 2.17



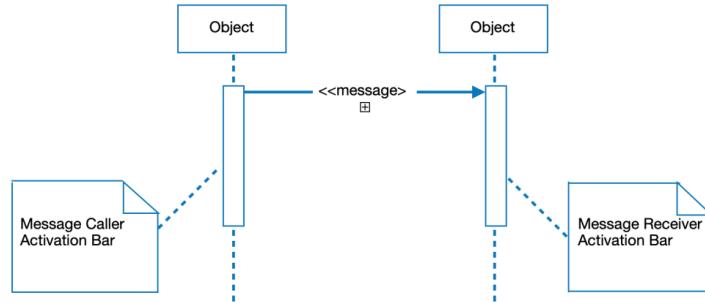
ภาพที่ 2.17 สัญลักษณ์ Participant destruction message
ที่มา: <https://devjourneys.com/2020/09/13/sequence-diagram/>

- Reflexive message เมื่อวัตถุส่งข้อความหาตัวเอง จะเรียกว่า reflexive message และงข้อความประเภทนี้โดยการใช้ message arrow ที่เริ่มจากจบที่เส้นชีวิตเดียวกันอย่างตัวอย่างด้านล่างนี้



ภาพที่ 2.18 Reflexive message
ที่มา: <https://devjourneys.com/2020/09/13/sequence-diagram/>

- Comment ในแผนภาพ UML ปกติแล้วจะอนุญาตให้มีเครื่องหมาย comment เพื่อแสดงความคิดเห็นได้ การแสดงความคิดเห็นจะอยู่ในกล่องสี่เหลี่ยมพื้นผ้าที่พับมุมบนด้านหนึ่ง ความคิดเห็นสามารถถูกเชื่อมโยงกับวัตถุที่เกี่ยวข้องด้วยเส้นประได้ ดังภาพที่ 2.19



ภาพที่ 2.19 Comment

ที่มา: <https://devjourneys.com/2020/09/13/sequence-diagram/>

- Sequence Fragment คือกล่องที่มีเครื่องหมายแสดง section การโต้ตอบระหว่างวัตถุ sequence diagram Fragment จะใช้ในแผนภาพที่มีการโต้ตอบของวัตถุที่ซับซ้อน เช่น การให้แบบ alternative หรือ loop ที่มีวิธีที่เป็นโครงสร้างมากขึ้น ด้านบนซ้ายของกล่องจะแสดง fragment operator ระบุว่า fragment นี้เป็น fragment ประเภทใด

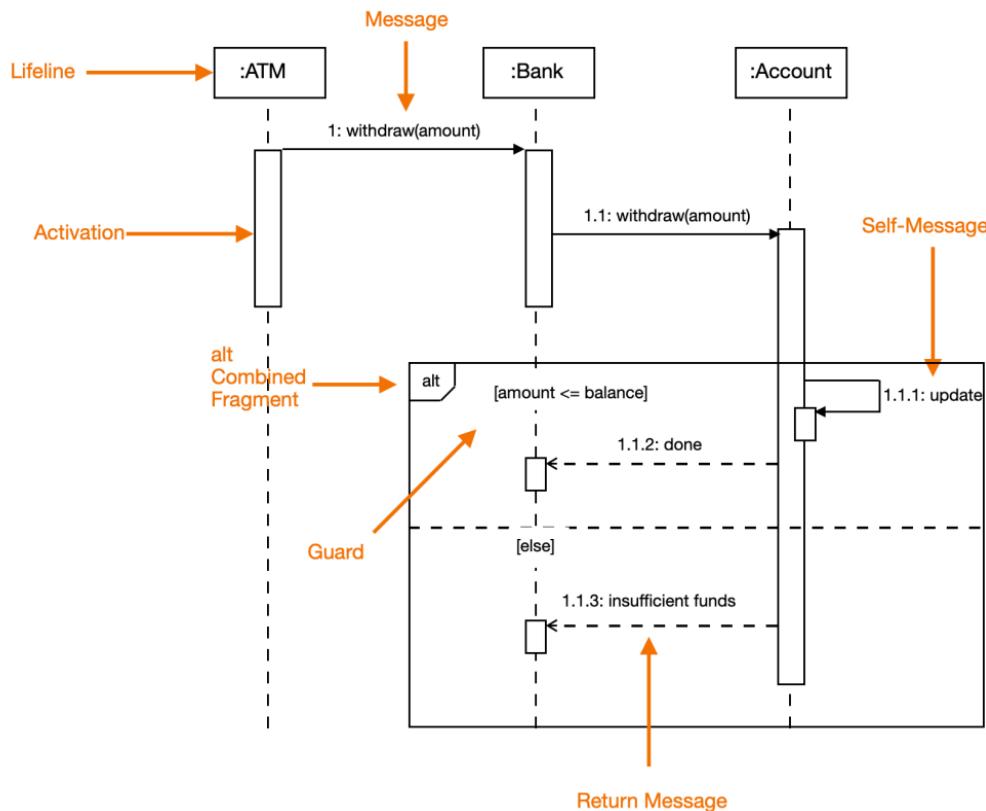
- Alternatives

- Alternative combination fragment ถูกใช้เมื่อมีตัวเลือกให้เลือกตั้งแต่ 2

ตัวเลือกขึ้นไป ใช้ตรรกะแบบ “if then else”

- Alternative fragment แสดงโดยเฟรมหรือกล่องสี่เหลี่ยมพื้นผ้าขนาดใหญ่ มี

‘alt’ ซึ่งเป็น fragment operator ระบุเป็นชื่อกล่องในมุมซ้ายด้านบนของกล่อง ในการเลือกตัวเลือกที่มากกว่าหนึ่ง ข้างในกล่องสี่เหลี่ยมพื้นผ้าจะถูกแบ่งออกเป็นส่วนๆ เรียกว่า interaction operand โดยใช้เส้นประเป็นตัวแบ่ง แต่ละ operand จะมี Guard ระบุไว้ที่ด้านบนของแต่ละ operand ดังภาพที่ 2.20



ภาพที่ 2.20 Alternatives

ที่มา: <https://devjourneys.com/2020/09/13/sequence-diagram/>

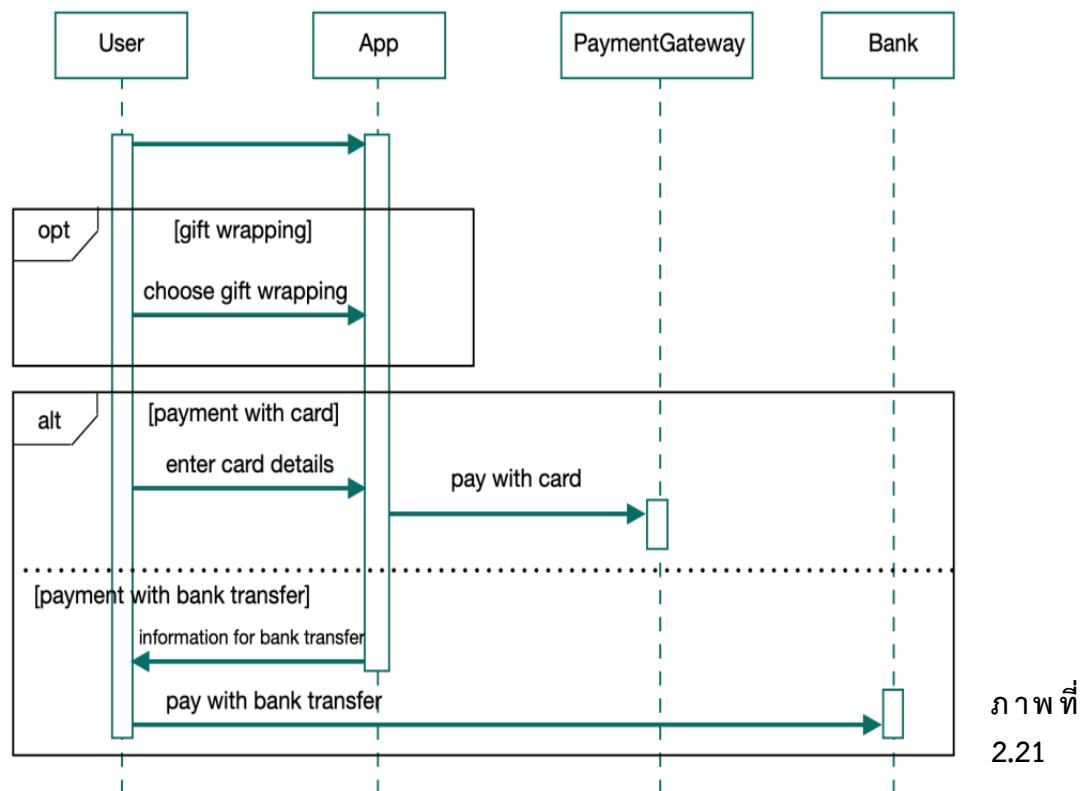
- Options

Option Combination fragment ถูกใช้เพื่อแสดงถึงลำดับเหตุการณ์ที่เกิดขึ้นภายใต้เงื่อนไขเดียวกัน ไม่ เช่นนั้นเหตุการณ์นั้นจะไม่สามารถเกิดขึ้นได้ ใช้ตรรกะแบบ ‘if then’ Option fragment สามารถแสดงได้โดยกล่องสีเหลืองผืนผ้าเช่นเดียวกับ Alternative fragment ส่วน fragment operator จะใช้ ‘opt’ ความแตกต่างระหว่าง Option fragment และ Alternative fragment คือ Option fragment จะไม่แบ่งออกเป็นแต่ละ Operand ส่วน Option Guard จะวางอยู่ด้านบนซ้ายในกล่อง ความแตกต่างระหว่าง alt และ opt fragment ใน Sequence diagram

alt ถูกใช้เพื่ออธิบายทางเลือกที่สามารถเลือกได้ในการให้ผลของเหตุการณ์ ทางเลือกเดียวเท่านั้นที่จะถูก executed

opt ถูกใช้เพื่อแสดงถึงขั้นตอนทางเลือกในการให้ผลของเหตุการณ์ หรือ workflow

เช่น ในร้านค้าออนไลน์ อาจใช้ opt เพื่อกำหนดว่าลูกค้าสามารถเลือกที่จะห่อของขวัญได้ หากต้องการและใช้ alt เพื่อกำหนดว่ามีวิธีจ่ายเงิน 2 วิธี คือ จ่ายผ่านบัตรเครดิตหรือโอนเงินผ่านธนาคารดังภาพที่ 2.21



Options

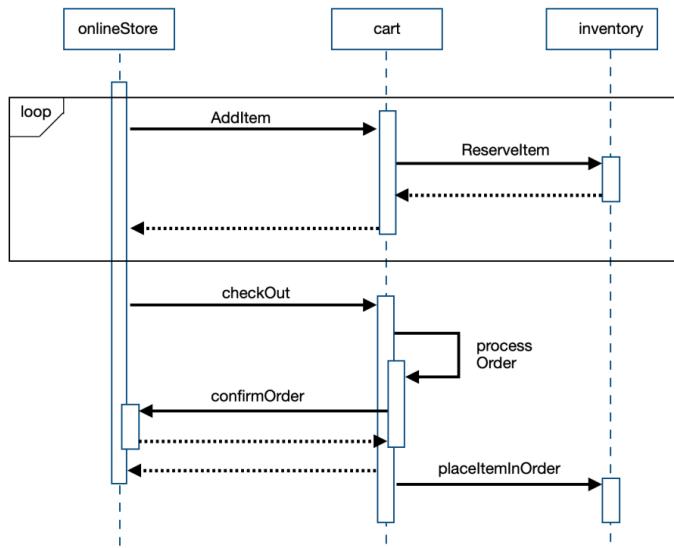
ที่มา: <https://devjourneys.com/2020/09/13/sequence-diagram/>

- Loops

Loop fragment ถูกใช้เพื่อแสดงลำดับเหตุการณ์ที่เกิดขึ้นซ้ำ โดยมี ‘loop’ เป็น fragment operation และ guard condition ระบุที่มุ่งด้านซ้ายของกล่อง guard ใน loop fragment สามารถมีเงื่อนไขพิเศษเพิ่มได้อีกสองเงื่อนไข คือ minimum iterations (minint = [the number])

maximum iterations (maxint = [the number])

หากใช้เงื่อนไข minimum iterations guard ลูปต้องแสดงผลไม่น้อยกว่าจำนวนที่ระบุใน minimum iteration แต่หากใช้เงื่อนไข maximum iteration guard ลูปก็ต้องแสดงผลไม่เกินจำนวนที่ระบุใน maximum iteration เช่นกัน loop fragment จะทำงานช้ากว่าค่าของ guard จะเป็น false ดังภาพที่ 2.22

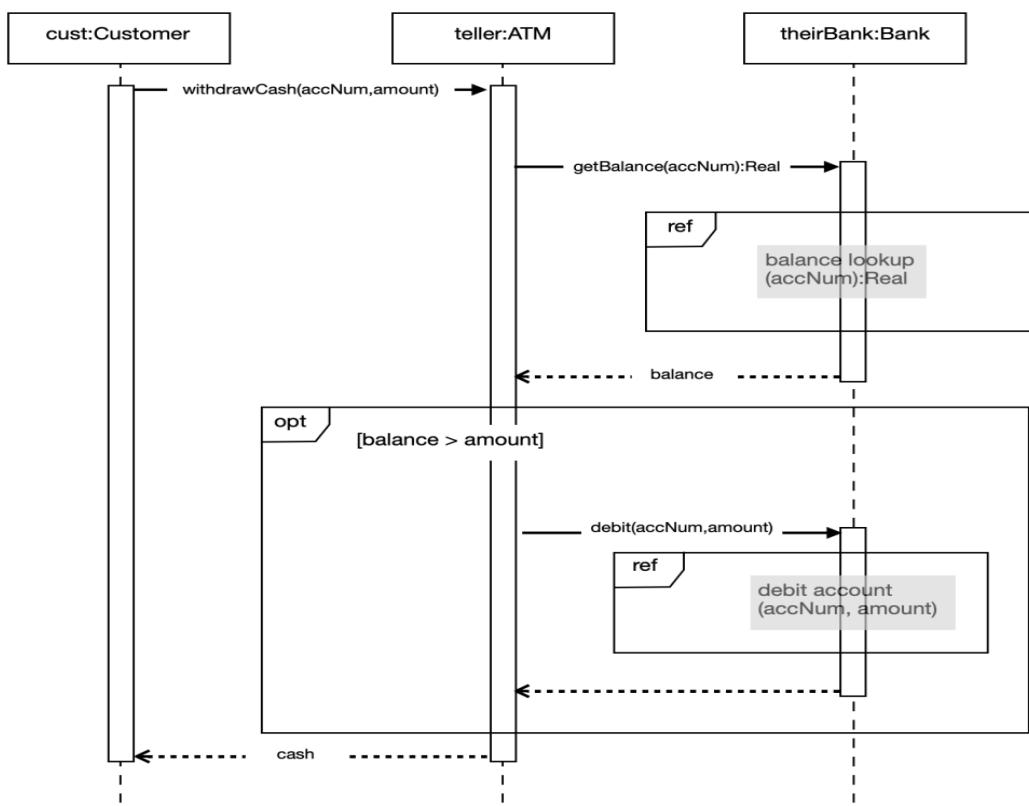


ภาพที่ 2.22 Loops

ที่มา: <https://devjourneys.com/2020/09/13/sequence-diagram/>

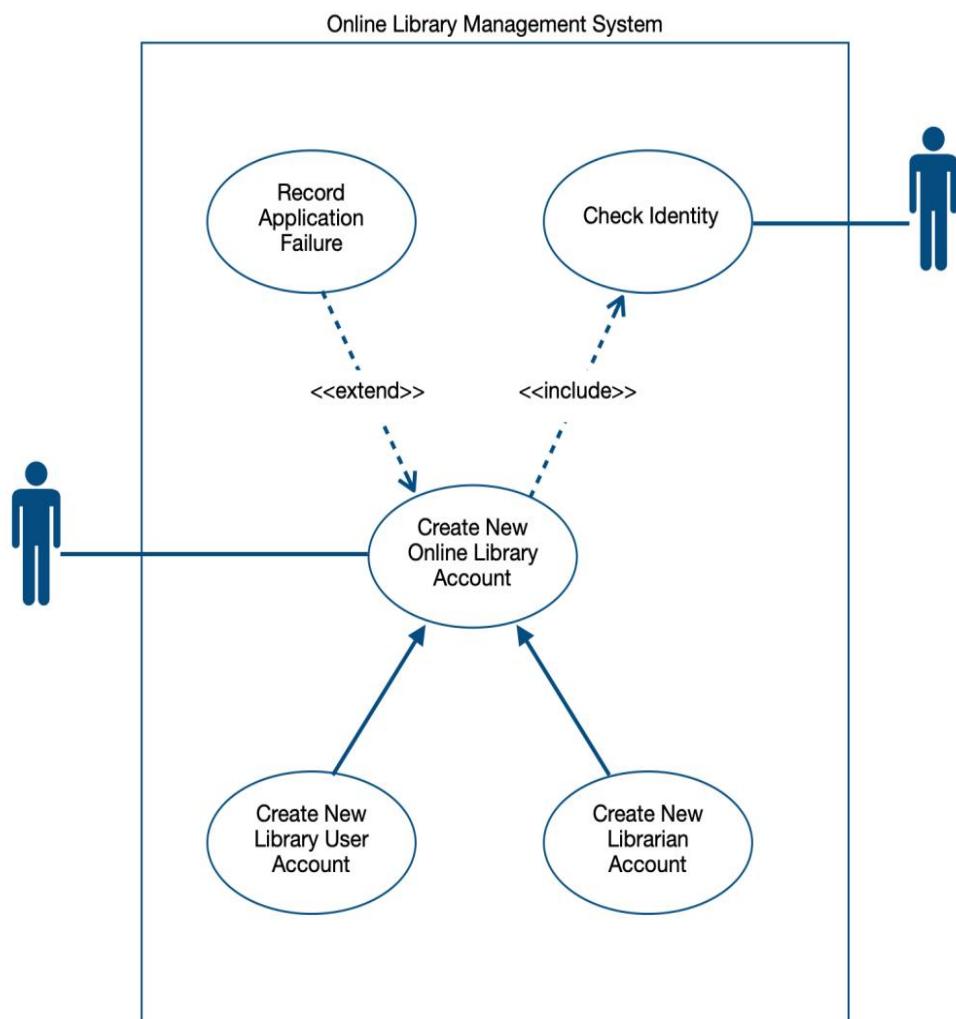
-Reference fragment

สามารถใช้ Reference fragment เพื่อจัดการกับขนาดของ sequence diagram ที่มีขนาดใหญ่ ref fragment จะช่วยให้ทำงานส่วนของ sequence diagram มาใช้กับส่วนอื่นๆได้ หรือ เรียกว่า สามารถอ้างถึงส่วนอื่นๆของ sequence diagram ได้โดยใช้ ref fragment การแสดงถึง ref fragment จะต้องกำหนด ‘ref’ เป็นชื่อกล่อง หรือ fragment operation และชื่อของ sequence diagram ที่ถูกอ้างถึงข้างในกล่อง ดังภาพที่ 2.23



ภาพที่ 2.23 Reference fragment
ที่มา: <https://devjourneys.com/2020/09/13/sequence-diagram/>

วิธีวิเคราะห์ Sequence Diagram เนื่องจาก sequence diagram แสดงลำดับการให้ผลของเหตุการณ์ในหนึ่ง Use Case การให้ผลไปของข้อความหรือ Message ใน Sequence Diagram จะเป็นไปตามทิศทางของ Use Case นั้นๆ ดังนั้นก่อนวิเคราะห์ Sequence Diagram จะต้องวิเคราะห์ Use Case ก่อน หรือ ต้องมีการตัดสินใจก่อนว่า การโต้ตอบหรือเหตุการณ์ใดบ้างที่จะระบุถึง ดังในตัวอย่างภาพที่ 2.24 Use Case ของ Online Library Management System เลือกว่าจะวิเคราะห์ Sequence Diagram ที่ขั้นตอนย่อยได้ ในตัวอย่างจะเลือกขั้นตอน ‘Create New User Account’ ของ ‘Create New Online Library Account’



ภาพที่ 2.24 ตัวอย่าง Online Library Management system
ที่มา: <https://devjourneys.com/2020/09/13/sequence-diagram/>

ก่อนที่จะวาดไดอะแกรม สิ่งสำคัญคือต้องระบุวัตถุ (Object) หรือ Actor ที่มีเกี่ยวข้องสำหรับการสร้าง new user account ซึ่งจะลิสต์ออกมาได้ดังนี้

- Librarian
- Online Library Management system
- User credential database
- Email system

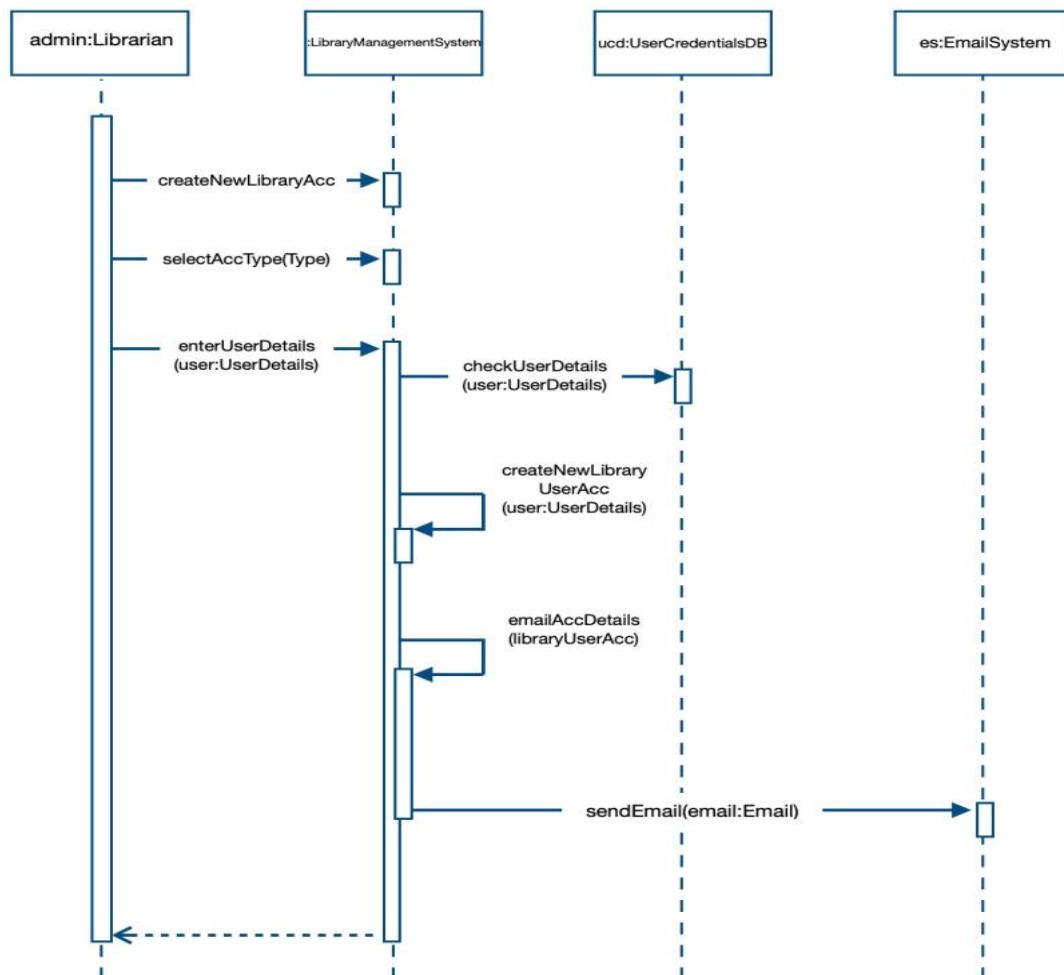
เมื่อระบุวัตถุหรือ Actor ที่เกี่ยวข้องได้แล้ว สิ่งต่อมาคือต้องเขียนรายละเอียดเชิงรายว่า Use Case นี้ มีกระบวนการใดบ้าง หากสามารถอธิบายได้ ก็จะสามารถทราบได้ว่าแต่ละส่วนมีการตอบโต้ หรือ Interaction อย่างไร จากนั้นการตอบโต้เหล่านี้จะถูกจับเอาไปใส่ไว้ใน Sequence Diagram ตัวอย่างเหตุการณ์ที่เกิดขึ้นใน Use Case ‘Create New Library User Account’

- Librarian ร้องขอระบบให้สร้าง online library account ใหม่
- Librarian เลือกประเภทของ user account
- Librarian ใส่รายละเอียดของผู้ใช้งาน
- รายละเอียดของผู้ใช้งานจะถูกตรวจสอบโดย User Credential Database

(หรือฐานข้อมูลนั้นแหล่ง)

- บัญชีผู้ใช้ใหม่ถูกสร้าง
- รวมข้อมูลทั้งหมดสำหรับบัญชีใหม่ และส่งอีเมลให้เจ้าของบัญชี

แต่ละลำดับเหตุการณ์ดังกล่าว ทำให้สามารถระบุได้ว่าข้อความใดจะเป็นข้อความที่ใช้ในการโต้ตอบระหว่างวัตถุใน Sequence Diagram



ภาพที่ 2.25 ตัวอย่าง Sequence Diagram
 ที่มา: <https://devjourneys.com/2020/09/13/sequence-diagram/>

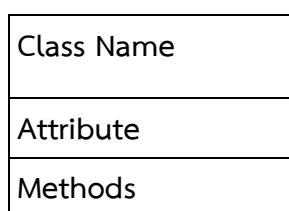
2.4.5 Class Diagram คือแผนภาพที่ใช้แสดง Class และความสัมพันธ์ในแต่ต่างๆ (Relation) ระหว่าง Class เหล่านั้น ซึ่งความสัมพันธ์ที่กล่าวถึงใน Class Diagram นี้ถือเป็นความสัมพันธ์เชิงสถิตย์ (Static Relationship) หมายถึงความสัมพันธ์ที่มิอยู่แล้วเป็นปกติในระหว่าง Class ต่างๆ ไม่ใช่ความสัมพันธ์ที่เกิดขึ้นเนื่องจากกิจกรรมต่างๆ ซึ่งเรียกว่าความสัมพันธ์เชิงกิจกรรม (Dynamic Relationship) เส้นที่ปรากฏใน Class Diagram นั้นประกอบด้วยกลุ่มของ Class และกลุ่มของ Relationship โดยสัญลักษณ์ที่ใช้ในการแสดง Class นั้นจะแทนด้วยสีเหลี่ยมแบ่งออกเป็น 3 ส่วน โดยแต่ละส่วนนั้น (จากบนลงล่าง) จะใช้ในการแสดง ชื่อของ Class, Attribute และฟังก์ชันต่างๆตามลำดับ

1) ความสัมพันธ์ระหว่าง Class

ความสัมพันธ์ระหว่าง Class (Relationship) คือ ความสัมพันธ์ระหว่าง Class ที่ทำงานร่วมกัน สามารถจำแนกได้ดังนี้

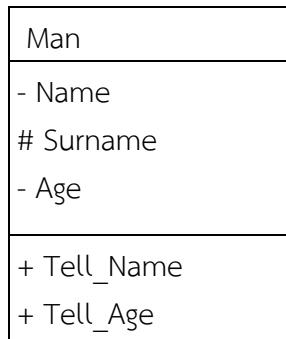
- ความสัมพันธ์แบบพึ่งพา (Dependency) เช่น “Class ลูกค้า” กับ “Class ขายสินค้า” กล่าวได้ว่า “Class ขายสินค้า” ขึ้นอยู่กับ “Class ลูกค้า” เพราะเมื่อลูกค้ามีการเปลี่ยนแปลงคำสั่งซื้อ หรือคำสั่งผลิตรายการขายก็จะต้องถูกเปลี่ยนแปลง (Update) ตามลูกค้า
- ความสัมพันธ์แบบสืบทอดคุณสมบัติ (Inheritance) เช่น “Class แม่” (super class) สืบทอดคุณลักษณะเฉพาะที่ตนมีอยู่ไปยัง “Class ลูก” (sub class)
- ความสัมพันธ์แบบร่วมกัน (Association) คือความสัมพันธ์ที่เกี่ยวเนื่องมีความสัมพันธ์ซึ้งกันและกัน เช่น “Class นักเรียน” สัมพันธ์กับ “Class รายวิชา” ในเรื่องของการลงทะเบียนเรียน

2) องค์ประกอบของ Class diagram



ภาพที่ 2.26 องค์ประกอบของ Class diagram
 ที่มา: <https://bloger-classdiagram.blogspot.com/p/class-diagram.html>

3) สัญลักษณ์ของ Class diagram



ภาพที่ 2.27 สัญลักษณ์ของ Class diagram

ที่มา: <https://bloger-classdiagram.blogspot.com/p/class-diagram.html>

ในการเขียนสัญลักษณ์แทน Class สิ่งที่ต้องคำนึงถึงอีกสิ่งหนึ่งคือระดับการเข้าถึงเรียก สัญลักษณ์ที่ใช้แทนการเข้าถึงนี้ว่า Visibility แบ่งออกได้เป็น 3 ประเภท ได้แก่

3.1 Private เขียนแทนด้วยสัญลักษณ์ - หมายถึง Attribute หรือ พังก์ชันที่ไม่สามารถมองเห็นได้จากภายนอก แต่สามารถมองเห็นได้จากภายในตัวของ Class เองเท่านั้น

3.2 Protect เขียนแทนด้วยสัญลักษณ์ #หมายถึง Attribute หรือพังก์ชันที่ส่วนใหญ่สำหรับการทำ Inheritance โดยเฉพาะ Attribute หรือพังก์ชันเหล่านี้จะเป็นของ Superclass เมื่อทำการ Inheritance แล้ว Attribute หรือพังก์ชันที่มี Visibility แบบ Protect จะกลายไปเป็นPrivate Attribute/ พังก์ชันหรือ Protected ขึ้นอยู่กับภาษา Programming ที่นำไปใช้

3.3 Public เขียนแทนด้วยสัญลักษณ์ + หมายถึง Attribute หรือ พังก์ชัน ที่สามารถมองเห็นได้จากภายนอก และสามารถเข้าไปเปลี่ยนค่า อ่านค่าหรือเรียกใช้งาน Attribute หรือ พังก์ชัน นั้นได้ทันทีโดยอิสระจากภายนอก (โดยทั่วไปแล้ว Visibility แบบ Public นักจะใช้กับพังก์ชันมากกว่า Attribute)

4) ความสัมพันธ์ระหว่าง Object ประกอบด้วย

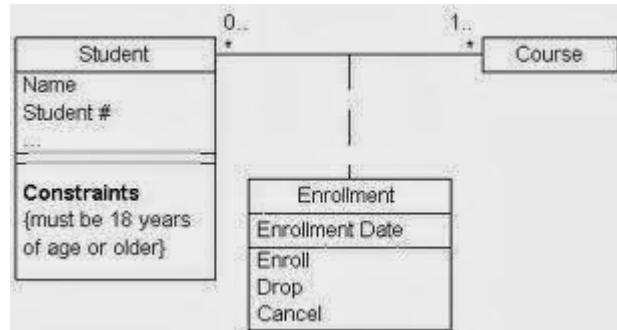
4.1 Association

4.2 Aggregation

4.3 Composition

4.4 Generalization

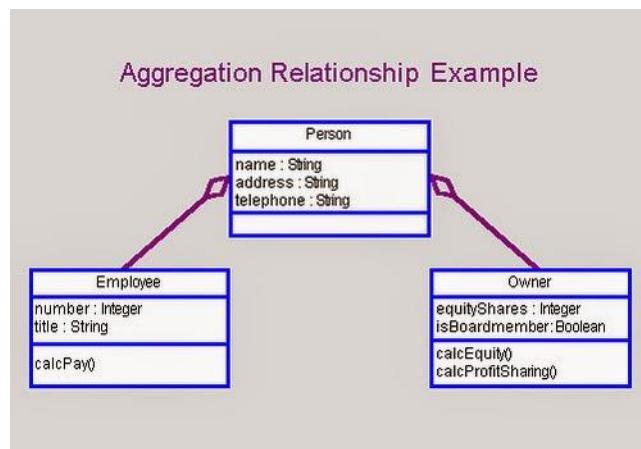
4.1 Association เป็นความสัมพันธ์ระหว่าง Object หรือ Class แบบ 2 ทิศทาง



ภาพที่ 2.28 Aggregation

ที่มา: <https://bloger-classdiagram.blogspot.com/p/class-diagram.html>

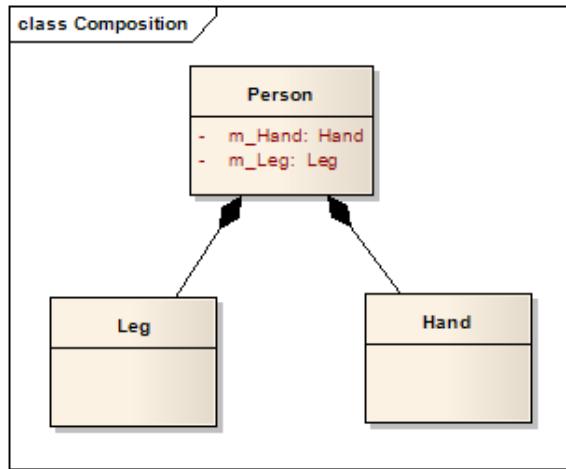
4.2 Aggregation เป็นความสัมพันธ์ระหว่าง Object หรือ Class แบบ “Whole-Part” หรือ “is part of” โดยจะมี Class ที่ใหญ่ที่สุดที่เป็น Object หลัก และมี Class อื่นเป็นส่วนประกอบ



ภาพที่ 2.29 Aggregation Relationship Example

ที่มา: <https://bloger-classdiagram.blogspot.com/p/class-diagram.html>

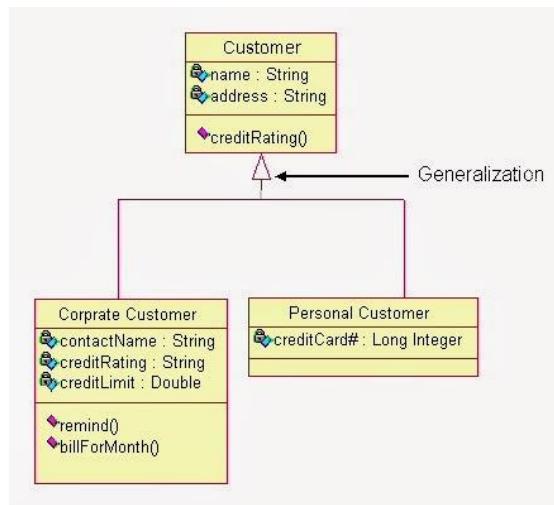
4.3 Composition เป็นความสัมพันธ์ระหว่าง Object หรือ Class แบบขึ้นต่อ กันและมีความเกี่ยวข้องกันเสมอ โดยจะมี Class ซึ่งเป็นองค์ประกอบของ Class อื่นที่ใหญ่กว่า เมื่อ Class ที่ใหญ่กว่าถูกทำลาย Class ที่เป็นองค์ประกอบก็จะถูกทำลายไปด้วย



ภาพที่ 2.30 Composition

ที่มา: <https://bloger-classdiagram.blogspot.com/p/class-diagram.html>

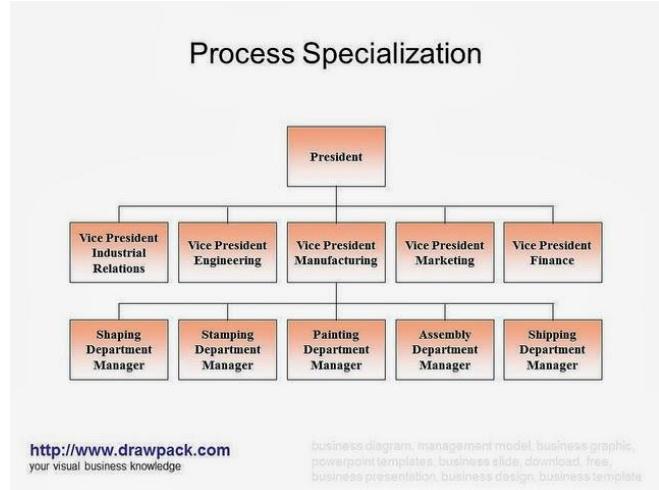
4.4 Generalization เป็นความสัมพันธ์ระหว่าง Object หรือ Class ในลักษณะของการสืบทอดคุณสมบัติจาก Class หนึ่ง (Super class) ไปยังอีก Class หนึ่ง (Subclass)



ภาพที่ 2.31 Generalization

ที่มา: <https://bloger-classdiagram.blogspot.com/p/class-diagram.html>

4.5 Specialization คือกระบวนการที่ตรงกันข้ามกับ กระบวนการ Generalization Abstraction กล่าวคือ ถ้าต้องการสร้าง Class ใหม่ โดยอาศัย Concept ของ Class เก่าบางส่วน



ภาพที่ 2.32 Specialization
ที่มา: <https://bloger-classdiagram.blogspot.com/p/class-diagram.html>

5. หลักในการสร้าง Class Diagram

สิ่งที่ต้องคำนึงถึงสำหรับการจำลอง Class และ Relationship ต่างๆ ใน

Class Diagram ใน OOA คือ

- Class ทั้งหมดที่ต้องมีอยู่ในระบบหรือใน Real World
- ต้องมีอยู่ครบ ไม่ขาดหาย
- ไม่มากเกินความจำเป็น

5.1) กำหนดกรอบของ Problem Domain ให้ชัดเจน

- ให้ยึดถือ Problem Domain นี้เป็นบรรทัดฐานในการวิเคราะห์ระบบ
- เขียน Use Case Diagram ของ Problem Domain
- พิจารณาว่า ในแต่ละ Use Case จะมี Objects อะไรอยู่บ้าง

5.2) พิจารณาหา Objects ที่สามารถจับต้องได้เห็นได้สัมผัสได้ ซึ่งเรียกว่า

Tangible Objects

5.3) พิจารณาหา Objects ที่ไม่สามารถจับต้องได้ซึ่งเรียกว่า Intangible Objects

5.4) ใช้ Classification Abstraction เพื่อแยกแยะและสร้าง Class จาก Objects ที่มีอยู่

- พยายามหา Attributes และ Functions ของ Class เท่าที่จะหาได้
- วาด Class ทั้งหมดที่ได้ ลงใน Class Diagram

5.5) หา Aggregation Abstraction (โดยพิจารณาการเป็นส่วนประกอบ)

- เพิ่มเติมสัญลักษณ์

- ใส่ Cardinality ให้ถูกต้อง

5.6) ใช้ Generalization มาพิจารณา

- เพิ่มเติมสัญลักษณ์

- อาจเกิด Class ใหม่เพื่อเป็น Generalized Class ได้

5.7) ใช้ Association มาพิจารณา

- เพิ่มเติมสัญลักษณ์

- พิจารณาประเภทของความสัมพันธ์และ Cardinality ให้ถูกต้อง

5.8) พิจารณา Class Diagram ว่ามี Class หรือ กลุ่มของ Class ที่ไม่มีความสัมพันธ์กับ Class อื่นๆ หรือไม่

- อาจจะพบ Class ที่ไม่จำเป็นสำหรับระบบ

- อาจจะขาด Class อื่นๆที่จำเป็นในระบบ

2.4.6 Activity Diagram คือแผนภาพกิจกรรมใช้อธิบายกิจกรรมที่เกิดขึ้นในลักษณะการดำเนินการ (Workflow) Activity Diagram จะมีลักษณะเดียวกับ Flowchart (แสดงขั้นตอนการทำงานของระบบ) โดยขั้นตอนในการทำงานแต่ละขั้นจะเรียกว่า Activity

1) การใช้Activity Diagram

- อธิบาย กระแสการทำงาน (Workflow)

- แสดงขั้นตอนการทำงานของระบบ

Activity อาจเป็นการทำงานต่างๆ ได้แก่

- การคำนวณผลลัพธ์บางอย่าง

- การเปลี่ยนแปลงสถานะ (State) ของระบบ

- การส่งค่ากลับคืน

- การส่งสัญญาณ

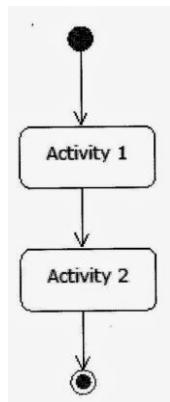
- การเรียกใช้ Operation (Method) อื่นๆ เพื่อทำงาน

- การสร้าง หรือ ทำลายวัตถุ

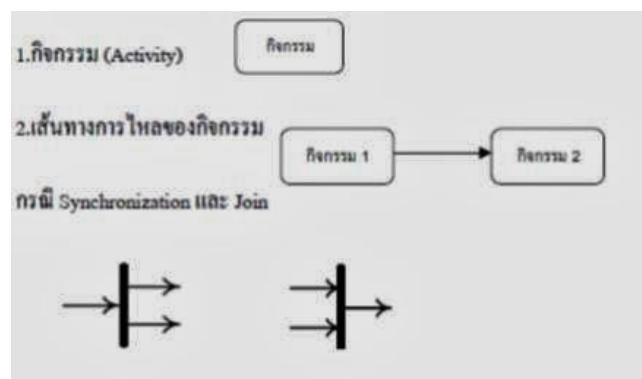
2) ลักษณะของ Activity Diagram

- Activity Diagram จะต้องมีจุดเริ่มต้นกับจุดสิ้นสุด และในระหว่างจุดเริ่มต้น กับ จุดสิ้นสุดจะมีขั้นตอนหรือ Activity ต่างๆ ของระบบ

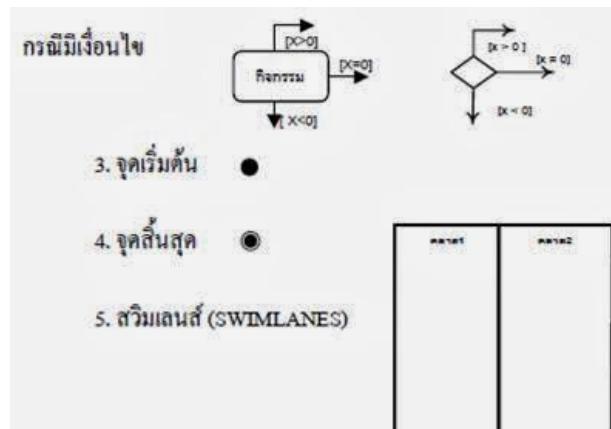
- ปกติแล้วจะเขียน Activity Diagram โดยอ่านจากด้านบนลงล่าง (ดังภาพที่ 2.33)



ภาพที่ 2.33 Activity Diagram แผนภาพแสดงการไหลของข้อมูล
ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>

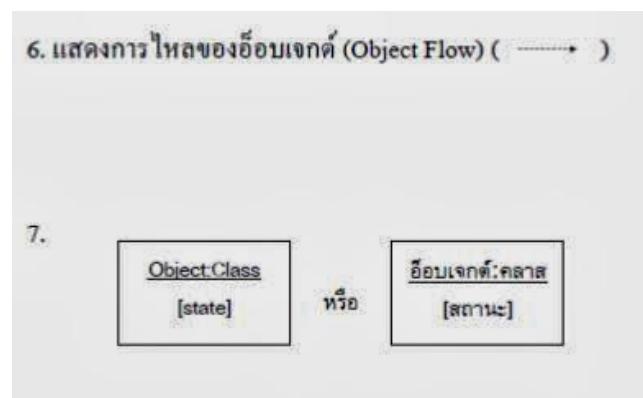


ภาพที่ 2.34 สัญลักษณ์ที่ใช้ใน Activity Diagram
ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>



ภาพที่ 2.35 สัญลักษณ์ที่ใช้ใน Activity Diagram

ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>



ภาพที่ 2.36 สัญลักษณ์ที่ใช้ใน Activity Diagram

ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>

3) ขั้นตอนในการเขียน Activity Diagram

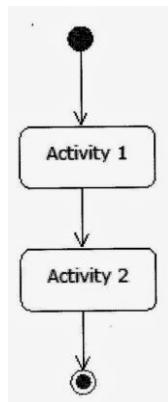
- พิจารณา กิจกรรมต่างๆ ที่ที่ได้จากผลการวิเคราะห์ที่ควรอธิบาย
- พิจารณา กิจกรรมย่อยที่เกิดขึ้น เงื่อนไขหรือกรณีต่างๆ ที่เกิดขึ้นเมื่อเป็นไปตาม

เงื่อนไข

- เรียงลำดับกิจกรรมที่เกิดก่อนหลัง
- เขียนกิจกรรมย่อยด้วยสัญลักษณ์แสดงกิจกรรม
- เขียนจุดเริ่มต้น
- เขียนจุดสิ้นสุด

4) รูปแบบการใช้ Activity Diagram

4.1. แบบทั่วไป



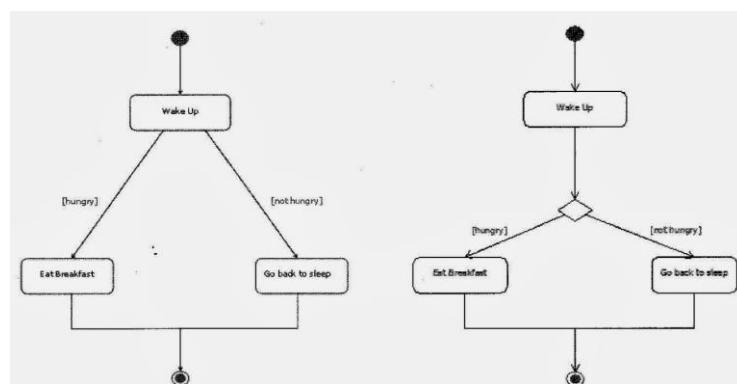
ภาพที่ 2.37 Activity Diagram แบบทั่วไป

ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>

4.2) แบบมีทางเลือกให้ตัดสินใจ การกำหนดทางเลือกให้แก่ Activity Diagram

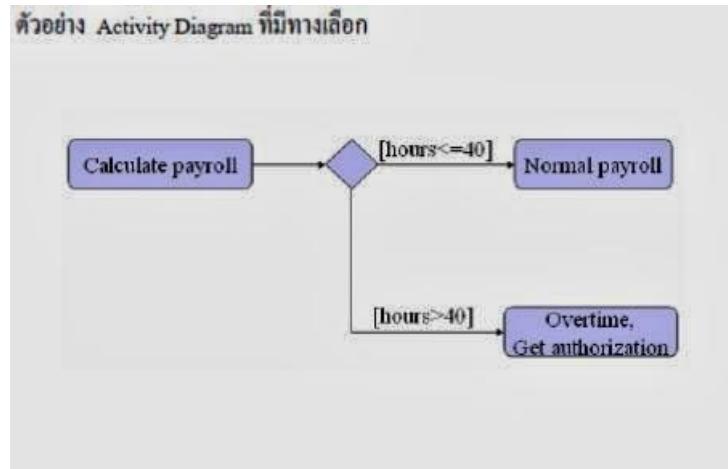
ทำได้ 2 วิธี

- ลากลูกศรของแต่ละทางเลือกไปยัง Activity ผลลัพธ์ของทางเลือกด้วยตรง
- ลากลูกศรของแต่ละทางเลือกผ่านรูปสี่เหลี่ยมขนมเปียกปูนก่อน



ภาพที่ 2.38 Activity Diagram แบบมีทางเลือกให้ตัดสินใจ

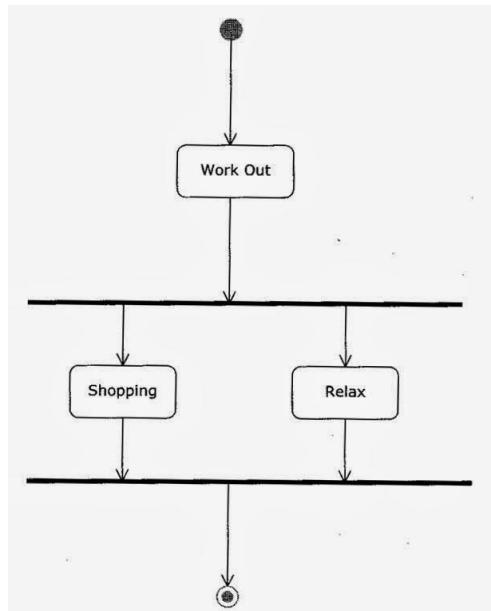
ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>



ภาพที่ 2.39 ตัวอย่างActivity Diagram

ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>

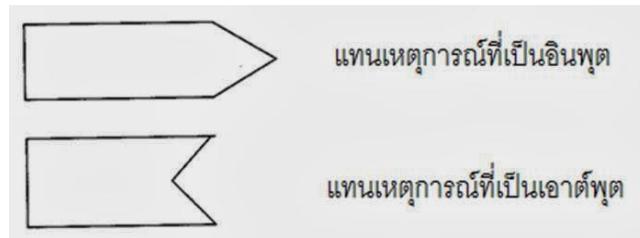
4.3) แบบมีการทำงานพร้อมๆกันหลายงาน ให้ใช้เส้นตรงแนวโน้มบนเส้นหนาที่เรียกว่าSwim Lanes มาเป็นสัญลักษณ์ที่ใช้จัดกลุ่มงานที่มีการทำงานพร้อมๆกันหรือการทำกิจกรรมในลักษณะคู่ขนาน



ภาพที่ 2.40 สัญลักษณ์ที่ใช้ใน Activity Diagram

ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>

4.4) แบบการส่งสัญญาณ ในกระบวนการทำงาน อาจเป็นไปได้ว่าจะมีการส่งสัญญาณ บางอย่างในระหว่างการทำงาน เมื่อเกิดการส่ง - รับ สัญญาณ เรียกว่าเกิด Activity ได้เช่นกัน

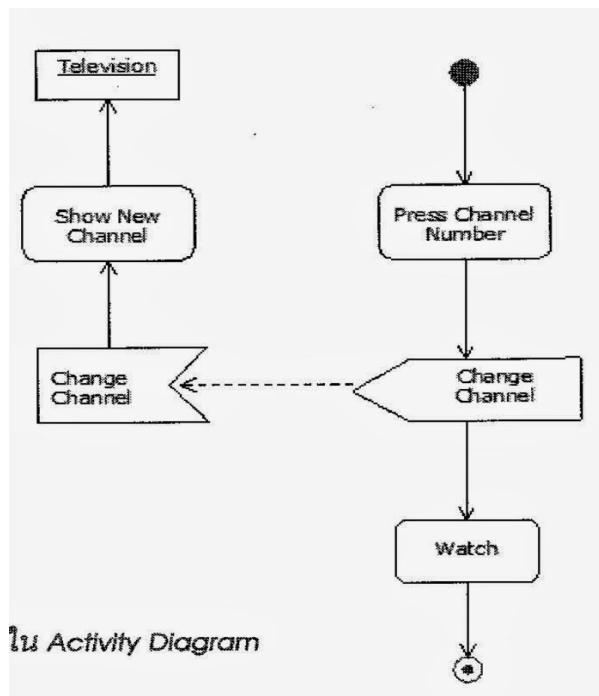


ภาพที่ 2.41 Activity Diagram แบบการส่งสัญญาณ

ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>

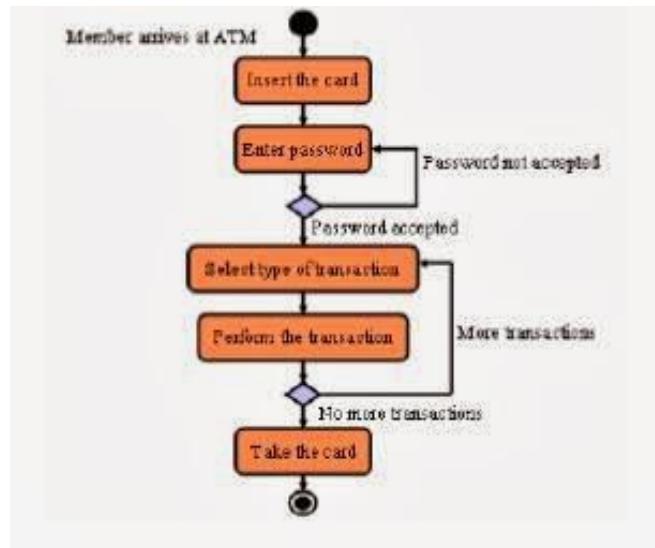
5) การใช้ Activity Diagram แสดงการส่งสัญญาณ

ตัวอย่างการใช้ Activity Diagram แสดงการส่งสัญญาณที่เป็นการแสดงความสัมพันธ์ระหว่าง Activity ทั้งสอง ภายใต้เหตุการณ์เดียวกันโดยระบบที่สนใจ คือ การกดปุ่มรีโมทคอนโทรลเพื่อเปลี่ยนช่อง โทรทัศน์



ภาพที่ 2.42 Activity Diagram แสดงการส่งสัญญาณ

ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>



ภาพที่ 2.43 ตัวอย่าง Activity Diagram ของ ATM

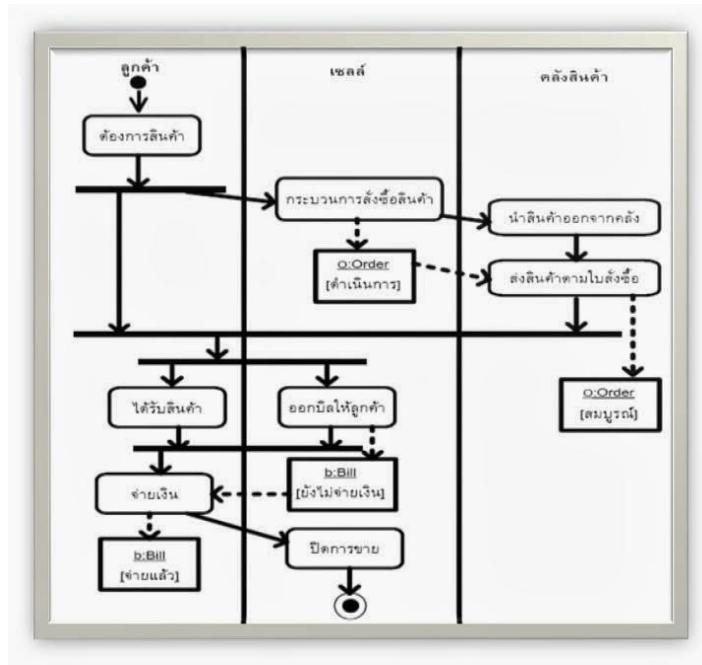
ที่มา: <https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>

โดยแบ่งการทำงานให้เป็นสัดส่วนด้วย Swim lanes

- คุณลักษณะอีกอย่างหนึ่ง คือ ความสามารถแสดงให้เห็นได้ว่าใครเป็นผู้มีหน้าที่รับผิดชอบในแต่ละ Activity ในกระบวนการทำงานหนึ่งๆ

- หลักการของการแสดงหน้าที่ จำทำโดยการแบ่งกลุ่มของการรับผิดชอบเป็นกลุ่มๆ ซึ่งเปรียบเหมือนการแบ่งว่ายน้ำ เรียกว่า泳道 (Swim lanes)

- ในแต่ละ Swim lanes จะมีการกำหนดชื่อกับเอาไว้ เช่นกระบวนการของการสั่งซื้อสินค้า อาจแบ่งกลุ่มของคนที่มีส่วนเกี่ยวข้องเป็น 3 ส่วน ได้แก่ ลูกค้า, ฝ่ายขาย, คลังสินค้า



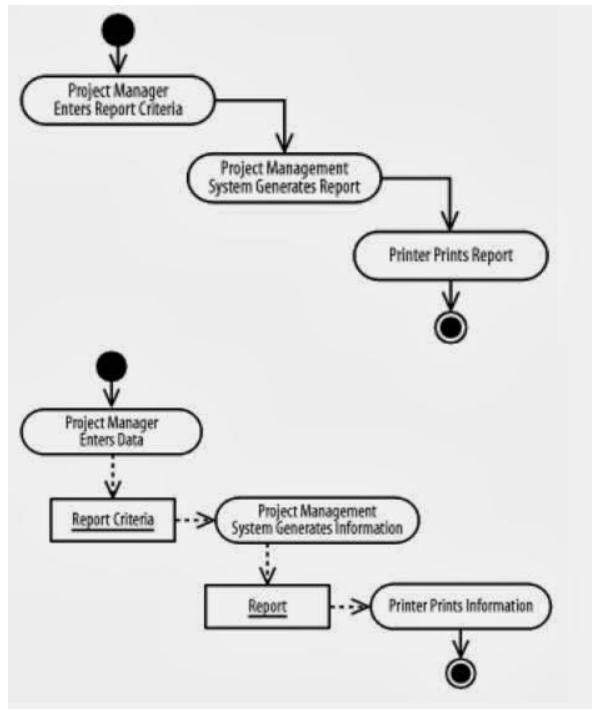
ภาพที่ 2.44 แบ่งการทำงานให้เป็นสัดส่วนด้วย Swim lanes

ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>

- Activity Diagram : Transitions

(ภาพบน) Control-flow transitions ใช้เพื่อเรียงลำดับของการเกิด Activity โดยจะเริ่มทำ Action ถัดไปก็ต่อเมื่อ Action ก่อนหน้าทำงานเสร็จเรียบร้อย

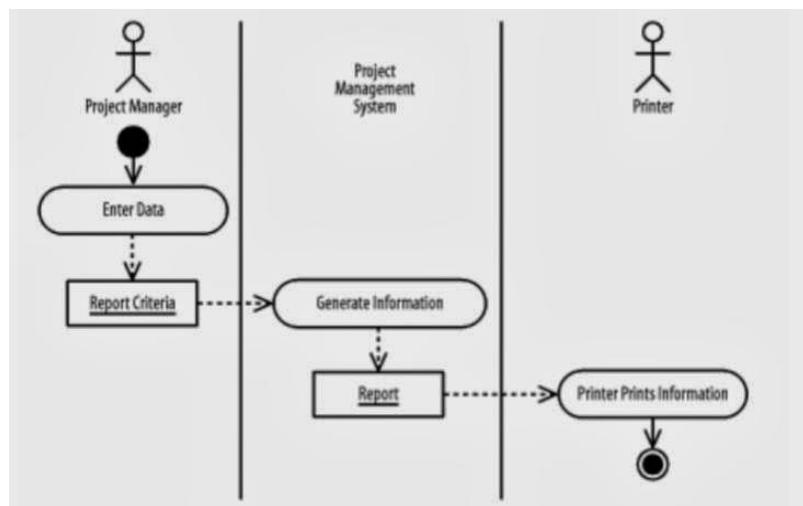
(ภาพล่าง) Object-flow transitions ใช้เพื่อระบุ Input หรือ Output ที่เกิดขึ้นจากการทำงานใน Action นั้นโดย Input /Output จะแสดงเป็น Object



ภาพที่ 2.45 Activity Diagram : Transitions

ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>

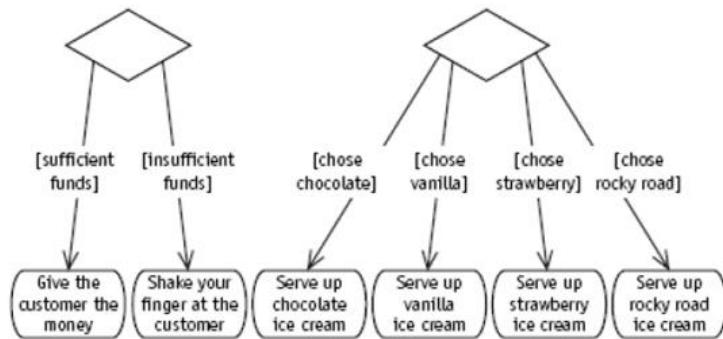
- Activity Diagram : Swim lanes กิจกรรมในการทำงาน สามารถแบ่งหน่วยงานที่รับผิดชอบได้ด้วย Swim lanes



ภาพที่ 2.46 Activity Diagram : Swim lanes

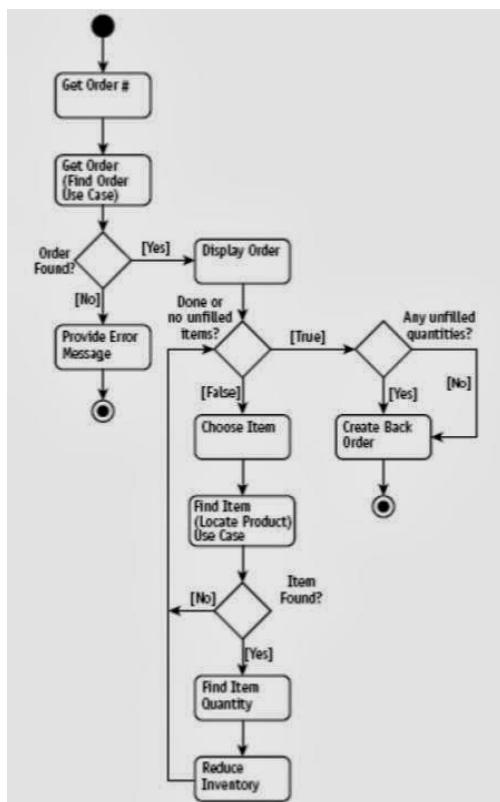
ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>

- Activity Diagram : Decision แทนด้วยสัญลักษณ์สี่เหลี่ยมข้าวหลามตัด พร้อมระบุเงื่อนไขของแต่ละกรณีเอาไว้



ภาพที่ 2.47 Activity Diagram : Decision

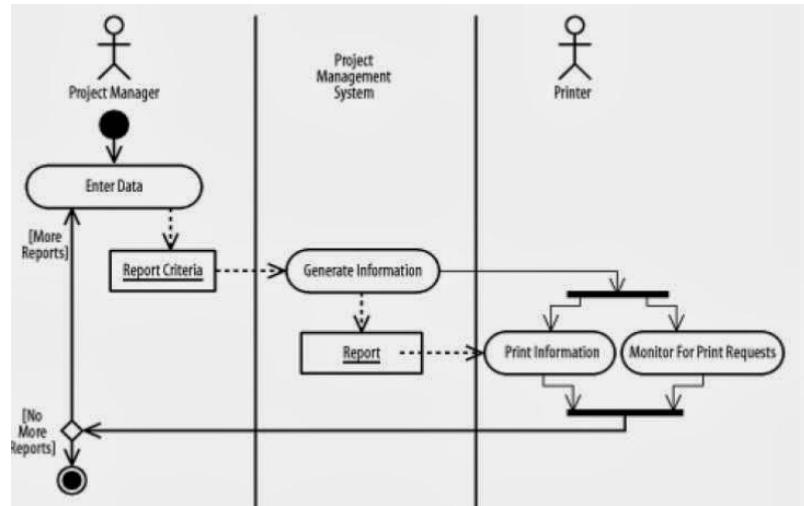
ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>



ภาพที่ 2.48 ตัวอย่างActivity Diagram : Decision

ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>

- Activity Diagram : Concurrency เป็นการแสดงการทำงานที่สามารถทำกิจกรรมได้พร้อมๆ กันได้



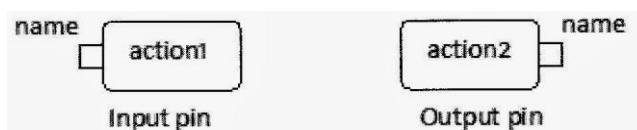
ภาพที่ 2.47 Activity Diagram : Concurrency

ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>

6) การระบุส่วนของข้อมูลให้แก่กิจกรรม

- โดยปกติกิจกรรมการทำงานมักเกี่ยวข้องกับข้อมูล เช่น การสร้าง ลบ หรือ การโยกย้าย ข้อมูลเป็นต้น

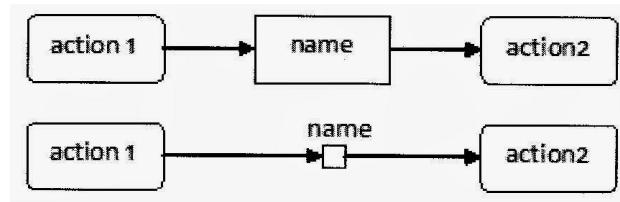
- Activity Diagram จึงมีส่วนที่เรียกว่า Input Pin และ Output Pin สำหรับการแสดงส่วนที่เป็นข้อมูล Input และ Output
 - Pin จะมีลักษณะเป็นรูปสี่เหลี่ยมเล็กๆ ที่วางไว้ก่อนหรือหลังรูปสี่เหลี่ยม มุมโค้งของ Activity เพื่อแสดง Input และ Output ของกิจกรรม



ภาพที่ 2.48 การระบุส่วนของข้อมูลให้แก่กิจกรรม

ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>

- ในกรณีที่ข้อมูล Input จากกิจกรรมหนึ่งเป็น Output ของอีกกิจกรรมหนึ่ง การแสดงความสัมพันธ์ระหว่างกิจกรรมดังกล่าว สามารถเขียนได้ 2 แบบ ดังนี้

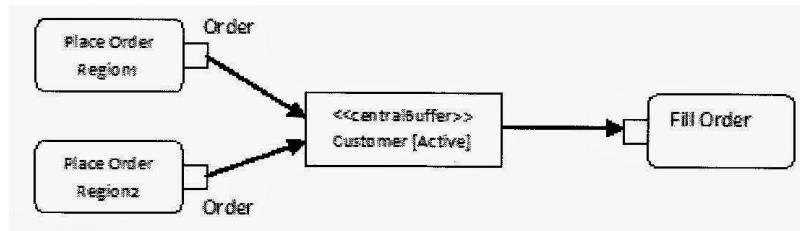


ภาพที่ 2.49 การแสดงความสัมพันธ์ระหว่างกิจกรรม 2 แบบ

ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>

7) การจัดระเบียบข้อมูล

- ในบางครั้งข้อมูลอาจมาจากต้นทางหลายแห่ง หรือมาจากต้นทางเดียวกัน แต่มีการส่งข้อมูลมาเรื่อยๆ อย่างต่อเนื่อง ดังนั้นจึงต้องจัดเรียงข้อมูลเหล่านั้นในระหว่างกระบวนการทำงานหนึ่งๆ โดยใช้สัญลักษณ์ <<centralBuffer>>

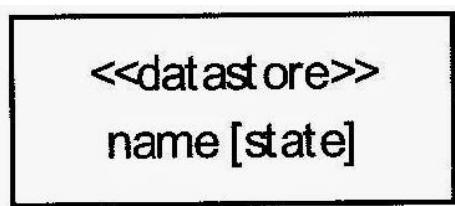


ภาพที่ 2.50 การจัดระเบียบข้อมูล

ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>

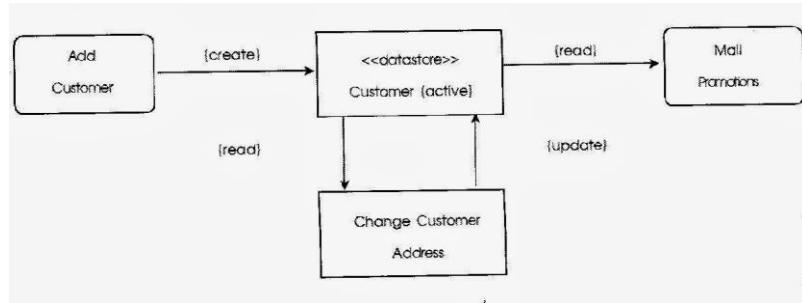
8) การสร้างที่พัก/เก็บข้อมูล

- ระหว่างกิจกรรมหนึ่งๆ อาจมีการสร้าง ลบ โยกย้ายข้อมูล รวมถึงมีการเรียกใช้ข้อมูลเพื่อ ประมวลผลบางอย่าง บางครั้งข้อมูลเหล่านี้จะเกิดขึ้นช่วงเวลาในระหว่างการทำงาน เมื่อการประมวลผลเสร็จสิ้น ข้อมูลนั้นจะหายไป หรือในบางครั้งอาจต้องเก็บข้อมูลดังกล่าวไว้เพื่อการทากิจกรรมอื่นๆ ต่อไป ไม่ว่ากรณีใดก็ตามต้องมีที่สำหรับพัก/เก็บข้อมูลนั้นเอาไว้ที่ Data Store โดยใน Activity Diagram จะใช้สัญลักษณ์ <<datastore>>



ภาพที่ 2.51 จะใช้สัญลักษณ์ <<datastore>>

ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>



ภาพที่ 2.52 ตัวอย่างการนา Data Store มาใช้งาน

ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>

8) คุณสมบัติของ Activity Diagram ที่ดี ได้แก่

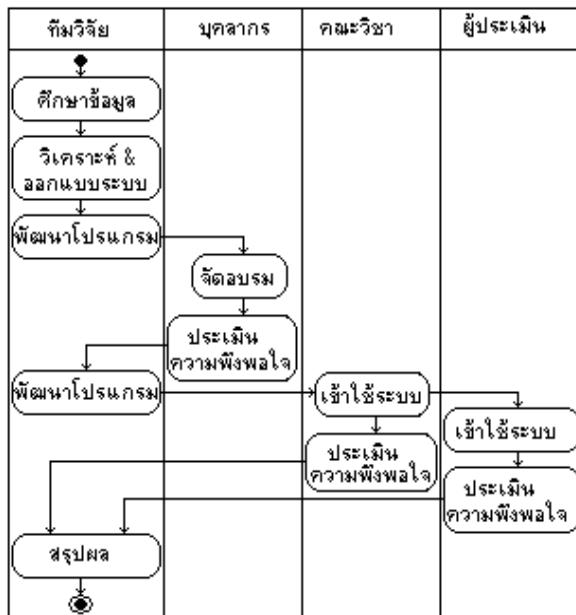
- มุ่งเน้นการติดต่อสื่อสารของระบบในเชิงไดนามิก
- เนพาะอีลิเมนต์ที่มีความสำคัญต่อกระบวนการทางานเท่านั้น
- แสดงรายละเอียดในแต่ละระดับการทำงาน โดยเลือกแสดง เนพาะที่มีความสำคัญต่อการเข้าใจการทำงานของระบบเท่านั้น

- ถ้าการทำงานส่วนใดมีความสำคัญ ก็ควรเขียน Activity Diagram ไม่ควรละเว้าไว้หรือแสดงเพียงอย่างย่อๆ

ตัวอย่าง Activity Diagram

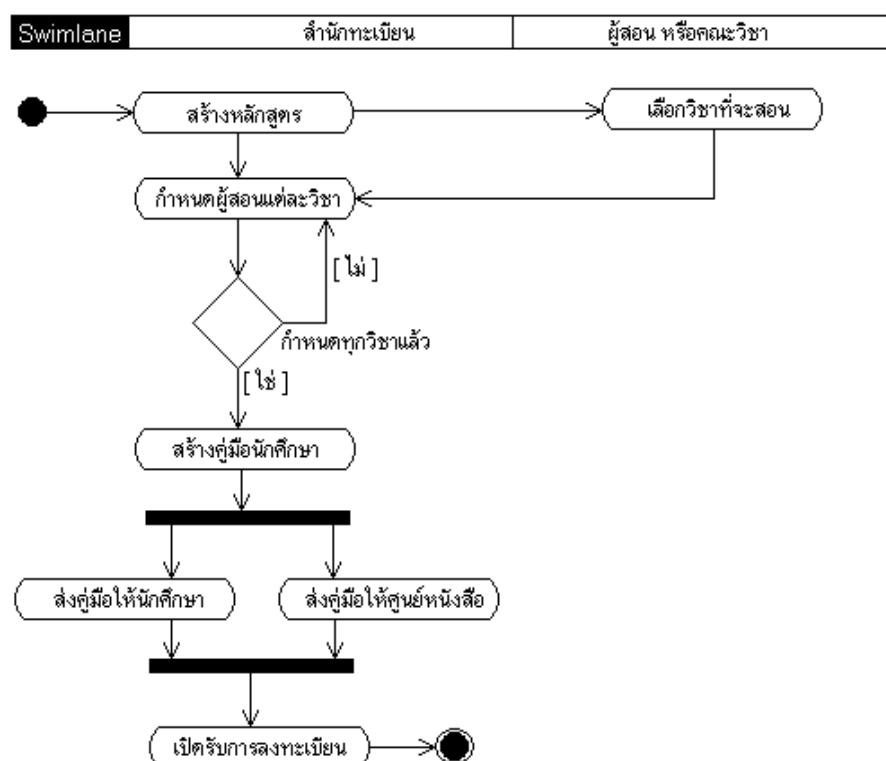
- เรื่อง Research Process (กระบวนการวิจัย)
- ตัวอย่างการลงทะเบียนเรียน

Activity Diagram : Research Process



ภาพที่ 2.53 ตัวอย่าง Research Process (กระบวนการวิจัย)

ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>



ภาพที่ 2.54 ตัวอย่างการลงทะเบียนเรียน

ที่มา:<https://infomation54.blogspot.com/2014/01/activity-diagram-workflow-activity.html>

2.5 โปรแกรมและภาษาที่ใช้พัฒนาโปรแกรม

2.5.1 Visual Studio Code

VS Code หรือ Visual Studio Code จากบริษัทไมโครซอฟต์ เป็นโปรแกรมประเภท Editor ใช้ในการแก้ไขโค้ดที่มีขนาดเล็ก แต่มีประสิทธิภาพสูง เป็น OpenSource โปรแกรมจึงสามารถนำมาใช้งานได้โดยไม่มีค่าใช้จ่าย เหมาะสำหรับนักพัฒนาโปรแกรมที่ต้องการใช้งานหลายแพลตฟอร์ม รองรับการใช้งานทั้งบน Windows , macOS และ Linux รองรับหลายภาษาทั้ง JavaScript, TypeScript และ Node.js ในตัว และสามารถเชื่อมต่อกับ Git ได้ง่าย สามารถนำมาใช้งานได้ง่ายไม่ซับซ้อน มีเครื่องมือและส่วนขยายต่าง ๆ ให้เลือกใช้มากมาย รองรับการเปิดใช้งานภาษาอื่น ๆ ทั้ง ภาษา C++ , C# , Java , Python , PHP หรือ Go สามารถปรับเปลี่ยน Themes ได้ มีส่วน Debugger และ Commands เป็นต้น ซึ่งบทความนี้จะเป็นการสอน วิธีการใช้งาน Visual Studio Code เป็นต้น มาเริ่มกันเลย

2.5.2 Flutter

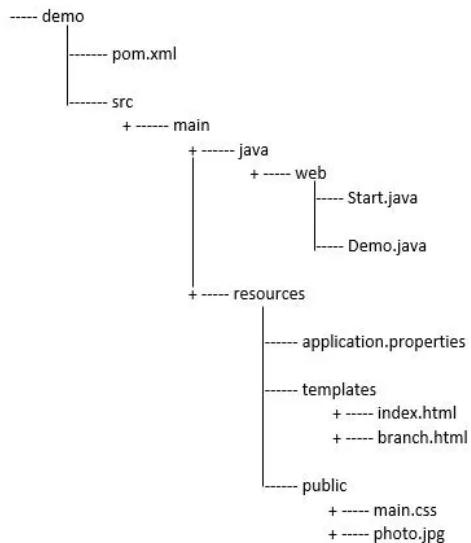
คือเฟรมเวิร์ก ที่ใช้สร้าง UI สำหรับ โมบายแอปพลิเคชัน ที่สามารถทำงานข้ามแพลตฟอร์ม ได้ทั้ง iOS และ Android ในเวลาเดียวกัน โดยภาษาที่ใช้ใน 플ัตเตอร์ นั้นจะเป็นภาษาดาร์ต ซึ่งถูกพัฒนาโดย Google และที่สำคัญคือเป็น open source ที่สามารถใช้งานได้แบบฟรี ๆ อีกด้วย ซึ่งหากสังเกตจากตัวอย่างด้านบน จะเห็นว่า 플ัตเตอร์ นั้นจะมี Widget พื้นฐานมาให้ เพื่อทำให้การออกแบบ UI มีความง่าย และสะดวกยิ่งขึ้น โดย Widget พื้นฐานของ 플ัตเตอร์ หลัก ๆ จะมีอยู่ 2 ชนิดคือ StatelessWidget และ StatefulWidget โดยที่ StatelessWidget จะใช้สร้าง Widget ที่ไม่มีการจัดการสถานะการทำงานใด ๆ เช่น การแสดงข้อความ, Icon หรือรูปภาพที่ไม่มี แอนิเมชัน เข้ามาเกี่ยวข้อง เป็นต้น ส่วน StatefulWidget จะใช้สร้าง Widget ที่มีการจัดการสถานะการทำงานต่าง ๆ เช่น การสร้าง ไอคอน ที่มีการใส่ แอนิเมชัน ให้สามารถยับไปมาได้, บุํกกดต่าง ๆ บนหน้า UI เป็นต้น

จุดเด่นหลักๆ ของ 플ัตเตอร์ คือ ระบบ Hot Reload โดยมีการทดสอบ, การสร้าง, การ add features หรือการกระทำต่าง ๆ กับ UI จะต้องมีการ reload เพื่อให้หน้า UI update ซึ่งระบบ Hot Reload จะเข้ามาช่วยในส่วนของการ reload โดยจุดเด่นของระบบนี้คือการย่นระยะเวลาที่ใช้ในการ reload ให้เหลือเพียงเสี้ยววินาทีเท่านั้น ทำให้การพัฒนา UI ของ แอปพลิเคชัน มีความรวดเร็วขึ้นอย่างมาก และยังมีจุดเด่นอีก 1 ที่ช่วยให้การพัฒนาเป็นไปได้ง่ายขึ้นไม่ว่าจะเป็น Build-In ที่ช่วยในการออกแบบ UI ให้มีความสวยงามยิ่งขึ้นอย่าง Material Design และ Cupertino (iOS-flavor), มี Framework ที่ช่วยให้การทำ แอนิเมชัน ต่าง ๆ หรือ gesture ของ UI เป็นเรื่องง่ายยิ่งขึ้น และยังสามารถใช้งานร่วมกับ IDE ที่กำลังเป็นที่นิยมอยู่ในปัจจุบันอย่าง VS Code และ Android Studio ได้อีกด้วย

จากที่กล่าวมาว่าจุดเด่นต่าง ๆ ของ Flutter มีอะไรบ้าง คราวนี้เราจะมากล่าวถึงข้อเสียกันบ้าง โดยข้อเสียหลัก ๆ ที่พบคือ การใช้ภาษา dart ใน การเขียน ซึ่งคนส่วนใหญ่อาจจะยังไม่คุ้นเคยกับ syntax ของภาษาที่สักเท่าไร ประกอบกับ community ยังเล็กอยู่เนื่องจาก Flutter ยังเปิดตัวมาได้ไม่นานนักเมื่อเทียบกับ Framework ตัวอื่น ๆ อย่าง React Native ที่มี community ค่อนข้างใหญ่ จึงทำให้ document ต่าง ๆ ยังไม่ยอดเยี่ยมเท่าที่ควร ทำให้เวลาไม่ปัญหาเกี่ยวกับการใช้งานอาจจะต้องมาบัง礙มหาวิชีแก้กันนานพอสมควร

2.5.3 Spring Boot

Spring Boot คือ Framework ชนิดหนึ่งใน Spring มีความสามารถในการสร้าง Web application หรือ Web service ได้ง่ายขึ้น เนื่องจาก Spring Boot มี Auto Configuration ซึ่งมีความสามารถในการลดความยุ่งยากในการกำหนดค่าต่างๆ และสามารถใช้งานได้ทันที เนื่องจาก Spring Boot มี Java Web Server ที่ built-in ให้ คือ Tomcat ทำให้ง่ายต่อการใช้งาน โดยมี Port default คือ 8080 ซึ่งสามารถแก้ไขเปลี่ยน Port ได้ที่ไฟล์ application.properties ซึ่งโครงสร้างการเก็บไฟล์ของ Spring Boot มีดังนี้



ภาพที่ 2.55 application.properties

ที่มา: <https://medium.com/@Teerawat.amo/spring-boot>

Spring Boot จะเริ่มต้นทำงานที่ Start.java ซึ่งมีรายละเอียดดังนี้

```
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class DemoApplication {

    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
    }
}
```

ภาพที่ 2.56 Spring Boot Start.java

ที่มา: <https://medium.com/@Teerawat.amo/spring-boot>

การระบุ @SpringBootApplication คือ การกำหนดที่เท่ากับการใช้ 3 Annotation ต่อไปนี้

- @Configuration ใช้การกำหนดค่าด้วยตนเอง
- @EnableAutoConfiguration ใช้การกำหนดค่าอัตโนมัติ
- @ComponentScan ค้นหา class ที่เป็น component อัตโนมัติ เช่น @Controller, @Service, @Component, @Repository เป็นต้น

ต่อมาจะเป็นส่วนของ Controller คือ ไฟล์ Demo.java ยกตัวอย่างในการทำ Web service แบบ RESTful โดยทำงานผ่าน verb ของ HTTP เช่น Get , Post , Put , Delete อย่างเช่นในรูปนี้ เมื่อมี request ไปที่ <http://localhost:8080/helloworld> เมื่อ web browser จะแสดงข้อความว่า “Hello World!” ขึ้นมา ต่อมาจะเป็นส่วนของ Controller คือ ไฟล์ Demo.java โดยจะขยายตัวอย่างเป็นการทำ Web service แบบ RESTful โดยทำงานผ่าน verb ของ HTTP เช่น Get , Post , Put , Delete อย่างเช่นในรูปนี้ เมื่อมี request ไปที่ <http://localhost:8080/helloworld> เมื่อ web browser จะแสดงข้อความว่า “Hello World!” ขึ้นมา

```

    @SpringBootApplication
    @RestController
    public class DemoApplication {

        @GetMapping("/helloworld")
        public String hello() {
            return "Hello World!";
        }
    }

```

ภาพที่ 2.57 <http://localhost:8080/helloworld>
 ที่มา: <https://medium.com/@Teerawat.amo/spring-boot>

2.5.4 ภาษา Dart

ดาร์ต นั้นเป็นภาษาโปรแกรมที่เอาไว้สำหรับสร้างแอพพลิเคชันบนแพลตฟอร์มที่หลากหลาย โดยได้ทั้ง mobile, desktop, server และก็ web สิ่งที่เป็นที่นิยมที่สุดที่ทำให้คนสนใจเรียนภาษา Dart กัน ก็คือเพื่อที่จะเอาไปใช้ร่วมกับ Flutter ที่เป็นเครื่องมือช่วยสร้าง UI ของ Google ซึ่งใช้ได้ทั้งกับ Android และ iOS หรือจะเป็นใน Desktop กับ Web ก็ยังได้ ภาษา Dart นี้ถูกสร้างโดย Google และปล่อยให้ใช้งานแบบ open source ทำให้ทุกคนสามารถนำไปใช้งานได้ฟรีๆ และการที่ Dart ถูกออกแบบมาให้ใช้งานได้ง่ายและมีประสิทธิภาพแบบภาษาเชิงวัตถุอื่นๆอย่าง Java C# C++ จึงเป็นตัวเลือกภาษาที่น่าสนใจในการศึกษาเป็นภาษาแรกอีกด้วย

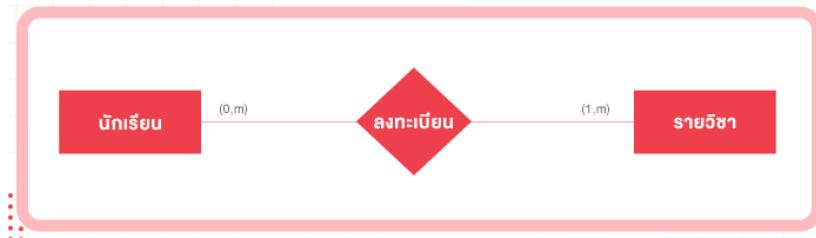
2.6 ระบบจัดการฐานข้อมูล

2.6.1) ER-Diagram

คือ Entity–relationship model (ER model) หรือที่นิยมเรียกว่า E-R Model เป็น Diagram ที่จะช่วยอธิบายโครงสร้าง Database ของระบบต่างๆที่ออกแบบมา อธิบายความสัมพันธ์ (Relationship) ของแต่ละ Entity รวมถึง attributes ของ Entity นั้นๆ ถ้าอธิบายในมุมของ DBMS Entity คือ table และ attributes คือ field ที่อยู่ใน table นั้นเองครับ ผลการออกแบบโดยใช้ E-R Model สามารถแสดงได้ด้วยการเขียนแผนภาพที่เรียกว่า Entity Relationship Diagram(ERD) ซึ่งถือว่าเป็นเครื่องมือที่ใช้อธิบายองค์ประกอบและข้อกำหนดของฐานข้อมูล ที่นักวิเคราะห์และออกแบบระบบใช้เป็นสื่อกลางในการสื่อสารระหว่างผู้ใช้และนักพัฒนาโปรแกรม เนื่องจากมีสัญลักษณ์ที่สื่อความหมายให้เข้าใจได้ง่าย ซึ่งในปัจจุบันมี Tool ที่สามารถแปลงจาก ER-Diagram กลยายนเป็น Database ได้ในภายหลังด้วย เป็นอะไรที่สะดวกมากเลยใช่ไหมล่ะ โดยจะมีองค์ประกอบหลักๆอยู่ 3 ส่วนคือ

- Entity
- Attribute
- Relationship

Simple ER-Diagram



ภาพที่ 2.58 ตัวอย่าง Simple ER-Diagram
ที่มา: <https://blog.clicknext.com/what-is-er-diagram/>

จากภาพที่ 2.55 Diagram ด้านบนเป็นตัวอย่าง ER-Diagram แบบง่ายๆที่ยกมาให้เห็นภาพกันก่อน เป็นความสัมพันธ์ระหว่าง 2 Entity Student กับ Course โดยความสัมพันธ์ที่มีคือ นักเรียน 1 คนสามารถลงทะเบียนได้ตั้งแต่ 1 ถึงหลายรายวิชา และใน 1 วิชาของรับนักเรียนได้หลายคนเป็นความสัมพันธ์แบบ Many-to-Many

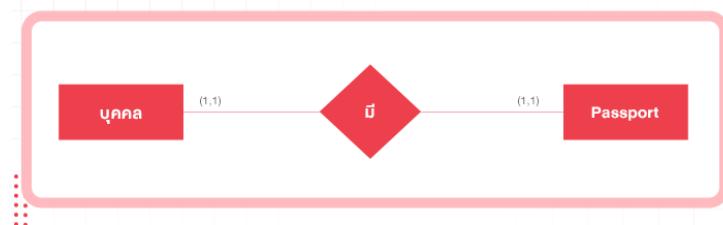
- 1) Component of an ER-Diagram ที่นำมาทำความรู้จักองค์ประกอบต่างๆของ ER Diagram กัน
- 1.1) Entity เอนติตี้(Entity) หมายถึงกลุ่มของสิ่งต่างๆที่สนใจจะเก็บข้อมูลไว้ในฐานข้อมูล ซึ่งอาจจะเป็น บุคคล สถานที่ การกระทำ หรือกิจกรรมต่าง ๆ ซึ่งสัญลักษณ์ที่ใช้ใน ERD คือ สี่เหลี่ยมผืนผ้า ตัวอย่างของเอนติตี้ ได้แก่
 - เอนติตี้ที่เป็น บุคคล เช่น พนักงาน , นักศึกษา , อาจารย์ , แพทย์ , พยาบาล , ผู้ป่วย , นักบิน , พนักงานขับรถ เป็นต้น
 - เอนติตี้ที่เป็น สถานที่ เช่น ประเทศ , จังหวัด , อำเภอ , น้ำตก , ภูเขา , โรงแรม , ห้องพัก , ห้องเช่า , ห้องเรียน เป็นต้น

1.2) Attribute และทริบิวต์(Attribute) หมายถึง ลักษณะหรือคุณสมบัติที่นำมาอธิบาย Entity และ ความสัมพันธ์ ตัวอย่างของแอทริบิวต์ของ Entity ซึ่งสัญลักษณ์ที่ใช้ใน ERD คือวงรี สำหรับ แอทริบิวต์ที่ถูกกำหนดให้ทำหน้าที่เป็นคีย์หลัก มีค่าได้เพียงค่าเดียวห้ามซ้ำกัน(primary key) ของ Entity ก็จะชี้ด้วยเส้นทึบใต้ชื่อของแอทริบิวต์ เพื่อแสดงให้รู้ว่าเป็นคีย์หลัก เช่น

- แอทริบิวต์ของ Entity “นักศึกษา” ได้แก่ รหัสนักศึกษา , คำนำหน้าชื่อ , ชื่อ , นามสกุล , วันเกิด , โปรแกรมวิชาที่สังกัด , เกรดเฉลี่ยสะสม
- แอทริบิวต์ของ Entity “ผู้ป่วย” ได้แก่ รหัสผู้ป่วย , ชื่อ , นามสกุล , สถานภาพ , วันที่เข้ารักษาครั้งแรก , ที่อยู่ , โทรศัพท์

1.3) Relationship ความสัมพันธ์ (Relationship) หมายถึง ความสัมพันธ์ระหว่าง Entity ต่าง ๆ ซึ่ง สัญลักษณ์ที่ใช้ใน ERD คือสี่เหลี่ยมผืนผ้า มีอยู่ด้วยกัน 4 แบบ ดังนี้

1.4) One-to-One Relationship หรือ 1 : 1 จากภาพที่ 2.59 เป็นการแสดงความสัมพันธ์ ของจำนวนข้อมูลของ Entity A ว่า ข้อมูล 1 รายการ มีความสัมพันธ์กับข้อมูล Entity B ได้ไม่เกิน 1 รายการ ตัวอย่าง เช่น บุคคล 1 คน จะสามารถมี passport ได้ 1 ใน และในขณะเดียวกัน passport 1 ใบมีข้อมูลได้แค่ 1 คนเท่านั้น



ภาพที่ 2.59 ตัวอย่าง One-to-One Relationship

ที่มา: <https://blog.clicknext.com/what-is-er-diagram/>

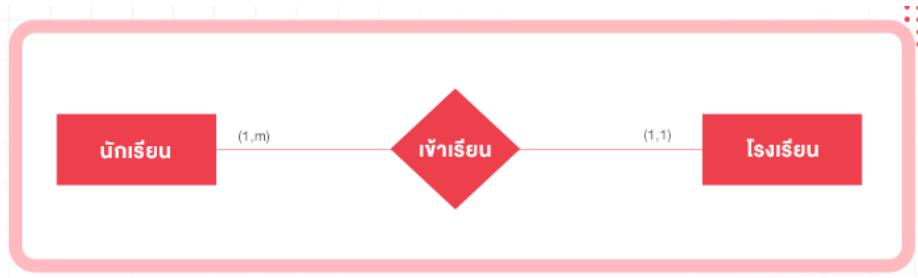
1.5) One-to-Many Relationship หรือ 1 : N จากภาพที่ 2.60 เป็นการแสดงความสัมพันธ์ ของจำนวนข้อมูลของ Entity A ว่า ข้อมูล 1 รายการ มีความสัมพันธ์กับข้อมูล Entity B ได้มากกว่า 1 รายการ ตัวอย่างเช่น อาจารย์ 1 คน จะสามารถมีนักศึกษาที่ปรึกษาได้มากกว่า 1 คน และในขณะเดียวกัน นักศึกษาแต่ละคนต้องมีอาจารย์ที่ปรึกษาคนใดคนหนึ่งเท่านั้น ดังภาพที่ 2.60



ภาพที่ 2.60 ตัวอย่าง One-to-Many Relationship

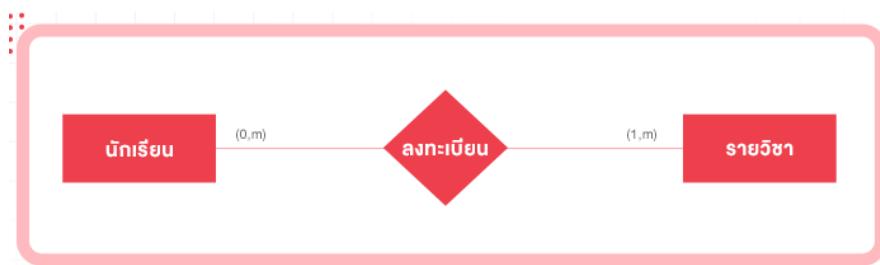
ที่มา: <https://blog.clicknext.com/what-is-er-diagram/>

1.6) Many-to-One Relationship หรือ N : 1 จากภาพที่ 2.61 เป็นการแสดงความสัมพันธ์ ของจำนวนข้อมูลของ Entity A ว่า ข้อมูล 1 รายการ มีความสัมพันธ์กับข้อมูล Entity B ได้แค่ 1 รายการ ในขณะที่ข้อมูล Entity B มีความสัมพันธ์กับ Entity A ได้มากกว่า 1 รายการ ตัวอย่างเช่น นักเรียน 1 คน จะสามารถเข้าเรียนที่โรงเรียนได้แค่ 1 โรงเรียนเท่านั้น แต่ในขณะเดียวกันโรงเรียน 1 โรงเรียนมีจำนวนนักเรียนได้หลายคน



ภาพที่ 2.61 ตัวอย่าง Many-to-One Relationship
ที่มา: <https://blog.clicknext.com/what-is-er-diagram/>

1.7) Many-to-Many Relationship หรือ M : N จากภาพที่ 2.62 เป็นการแสดงความสัมพันธ์ของจำนวนข้อมูลของ Entity A ว่า ข้อมูล 1 รายการ มีความสัมพันธ์กับข้อมูลเอนติตี้ B ได้แค่หลายรายการ ในขณะที่ข้อมูล Entity B มีความสัมพันธ์กับ Entity A ได้มากกว่า 1 รายการ เช่นเดียว ตัวอย่างเช่น นักเรียน 1 คนสามารถลงทะเบียนได้หลายรายวิชา และใน 1 วิชารองรับนักเรียนได้หลายคน



ภาพที่ 2.62 ตัวอย่าง Many-to-Many Relationship
ที่มา: <https://blog.clicknext.com/what-is-er-diagram/>

2.6.2) พจนานุกรมข้อมูล

คือ พจนานุกรมข้อมูล ที่แสดงรายละเอียดตารางข้อมูลต่างๆ ในฐานข้อมูล (Database) ซึ่งประกอบด้วยชื่อรีเลชัน (Relation Name), แอตทริบิวต์ (Attribute), ชื่อแทน (Aliases Name), รายละเอียดข้อมูล (Data Description), แอตทริบิวเตโดเมน (Attribute Domain), ฯลฯ ทำให้สามารถค้นหารายละเอียดที่ต้องการได้สะดวกมากยิ่งขึ้น พจนานุกรมข้อมูลเป็นการแสดงผลระหว่างรูปแบบของพจนานุกรมโดยทั่วไปและรูปแบบของข้อมูลในระบบงานคอมพิวเตอร์ เพื่ออธิบายชนิดของข้อมูลแต่ละตัวว่าเป็น ตัวเลข อักษร ข้อความ หรือวันที่ เป็นต้น เพื่อช่วยในการอธิบายรายละเอียดต่างๆ ในการอ้างอิงหรือค้นหาที่เกี่ยวกับข้อมูล หรือจะเรียกง่ายๆ ว่า Data Dictionary คือ เอกสารที่ใช้อธิบายฐานข้อมูลหรือการจัดเก็บฐานข้อมูล ซึ่ง Data Dictionary มีประโยชน์ ดังนี้

- จัดเก็บรายละเอียดข้อมูล
- แสดงความหมายที่เกี่ยวกับระบบ
- ทำเอกสารที่บอกรุณเล็กขณะของระบบ

- หาข้อบกพร่องและสิ่งที่หายไปจากระบบ

ส่วนประกอบของ Data Dictionary

1. ข้อมูลย่อย (Data Element) : ส่วนประกอบพื้นที่ ที่ไม่สามารถแบ่งให้เล็กลงได้อีก
2. โครงสร้างข้อมูล (Data Structure) : สร้างขึ้นโดยการนำส่วนย่อยของข้อมูล ตั้งแต่ ตัวขึ้นไป ที่สัมพันธ์กันมาร่วมเข้าด้วยกัน

สัญลักษณ์ที่ใช้ในพจนานุกรมข้อมูล ได้แก่

- = หมายถึง เท่ากับ
- + หมายถึง และ
- { } หมายถึง มีการซ้ำของส่วนย่อยข้อมูล
- [] หมายถึง ทางเลือกให้เลือกส่วนย่อยของข้อมูลตัวใดตัวหนึ่ง
- () หมายถึง การเกิดขึ้นเป็นกรณีพิเศษ จะปรากฏหรือไม่ปรากฏก็ได้

2.6.3) マイแอสคิวเอล

คือ ระบบจัดการฐานข้อมูล หรือ Database Management System (DBMS) แบบฐานข้อมูลเชิงสัมพันธ์ หรือ Relational Database Management System (RDBMS) ซึ่งเป็นระบบฐานข้อมูลที่จัดเก็บรวบรวมข้อมูลในรูปแบบตาราง โดยมีการแบ่งข้อมูลออกเป็นแถว (Row) และในแต่ละแถวแบ่งออกเป็นคอลัมน์ (Column) เพื่อเชื่อมโยงระหว่างข้อมูลในตารางกับข้อมูลในคอลัมน์ที่กำหนด แทนการเก็บข้อมูลที่แยกออกจากกัน โดยไม่มีความเชื่อมโยงกัน ซึ่งประกอบด้วยข้อมูล (Attribute) ที่มีความสัมพันธ์เชื่อมโยงกัน (Relation) โดยใช้ RDBMS Tools สำหรับการควบคุมและจัดเก็บฐานข้อมูลที่จำเป็น ทำให้นำไปประยุกต์ใช้งานได้ง่าย ช่วยเพิ่มประสิทธิภาพในการทำงานให้มีความยืดหยุ่นและรวดเร็วได้มากยิ่งขึ้น รวมถึงเชื่อมโยงข้อมูล ที่จัดแบ่งกลุ่มข้อมูลแต่ละประเภทได้ตามต้องการ จึงทำให้ MySQL เป็นโปรแกรมระบบจัดฐานข้อมูลที่ได้รับความนิยมสูง

โปรแกรมนี้เป็น Open Source ที่ถูกพัฒนาขึ้นจาก MySQL AB ในประเทศสวีเดน โดยชาวสวีเดน 2 คน คือ David Axmark และ Allan Larsson ร่วมกับชาวพินแลนด์ Michael Monty Widenius ซึ่งต่อมาในปี ค.ศ. 2008 ถูกซื้อกิจการโดย Sun Microsystems และภายหลัง Oracle Corporation ได้เข้าซื้อกิจการในปี ค.ศ. 2010

มีหน้าที่จัดเก็บข้อมูลอย่างเป็นระบบ รองรับคำสั่งภาษา Structured Query Language หรือ SQL เพื่อจัดการกับฐานข้อมูลโดยเฉพาะ เป็นภาษามาตรฐานระบบฐานข้อมูลเชิงสัมพันธ์และเป็นระบบเปิด (Open System) ที่มีโครงสร้างของภาษาที่เข้าใจง่าย ไม่ซับซ้อน และนิยมใช้งานร่วมกับภาษาโปรแกรม PHP รวมถึงภาษาอื่น ๆ ที่สามารถทำงานร่วมกับฐานข้อมูล MySQL ได้หลากหลาย เช่น C, C++, Python, Java เป็นต้น อีกทั้ง MySQL ยังได้รับการออกแบบและปรับให้มีความเหมาะสมสำหรับการพัฒนา Website และ Web Application ทำให้สามารถรองรับการทำงานได้ทุกแพลตฟอร์ม รวมถึงการอนุญาตให้ผู้ใช้งานหลายคนสามารถใช้งานพร้อมกันได้ (Multi-user) นอกจากนี้ยังสามารถจัดการและสร้างฐานข้อมูลจำนวนมากรวมถึงประมวลผลหลาย ๆ งานได้พร้อมกัน (Multi-threaded) อย่างสมบูรณ์ จึงทำให้ MySQL เป็นตัวเลือกยอดนิยมสำหรับธุรกิจการพาณิชย์อิเล็กทรอนิกส์ หรือ Electronic Commerce (E-Commerce) และเหมาะสม

สำหรับการนำไปใช้งานสร้างเว็บไซต์ทั่วไป เพราะมีความแม่นยำ ครบครัน ช่วยให้เข้าถึงข้อมูลได้อย่างรวดเร็ว อีกทั้งยังมีความน่าเชื่อถือสูง และยังมีโปรแกรมเสริมช่วยจัดฐานข้อมูลที่ใช้งานง่าย เช่น Mysql Admin, phpMyAdmin เป็นต้น

1.1) ระบบใช้งาน

มีให้เลือกใช้งาน 2 รุ่น ได้แก่ MySQL Community Edition ที่เป็นเวอร์ชันฟรี ซึ่งเป็น Open Source และ MySQL Enterprise Edition ที่มีคุณสมบัติมากกว่าและการสนับสนุนด้านเทคนิคที่ครอบคลุม รวมถึงยังได้รับอนุญาตให้ใช้ในเชิงพาณิชย์ โดย MySQL เป็นตัวเลือกยอดนิยมสำหรับเว็บไซต์ขนาดใหญ่และ Web Application เนื่องจากสามารถรองรับการรับส่งข้อมูลในระดับสูง รวมถึงยังมีคุณสมบัติที่ช่วยเพิ่มประสิทธิภาพการทำงาน เช่น กระบวนการจัดเก็บข้อมูล (Store Procedure), กระบวนการทำงานแบบอัตโนมัติ (Database Trigger), มุมมองฐานข้อมูล (Database View) และการรวมระบบฐานข้อมูล (Database Schema) เป็นต้น

โดย MySQL ถูกนำไปใช้ในองค์กรหรือกลุ่มธุรกิจชั้นนำต่าง ๆ มากมาย เพราะสามารถปรับใช้ให้เข้ากับความต้องการของแต่ละองค์กรได้อย่างมีประสิทธิภาพ เช่น

- การจัดเก็บข้อมูลสำหรับ Website
- การจัดเก็บข้อมูลสำหรับ Mobile Application
- การจัดเก็บข้อมูลสำหรับ Application สำหรับองค์กร
- การจัดเก็บข้อมูลทางการแพทย์
- การจัดเก็บข้อมูลทางการเงิน
- การจัดเก็บและสร้างฐานข้อมูลของลูกค้า

2.7 การทดสอบซอฟต์แวร์

การทดสอบซอฟต์แวร์คือ กระบวนการการประเมินและปรับปรุงคุณภาพซอฟต์แวร์ของบริษัท เพื่อค้นหาข้อผิดพลาดหรือข้อบกพร่องของซอฟต์แวร์ให้ปรากฏออกมาเพื่อหาแนวทางอุดช่องโหว่ของปัญหาที่เกิดขึ้น หรืออาจเกิดขึ้นให้ได้นั่นเอง สำหรับอาชีพ Tester หรือ Software tester หรือ Quality Assurance Engineer หรือ Quality Engineer นั้น จริงๆ ทั้งหมดนี้ก็คืออาชีพเดียวกันทั้งหมด โดยพื้นฐานหน้าที่ของอาชีพนี้ก็คือ คนที่มีหน้าที่ตรวจสอบคุณภาพของ software ที่ถูกผลิตขึ้นมาโดย programmer โดยทำการทดสอบระบบต่างๆ และตรวจสอบหาข้อผิดพลาดของซอฟต์แวร์ เพื่อเช็คดูให้ดีว่าซอฟต์แวร์ที่เราส่งออกไปให้ลูกค้านั้นมีข้อผิดพลาดอะไรตรงไหนหรือไม่ เพื่อหารือแก้ไขปัญหาให้ทันท่วงที ซึ่งหน้าที่หลักๆ ก็จะนั่งทดสอบระบบและทดสอบซอฟต์แวร์ทั้งวัน เพื่อป้องกันไม่ให้เกิด Bug หรือข้อผิดพลาดตามมาเมื่อซอฟต์แวร์ไปถึงมือของลูกค้าแล้ว ซึ่งจริงๆ แล้ว คนที่เป็น Software Tester เค้าก็ไม่ได้มีหน้าที่นั่งทดสอบทั้งวัน ขนาดนั้นหรอก เพราะในระหว่างวันก็ยังต้องมีหน้าที่อื่นๆ แทรกเข้ามาเพิ่มอีกด้วยเช่นกัน

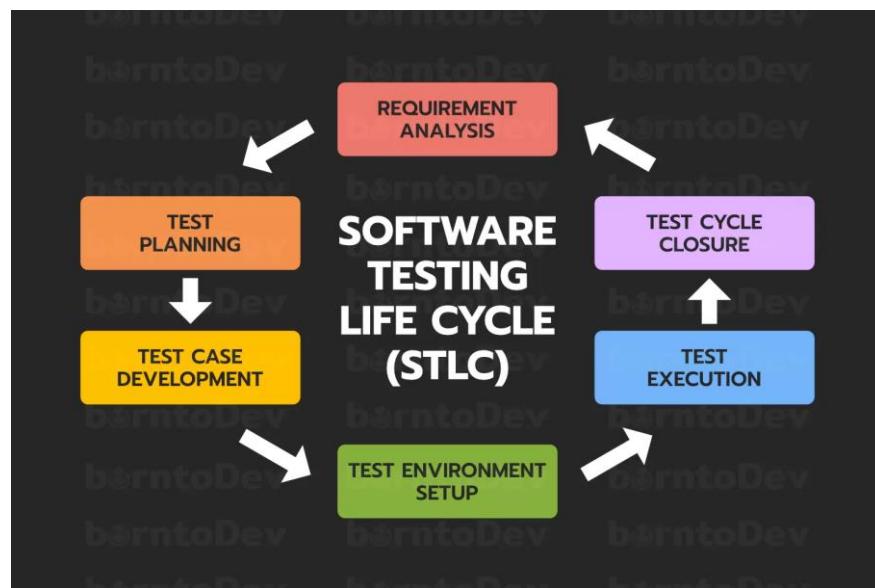
ประเภทของ Software Testing แบ่งออกเป็น 3 ตัวด้วยกันได้แก่

1) Functional Testing การทดสอบถึงฟังก์ชันในการทำงานของระบบว่าสามารถทำงานได้จริง หรือไม่ ตามเอกสารความต้องการที่ระบุไว้ โดยจะวิเคราะห์ถึงในส่วนต่าง ๆ เช่น User Interface, Database, Security และอื่น ๆ เช่น ทดสอบระบบฝากรเงินว่าสามารถฝากรเงินได้จริงหรือไม่ เป็นต้น

2) Non-Functional Testing การทดสอบสิ่งที่ตรงกันข้ามกับ Functional Testing ซึ่งเป็นการทดสอบถึงสิ่งที่ไม่สามารถเขียนเป็นฟังก์ชันได้ที่ได้รับตามเอกสารความต้องการ เช่น Performance Testing ที่จะทดสอบถึงความเร็ว การตอบสนอง และปริมาณการใช้ทรัพยากรต่าง ๆ เช่น หน้าเว็บต้องเปิดด้วยความเร็วไม่เกิน 3 วินาที หรือจะเป็นส่วน Usability ที่วัดในความพึงพอใจของผู้ใช้ก็จัดอยู่ในประเภทนี้เช่นกัน

3) Maintenance Testing ตรวจสอบว่าระบบรองรับในการบำรุงรักษาหรือไม่ ซึ่งมีเป้าหมายเพื่อค่อยค้นหาข้อผิดพลาดหรือจุดบกพร่องต่าง ๆ ของระบบที่อาจเกิดขึ้นหลังจากมีการเปลี่ยนแปลง เช่น มีการอัปเดตเฟลอร์ใหม่เข้ามาภายในแอปพลิเคชัน ก็จะต้องมีการทดสอบเพื่อให้มั่นใจว่าจะไม่ส่งผลเสียหรือขัดแย้งในโค้ดที่ทำงานอยู่

ในขั้นตอนของ Software Testing จะมีความคล้ายกับ Software Development Life Cycle (SDLC) โดยจะมีชื่อว่า “Software Testing Life Cycle (STLC)”



ภาพที่ 2.63 ตัวอย่าง Software Testing Life Cycle (STLC)
ที่มา: <https://www.borntodev.com/2023/09/19/>

ประกอบไปด้วย 6 กระบวนการหลัก ๆ ดังนี้

ขั้นที่ 1 : Requirement Analysis : การวิเคราะห์ความต้องการที่ได้รับว่ามีอะไรที่สามารถทดสอบได้บ้าง จะทดสอบกันอย่างไร เป็นต้น

ขั้นที่ 2 : Test Planning : วางแผนกลยุทธ์ในการทดสอบว่าจะทำได้อย่างไร ด้วยการกำหนด Objective (จุดหมาย), Resource (ทรัพยากร)

ขั้นที่ 3 : Test Case Development : เขียน Test Case ในการทดสอบแต่ละส่วนโดยจะทดสอบนอกเหนือจากที่เอกสารความต้องการระบุไว้

ขั้นที่ 4 : Test Environment Setup : ทดสอบสภาพแวดล้อมต่าง ๆ เช่น ผ่านอุปกรณ์อะไร , Setup อย่างไร เป็นต้น

ขั้นที่ 5 : Test Execution : ทำการทดสอบตาม Environment ที่ได้กำหนดไว้

ขั้นที่ 6 : Test Cycle Closure : ขั้นตอนสุดท้ายเป็นการทำสรุปการทดสอบว่ามีอะไรเกิดขึ้นบ้าง ระดับของการทดสอบ Software ควรที่จะมีอะไรบ้างสามารถแบ่งออกเป็นหลัก ๆ ได้ตามนี้

1. Module Testing หรือ Unit Testing การทดสอบในแต่ละส่วนของโปรแกรมเพื่อตรวจสอบ ความถูกต้องและยืนยันว่าแต่ละหน่วยทำงานได้อย่างถูกต้องเพื่อป้องกันข้อผิดพลาดที่อาจเกิดขึ้น ถ้าหากไม่มีการทดสอบในส่วนนี้โปรแกรมของเราก็อาจจะเกิดข้อบกพร่องในการทำงานได้

2. Integration Testing การทดสอบในภาพรวมว่าส่วนต่าง ๆ ของโปรแกรมสามารถทำงานร่วมกันได้โดยไม่มีข้อผิดพลาด รวมไปถึงการติดต่อสื่อสารระหว่างส่วนต่าง ๆ เช่น การทดสอบเรียกใช้ฟังก์ชันข้าม Module เป็นต้น ถ้าหากไม่มีการทดสอบนี้ก็จะส่งผลในการทำงานโดยรวมของโปรแกรมที่อาจมีข้อบกพร่องอยู่ในบางส่วน

3. System Testing การทดสอบในการทำงานของระบบว่าถูกต้องตามความต้องการหรือไม่ โดยจะตรวจสอบตั้งแต่ Performance , Load , Reliability รวมไปถึง Security เพื่อจุดหมายในการวัดผลการทำงานของระบบก่อนที่จะส่งมอบให้ผู้ใช้งานจริงทดสอบในลำดับถัดไป หากไม่มีการทดสอบในขั้นตอนนี้ก็จะส่งผลเสียเป็นอย่างสูงหากผู้ใช้ได้ลองทดสอบจริงแล้วตรวจพบข้อผิดพลาดด้วยตัวเองซึ่งควรจะมีข้อผิดพลาดให้น้อยที่สุดก่อนถึงมือของผู้ใช้งานจริง

4. Acceptance Testing (UAT) การทดสอบขั้นสุดท้ายเพื่อตรวจสอบว่าระบบสามารถปฏิบัติตามความต้องการของผู้ใช้และองค์กร โดยจะเน้นในการทดสอบซึ่งให้ผู้ใช้หรือลูกค้ามาทดสอบด้วยตัวเองว่าระบบสามารถทำงานได้จริงตามที่ต้องการหรือไม่ ซึ่งการทดสอบนี้จำเป็นที่จะต้องมีหากว่าไม่มีแล้วผู้ใช้หรือองค์กรจะมั่นใจได้อย่างไร ว่าระบบสามารถทำงานได้ตรงตามความต้องการจริง

2.8 งานวิจัยที่เกี่ยวข้อง

นายณัฐวุฒิ แย้มมาศ (2564) ได้พัฒนาระบบแนะนำแหล่งท่องเที่ยวในเขตทหารของประเทศไทย ของซึ่งเป็นแอปพลิเคชันบนระบบปฏิบัติการ แอนดรอยด์ (Android OS) และมีจุดประสงค์ในการเผยแพร่ข้อมูลเกี่ยวกับแหล่งท่องเที่ยวภายในเขต ทหารที่น่าสนใจในแต่ละทุกภาคส่วนในประเทศไทย แอปพลิเคชันนี้มีการใช้ในรูปแบบ ได้แก่ภาษาไทย โดยการทำงานของแอปพลิเคชันจะมีเนื้อหาต่างๆ เกี่ยวกับแหล่งท่องเที่ยวที่มีการแบ่งภาคส่วน มี ทั้งหมด 3 ส่วนได้แก่ กรมท่าอากาศยาน กรมท่าอากาศยาน และกรมท่าอากาศยาน แต่ละประเภทแสดงแหล่งท่องเที่ยวหลากหลายและจักน่าสนใจโดยรอบ โดยแสดงข้อมูลรายละเอียดของพื้นที่ เช่น ประวัติของพื้นที่ แหล่งท่องเที่ยวที่น่าสนใจ รวมทั้งรูปภาพ วิดีโอตัวอย่าง และตำแหน่งของพื้นที่ที่นำทางได้ ในแอป พลิกะชันจะมีการเรียกใช้งาน Google Map เพื่อนำทางไปยังตำแหน่งของแหล่งท่องเที่ยวนั้นๆ อีกทั้ง ผู้ใช้สามารถโหลดให้คะแนนความพึงพอใจของแหล่งท่องเที่ยวที่ได้ชมอีกด้วย การพัฒนาระบบแนะนำ แหล่งท่องเที่ยวในเขตทหารของประเทศไทย บนสมาร์ทโฟนระบบปฏิบัติการแอนดรอยด์ ผู้พัฒนาได้ใช้ภาษา Java ในการสั่งการทำงานของแอปพลิเคชัน และใช้ภาษา Extensible Markup Language (XML) ในการออกแบบและจัดรูปแบบหน้าจอแอปพลิเคชันบนโปรแกรมแอนดรอยด์สตูดิโอ (Android Studio) และใช้ระบบ GPS และ Google Map เป็นส่วนหนึ่งในการพัฒนา Copyright ©

นายอภิวัฒน์ เพียงจันตา (2565) ได้พัฒนาแอปพลิเคชันแนะนำสถานที่ท่องเที่ยวและที่พักในอำเภอจอมทอง จังหวัด เชียงใหม่บนระบบปฏิบัติการแอนดรอยด์ มีวัตถุประสงค์เพื่อ พัฒนาแอปพลิเคชันแนะนำสถานที่ ท่องเที่ยวและที่พัก เครื่องมือที่ใช้ในการศึกษา ประกอบด้วย ภาษา ไออ้อนิก เฟรมเวิร์ค (Ionic Framework) ภาษา แองกูล่า (Angular) ภาษา โหนดเจอส (Node.js) ระบบปฏิบัติการ Windows และระบบปฏิบัติการ แอนดรอยด์ (Android) ระบบฐานข้อมูล MySQL ระบบเครือข่ายอินเทอร์เน็ต JavaScript และ จากการศึกษาพบว่า ระบบที่พัฒนาขึ้นสามารถอ่านความสนใจของผู้ใช้ได้แม่นยำ สามารถติดตามสถานที่ท่องเที่ยวและสถานที่พัก รวมถึงสามารถแนะนำสถานที่ท่องเที่ยวที่อยู่ใกล้ๆ กับสถานที่พัก ให้แก่ผู้ใช้ สามารถตรวจสอบสถานที่ท่องเที่ยวและสถานที่พักในอำเภอจอมทอง จังหวัดเชียงใหม่บนระบบปฏิบัติการแอนดรอยด์ ที่พัฒนาขึ้นเน้นเกี่ยวกับการแสดงรายละเอียดและสามารถนำทางไปยังสถานที่ต่างๆ สามารถนำพาไปใช้ ได้จริงและคาดว่าจะช่วยอำนวยความสะดวกให้แก่นักท่องเที่ยวสามารถได้ใช้งานแอปพลิเคชันนี้และ เพื่อเป็นต้นแบบในการพัฒนาแอปพลิเคชันแนะนำสถานที่ท่องเที่ยวและที่พักในอำเภอจอมทอง จังหวัดเชียงใหม่บนระบบปฏิบัติการแอนดรอยด์ ต่อไปในอนาคต

นายภาครช ชุมใจ (2565) ได้พัฒนาแอปพลิเคชันแนะนำโรงพยาบาลสัตว์ในเขตตัวเมืองเชียงใหม่ ซึ่ง เป็นแอปพลิเคชันบนระบบปฏิบัติการ แอนดรอยด์ (Android OS) มีวัตถุประสงค์เพื่ออำนวยความสะดวกให้แก่เจ้าของสัตว์เลี้ยงที่ต้องการนำสัตว์เข้าใช้ บริการโรงพยาบาลสัตว์ โดยการทำงานของแอปพลิเคชันจะมีเนื้อหาเกี่ยวกับ โรงพยาบาลสัตว์ทั้งหมด 15 แห่ง ในเขตตัวเมืองจังหวัดเชียงใหม่ ซึ่งจะแสดงข้อมูลของโรงพยาบาลสัตว์ บริการ เวลาเปิด เวลาปิด พิกัดตำแหน่ง ช่องทางการติดต่อ แอปพลิเคชันจะมีการเรียกใช้ งาน Google Map เพื่อใช้คำแนะนำที่ตั้งโดยรวมของ โรงพยาบาลสัตว์และใช้นำทางไปยังโรงพยาบาลสัตว์ได้ และการพัฒนาแอปพลิเคชันแนะนำโรงพยาบาลสัตว์ ในเขตตัวเมืองเชียงใหม่ บนสมาร์ทโฟนแอนดรอยด์ ผู้พัฒนาได้ใช้โปรแกรมวิชวล สตูดิโอ (Visual Studio) ซึ่ง ใช้ภาษา Kotlinในการออกแบบและพัฒนา

นายภาณุวัฒน์ พองเมฆ (2564) ได้พัฒนาแอปพลิเคชันแนะนำสถานที่ท่องเที่ยวใน อำเภอแม่สะเรียง จังหวัดแม่ฮ่องสอน บนปฏิบัติการแอนดรอยด์ ผู้วิจัยได้พัฒนาขึ้นเพื่ออำนวยความสะดวกต่อผู้ใช้ โดยพัฒนาให้มีการเข้าถึงข้อมูลและรายละเอียดของสถานที่ท่องเที่ยวได้อย่างรวดเร็ว และเพื่อให้ผู้ใช้สามารถในการเดินทางมาเที่ยวได้ด้วยตัวเอง โดยการทำงานของระบบจะทำงานผ่านเครือข่ายอินเทอร์เน็ต และแสดงผลในอุปกรณ์เคลื่อนที่ โดยใช้โปรแกรมมอนゴลีบี (MongoDB) ในการจัดเก็บฐานข้อมูลในรูปแบบฐานข้อมูลเชิงสัมพันธ์ ใช้ (API) Nest JS ในการประมวลผลการทำงานของแอปพลิเคชันในส่วนต่างๆ เช่น การดึงข้อมูลจาก เพิ่ม ลบ แก้ไข ฐานข้อมูล ในตัวแอปพลิเคชันจะใช้ ภาษา Dart ในการพัฒนา ภาษา Dart ยังสามารถทำงานได้ทั้ง Android และ IOS การทำงาน ในตัวแอปพลิเคชันสามารถอ่านทางยังสถานที่ท่องเที่ยว ที่ผู้ใช้งานสนใจได้จากการทดสอบระบบพบว่า ระบบสามารถใช้งานได้ดีและสามารถนำ ไปใช้งานได้จริง ทำให้มีการประชาสัมพันธ์ว่าสารรายละเอียดต่างๆ เปิดกว้างยิ่งขึ้น

นางสาวอธิชา จิตตางกูร (2563) ได้พัฒนาแอปพลิเคชันแนะนำสถานที่ท่องเที่ยวอำเภอเชียงของ จังหวัดเชียงราย บนระบบปฏิบัติการแอนดรอยด์ มีวัตถุประสงค์เพื่ออำนวยความสะดวกแก่นักท่องเที่ยวที่เข้ามาเยี่ยมชม และส่งเสริมการท่องเที่ยวให้แก่อำเภอเชียงของ จังหวัดเชียงราย โดยจะมีวิธีดำเนินการคือ ในขั้นตอนการเก็บรวบรวมข้อมูลจะค้นคว้าข้อมูลผ่านเว็บไซต์ และเอกสารที่มีเนื้อหาเกี่ยวกับระบบปฏิบัติการ แอนดรอยด์ และรวบรวมข้อมูลเกี่ยวกับสถานที่ท่องเที่ยว ณ เทศบาลเชียงของ และใช้โปรแกรมแอนดรอยด์

สตูดิโอ (Android Studio) ซึ่งใช้ภาษาจาวา (Java) ในการใช้งาน โดยมีเทคนิคการเรียกใช้ ไลบรารี (Library) ต่าง ๆ มาช่วยในการออกแบบแอปพลิเคชันตัวแอปพลิเคชันจะทำการเชื่อมต่อกับภูเก็ต แมพ เอฟีไอ (Google Maps API) เป็นตัวแสดงแผนที่ โดยผ่านทำงานของภูเก็ต แมพ (Google Maps) และสามารถนำทางไปยังสถานที่ต่าง ๆ ได้อย่างถูกต้องสามารถใช้งานแอนดรอยด์เวอร์ชั่น 6.0 ขึ้นไป เมื่อพัฒนาเสร็จแล้วนำไปใช้กับกลุ่มตัวอย่างคือ นักศึกษา บุคลากรและผู้ใช้งานทั่วไปจำนวน 80 คน โดยใช้แบบสอบถามความพึงพอใจในการใช้งานแอปพลิเคชันในการประเมินมีผลตอบรับในด้านของผู้ใช้งาน โดยรวมมีความพึงพอใจในการใช้งานแอปพลิเคชัน

บทที่ 3

การวิเคราะห์และการออกแบบระบบ

การวิเคราะห์และการออกแบบ โมบายแอปพลิเคชันและเว็บแอปพลิเคชัน ช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ด ประกอบด้วยรายละเอียดดังนี้

3.1 วิเคราะห์ระบบงาน

3.2 ยูเคส ไดอะแกรม (Use Case diagram)

3.3 คลาสไดอะแกรม (Class Diagram)

3.4 ซีเควนซ์ ไดอะแกรม (Sequence Diagram)

3.5 แอคทิวิตี้ ไดอะแกรม (Activity Diagram)

3.6 อี-อาร์ ไดอะแกรม (Er - Diagram)

3.7 ออกแบบฐานข้อมูลเชิงกายภาพ (Physical Design)

3.1 วิเคราะห์ระบบงาน

3.1.1 วิเคราะห์งานแบบเดิม

ในปัจจุบันนั้นการรู้จักผู้สูงอายุเป็นเรื่องยากเนื่องจากวิถีการใช้ชีวิตของคนในปัจจุบันค่อนข้างที่จะเป็นอะไรที่ใหม่ๆ ถ้าเทียบกับการใช้ชีวิตเมื่อ 10 ปี ที่แล้วเนื่องจากวิถีการใช้ชีวิตของผู้คนสมัยนี้จะเป็นการพบร่องกันที่ไม่ใช่แบบพบกันต่อหน้าแต่จะเป็นออนไลน์ส่วนใหญ่ทำให้ผู้ใหญ่ที่อายุตั้งแต่ 40 ปีขึ้นไปนั้นแทบจะไม่รู้จักกัน เพราะคนที่อายุมากกว่า 40 ปีนั้นบางคนไม่สามารถใช้งานเทคโนโลยีสมัยใหม่ได้ถัดนักเข่นเดียว กับผู้สูงอายุที่ในบางครั้งเขามีได้รู้จักคนสมัยใหม่เลย เพราะคนสมัยนั้น

ค่อนข้างที่จะเปลี่ยนวิถีชีวิตต่างจากเมื่อก่อนทำให้เราเห็นถึงปัญหาที่จะตามมาของผู้ป่วยที่สูงวัยที่อาจมีความจำที่เสื่อมถอยตามอายุทำให้ยากที่จะจำอะไรได้หลายอย่าง

3.1.2 วิเคราะห์ระบบงานใหม่

จากในปัญหาที่กล่าวนั้นทำให้เราเล็งเห็นระบบใหม่ที่จะช่วยในการแก้ไขปัญหาที่กล่าวมา ข้างต้นทำให้เล็งเห็นปัญหาระหว่างคนรุ่นใหม่กับผู้สูงอายุ ผู้วิจัยจึงได้คิดค้นแอปพลิเคชันที่จะช่วยสามารถพาผู้สูงอายุกลับบ้านได้ซึ่งจะใช้ข้อมูลของผู้สูงอายุจากโรงพยาบาลที่รักษาโดยใช้ข้อมูลที่อยู่ที่ผู้ใช้งานหรือผู้ป่วยกรอกกับทางโรงพยาบาลกรณีที่ผู้ป่วยออกใบพื้นที่ใกล้จะพื้นที่ผู้เยี่ยมชมสามารถใช้งานคิวอาร์โค้ดโดยจะสามารถร้องขอข้อมูลผู้ป่วยได้โดยเจ้าหน้าที่หรือแอดมินจะสามารถเห็นคำร้องขอเข้าถึงข้อมูลผู้ป่วยได้โดยผู้ดูแลระบบสามารถเห็นการร้องขอได้และสามารถทำการอนุมัติให้เข้าถึงข้อมูลได้โดยผู้เยี่ยมชมต้องร้องขอผ่านคิวอาร์โค้ดที่ติดอยู่กับผู้ใช้งานพาผู้ใช้งานกลับบ้านโดยข้อมูลที่เข้าถึงได้ทั่วไปและข้อมูลสถานที่ตั้งบ้านพักอาศัยของผู้ใช้

3.2 ยูเคส ไดอะแกรม (Use Case Diagram)

เป็นแผนภาพที่ใช้แสดงให้ทราบว่าระบบการทำงานของแอปพลิเคชันหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ด มีระบบการทำงานอย่างไรโดยใช้ยูสเคสไดอะแกรม เข้ามาช่วยในการอธิบาย



ภาพที่ 3.1 ยูสเคส ไดอะแกรม (USE CASE DIAGRAM) แอปพลิเคชันช่วยเหลือทางกลับบ้ายด้วยคิวอาร์โค้ด

ตารางที่ 3.1 แสดงคำอธิบายยุสเคส : LOGIN

| | |
|----------------------|--|
| Project : | แอปพลิเคชันช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ด |
| Use Case Name: | Login |
| Actor: | Admin |
| Use Case Referenced: | Login |
| Basic Flow: | <ol style="list-style-type: none"> 1. Use case จะเริ่มเมื่อผู้ใช้งานเปิดแอปพลิเคชัน 2. แสดงหน้าเข้าสู่ระบบ 3. ทำการเข้าสู่ระบบ เพื่อใช้งานแอปพลิเคชัน 4. ยุสเคสสิ้นสุดการทำงาน |
| Alternate Flow : | - |
| Pre-Condition(s): | 1.แอดมินต้องมีชื่อในระบบ |
| Post Condition(s): | 1.เข้าสู่หน้าเมนูหลัก |

ตารางที่ 3.2 แสดงคำอธิบายสูญสคेट: QR-CODE CHECKING

| | |
|----------------------|--|
| Project : | แอปพลิเคชันช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ด |
| Use Case Name: | Qr-code checking |
| Actor: | Admin |
| Use Case Referenced: | Qr-code checking |
| Basic Flow: | <p>1. ทำการเช็คว่าข้อมูลในQrcode ที่สแกนนั้นมีข้อมูลตรงกับฐานข้อมูลหรือไม่</p> <p>2. ถ้าผ่านสามารถดำเนินการต่อได้</p> <p>3. Use case สิ้นสุด</p> |
| Alternate Flow : | - |
| Pre-Condition(s): | <p>1. Use case จะเริ่มเมื่อตัวผู้เยี่ยมชมนั้นเข้าใช้งาน Qrcodescaner</p> <p>2. Use case เป็นแบบออโต้จะไม่ทำงานหากไม่มีการสแกนเกิดขึ้น</p> |
| Post Condition(s): | 1. ไปหน้าใส่ข้อมูลสำหรับผู้เยี่ยมชม |

ตารางที่ 3.3 แสดงคำอธิบายยุสเคส: MANAGE USER

| | |
|----------------------|--|
| Project : | แอปพลิเคชันช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ด |
| Use Case Name: | Manage User |
| Actor: | admin |
| Use Case Referenced: | Manage User |
| Basic Flow: | <ol style="list-style-type: none"> 1. Use case จะเริ่มเมื่อผู้ดูแลต้องการจัดการแก้ไขข้อมูล 2. ผู้ดูแลระบบ เลือกเมนู แก้ไข เพิ่ม ลบ 3. สามารถเลือกแก้ไข เพิ่ม หรือลบ ข้อมูลของผู้ป่วยได้ 3. ยุสเคสสิ้นสุดการทำงาน |
| Alternate Flow : | - |
| Pre-Condition(s): | <ol style="list-style-type: none"> 1. แอดมินล็อคอินผ่านแล้ว |
| Post Condition(s): | <ol style="list-style-type: none"> 1. แสดงหน้าเมนู จัดการข้อมูล 2. แสดงหน้าเมนู เลือกเมนู เพิ่ม ลบ แก้ไข 3. แสดงข้อมูลของผู้ป่วยที่สามารถ เพิ่ม ลบ แก้ไขได้ |

ตารางที่ 3.4 แสดงคำอธิบายยุสเคส: PUSH NOTIFICATION

| | |
|----------------------|---|
| Project : | แอปพลิเคชันช่วยเหลือทางกลับบ้านด้วยคิวอาร์โค้ด |
| Use Case Name: | Push Notifitcation |
| Actor: | admin |
| Use Case Referenced: | Push Notifitcation |
| Basic Flow: | <ol style="list-style-type: none"> 1. Use case จะเริ่มเมื่อมีการร้องขอเข้าถึงข้อมูล 2. ตัวระบบแจ้งเตือน จะแจ้งว่ามีการร้องขอเข้าถึงข้อมูล ไปยัง 3. ยุสเคสสิ้นสุดการทำงาน |
| Alternate Flow : | - |
| Pre-Condition(s): | <ol style="list-style-type: none"> 1. ตัวผู้เยี่ยมชมได้กรอกข้อมูลแล้ว |
| Post Condition(s): | <ol style="list-style-type: none"> 1. มีการแจ้งเตือนว่าได้รับสิทธิหรือไม่ |

ตารางที่ 3.5 แสดงคำอธิบายยุสเคส: REGISTER GUEST DATA

| | |
|----------------------|---|
| Project : | แอปพลิเคชันช่วยเหลือทางกลับบ้านด้วยคิวอาร์โค้ด |
| Use Case Name: | Register Guest data |
| Actor: | Guest |
| Use Case Referenced: | Register Guest data |
| Basic Flow: | <ol style="list-style-type: none"> 1. Use case เริ่มโดยการให้ผู้เยี่ยมชมกรอกข้อมูล 2. ผู้เยี่ยมชม กรอก เบอร์โทรศัพท์และมีการถ่ายรูปกับผู้ป่วย 3. ยุสเคสเสร็จสิ้นการทำงาน |
| Alternate Flow : | - |
| Pre-Condition(s): | <ol style="list-style-type: none"> 1. ต้องผ่านหน้าต่าง qrcode scan และ |
| Post Condition(s): | <ol style="list-style-type: none"> 1. มีการไปยังหน้าถัดไป 2. ข้อมูลขึ้นใน Record |

ตารางที่ 3.6 แสดงหน้าคำอธิบายยูสเคส: QR-CODE GENERATE

| | |
|----------------------|--|
| Project : | แอปพลิเคชันช่วยเหลือทางกลับบ้านด้วยคิวอาร์โค้ด |
| Use Case Name: | Qr-code Generate |
| Actor: | Admin |
| Use Case Referenced: | Qr-code Generate |
| Basic Flow: | <p>1. Use case จะเริ่มเมื่อแอดมินเปิดหน้าต่าง Qr-code Generate</p> <p>2. มีรายการเลือกในการเลือกสำหรับคนไข้ที่การต้องเจน qr-code ในการใช้ในการทำป้ายประจำตัว ผู้ป่วย</p> <p>3. ยูสเคสสิ้นสุด</p> |
| Alternate Flow : | - |
| Pre-Condition(s): | 1. มีผู้ป่วยในระบบ |
| Post Condition(s): | 1. มีการเจน Qrcode เกิดขึ้น |

ตารางที่ 3.7 แสดงคำอธิบายยูสเคส: QR CODE SCANNER

| | |
|----------------------|---|
| Project : | แอปพลิเคชันช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ด |
| Use Case Name: | Qrcode Scanner |
| Actor: | guest |
| Use Case Referenced: | Qrcode Scanner |
| Basic Flow: | <ol style="list-style-type: none"> 1. Use case จะเริ่มเมื่อผู้เยี่ยมชมเข้าหน้าไซต์ 2. กดเข้าบุํฟผู้เยี่ยมชม 3. แสดงหน้าสแกน QR CODE 4. ดำเนินการสแกน 5. รอการดำเนินการจากเจ้าหน้าที่ 6. ยูสเคสสิ้นสุดการทำงาน |
| Alternate Flow : | - |
| Pre-Condition(s): | - |
| Post Condition(s): | <ol style="list-style-type: none"> 1. เข้าหน้าสแกน 2. มีการสแกน |

ตารางที่ 3.8 แสดงคำอธิบายยูสเคส : VIEW USER DATA

| | |
|----------------------|--|
| Project : | แอปพลิเคชันช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ด |
| Use Case Name: | View user Data |
| Actor: | Guest |
| Use Case Referenced: | View user Data |
| Basic Flow: | <ol style="list-style-type: none"> 1. Use case จะเริ่มเมื่อได้รับการยืนยันจากผู้ดูแลแล้ว 2. แสดงหน้าต่างข้อมูลส่วนตัวผู้ใช้งาน 3. ยุสเคสสิ้นสุดการทำงาน |
| Alternate Flow : | - |
| Pre-Condition(s): | <ol style="list-style-type: none"> 1. ผู้เยี่ยมชมต้องทำการสแกน QR CODE ผ่าน 2. ผู้ใช้งานเข้าสู่ระบบ |
| Post Condition(s): | <ol style="list-style-type: none"> 1. แสดงหน้าจอข้อมูลของผู้ใช้งาน |

ตารางที่ 3.9 แสดงคำอธิบายยูสเคส : View Map

| | |
|----------------------|---|
| Project : | แอปพลิเคชันช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ด |
| Use Case Name: | View Map |
| Actor: | guest |
| Use Case Referenced: | View Map |
| Basic Flow: | <ol style="list-style-type: none"> 1. Use case จะเริ่มเมื่อได้ดูข้อมูลส่วนตัวของผู้ใช้งานแล้ว 2. แสดงปุ่มสำหรับนำทางกลับบ้าน 3. แสดงหน้าจอ Google Map และนำทาง 4. ยูสเคสสิ้นสุดการทำงาน |
| Alternate Flow : | - |
| Pre-Condition(s): | <ol style="list-style-type: none"> 1. ผู้เยี่ยมชมต้องผ่านการยืนยันจากเจ้าหน้าที่ |
| Post Condition(s): | <ol style="list-style-type: none"> 1. แสดงหน้าต่าง Google Map 2. สามารถนำทางได้จริง |

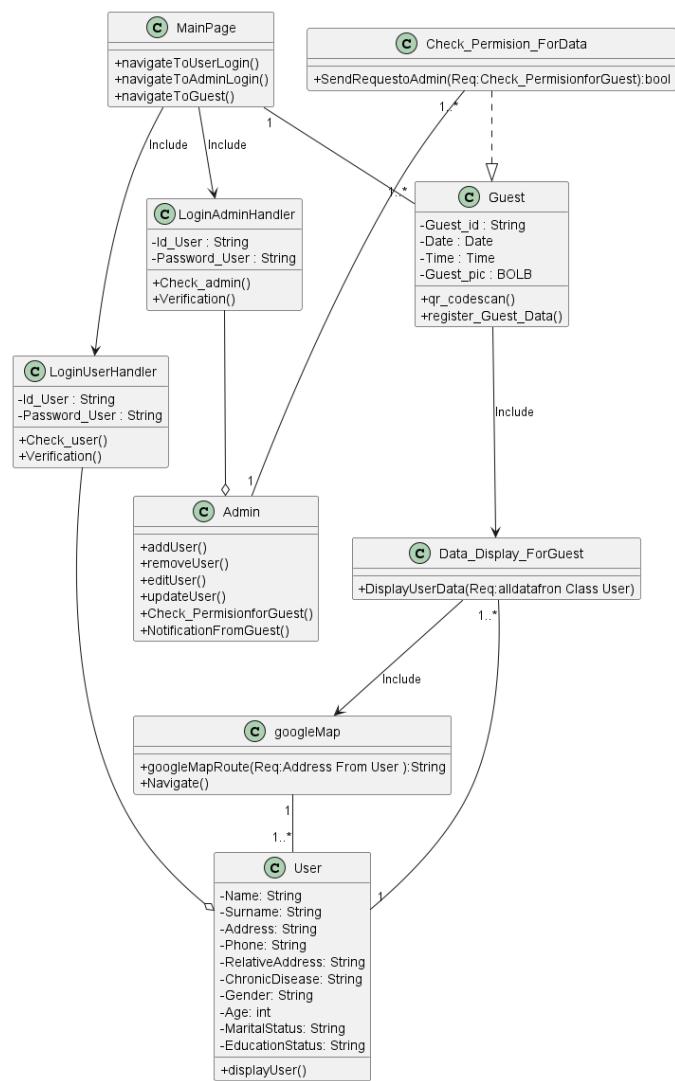
ตารางที่ 3.10 แสดงคำอธิบายยุสเคส : Wait for Permission

| | |
|----------------------|---|
| Project : | แอปพลิเคชันช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ด |
| Use Case Name: | Wait for Permission |
| Actor: | Guest , Admin |
| Use Case Referenced: | Wait for Permission |
| Basic Flow: | <ol style="list-style-type: none"> 1. Use case จะเริ่มเมื่อได้ผู้เยี่ยมชมสแกนคิวอาร์โค้ด 2. แสดงจอรอดำเนินการ 3. ยูสเคสสิ้นสุดการทำงาน |
| Alternate Flow : | - |
| Pre-Condition(s): | 1. ผู้เยี่ยมชมใช้งานสแกนคิวอาร์โค้ด |
| Post Condition(s): | 1. แสดงหน้าต่าง รอ |

ตารางที่ 3.11 แสดงคำอธิบายยูสเคส : Permission Denied

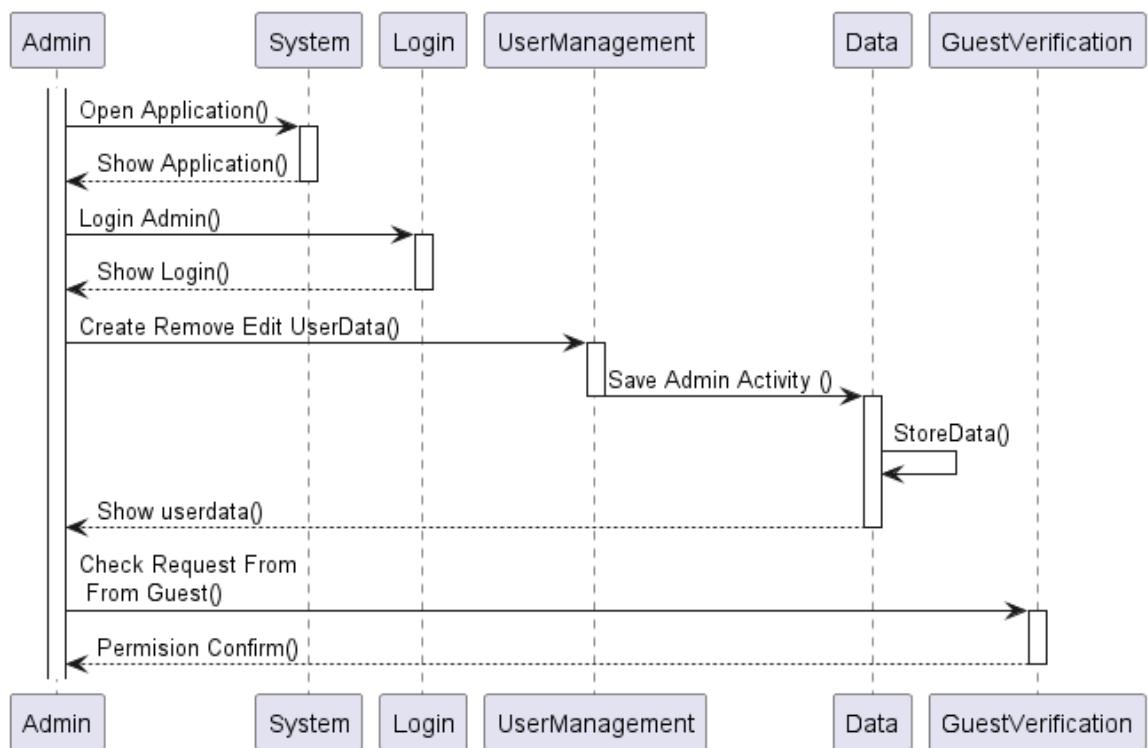
| | |
|----------------------|---|
| Project : | แอปพลิเคชันช่วยหาเส้นทางกลับบ้านด้วยคิวอาร์โค้ด |
| Use Case Name: | Permission Denied |
| Actor: | Admin |
| Use Case Referenced: | Permission Denied |
| Basic Flow: | <ol style="list-style-type: none"> 1. Use case จะเริ่มเมื่อผู้ดูแลระบบไม่ยืนยันการขอเข้าดูข้อมูลส่วนตัว 2. มีการยกเลิกการร้องขอข้อมูล 3. ยูสเคสสิ้นสุดการทำงาน |
| Alternate Flow : | - |
| Pre-Condition(s): | <ol style="list-style-type: none"> 1. มีการร้องขอเข้าดูข้อมูลผู้ป่วย |
| Post Condition(s): | <ol style="list-style-type: none"> 1. มีการยกเลิกเข้าถึงข้อมูล |

3.3 คลาสโดยแกรม (class diagram)

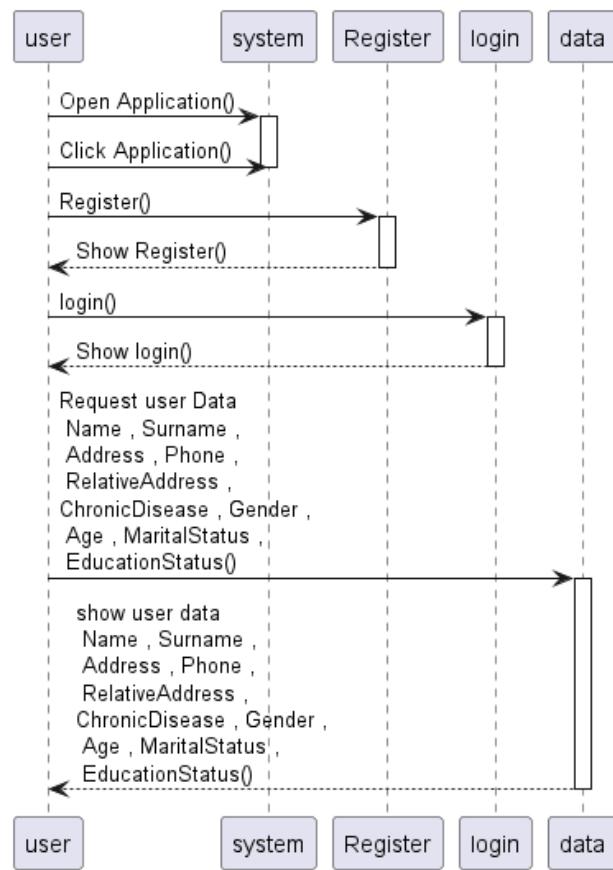


ภาพที่ 3.2 ภาพตัวอย่าง คลาสโดยแกรมของแอปพลิเคชัน ช่วยทางกลับบ้านด้วยคิวอาร์โค้ด

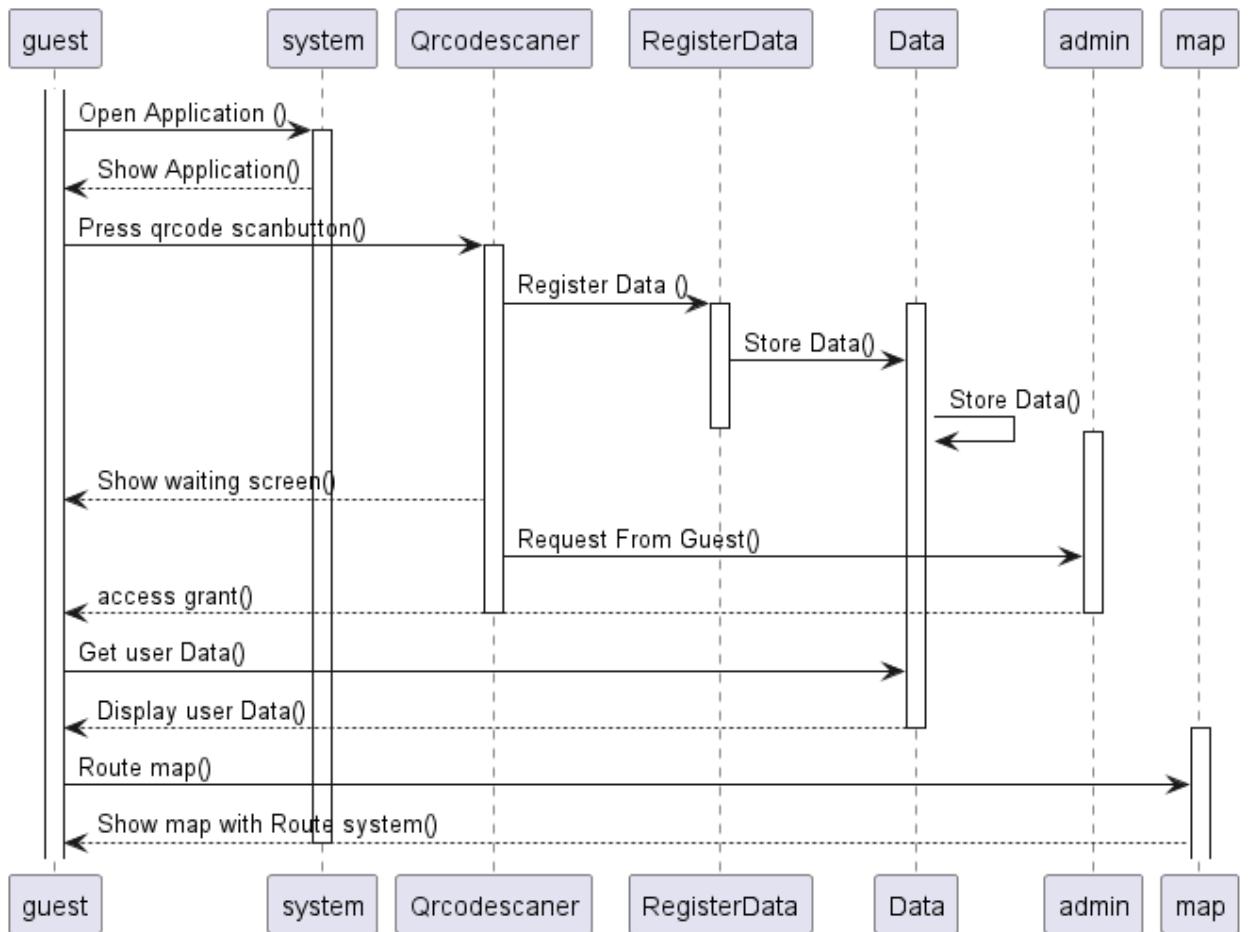
3.4 ซีเควนซ์โดยแกรม (Sequence Diagram)



ภาพที่ 3.3 ภาพซีเควนซ์โดยแกรม แออดมิน (ADMIN) ของแอปพลิเคชันช่วยเหลือทั้งบ้านด้วย
คิวอาร์โค้ด

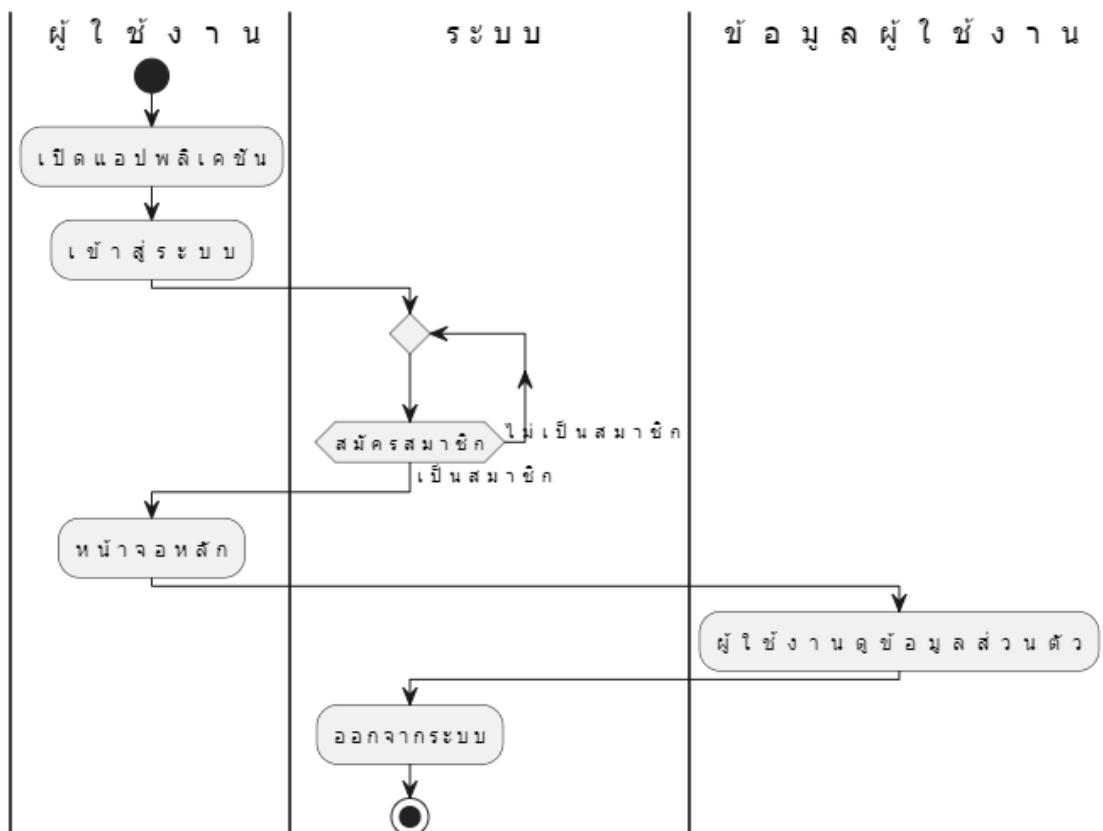


ภาพที่ 3.4 ภาพชีวิตรุ่นชี้ไดอะแกรม ยูซเซอร์ (USER) ของแอปพลิเคชันช่วยทางกลับบ้านด้วยคิวอาร์โค้ด

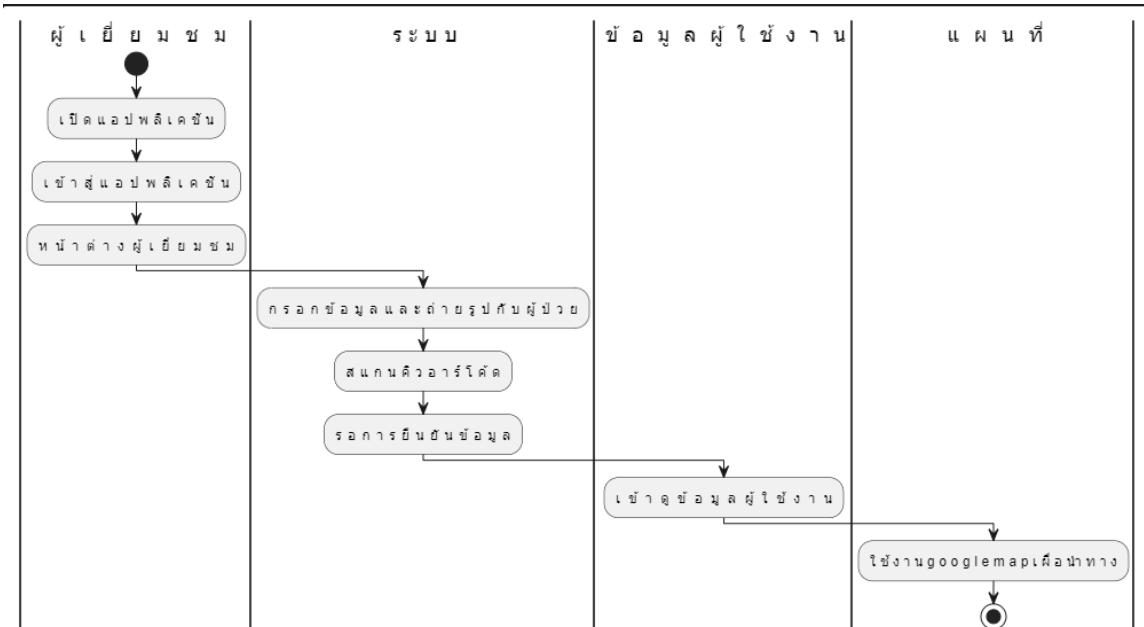


ภาพที่ 3.5 ภาพซีเควนซ์โดยแกรม ผู้เยี่ยมชม (GUEST) ของแอปพลิเคชันช่วยทางกลับบ้าน ด้วยคิวอาร์โค้ด

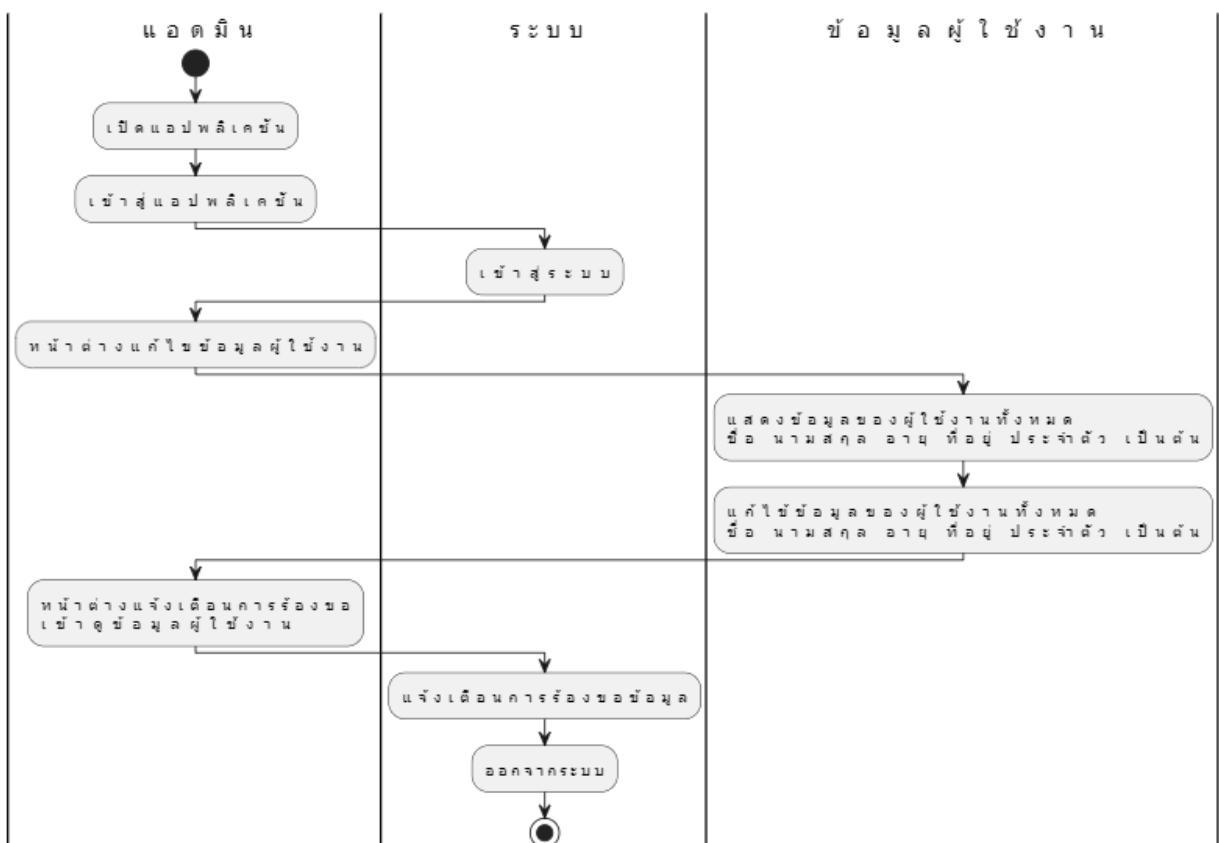
3.5 แอคทิวิตี้ ไดอะแกรม (Activity Diagram)



ภาพที่ 3.6 ภาพแอคทิวิตี้ ไดอะแกรม ผู้ใช้งาน (USER) ของแอปพลิเคชันช่วยทางกลับบ้านด้วยคิวอาร์โค้ด

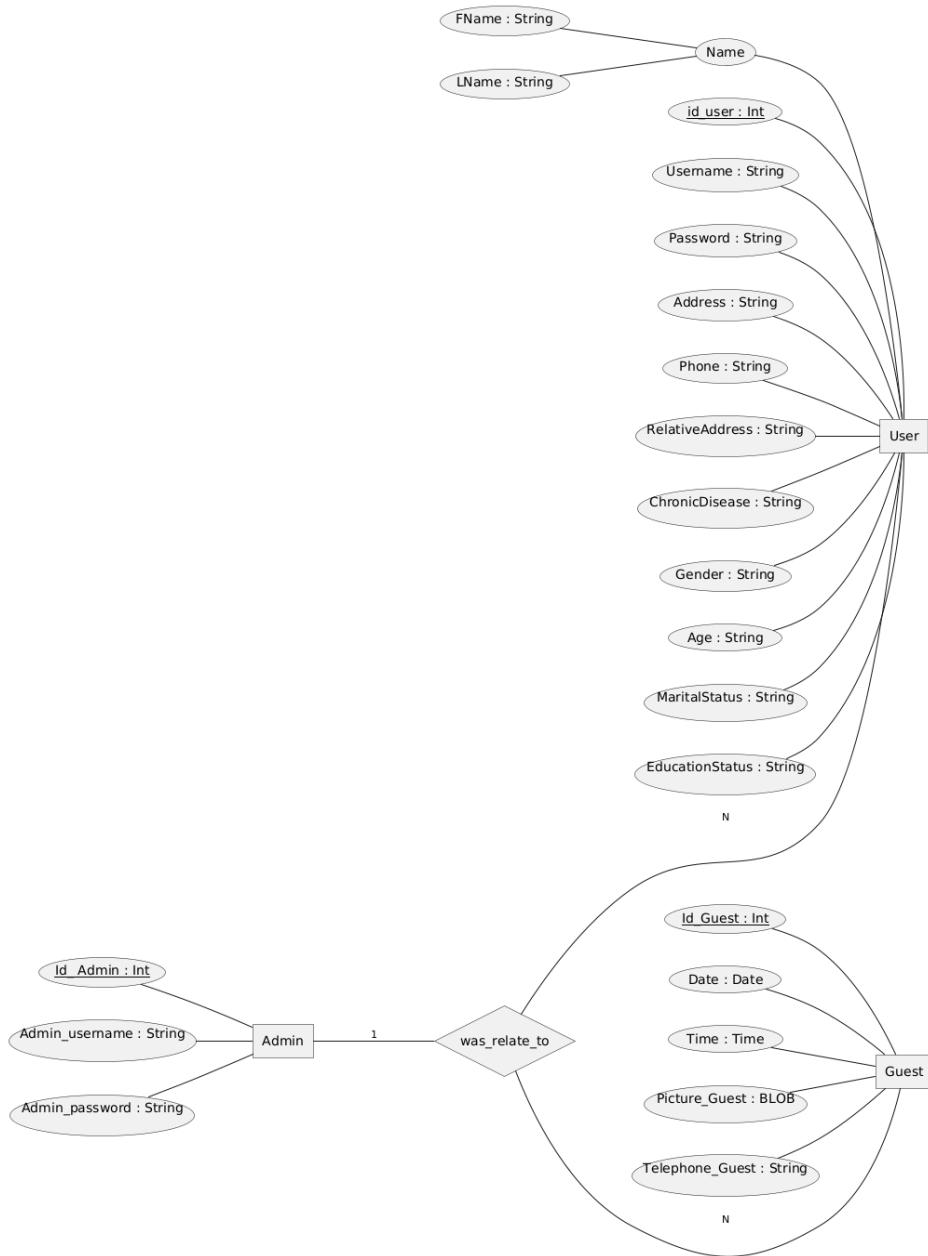


ภาพที่ 3.7 ภาพแอคทิวิตี้ ไดอะแกรม ผู้เยี่ยมชม (GUEST) ของแอปพลิเคชันช่วยทางกลับบ้าน ด้วยคิวอาร์โค้ด



ภาพที่ 3.8 ภาพแอคทิวิตี้ ไดอะแกรม แออดมิน (ADMIN) ของแอปพลิเคชันช่วยเหลือทางกลับบ้าน ด้วยคิวอาร์โค้ด

3.6 อี-อาร์ ไดอะแกรม (Er - Diagram)



ภาพที่ 3.9 ภาพอี-อาร์ ไดอะแกรมของ ยูสเซอร์ทั้งหมด ของแอปพลิเคชันช่วยเหลือทั้งบ้าน
ด้วยคิวอาร์โค้ด

3.7 ออกแบบฐานข้อมูลเชิงกายภาพ (Physical Design)

ตารางที่ 3.14 ตารางแสดงลำดับเพิ่มข้อมูล

| แหล่งข้อมูล | ชื่อเพิ่มข้อมูล ภาษาอังกฤษ | ชื่อเพิ่มข้อมูล ภาษาไทย | ชนิด | คำอธิบาย |
|-------------|-------------------------------|----------------------------|-------------|---------------------------|
| D1 | User | ผู้ใช้งาน | Master File | เก็บข้อมูลผู้ใช้งาน |
| D2 | Admin | ผู้ดูแลระบบ | Master File | เก็บข้อมูลผู้ดูแล ระบบ |
| D3 | Guest | ผู้เยี่ยมชม | Master File | เก็บข้อมูลผู้เยี่ยม ชม |

แหล่งข้อมูล : D1

ชื่อแฟ้มข้อมูล : User

ชนิดแฟ้มข้อมูล : Master File

คำอธิบายแฟ้ม : เก็บข้อมูลผู้ใช้งาน

ตารางที่ 3.15 แสดงตาราง USER

| ชื่อแອทีบีวีต์ | ชนิดข้อมูล | ขนาด (ไบต์) | รูปแบบ | ช่วงข้อมูล | ป้อมข้อมูล (Y/N) | คีย์หลัก หรือ คีย์นอก | ตารางอ้างอิง |
|----------------|------------|----------------|---------|-------------------|---------------------|-----------------------------|--------------|
| User_ID | Varchar | 7 | NNNNNNX | a-z,0-9 | Y | PK | |
| FName | Varchar | 30 | X(30) | A-Z, a-z, ก-ฮ, | Y | | |
| LName | Varchar | 30 | X(30) | A-Z, a-z, ก-ฮ | Y | | |
| Username | Varchar | 30 | X(30) | A-Z, a-z, | Y | | |

| | | | | | | | |
|---------------------|---------|-----|--------|----------------------|---|--|--|
| | | | | ଫ-ୟ,୦-୨ | | | |
| Password | Varchar | 30 | X(30) | A-Z, a-z, 0-9 | Y | | |
| Address | Varchar | 100 | X(100) | A-Z, a-z, ଫ-ୟ,୦-୨ | Y | | |
| Phone | Varchar | 12 | N(12) | 0-9 | Y | | |
| Relative Address | Varchar | 100 | X(100) | A-Z, a-z, ଫ-ୟ, | Y | | |
| ChronicDisease | Varchar | 30 | X(30) | A-Z, a-z, ଫ-ୟ, | Y | | |
| Gender | Varchar | 30 | X(30) | A-Z, a-z, ଫ-ୟ, | Y | | |
| Age | Integer | 3 | NNN | 0-9 | Y | | |
| MaritalStatus | Varchar | 30 | X(30) | A-Z, a-z, | Y | | |

| | | | | | | | |
|-----------------|---------|----|-------|-------------------|---|--|--|
| | | | | ก-ฮ, | | | |
| EducationStatus | Varchar | 30 | X(30) | A-Z, a-z, ก-ฮ, | Y | | |

ตารางที่ 3.16 ตารางแสดงตัวอย่างของ USER

| USER_ID | FNAME | LNAME | USERNAME | PASSWORD | ADDRESS | PHONE | RELATIVE ADDRESS | CHRONICDI SEASE | GENDER | AGE | EDUCATION STATUS |
|-------------|-------|----------|----------|-----------|---------------------|----------------|------------------|-----------------|--------|-----|------------------|
| 000001 H | จิตดี | ศรีสว่าง | JIDEE99 | 86425YY | ที่ อยู่... ฯ | 0863576 209 | ที่อยู่... ฯ | - | FEMALE | 26 | ปริญญาตรี |
| 000002 H | ทิศ | ชายตรง | THID9966 | You896632 | ที่ อยู่... ฯ | 0873556 209 | ที่อยู่... ฯ | ความดันสูง | MALE | 60 | |

แหล่งข้อมูล : D2

ชื่อแฟ้มข้อมูล : Admin

ชนิดแฟ้มข้อมูล : Master File

คำอธิบายแฟ้ม : เก็บข้อมูลผู้ดูแลระบบ

| ชื่อแอทริบิวต์ | ชนิดข้อมูล | ขนาด (ไบต์) | รูปแบบ | ช่วงข้อมูล | ป้องข้อมูล (Y/N) | คีย์หลัก | ตารางอ้างอิง |
|----------------|------------|----------------|--------|----------------------|---------------------|----------|--------------|
| Id_Admin | Varchar | 5 | NNNXX | a-z,0-9 | Y | PK | |
| Admin_Username | Varchar | 30 | X(30) | A-Z, a-z, ก-ჟ,0-9 | Y | | |
| Admin_Password | Varchar | 30 | X(30) | A-Z, a-z, ก-ჟ,0-9 | Y | | |

ตารางที่ 3.17 ตารางแสดงตัวอย่าง ADMIN

| ID_ADMIN | ADMIN_USERNAME | ADMIN_PASSWORD |
|----------|----------------|----------------|
| 001NH | KIM90006 | ILEOFFF236 |
| 002NH | LIM5864 | CIM66332 |

แหล่งข้อมูล : D3

ชื่อแฟ้มข้อมูล : Guest

ชนิดแฟ้มข้อมูล : Master File

คำอธิบายแฟ้ม : เก็บข้อมูลผู้เยี่ยมชม

| ชื่อแอทริบิวต์ | ชนิดข้อมูล | ขนาด (ไบต์) | รูปแบบ | ช่วงข้อมูล | ป้อมข้อมูล (Y/N) | คีย์หลัก หรือ คีย์นอก | ตารางอ้างอิง |
|----------------|------------|----------------|----------|------------|---------------------|-----------------------------|--------------|
| Id_Guest | Varchar | 7 | XXXXXXXX | a-z,0-9 | Y | PK | |

| | | | | | | | |
|-----------------|---------|-------|----------|---------|---|--|---|
| Telephone_guest | Varchar | 10 | X(10) | 0-9 | Y | | |
| Picture_guest | BLOB | 18000 | X(18000) | a-z,0-9 | Y | | ใช้วิธีการแปลง เลขฐานสอง เป็นรูปภาพ |
| Date | Date | 10 | X(10) | 0-9 | Y | | |
| Time | Varchar | 10 | X(10) | 0-9 | Y | | |

ตารางที่ 3.17 ตารางแสดงตัวอย่าง GUEST

| ID_GUEST | Telephone_guest | Picure_guest | DATE | TIME |
|----------|-----------------|--------------|------------|-------|
| 0000001 | 0963579791 | 100100011110 | 12/2/2564 | 10:20 |
| 0000002 | 0855528641 | 100100011101 | 26/12/2564 | 21:00 |