ISTEP

ระบบจดบันทึกรายรับรายจ่าย

MONEY MOVEMENT

ชื่อของ GitHub repository

money_movement

นายพัสกร จุลพล 54010907

นายสุรพงศ์ เท่าเทียมตน 54011423

Object-oriented Analysis and Design

ภาคการศึกษาที่ 1/ 2557

สารบัญ

บทคัดย่อ

การออมเงินเป็นสิ่งสำคัญเพราะการทำธุรกรรมต่างๆ เช่น การลงทุน การเริ่มต้นเปิดกิจการ การซื้อบ้าน การซื้อรถยนต์ ต้องใช้เงินจำนวนมาก ดังนั้นหากมีเงินเหลือจากการใช้จ่ายในชีวิตประจำวันจะทำให้มีเงินออมมาขึ้นตามไปด้วย ซึ่งเป็นที่มาว่าหากสามารถสร้างตัวช่วยในการออมขึ้นมาก็จะมีประโยชน์ให้ผู้ใช้ จึงเป็นเหตุผลที่ทำ Money Movement ขึ้นมา Money Movement เป็น web application เกี่ยวกับการจดบันทึกรายรับรายจ่ายของผู้ใช้เพื่อให้ผู้ใช้สามารถรู้ว่าตนเองใช้เงินไปกับอะไรบ้างมีจำนวนสัดส่วนอย่างไร จะได้นำไปใช้ประกอบการตัดสินใจในการลดรายจ่ายหรือหาวิธีเพิ่มรายได้เพื่อให้สามารถออมเงินได้มากขึ้น

บทนำและแรงจูงใจ

การออมเงินเป็นสิ่งสำคัญเพราะการทำธุรกรรมต่างๆ เช่น การลงทุน การเริ่มต้นเปิดกิจการ การซื้อบ้าน การซื้อรถยนต์ ฯลฯ ต้องใช้เงินเป็นจำนวนมาก ซึ่งหากขาดการออมที่ดีจะทำให้ไม่สามารถทำสิ่งทีว่ามาเหล่านี้ได้โดยง่ายซึ่งอาจจะต้องทำการกู้เงินในจำนวนที่มากขึ้นทำให้เกิดปัญหาหนี้สินตามมา ซึ่งเป็นที่มาว่าหากสามารถสร้างตัวช่วยในการออมขึ้นมาก็จะมีประโยชน์ให้คนจำนวนมากได้ ดังนั้นจึงเป็นเหตุผลที่ทำ Money Movement ขึ้นมา

Money Movement เป็น web application เกี่ยวกับการจดบันทึกรายรับรายจ่ายของผู้ใช้เพื่อให้ผู้ใช้สามารถรู้ว่าตนเองใช้เงินไปกับอะไรบ้างมีจำนวนสัดส่วนอย่างไร จะได้นำไปใช้ประกอบการตัดสินใจในการลดรายจ่ายหรือหาวิธีเพิ่มรายไดhเพื่อให้สามารถออมเงินได้มากขึ้น โดยสิ่งที่ Money Movement ทำได้มีดังนี้

1. สามารถเก็บรายรับและรายจ่ายตามผู้ใช้กรอกได้โดยสามารถแบ่งเป็นหมวดหมู่ตามที่ผู้ใช้กำหนดหมวดหมู่ขึ้นมา
2. สามารถแสดงรายละเอียดรายรับรายจ่ายของผู้ใช้ออกมาในลักษณะต่างๆ เช่น กราฟ แผนภูมิวงกลม เป็นต้น
3. สามารถนำข้อมูลรายรับรายจ่ายมาวิเคราะห์ว่ามีแนวโน้มการเพิ่มขึ้นหรือลดลงของรายรับรายจ่ายในหมวดหมู่ใดในกรอบเวลาที่กำหนด
4. สามารถเปรียบเทียบค่าใช้จ่ายของตนเองกับค่า average ของคนอื่นๆ ที่อยู่ระบบ หรือคนใน กลุ่ม/ประเภทเดียวกัน เช่น ช่วงอายุ อาชีพ

วิธีการใช้งาน Money Movement

1. ผู้ใช้ต้องสมัครสมาชิกกับตัว web application
2. ผู้ใช้ต้องกำหนดหมวดหมู่รายรับรายจ่ายของตนว่ามีหมวดหมู่อะไรบ้างตามความต้องการของผู้ใช้
3. ผู้ใช้ต้องกรอกรายรับรายจ่ายของตัวเองลงในข่องว่าอยู่ในหมวดหมู่อะไรให้ผ่านหน้า Web application
4. เมื่อผู้ใช้อยากดูรายละเอียดรายรับรายจ่ายของผู้ใช้ ผู้ใช้ก็กำหนดรอบเวลาและรูปแบบการแสดงผลที่ผู้ใช้อยากจะทราบลงไป web application ก็จะแสดงรายละเอียดออกมาในรูปแบบที่ต้องการ

5. เมื่อผู้ใช้อยากจะทำการวิเคราะห์แนวโน้มของรายรับรายจ่ายก็เลือกช่วงเวลาที่ต้องการเช่น รายวัน รายสัปดาห์ รายเดือน จากนั้น web application จะแสดงแนวโน้มออกมาเป็นกราฟว่ามีอะไรกับเพิ่มขึ้นหรือลดลง ในช่วงเวลานั้นมีเงินเก็บหรือไม่

งานที่เกี่ยวข้องหรือใกล้เคียง

- MINT เป็น web application ที่สามารถบันทึกรายรับรายจ่ายได้ สามารถทำงานได้หลากหลาย สามารถตั้งเป้าหมายการเก็บเกินได้ว่าจะเก็บเงินไปทำอะไร หรือ สามารถกำหนดเป้าหมายรายจ่ายได้ว่าจะใช้เงินกับส่วนนี้จำนวนเท่าไหร่

- Spendee เป็น application บนสมาร์ทโฟน (smartphone) ที่สามารถบันทึกรายรับรายจ่ายได้ โดยในส่วนของรายรับรายจ่ายที่ทำการบันทึกจะสามารถเลือกประเภทของที่มาได้ ซึ่งได้แบ่งเป็นหมวดหมู่ เช่น ค่ารถ ค่าท่องเที่ยว ค่าพักผ่อน เงินเดือน เป็นต้น และสามารถแสดงกราฟผลการใช้จ่ายออกเป็นรายอาทิตย์ รายเดือน รายไตรมาส รายปี

ซึ่งสิ่งที่ web application ทำได้เหมือนกันคือ สามารถบันทึกรายรับรายจ่ายได้ สามารถตั้งเป้าหมายในการใช้ได้ เลือกประเภทของที่มาโดยจัดเป็นหมวดหมู่ สามารถแสดงผลการใช้จ่ายต่างๆ ออกเป็นรายอาทิตย์ รายเดือน รายปีได้ สิ่งที่แตกต่างคือ Money Movement สามารถเปรียบเทียบข้อมูลระหว่างรายรัยรายจ่ายตัวผู้ใช้กับค่าเฉลี่ยของผู้ใช้ในกลุ่มเดียวกัน เช่น ช่วงอายุ อาชีพ รายได้

ผลการวิเคราะห์ความต้องการของผู้ใช้ระบบ

Functional requirements

1. ระบบสามารถให้ผู้ใช้ทำการสมัครสมาชิกและเข้าสู่ระบบได้
2. ระบบสามารถให้ผู้ใช้สามารถบันทึกรายรับ-รายจ่ายของตนได้
3. ระบบสามารถนำข้อมูลรายรับรายจ่ายของผู้ใช้มาแสดงผลออกมาเป็นรูปแบบต่างๆได้เช่น ตาราง กราฟวงกลม กราฟเส้น เป็นต้น
4. ระบบสามารถวิเคราะห์ข้อมูลข้อมูลรายรับรายจ่ายของผู้ใช้เพื่อหาแนวโน้มการเพิ่มขึ้นหรือลดลงของรายรับรายจ่ายในหมวดหมู่ต่างๆ

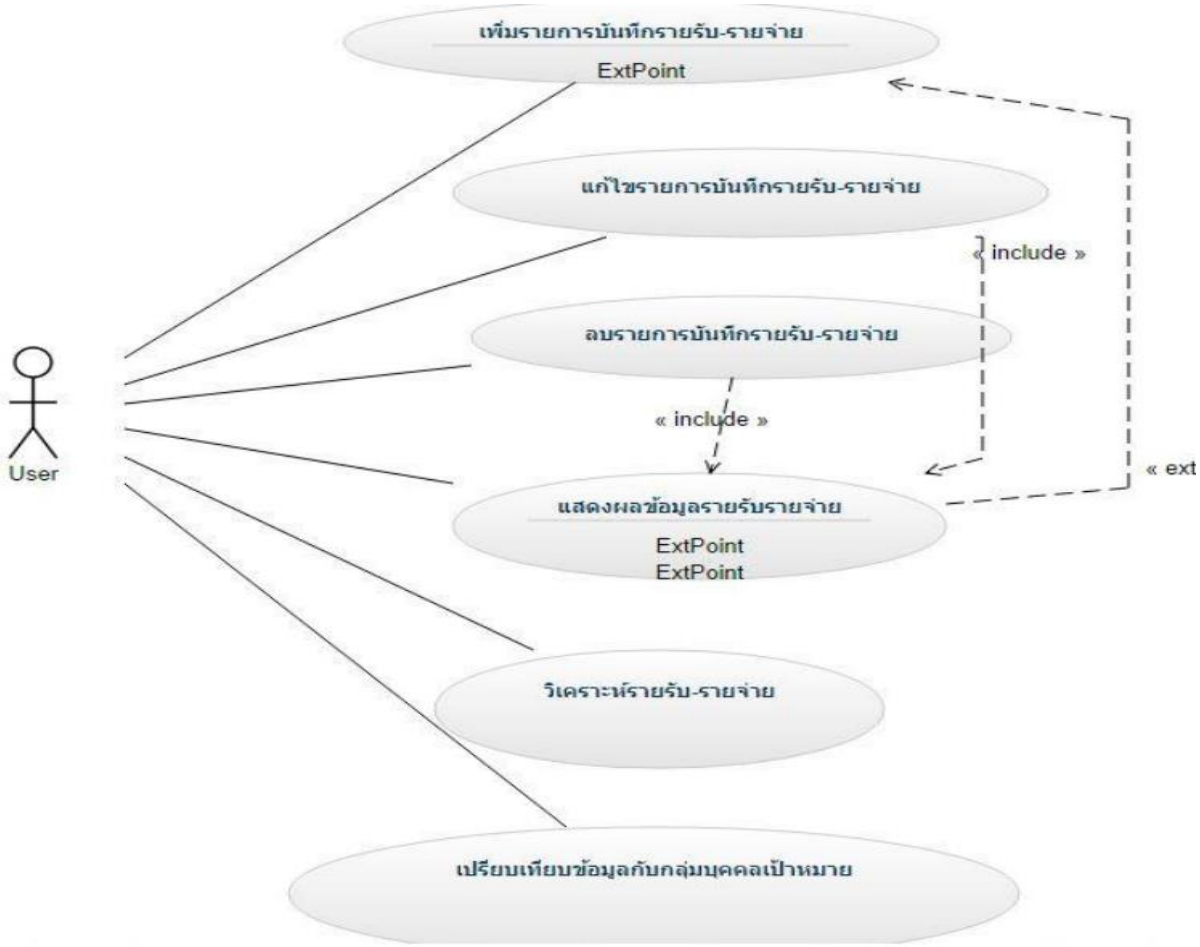
5. ระบบสามารถนำข้อมูลรายรับรายจ่ายของผู้ใช้มาเปรียบเทียบกับค่าเฉลี่ยรายรับรายจ่ายของผู้ใช้ที่อยู่ในกลุ่มเดียวเช่น ช่วงอายุ อาชีพ รายได้

Non-functional requirements

1. ระบบมีความสามารถในการให้บริการ (availability) 85 %
2. กราฟที่แสดงให้ผู้ใช้เห็นนั้นเป็นกราฟที่มีความสวยงามและทำให้เข้าใจง่าย
3. แบ่งหมวดหมู่ค่าใช้จ่ายและรายได้อย่างชัดเจนเข้าใจง่าย
4. การเพิ่มข้อมูลทำได้ง่าย เช่น มีการเติมข้อมูลให้แบบอัตโนมัติ (Auto complete)
5. ตอบสนองผู้ใช้ภายในเวลาไม่เกิน 2 นาที (กรณีที่ความเร็ว Internet ผู้ใช้นั้นเร็วในระดับมาตรฐาน)

แนวทางการใช้งาน

Use case diagram



Use case specification (1)

Use case name: เปรียบเทียบข้อมูลกับกลุ่มบุคคลเป้าหมาย

Use case purpose: ในส่วนนี้ผู้ใช้สามารถที่จะเปรียบเทียบข้อมูลการใช้จ่ายของตนเองกับกลุ่มบุคคลที่ช่วงอายุ อาชีพ

หรือรายได้ใกล้เคียงกัน เพื่อให้ผู้ใช้รู้ถึงศักยภาพในการออมของผู้ใช้เองเมื่อเทียบกับกลุ่มบุคคลดังที่กล่าวมาว่าเป็นอย่างไรบ้าง

Preconditions: ผู้ใช้จะต้องมีข้อมูลการบันทึกรายการต่างๆ ในช่วงเวลาที่ท าการเปรียบเทียบ

Postconditions: ในการแสดงผลการเปรียบเทียบจะต้องสามารถแสดงผลในรูปของกราฟได้หรืออาจจะส่งออก

(Export) ข้อมูลออกมาเป็นไฟล์ (file) นามสกุล .pdf ได้

Limitations: ในการเปรียบเทียบข้อมูลนั้นจะเป็นการเปรียบเทียบข้อมูลกับกลุ่มเป้าหมายที่มีเฉพาะภายในฐานข้อมูล

เท่านั้น

Assumptions: ภายในฐานข้อมูลมีกลุ่มผู้ใช้กลุ่มเป้าหมายในปริมาณมาก

Primary scenario:

M1: ผู้ใช้งานทำการเลือกประเภทกลุ่มเป้าหมายที่จะเปรียบเทียบด้วย

M2: ผู้ใช้งานทำการเลือกช่วงเวลาที่ต้องการที่จะทำการเปรียบเทียบข้อมูลรายรับ-รายจ่าย

M3: ระบบทำการเปรียบเทียบข้อมูลการใช้จ่ายระหว่างผู้ใช้กับกลุ่มบุคคลเป้าหมาย

M4: ระบบแสดงผลการเปรียบเทียบออกมาในรูปของกราฟ

M5: ถ้าผู้ใช้ต้องที่จะทำการส่งออกข้อมูลการเปรียบเทียบเป็นไฟล์ (file) นามสกุล .pdf

Alternate scenarios:

Condition: ผู้ใช้ไม่เลือกประเภทของกลุ่มบุคคลเป้าหมายที่จะทำการเปรียบเทียบ

D1: มีข้อความแจ้งเตือนให้ทราบว่าไม่ได้ทำการเลือกกลุ่มบุคคลเป้าหมาย

D2: กลับไปยัง M1 ให้ผู้ใช้ทำการเลือกประเภทของกลุ่มบุคคลเป้าหมายที่จะเปรียบเทียบใหม่ Condition: ผู้ใช้เลือกช่วงเวลาของการเปรียบเทียบไม่ถูกต้อง

D1: มีข้อความแจ้งเตือนให้ทราบว่าเลือกช่วงเวลาไม่ถูกต้อง

D2: กลับไปยัง M2 ให้ผู้ใช้ทำการเลือกช่วงเวลาที่จะเปรียบเทียบใหม่

Use case specification (2)

Use case name: วิเคราะห์รายรับ-รายจ่าย

Use case purpose: ในส่วนนี้เมื่อให้ผู้ใช้งานเรียกใช้ระบบจะทำการเรียกข้อมูลรายรับรายจ่ายของผู้ใช้ในช่วงเวลาที่ต้องการมาทำการวิเคราะห์เพื่อหาแนวโน้มว่าผู้ใช้กำลังใช้จ่ายเงินกับหมวดหมู่ใดเพิ่มขึ้น หรือหมวดหมู่ใดน้อยลง

Preconditions: ผู้ใช้ต้องเป็นสมาชิกของ web application

Postconditions: ระบบท าการวิเคราะห์ข้อมูลและแสดงผลให้กับผู้ใช้ได้ส าเร็จ

Limitations: การวิเคราะห์ข้อมูลรายรับรายจ่ายเป็นไปตามขั้นตอนวิธี (algorithm) แบบหนึ่ง ซึ่งหากเปลี่ยนขั้นตอนวิธีเป็นอีกรูปแบบหนึ่งผลวิเคราะห์ก็จะได้เป็นอีกรูปแบบหนึ่งดังนั้น ดังนั้นผลวิเคราะห์ที่ได้เป็นไปตามขั้นตอนวิธีซึ่งอาจไม่ตรงกับความเป็นจริงเสมอไป

Assumptions: ผู้ใช้มีการบันทึกข้อมูลจำนวนมากพอที่จะน าไปวิเคราะห์

Primary scenario:

M1: ผู้ใช้เลือกใช้การวิเคราะห์ข้อมูล

M2: ทำการเลือกช่วงเวลาที่ต้องการจะนำข้อมูลมาทำการวิเคราะห์ซึ่งประกอบไปด้วยจุดเริ่มต้นและจุดสิ้นสุด

M3: ระบบทำการดึงข้อมูลของผู้ใช้จากช่วงเวลาที่ต้องการจากฐานข้อมูล

M4: นำข้อมูลที่ได้มาทำการวิเคราะห์

M5: นำผลการวิเคราะห์มาแสดงผลให้ผู้ใช้ทราบ

Alternate scenarios:

Conditions: ผู้ใช้ทำการเลือกช่วงเวลาผิดปกติเช่น เลือกจุดเริ่มต้นเป็น 17 ต.ค. 2555 แต่เวลาจุดสิ้นสุดเป็น 17 ต.ค. 2554 เป็นต้น

D1: ทำการแจ้งเตือนให้ผู้ใช้ทราบว่าทำการกรอกช่วงเวลาผิดปกติ

D2: กลับไปส่วนที่ให้ผู้ใช้เลือกช่วงเวลาค่าใหม่

Conditions: ระบบไม่สามารถดึงข้อมูลออกจากฐานข้อมูลได้

D1: ทำการดึงข้อมูลใหม่

D2: หากยังไม่สามารถทำได้ในเวลาที่กำหนดให้กลับไปยัง M2 และแจ้งผู้ใช้ให้ทราบ

D3: หากสำเร็จก็ดำเนินการต่อตามขั้นตอน

Conditions: ระบบไม่สามารถวิเคราะห์ข้อมูลได้เนื่องจากข้อมูลไม่เพียงพอ

D1: แจ้งให้ผู้ใช้ทราบว่าไม่สามารถวิเคราะห์ข้อมูลให้ได้เนื่องจากข้อมูลไม่เพียงพอ

D2: กลับไปยัง M2 เพื่อให้ผู้ใช้เลือกช่วงเวลาค่าใหม่

สถาปัตยกรรมของระบบ

Problem Analysis

Abstraction

1. ผู้ใช้งานระบบ

ประกอบไปด้วย Username Password และข้อมูลส่วนตัวอื่นๆของผู้ใช้เพื่อใช้ในการระบบและเป็นข้อมูลในการวิเคราะห์

2. รายรับรายจ่ายของผู้ใช้

ประกอบไปด้วย เจ้าของ จำนวนเงิน หมวดหมู่ วันที่ ของรายรับรายจ่ายนั้น มีไว้เพื่อใช้ในการเป็นข้อมูลในการแสดงผล วิเคราะห์ เปรียบเทียบ ตามความต้องการของผู้ใช้

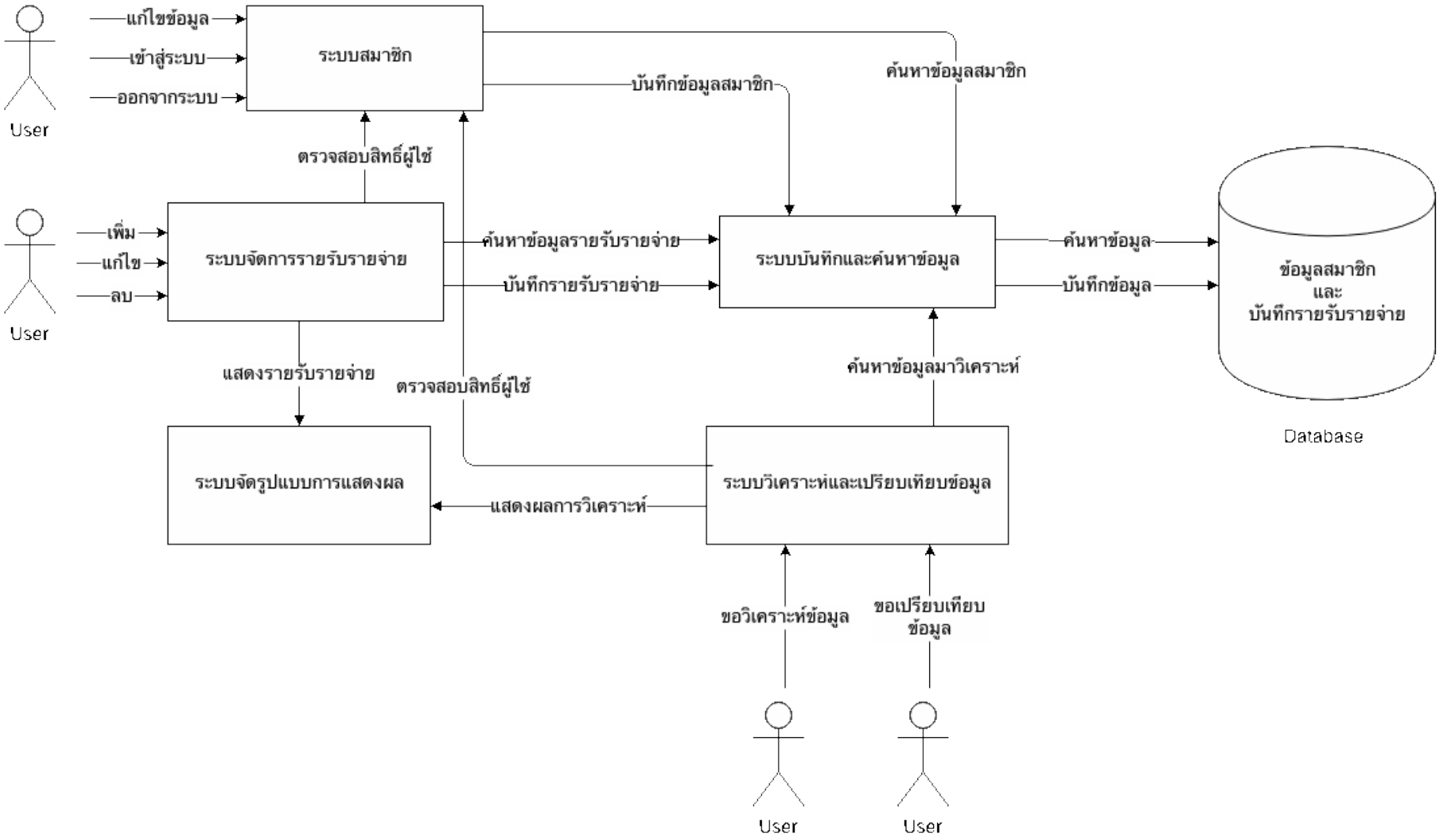
3. วิเคราะห์ข้อมูลรายรับรายจ่าย

ประกอบด้วยส่วนของการวิเคราะห์ข้อมูล เปรียบเทียบข้อมูล โดยจะนำเอาข้อมูลของการบันทึกรายรับรายจ่ายของผู้ใช้มาทำการวิเคราะห์ เปรียบเทียบและนำผลลัพท์ไปแสดงผลให้กับผู้ใช้

4. รูปแบบการแสดงผล

ในการแสดงผลให้กับผู้ใช้นั้นจะแสดงผลในรูปแบบของกราฟความสัมพันธ์ เช่น กราฟเส้น กราฟวงกลม หรืออาจจะแสดงในรูปแบบของลำดับของการจดบันทึกรายรับรายจ่าย (timeline)

Application Architecture



ระบบสมาชิก

มีหน้าที่ทำเกี่ยวกับการสมัครสมาชิก เข้าสู่ระบบ ออกจากระบบ การทำการตรวจสอบว่าผู้ใช้ทำการเข้าสู่ระบบแล้วหรือยัง เมื่อเรียกใช้ส่วนที่จำเป็นต้องผ่านการเข้าสู่ระบบจึงจะสามารถทำได้

ระบบจัดการรายรับรายจ่าย

มีหน้าที่ทำการจัดการเกี่ยวกับ เพิ่มข้อมูลรายรับรายจ่าย แก้ไขข้อมูลรายรับรายจ่าย ลบข้อมูลรายรับจ่าย

ระบบวิเคราะห์และเปรียบเทียบข้อมูล

มีหน้าที่จัดการการวิเคราะห์ข้อมูลรายรับรายจ่ายของผู้ใช้และเปรียบเทียบข้อมูลรายรับรายจ่ายกับค่าเฉลี่ยของผู้ใช้คนอื่นๆ

ระบบบันทึกและค้นหาข้อมูล

เป็นส่วนที่มีหน้าที่ทำการไปค้นหาข้อมูลและบันทึกข้อมูลจากที่ที่เก็บข้อมูลไว้

ระบบจัดรูปแบบการแสดงผล

เป็นส่วนที่มีหน้าที่จัดการข้อมูลให้ตรงตามเงื่อนไขกับการนำไปแสดงผล

Subsystem / Component

1. ระบบสมาชิก

1.1 ส่วนดูแลการเข้าสู่ระบบมีหน้าที่จัดการเกี่ยวกับการเข้าสู่ระบบของผู้ใช้

1.2 ส่วนดูแลการออกจากระบบมีหน้าที่จัดการเกี่ยวกับออกจากระบบ

1.3 ส่วนดูแลการแก้ไขข้อมูลผู้ใช้มีหน้าที่ทำการแก้ไขข้อมูลของผู้ใช้ตามความต้องการของผู้ใช้ เช่น password ชื่อ นามสกุล อาชีพ อายุ

1.4 ส่วนดูแลการสมัครสมาชิกมีหน้าที่ดูแลการสมัครสมาชิกของผู้ใช้

1.5 ส่วนตรวจสอบการอยู่ในระบบและสิทธิ์ของผู้ใช้ มีหน้าที่ตรวจสอบว่าผู้ใช้ยังอยู่ในระบบหรือไม่เมื่อทำการขอให้งานความสามารถต่างๆระบบ

2. ระบบจัดการรายรับรายจ่าย

2.1 ส่วนเพิ่มข้อมูลรายรับรายจ่าย มีหน้าที่เพิ่มข้อมูลรายรับรายจ่ายเข้าไปสู่ระบบ

2.2 ส่วนแก้ไขข้อมูลรายรับรายจ่าย มีหน้าที่แก้ไขข้อมูลรายรับรายจ่ายที่มีอยู่ในระบบตามความต้องการของผู้ใช้

2.3 ส่วนลบข้อมูลรายรับรายจ่าย มีหน้าที่ลบข้อมูลรายรับรายจ่ายที่มีอยู่ในระบบของผู้ใช้

3. ระบบวิเคราะห์และเปรียบเทียบข้อมูล

3.1 ส่วนวิเคราะห์ข้อมูลรายรับรายจ่ายของลูกค้า มีหน้าที่วิเคราะห์ข้อมูลรายรับรายจ่ายของลูกค้าว่ามีแนวโน้มเป็นอย่างไรตามขั้นตอนวิธีการ (Algorithm) ต่างๆ

3.2 ส่วนเปรียบเทียบข้อมูลรายรับรายจ่ายของลูกค้า มีหน้าที่เปรียบเทียบข้อมูลรายรับรายจ่ายของลูกค้ากับค่าเฉลี่ยรายรับรายจ่ายของคนอื่นๆ ในระบบตามขั้นตอนวิธีการ (Algorithm) ต่างๆ

4. ระบบจัดรูปแบบการแสดงผล

4.1 ส่วนแสดงผลในส่วนของการวิเคราะห์ข้อมูลรายรับรายจ่าย มีหน้าที่ทำการแสดงผลลัพธ์ของการวิเคราะห์ออกมาในรูปแบบต่างๆให้ผู้ใช้เห็น

4.2 ส่วนแสดงผลในส่วนของการเปรียบเทียบข้อมูลรายรับรายจ่าย มีหน้าที่ทำการแสดงผลลัพธ์การวิเคราะห์ออกมาในรูปแบบต่างๆให้ผู้ใช้เห็น

4.3 ส่วนแสดงผลข้อมูลรายรับรายจ่าย มีหน้าที่ทำการแสดงข้อมูลรายรับรายจ่ายออกมาในรูปแบบต่างๆ เช่น กราฟเส้น กราฟวงกลม ตาราง เป็นต้น

5. ระบบบันทึกและค้นหาข้อมูล

มีหน้าที่เป็นตัวกลางในการติดต่อกับแหล่งข้อมูลโดยมีหน้าที่หลักๆ คือ 1.ค้นหาข้อมูล 2.แก้ไขข้อมูล 3. เพิ่มข้อมูล 4.ลบข้อมูล

แผนภาพของคลาสหลัก

Domain classes

Class ที่เกี่ยวข้องกับระบบสมาชิกจะประกอบไปด้วย 5 Class

1. LoginManager จะทำการ implement interface LoginAble เพื่อนำไปใช้เป็นตัวที่ทำการเข้าสู่ระบบ

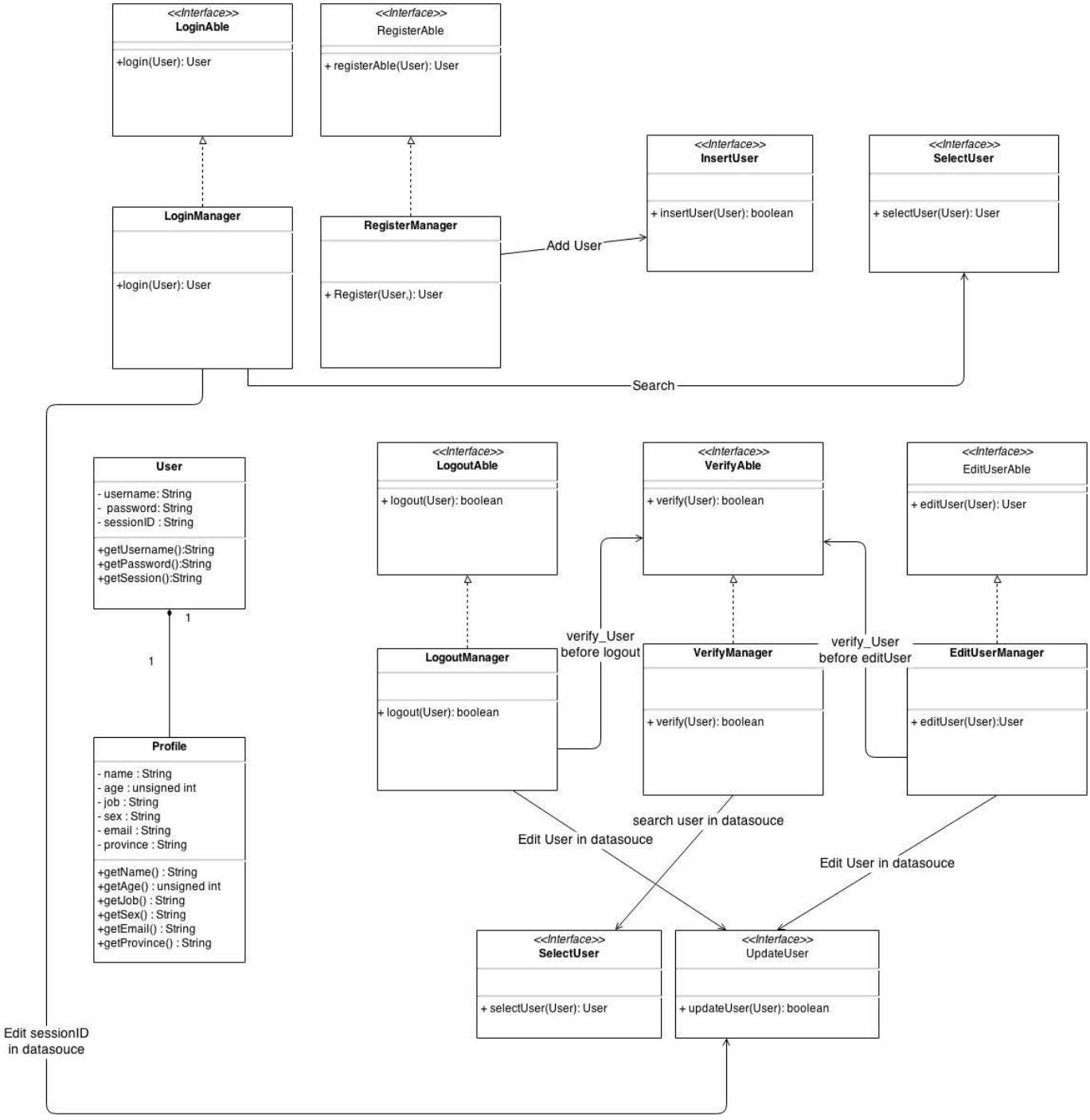
2. LogoutManager จะทำการ implement interface LogoutAble เพื่อนำไปใช้ในการออกจากระบบ โดย class เรียกใช้ class ที่ implement interface VerifyAble เพื่อตรวจสอบว่าผู้ใช้ที่จะสั่ง Logout นั้นมีสิทธิ์และอยู่ในระบบหรือไม่

3. VerifyManager จะทำการ implement interface VerifyAble เพื่อนำไปใช้ในการตรวจสอบผู้ใช้ว่าอยู่ในระบบหรือไม่มีสิทธิ์ที่จะทำการใดๆที่กำลังจะทำหรือไม่

4. EditUserManager จะทำการ implement interface EditUserManager เพื่อนำไปใช้ในการในการแก้ไขข้อมูลและบัญชีของผู้ใช้ โดย class นี้จะทำการเรียกใช้ interface VerifyAble เพื่อตรวจสอบสิทธิ์และสถานะการอยู่ในระบบ

5. RegisterManager จะทำการ implement interface RegisterAble เพื่อนำไปใช้ในการสมัครสมาชิกของผู้ใช้

ซึ่งทั้ง 5 class นี้จะเรียกใช้ class ที่ implement interface ที่เกี่ยวกับการติดต่อกับแหล่งข้อมูลเพื่อทำการค้นหาจากแหล่งข้อมูลโดยที่ทำการออกแบบให้ใช้เรียกใช้ผ่าน class ที่ implement interface แทนการเขียนติดต่อกับแหล่งข้อมูลใน class เลย เพราะต้องการความยืดหยุ่นเวลาการแก้ไขปรับเปลี่ยนการติดต่อกับแหล่งข้อมูล จะได้แก้ไขที่คลาสที่เรียกใช้ หรือทำการเรียกใช้ class ใหม่


<<Interface>>
LoginAble
+login(User): User
<<Interface>>
RegisterAble
+ registerAble(User): User
<<Interface>>
InsertUser
+ insertUser(User): boolean
<<Interface>>
SelectUser
+ selectUser(User): User
LoginManager
+login(User): User
RegisterManager
+ Register(User,): User
Add User
Search
User
- username: String
- password: String
- sessionID : String
+getUsername():String
+getPassword():String
+getSession():String
1
1
Profile
- name : String
- age : unsigned int
- job : String
- sex : String
- email : String
- province : String
+getName() : String
+getAge() : unsigned int
+getJob() : String
+getSex() : String
+getEmail() : String
+getProvince() : String
<<Interface>>
LogoutAble
+ logout(User): boolean
<<Interface>>
VerifyAble
+ verify(User): boolean
<<Interface>>
EditUserAble
+ editUser(User): User
LogoutManager
+ logout(User): boolean
verify_User
before logout
VerifyManager
+ verify(User): boolean
verify_User
before editUser
EditUserManager
+ editUser(User):User
search user in datasouce
Edit User in datasouce
Edit User in datasouce
<<Interface>>
SelectUser
+ selectUser(User): User
<<Interface>>
UpdateUser
+ updateUser(User): boolean
Edit sessionID
in datasouce

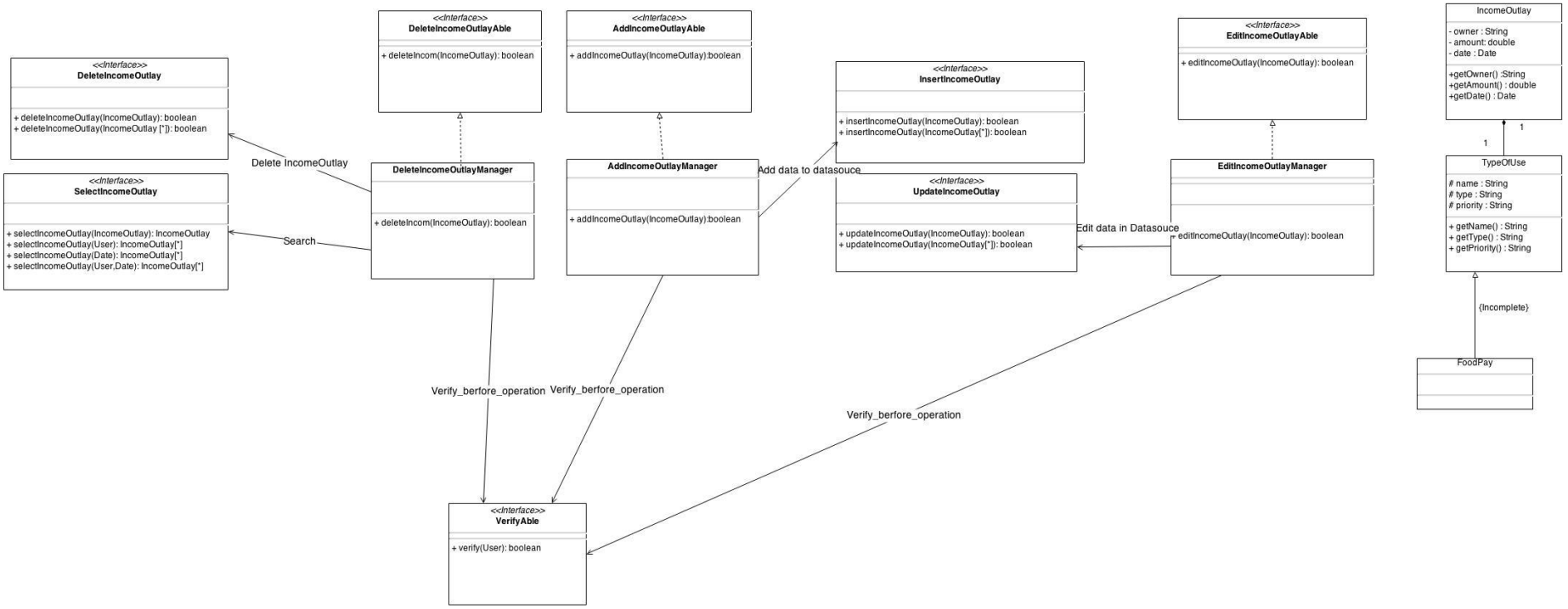
Class ที่เกี่ยวข้องกับส่วนจัดการรายรับรายจ่ายจะประกอบด้วย 3 class คือ

1. AddIncomeOutlayManager จะทำการ implement AddIncomeOutlayAble เพื่อใช้ในการเพิ่มรายรับรายจ่าย

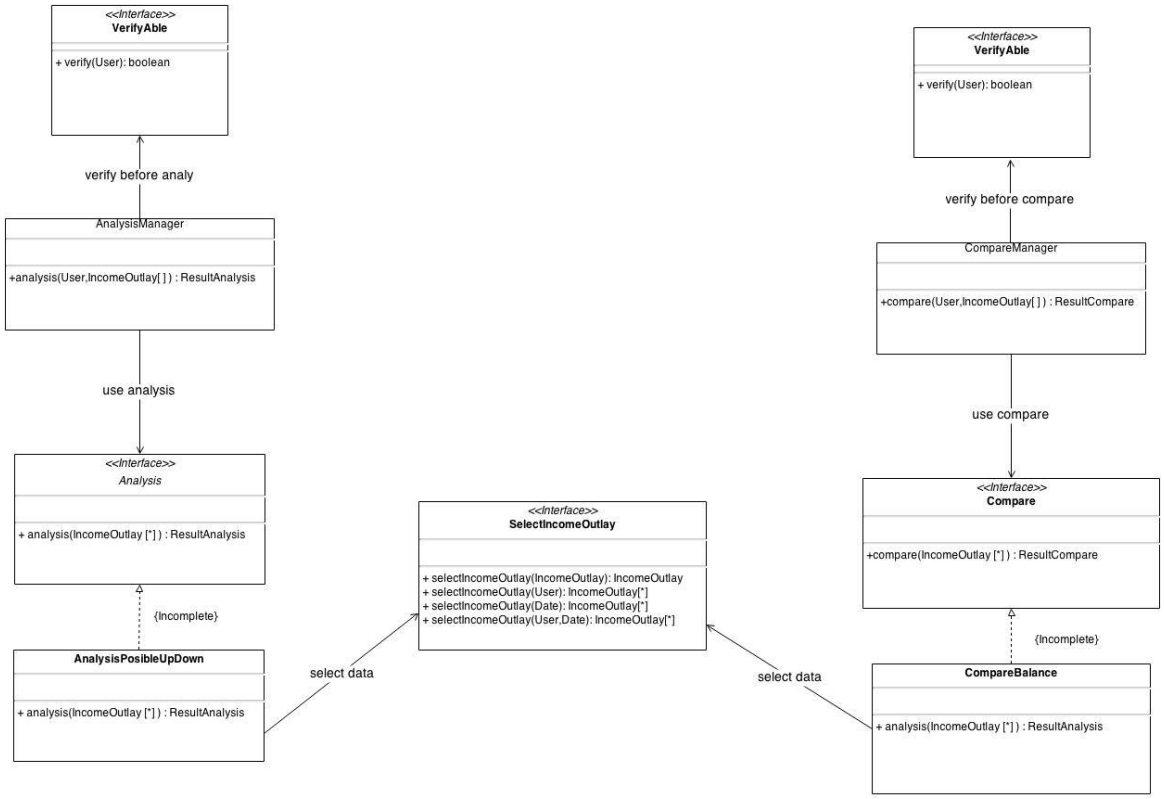
2. EditIncomeOutlayManager จะทำการ implement EditIncomeOutlayAble เพื่อใช้ในการแก้ไขรายรับรายจ่าย

3. DeleteIncomeOutlayManager จะทำการ implement DeleteIncomeOutlayAble เพื่อใช้ในการลบรายรับรายจ่าย

ซึ่งทั้ง 3 class นี้จะเรียกใช้ class ที่ implement interface VerifyAble และเรียกใช้ class ที่ implement interface ที่เกี่ยวกับการติดต่อกับแหล่งข้อมูลที่ทำอย่างนี้นั้นแทนการเขียนทำการตรวจสอบสิทธิ์ที่ตัว class เลยก็เพราะ ถ้ามีการเปลี่ยนรูปแบบการตรวจสอบสิทธ์ จะได้ไม่ต้องทำการแก้ไขที่ class นี้แต่ไปแก้ไขที่ class ที่ทำการ implement VerifyAble แทน หรือ ให้ class นี้เรียกใช้ class ที่ทำการ implement VerifyAble ที่มีการตรวจสอบสิทธิ์แบบใหม่



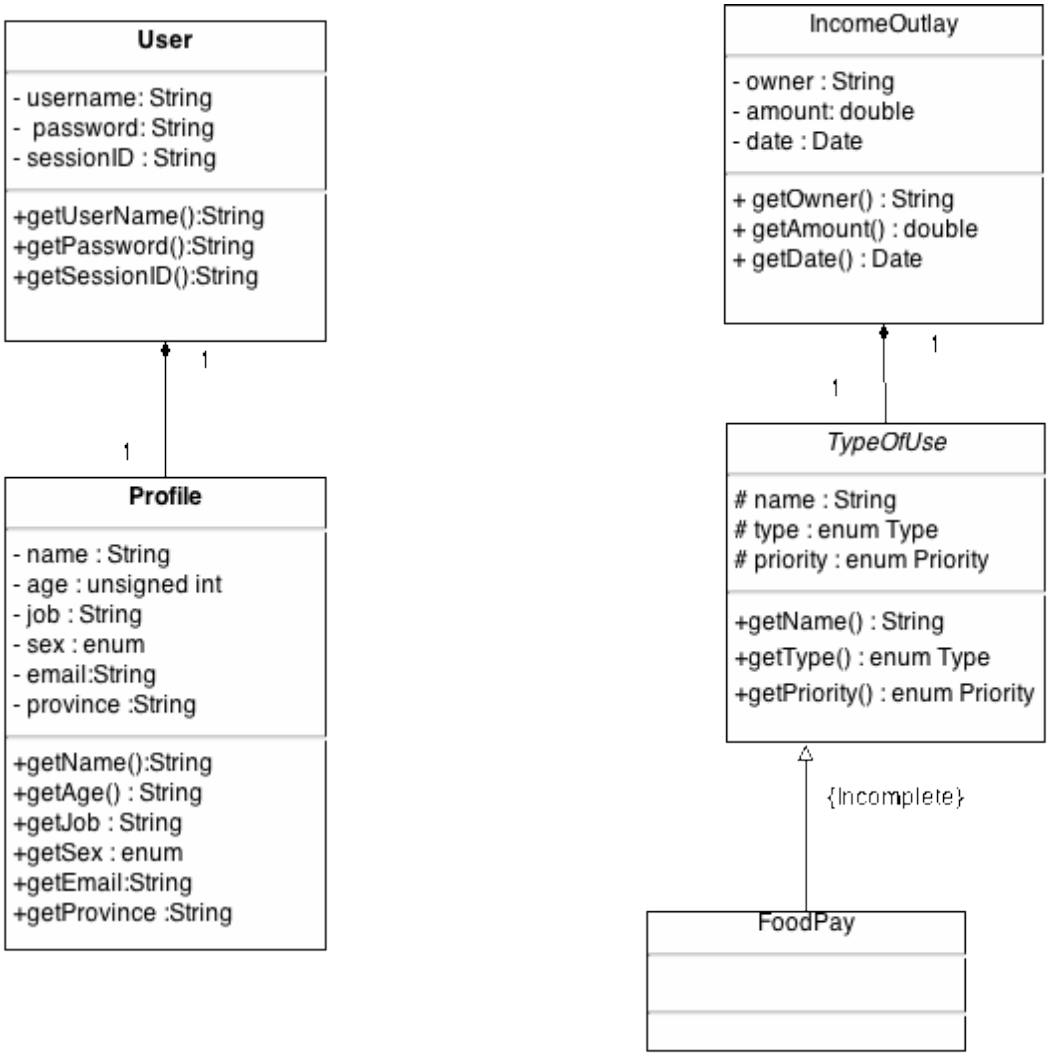
Class ที่เกี่ยวข้องกับส่วนวิเคราะห์ข้อมูล นั้นจะเป็น class ที่ทำการ implement interface Analysis และ Compare ซึ่งที่ออกแบบอย่างนี้เพราะอาจมีวิธีการวิเคราะห์ข้อมูลและเปรียบเทียบข้อมูลแบบอื่นๆอีกหลากหลายรูปแบบจึงทำ interface ให้คนภายนอกส่งข้อมูลมาวิเคราะห์หรือเปรียบเทียบอย่างเดียว ส่วนภายในเราจะใช้ขั้นตอนวิธีการ (Algorithm) ใหม่อย่างไรก็ได้เพียงแต่ต้อง implement interface Analysis หรือ Compare ก็พอ



Class ที่เกี่ยวข้องกับส่วนบันทึกและค้นหาข้อมูล นั้นจะเป็น class ที่ทำการ implement interface ที่อยู่ในภาพ โดยที่ออกแบบอย่างนี้ก็เพราะต้องการให้ยืดหยุ่นเราสามารถปรับเปลี่ยนการทำงานในการไปค้นหาหรือลบข้อมูลตามความต้องการที่เปลี่ยนแปลงได้

Class User และ Profile (ซ้าย) นั้น Class user จะประกอบไปด้วย Profile โดยมีความสัมพันธ์แบบ 1 ต่อ 1 คือ User หนึ่งตัวมีได้ Profile เดียวและ Profile นั้นมีเจ้าของได้คนเดียวซึ่งในที่นี้สามารถเพิ่ม operation ในการ get หรือ set ข้อมูลที่เป็น private ได้ตามความเห็นสมควรว่าจะให้เอาข้อมูลออกมาได้หรือไม่

Class IncomeOutlay และ TypeOfUse (ขวา) นั้น IncomeOutlay นั้นจะประกอบไปด้วย TypeOfuse โดยมีความสัมพันธ์แบบ 1 ต่อ 1 โดยที่ออกแบบอย่างนี้เพราะต้องการให้ TypeOfUse มีได้หลายชนิดไม่จำกัดอยู่เพียงไม่กี่ชนิดและสามารถสร้างขึ้นใหม่ได้เรื่อยๆโดยสร้าง class ที่สืบทอดจาก TypeOfUse



รายละเอียดการพัฒนาซอฟแวร์

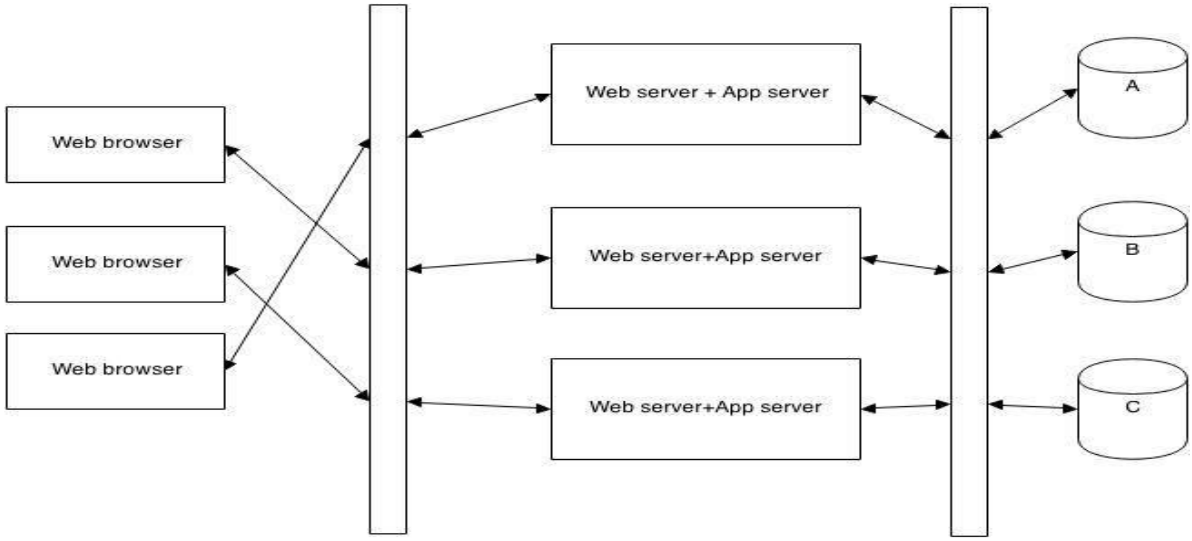
Deployment

ส่วนที่เป็นหน้าแสดงผล

ในส่วนนี้จะใช้ HTML, CSS, Java Script ในการทำส่วนการแสดงผล(Presentation) ที่จะนำไปใช้กับ Web- browser ของผู้ใช้โดยจะเลือกใช้ Bootstrap เพื่อช่วยในการจัดหน้าจอเพื่อให้ใช้แสดงได้ในหลายอุปกรณ์ เช่น คอมพิวเตอร์ มือถือ ที่มีขนาดหน้าจอแตกต่างกัน ใช้ AngularJS ที่เป็น API ของภาษา Java Script ซึ่งช่วยทำให้เขียนโปรแกรมได้สะดวกและง่ายขึ้นไม่ต้องทำการเขียนในหลายๆ ส่วนเอง

Web server

ในส่วนนี้จะใช้บริการ Web site ของ Microsoft Azure ที่มีไว้ให้ โดยทาง Azure จะสร้างเครื่องที่ทำการลงโปรแกรมที่จำเป็นต่างๆ ที่จำเป็นในการเป็นเครื่อง Server ไว้ให้ ตัวเราเพียงแค่ทำการส่งชุดคำสั่งของโปรแกรมของเราขึ้นไปบนตัวเครื่องเท่านั้น อีกทั้งตัวบริการนี้สามารถทำการเพิ่ม-ลดขนาด(Scale) โดยสร้างเครื่องใหม่ขึ้นมารับงานเมื่อเครื่อง Server ตัวเดียวเริ่มรับงานไม่ไหว ซึ่งสามารถตั้งกฎในการเพิ่มจำนวนเครื่องที่เป็น Instances ได้ง่ายโดยไม่จำเป็นต้องเขียนโปรแกรม โดยเราเลือกให้ตัว Web server ทำการลงตัว Apache tomcat และใช้ภาษา Java ในการเขียนส่วนที่เป็น Business logic และใช้ตัว Spring framework เป็นตัวช่วยในการติดต่อ



ส่วนของ Database

ในส่วนนี้จะใช้บริการของ SQL Database ของ Microsoft Azure ซึ่งมีให้อยู่แล้วอีกทั้งยังรับประกันความสามารถในการให้บริการ (Availability) อีกทั้งยังสามารถทำการ Scale up หรือ Scale down ได้ง่าย โดย

จะเอา SQL Database ที่ได้นั้นมาทำเป็น Distribute database โดยกระจายข้อมูลแบบ Horizontal ไปให้ทุก Database เพื่อแบ่งภาระงานในการเขียนและอ่านข้อมูล โดยส่วนที่ทำการเลือกที่จะส่งข้อมูลลง Database ว่าจะลง Database ไหนนั้นจะทำการดำเนินการสร้าง(Implement) ขึ้นมาเอง

Implementation plan

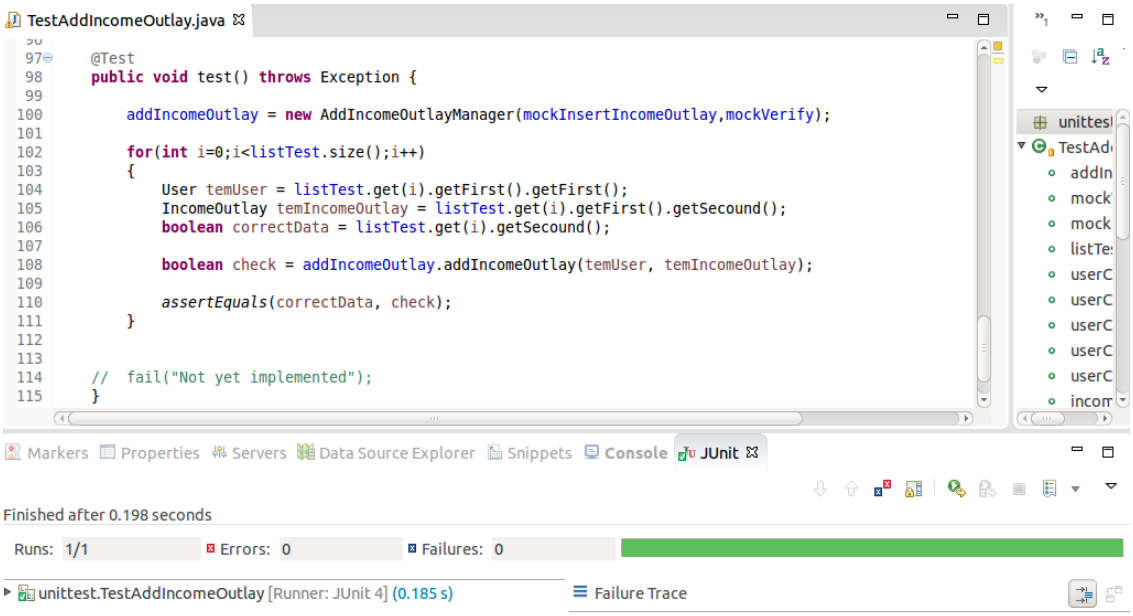
งานที่ทำ	ระยะเวลา/วัน	ผู้รับผิดชอบ
1. ออกแบบส่วนที่เป็น business logic ว่าแต่ละระบบนั้นจะมี Class อะไรบ้าง แต่ละ Class ติดต่อกันอย่างไร	5 (20-24 ก.ย. 57)	นายสุรพงศ์ เท่าเทียมตน
2. ออกแบบส่วนที่เป็นส่วนแสดงผลให้กับผู้ใช้ว่าควรจะจัดวางอย่างไร และทดลองใช้ Bootstrap สร้างส่วนแสดงผลขึ้นมา	6 (25-30 ก.ย. 57)	นายพัสกร จุลพล
3. สร้างและออกแบบฐานข้อมูล	2 (9-10 ต.ค. 57)	นายสุรพงศ์ เท่าเทียมตน
4. เขียนส่วนที่เป็นส่วนจัดรูปแบบแสดงผล และ ส่วนระบบสมาชิก	4 (11-14 ต.ค. 57)	นายพัสกร จุลพล
5. เขียนนส่วนแสดงผลที่เกี่ยวกับระบบสมาชิก เช่น หน้าเข้าสู่ระบบ หน้าประวัติสมาชิก	4 (15-18 ต.ค. 57)	นายสุรพงศ์ เท่าเทียมตน
6. เขียนส่วนจัดการรายรับรายจ่าย ส่วนบันทึกและค้นหาข้อมูล	4 (19-22 ต.ค. 57)	นายสุรพงศ์ เท่าเทียมตน
7. เขียนส่วนแสดงผลที่เกี่ยวกับการจัดการรายรับรายจ่าย เช่น หน้าบันทึกรายรับรายจ่าย หน้าค้นหา หน้าลบรายรับรายจ่าย	4 (23-26 ต.ค. 57)	นายพัสกร จุลพล
8. เขียนส่วนวิเคราะห์และเปรียบเทียบข้อมูล	4 (27-30 ต.ค. 57)	นายสุรพงศ์ เท่าเทียมตน
9. เขียนส่วนแสดงผลการวิเคราะห์และเปรียบเทียบ	4 (31 ต.ค.-3 พ.ย. 57)	นายพัสกร จุลพล
10. เริ่มทำการสร้าง Distribute database และทำส่วนที่ทำการแยกว่าจะเลือกค้นหาหรือบันทึกลง Database ตัวไหน	4 (4-7 พ.ย. 57)	นายสุรพงศ์ เท่าเทียมตน
11. การ Test แต่ละส่วนของระบบ	7 (8-14 พ.ย. 57)	นายพัสกร จุลพล

ผลการทดสอบซอฟแวร์

Unit Test

TestAddIncomeOutlay.java ไฟล์นี้ใช้ตรวจสอบเกี่ยวกับการเพิ่มบันทึกรายรับรายจ่ายว่ามีข้อผิดพลาดหรือไม่ โดย file test จะอยู่ใน directory ดังนี้

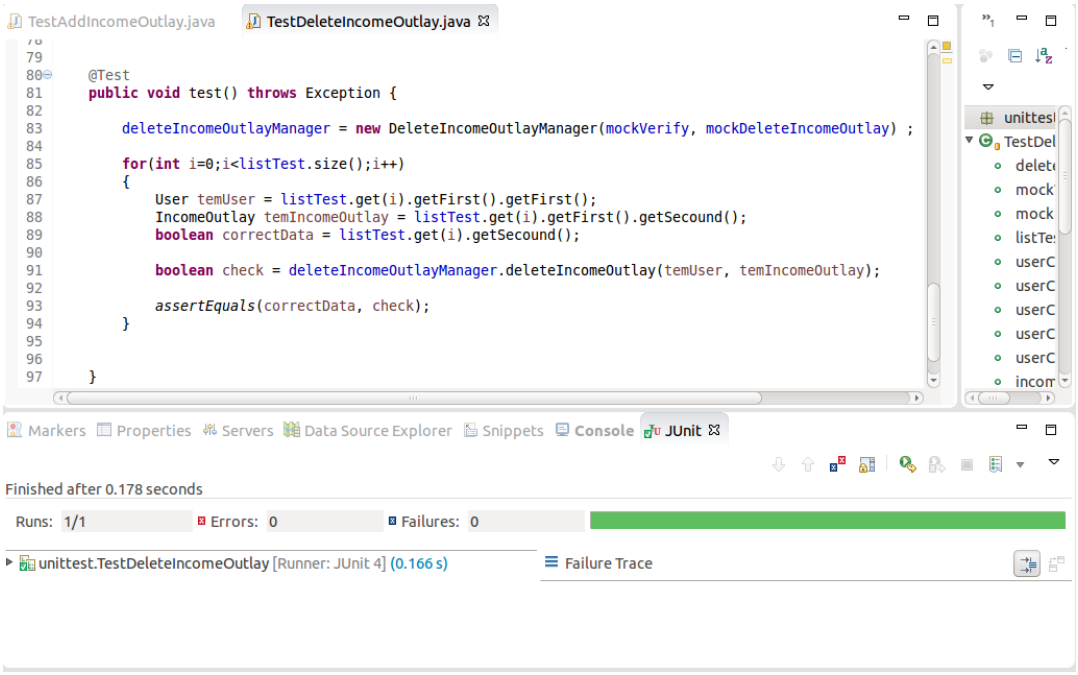
/ROOT/src/unittest/TestAddIncomeOutlay.java



จากการทดลอง unit test พบว่า Class AddIncomeOutlayManager สามารถผ่านการ unit test ได้ทุก case ที่ได้มีกำหนดไว้

TestDeleteIncomeOutlay.java ไฟล์นี้ใช้ตรวจสอบเกี่ยวกับการลบบันทึกรายรับรายจ่ายว่ามีข้อผิดพลาดหรือไม่ โดย file test จะอยู่ใน directory ดังนี้

/ROOT/src/unittest/TestDeleteIncomeOutlay.java



จากการทดลอง unit test พบว่า Class DeleteIncomeOutlayManager สามารถผ่านการ unit test ได้ทุก case ที่ได้มีกำหนดไว้

Evaluation

การทดลองที่ 1

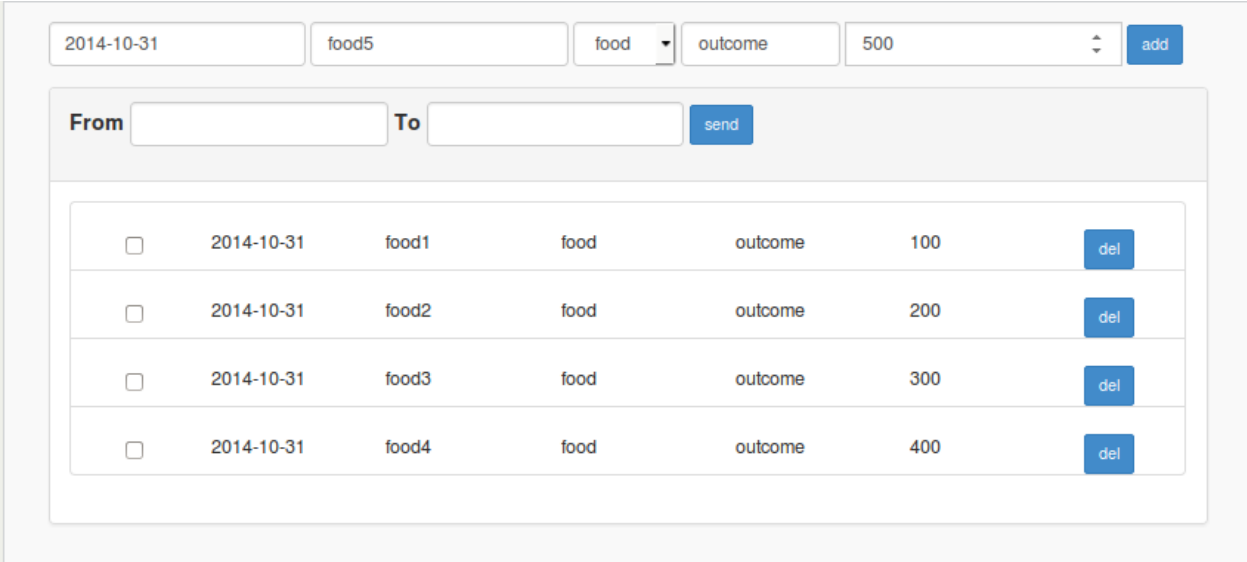
การบันทึกรายรับรายจ่าย

- **จุดประสงค์:** เพื่อดูว่าการบันทึกรายรับ-รายจ่ายของผู้ใช้นั้นถูกบันทึกลงระบบจริงๆ และเมื่อนำข้อมูลกลับมาแสดงผลนั้นถูกต้องตามที่บันทึกไปหรือไม่
- **สิ่งที่จะวัด:** ความถูกต้องของข้อมูลรายรับรายจ่าย เช่น วันที่บันทึก จำนวนเงิน ชนิดของข้อมูลรายรับรายจ่ายที่ใช้ เป็นต้น
- **วิธีทำและสิ่งที่ต้องใช้ในการทดลอง:**
  1. ทำการบันทึกข้อมูลรายรับรายจ่ายให้กับระบบ
  2. ตรวจสอบการแสดงผลที่หน้าจอว่าแสดงผลหลังจากบันทึกถูกต้องตามที่ต้องการหรือไม่
  3. ทำการออกจากระบบจากนั้นเข้าสู่ระบบอีกครั้งและเข้าไปดูส่วนแสดงการรายรับรายจ่ายว่าข้อมูลที่บันทึกไปนั้นยังอยู่และมีค่าถูกต้องหรือไม่
  4. บันทึกผลการทดลอง

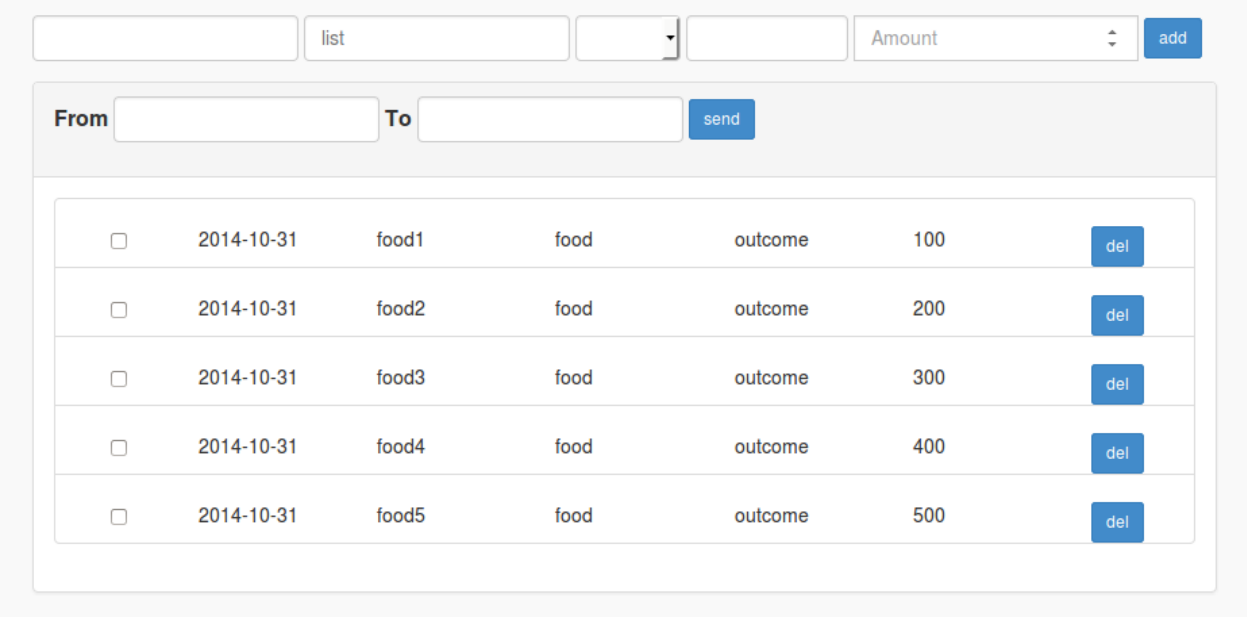
5. ทำการทดลองซ้ำอีกตามจำนวนครั้งที่ต้องการ

- ผลที่ได้จากการทดลอง:

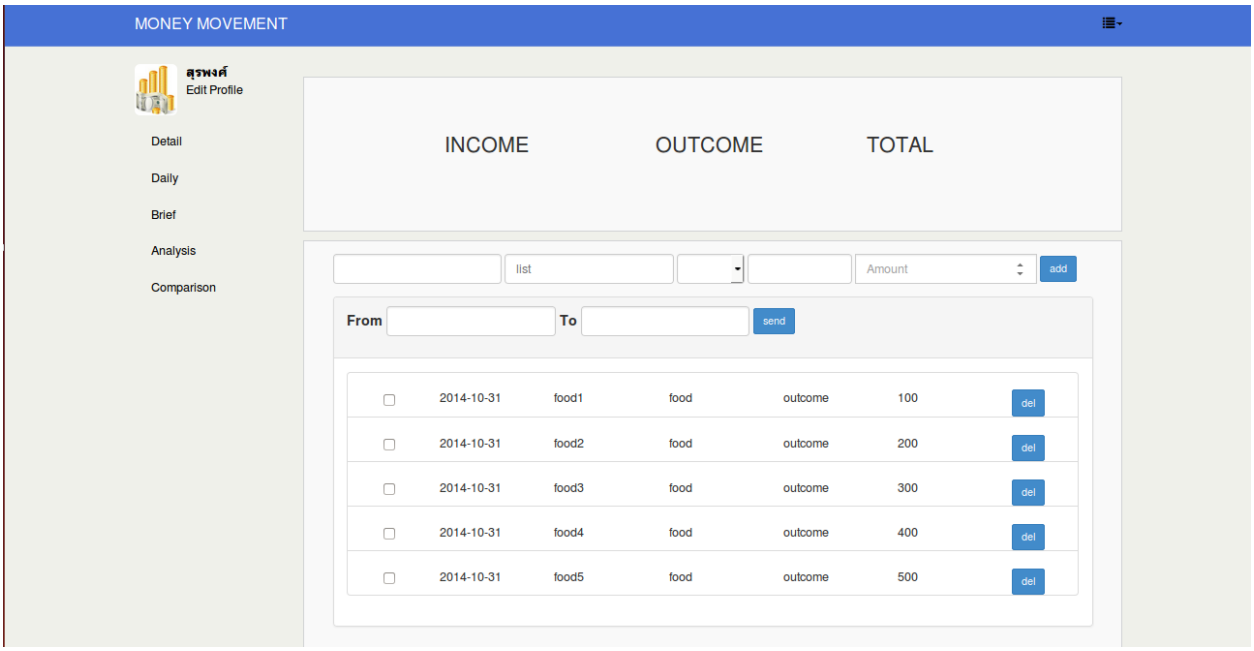
จากการทดลองที่จะทำการบันทึกรายรับรายจ่ายโดยจะทดลองบันทึกข้อมูลรายรับรายจ่าย 5 รายรับรายจ่ายโดยผลออกมาว่าสามารถบันทึกได้



รูปที่ 1 ภาพก่อนการบันทึกข้อมูลรายรับรายจ่ายที่ 5



รูปที่ 2 ภาพการบันทึกรายรับรายจ่ายที่ 5 ที่ทำการบันทึกได้สำเร็จ



รูปที่ 3 ภาพผลการทดลองการเข้ามาระบบอีกครั้งเพื่อดูข้อมูลรายรับรายจ่าย

และการทดลองออกจากระบบและเข้ามาดูข้อมูลใหม่ว่าข้อมูลที่บันทึกลงไปนั้นยังอยู่หรือไม่ซึ่งจากการทดลองพบว่าข้อมูลยังอยู่ครบ

และจากการทดลองซ้ำอีก 2 ครั้งเพื่อทดสอบว่าจะสามารถบันทึกข้อมูลได้หรือไม่นั้นการทดลองทุกครั้งก็สามารถบันทึกข้อมูลได้อย่างถูกต้องแม้ออกจากระบบกลับมาข้อมูลก็ยังอยู่

- สรุปผลการทดลอง:

จากผลการทดลองพบว่าการบันทึกรายรับรายจ่ายนั้นสามารถทำได้อย่างถูกต้อง และเมื่อออกจากระบบและกลับเข้ามาอีกครั้งข้อมูลยังสามารถแสดงได้อย่างถูกต้อง

การทดลองที่ 2

การวิเคราะห์รายรับรายจ่าย

- จุดประสงค์: เพื่อดูว่าการวิเคราะห์ข้อมูลรายรับรายจ่ายเป็นไปตามสูตรการวิเคราะห์ที่กำหนดให้หรือไม่

- สิ่งที่จะวัด: ความถูกต้องของผลลัพธ์การวิเคราะห์

- วิธีทำและสิ่งที่ต้องใช้การทดลอง:

1. ทำการวิเคราะห์ข้อมูลรายรับรายจ่ายที่กำหนดไว้เพื่อหาผลลัพธ์การวิเคราะห์

2. ทำการเปรียบเทียบผลลัพธ์จากการทดลองที่ได้จากระบบกับผลการวิเคราะห์ที่ถูกต้องว่าตรงกันหรือไม่
3. บันทึกผลการทดลอง
4. ทำการทดลองซ้ำอีกตามจำนวนครั้งที่ต้องการ

- ผลที่ได้จากการทดลอง:

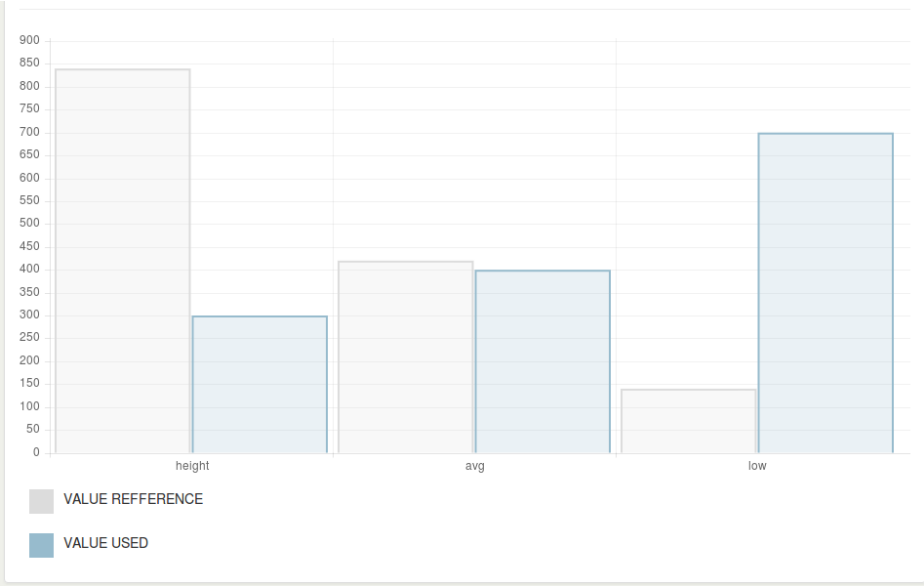
ในการทดลองนี้จะทำการทดลองวิเคราะห์ข้อมูลรายรับรายจ่ายโดยใช้หลักการคำนวณโดยอาศัยหลักการใช้เงินตามความสำคัญ โดยหลักการคือเราควรใช้เงินให้กับรายจ่ายที่สำคัญ 6 ส่วน รายจ่ายที่จำเป็นรองลงมา 3 ส่วน และรายรับรายจ่ายที่จำเป็นน้อย 1 ส่วน เช่น ถ้ารายจ่ายทั้งหมดเป็น 1000 รายจ่ายควรเป็นดังตาราง

	รายจ่ายที่มีความสำคัญ	รายจ่ายที่มีความสำคัญลงมา	รายจ่ายที่มีความจำเป็นน้อย
การแบ่งที่ควรเป็น	600	300	100

ในการทดลองนี้จะกำหนดให้ผู้ใช้เงินเป็นจำนวน 1400 โดยแบ่งเป็นตามตารางดังนี้

	รายจ่ายที่มีความสำคัญ	รายจ่ายที่มีความสำคัญลงมา	รายจ่ายที่มีความจำเป็นน้อย
รายจ่ายที่จ่ายจริง	300	400	700
การแบ่งที่ควรเป็น	840	420	140

ซึ่งจากทดลองพบว่าผลออกมาเป็นดังที่ต้องการดังภาพ

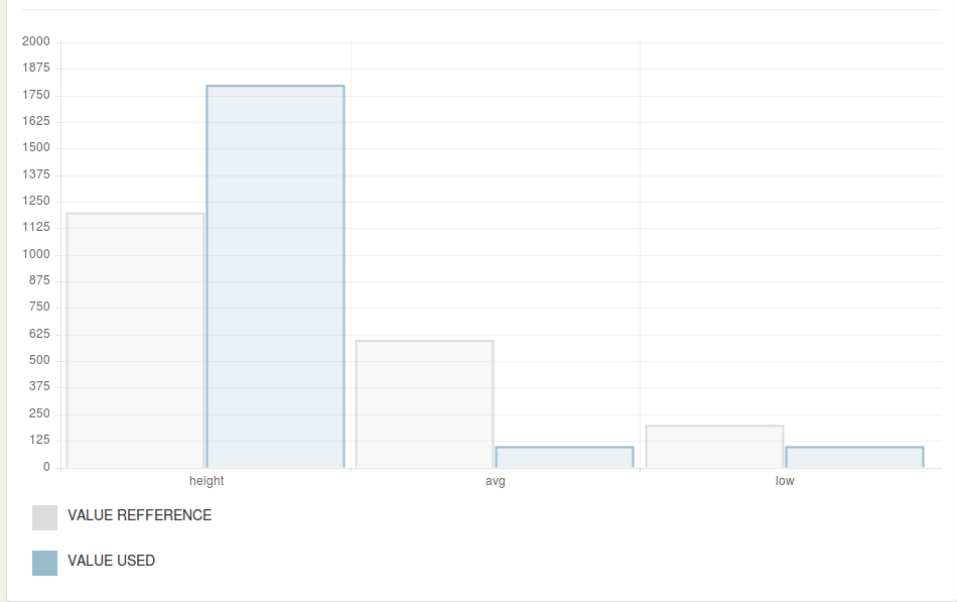


รูปที่ 4 ภาพผลการวิเคราะห์ข้อมูลการทดลองครั้งที่ 1

โดยจากภาพกราฟแท่งสีขาวหมายถึงรายจ่ายที่ควรจะเป็น กราฟแท่งสีฟ้าหมายถึงรายจ่ายที่ใช้ไปจริงๆซึ่งเป็นไปตามที่คาดไว้ และจากการทดลองอีก 2 ครั้งเพื่อตรวจสอบอีกครั้ง

ผลที่คาดหวังในการทดลองครั้งที่ 2

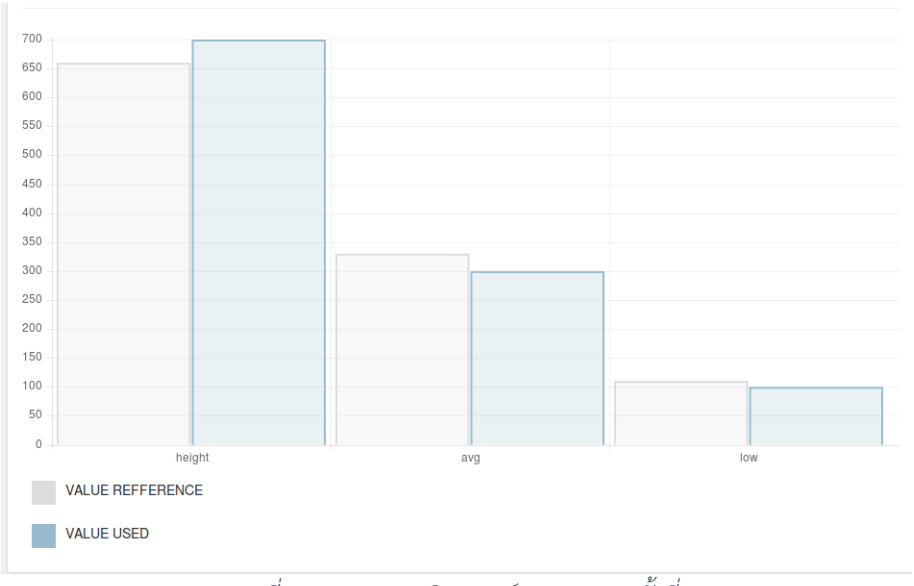
	รายจ่ายที่มีความสำคัญ	รายจ่ายที่มีความสำคัญลงมา	รายจ่ายที่มีความจำเป็นน้อย
รายจ่ายที่จ่ายจริง	1800	100	100
การแบ่งที่ควรเป็น	1200	600	200



รูปที่ 5 ภาพผลการวิเคราะห์การทดลองครั้งที่ 2

ผลที่คาดหวังในการทดลองครั้งที่ 3

	รายจ่ายที่มีความสำคัญ	รายจ่ายที่มีความสำคัญลงมา	รายจ่ายที่มีความจำเป็นน้อย
รายจ่ายที่จ่ายจริง	700	300	100
การแบ่งที่ควรเป็น	660	600	200



รูปที่ 6 ภาพผลการวิเคราะห์การทดลองครั้งที่ 3

- สรุปผลการทดลอง:

จากผลการทดลองสามารถสรุปได้ว่าการวิเคราะห์การทดลองเป็นไปตามสูตรการทดลองที่กำหนด

บทสรุป

Money Movement เป็น web application เกี่ยวกับการจดบันทึกรายรับรายจ่ายของผู้ใช้เพื่อให้ผู้ใช้สามารถรู้ว่าตนเองใช้เงินไปกับอะไรบ้างมีจำนวนสัดส่วนอย่างไร สามารถวิเคราะห์รายรับรายจ่ายของผู้ใช้งานได้ สามารถเปรียบเทียบรายรับรายจ่ายของตนกับค่าเฉลี่ยรายรับรายจ่ายของคนอื่นๆได้ ซึ่งจะนำมาแสดงผลในรูปแบบกราฟ แผนภูมิ เพื่อให้ผู้ใช้เข้าใจง่ายขึ้น และด้วยความสามารถที่ได้กล่าวไปผู้ใช้สามารถนำข้อมูลการวิเคราะห์และเปรียบเทียบ นำไปใช้ประกอบการตัดสินใจในการลดรายจ่าย หาวิธีเพิ่มรายได้ หรือ วางแผนการใช้เงิน

โดยในการพัฒนานั้น Money Movement ถูกออกแบบโดยใช้หลักการออกแบบเชิงวัตถุซึ่งเป็นความรู้ใหม่ที่ผู้จัดทำไม่มีความรู้มากนัก อีกทั้งยังต้องใช้เทคโนโลยีใหม่ๆ เช่น Cloud service ในการพัฒนา ซึ่งทำให้ต้องใช้เวลาในทำการศึกษาและทดลองเป็นอย่างมาก ทำให้การพัฒนาทำได้ช้าไม่ตรงตามเวลาที่ได้วางแผนไว้ในตอนแรก แต่ท้ายที่สุดผู้พัฒนาก็สามารถทำการพัฒนา Money Movement จนสามารถมาใช้งานจริงได้ ซึ่งจากการพัฒนา Money Movement นั้นทำให้ผู้ใช้ได้ความรู้ใหม่มากมาย เช่น การออกแบบเชิงวัตถุ ,Cloud service, json , angularjs , spring framework,การทำ unit test สุดท้ายนี้ผู้พัฒนาหวังเป็นอย่างยิ่งว่า Money Movement นั้นจะสามารถนำไปใช้ในชีวิตประจำวัน เพื่อช่วยแก้ปัญหาในการใช้จ่ายได้

บรรณานุกรม

Google Inc. 2010-2012. “AngularJS.”
[ระบบออนไลน์]. แหล่งที่มา https://angularjs.org/ (20 ตุลาคม 2557)

Twitter Inc. 2010. “Bootstrap.”
[ระบบออนไลน์]. แหล่งที่มา http://getbootstrap.com/ (4 ตุลาคม 2557)

Microsoft Corporation 2010 “Azure.”
[ระบบออนไลน์]. แหล่งที่มา http://azure.microsoft.com/en-us/ (2 กันยายน 2557)

W3shcool.com. 2012. “JSON.”
[ระบบออนไลน์]. แหล่งที่มา http://www.w3schools.com/json/default.asp (25 กันยายน 2557)

Pivotal Software, Inc. 2014 “spring framework”
[ระบบออนไลน์]. แหล่งที่มา http://spring.io/ (30 กันยายน 2557)