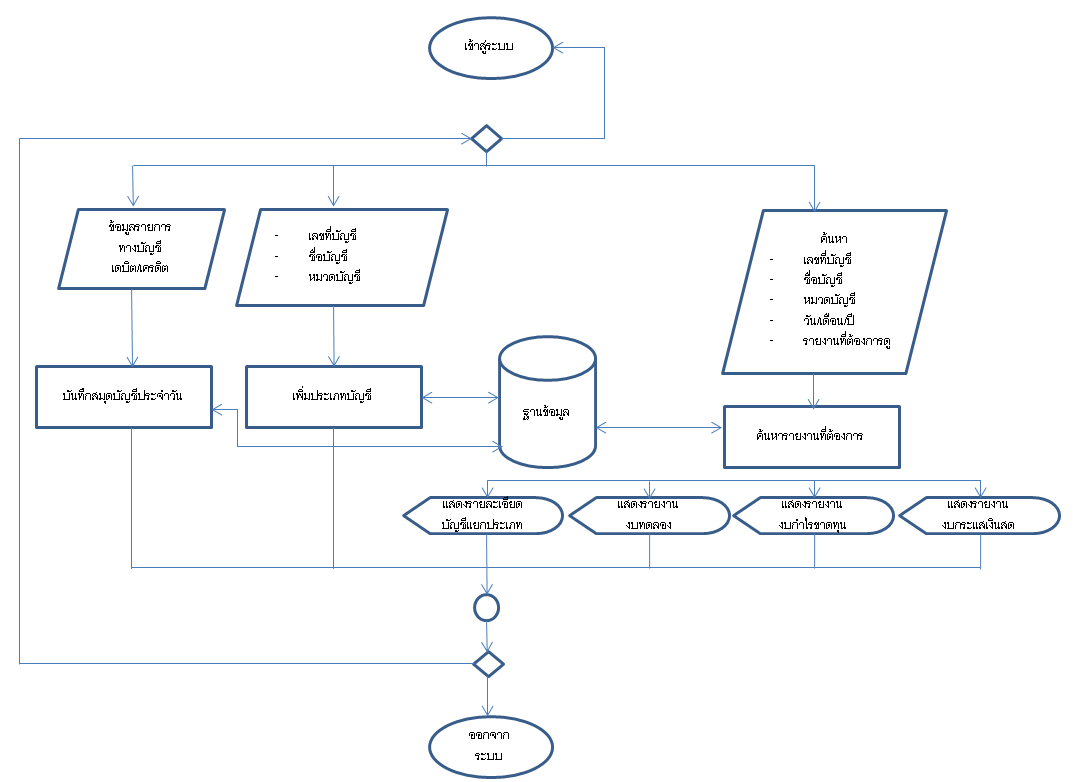
บทที่ 5

การพัฒนาระบบ

**Flowchart แผนโครงสร้างของกระบวนการทำงานของระบบ**

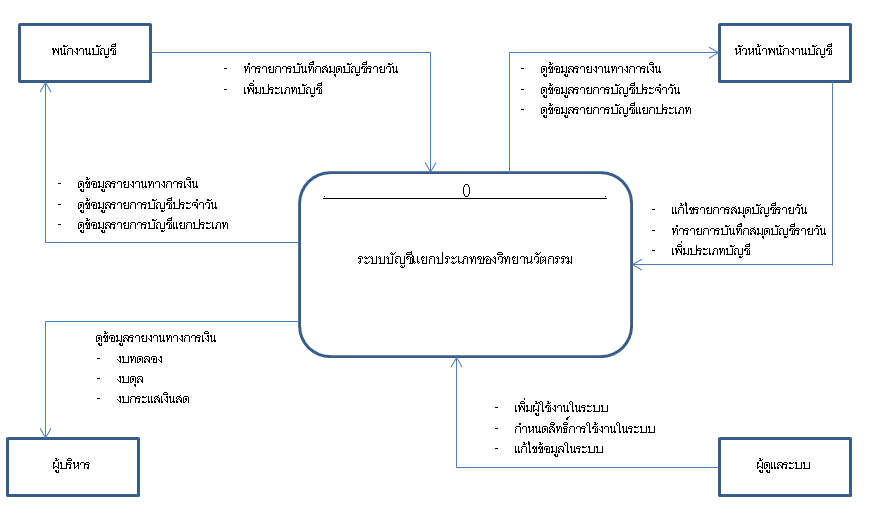
เป็นเครื่องมือแสดงขั้นตอน หรือกระบวนการทำงาน โดยใช้สัญลักษณ์ที่เป็นมาตรฐานเดียวกัน ซึ่งในสัญลักษณ์จะมีข้อความสั้น ๆ อธิบายข้อมูลที่ต้องใช้ ผลลัพธ์ หรือคำสั่งประมวลผลของขั้นตอนนั้น ๆ และเชื่อมโยงขั้นตอนเหล่านั้นด้วยเส้นที่มีลูกศรชี้ทิศทางการทำงานตั้งแต่เริ่มต้นจนจบกระบวนการ ซึ่งในกระบวนของระบบโปรแกรมบัญชีของวิทยาลัยจะมีรูปแบบดังนี้



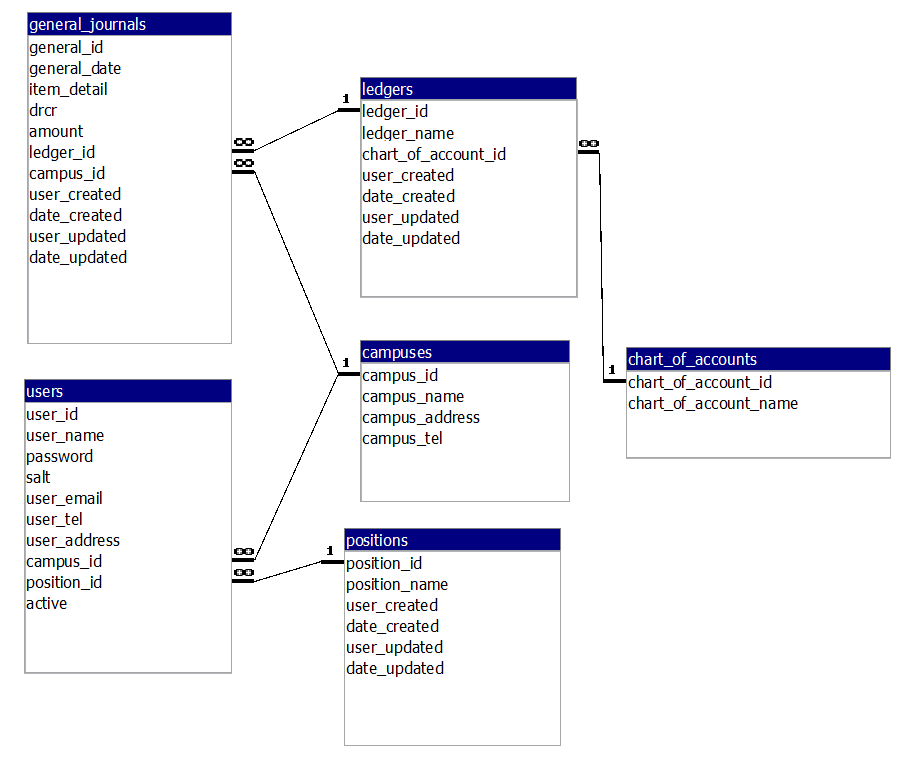
**แผนภาพกระแสของข้อมูล**

- Context Diagram ( Dataflow Diagram level 0)

แผนภาพกระแสข้อมูลระดับบนสุดที่แสดงภาพรวมการทำงานของระบบที่มีความสัมพันธ์กับสภาพแวดล้อมภายนอกระบบ และแสดงให้เห็นถึงรายละเอียดของ Process การทำงานหลักๆ ที่มีอยู่ภายในภาพรวมของระบบ (Context Diagram) ว่ามีขั้นตอนใดบ้าง



**การออกแบบฐานข้อมูล**



รายละเอียดตาราง

ตาราง : general\_journals  
คำอธิบาย : เป็นตารางที่ใช้เก็บข้อมูลรายการของสมุดบัญชีรายวัน

|  |  |  |
| --- | --- | --- |
| ชื่อฟิลด์ | ประเภทข้อมูล | คำอธิบาย |
| general\_id | Text | หมายเลขรายการ |
| general\_date | Date/Time | วันเดือนปี |
| item\_detail | Text | รายการ |
| drcr | Text | debit หรือ credit |
| amount | Currency | จำนวนเงิน |
| ledger\_id | Text | เลขที่บัญชี(แยกประเภท) |
| campus\_id | Integer | เลขที่วิทยาเขต(ระบุว่ารายการนี้เป็นของวิทยาเขตไหน) |
| user\_created | Text | คนที่สร้างรายการนี้ |
| date\_created | Date/Time | วันที่สร้างรายการนี้ |
| user\_updated | Text | คนที่แก้ไขรายการนี้ |
| date\_updated | Date/Time | วันที่แก้ไขรายการนี้ |

ตาราง : ledgers  
คำอธิบาย : เป็นตารางที่ใช้เก็บข้อมูลบัญชีแยกประเภท

|  |  |  |
| --- | --- | --- |
| ชื่อฟิลด์ | ประเภทข้อมูล | คำอธิบาย |
| ledger\_id | Text | เลขที่บัญชีแยกประเภท |
| ledger\_name | Text | ชื่อบัญชีแยกประเภท |
| chart\_of\_account\_id | Text | หมวดบัญชี |
| user\_created | Text | คนที่สร้างรายการนี้ |
| date\_created | Date/Time | วันที่สร้างรายการนี้ |
| user\_updated | Text | คนที่แก้ไขรายการนี้ |
| date\_updated | Date/Time | วันที่แก้ไขรายการนี้ |

ตาราง : chart\_of\_accounts  
คำอธิบาย : เป็นตารางที่ใช้เก็บข้อมูลเกี่ยวกับหมวดบัญชี

|  |  |  |
| --- | --- | --- |
| ชื่อฟิลด์ | ประเภทข้อมูล | คำอธิบาย |
| chart\_of\_account\_id | Text | รหัสหมวดบัญชี |
| chart\_of\_account\_name | Text | ชื่อหมวดบัญชี |

ตาราง : campuses  
คำอธิบาย : เป็นตารางที่ใช้เก็บข้อมูลวิทยาเขต

|  |  |  |
| --- | --- | --- |
| ชื่อฟิลด์ | ประเภทข้อมูล | คำอธิบาย |
| campus\_id | Integer | เลขที่วิทยาเขต |
| campus\_name | Text | ชื่อวิทยาเขต |
| campus\_address | Text | ที่อยู่วิทยาเขต |
| campus\_tel | Text | เบอร์โทรศัพท์วิทยาเขต |

ตาราง : users  
คำอธิบาย : เป็นตารางที่ใช้เก็บข้อมูลผู้ใช้งานในระบบ

|  |  |  |
| --- | --- | --- |
| ชื่อฟิลด์ | ประเภทข้อมูล | คำอธิบาย |
| user\_id | Text | ไอดีของผู้ใช้งาน |
| user\_name | Text | ชื่อของผู้ใช้งาน |
| password | Text | รหัสของผู้ใช้งาน |
| salt | Text | ข้อมูลสุ่มที่ใช้ในการเข้ารหัสของผู้ใช้งาน |
| user\_email | Text | อีเมล์ของผู้ใช้งาน |
| user\_tel | Text | เบอร์โทรของผู้ใช้งาน |
| user\_address | Text | ที่อยู่ของผู้ใช้งาน |
| campus\_id | Integer | รหัสวิทยาเขต |
| position\_id | Integer | รหัสตำแหน่งงาน |
| active | Text | สถานะการใช้งาน |

ตาราง : positions  
คำอธิบาย : เป็นตารางที่ใช้เก็บข้อมูลเกี่ยวกับตำแหน่งงาน

|  |  |  |
| --- | --- | --- |
| ชื่อฟิลด์ | ประเภทข้อมูล | คำอธิบาย |
| position\_id | Integer | รหัสตำแหน่ง |
| position\_name | Text | ชื่อตำแหน่ง |
| user\_created | Text | คนที่สร้างรายการนี้ |
| date\_created | Date/Time | วันที่สร้างรายการนี้ |
| user\_updated | Text | คนที่แก้ไขรายการนี้ |
| date\_updated | Date/Time | วันที่แก้ไขรายการนี้ |

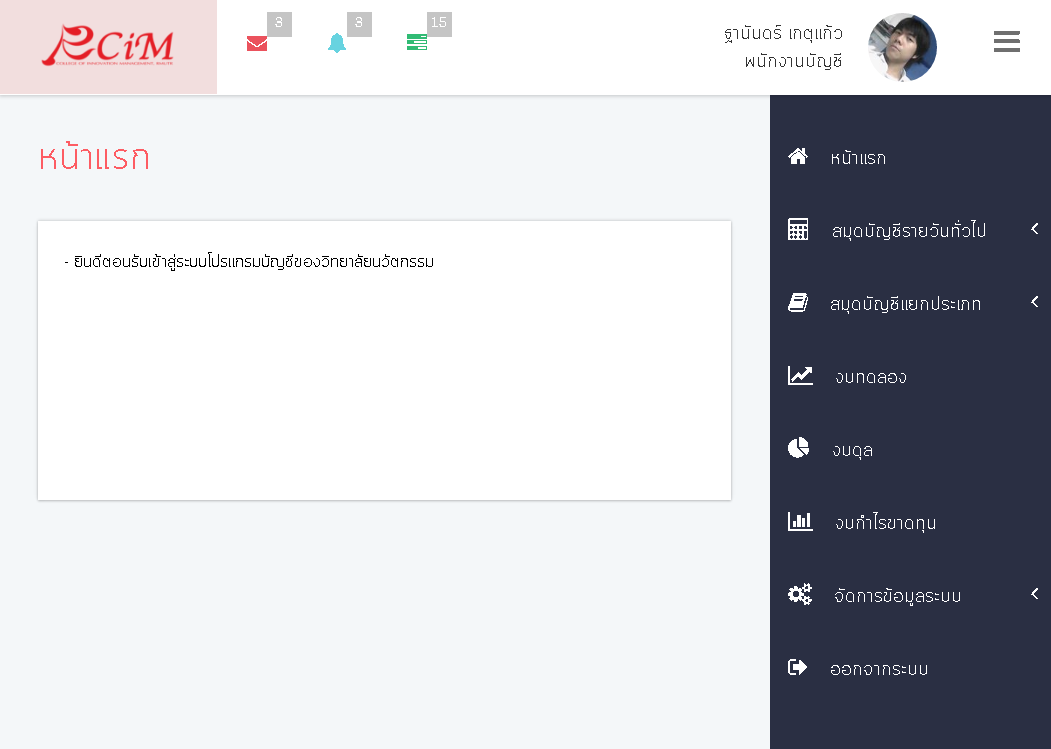
**การออกแบบหน้าจอการทำงานและรูปแบบของรายงาน**

ในการออกแบบ user interface จะใช้ Baxster Admin Panel เป็น Template หลักของ Application ซึ่งมีความสามารถในการทำ Responsive และมีรูปแบบที่สวยงาม ส่วนฟอนต์ภาษาไทยจะใช้ Cloud สร้างโดย Typomancer

หน้าจอ Login



หน้าจอหลัก

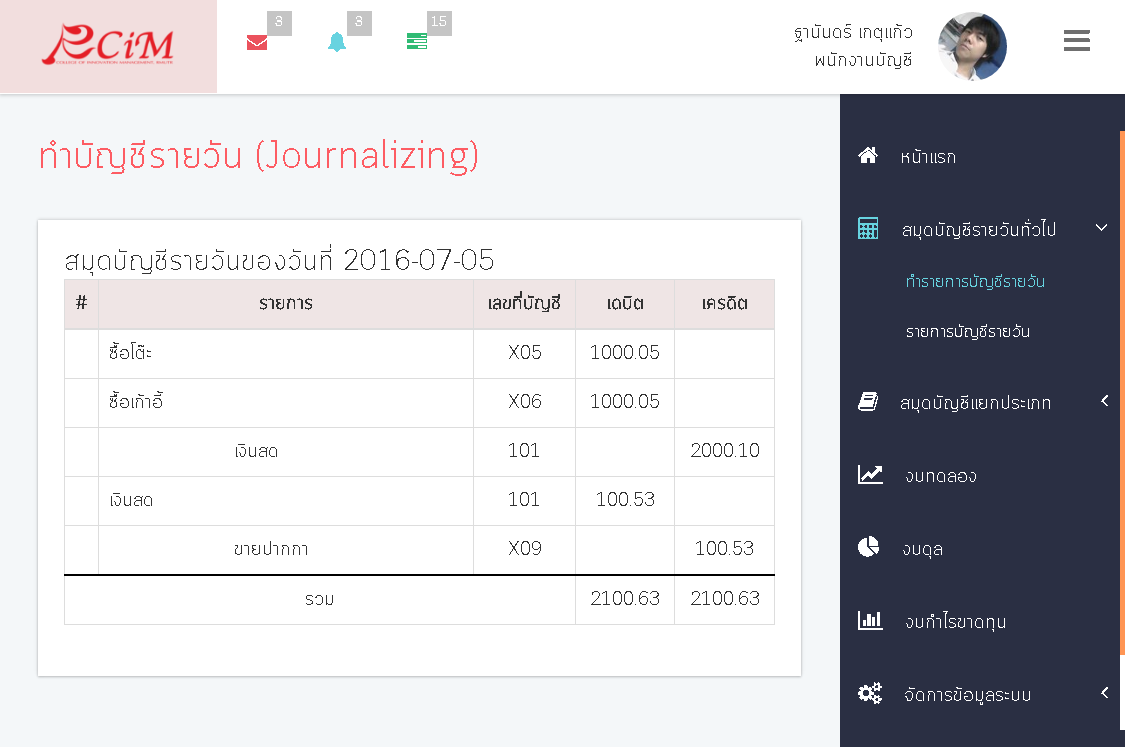


โดยในระบบจะมีการแบ่งเมนูการทำงานดังนี้

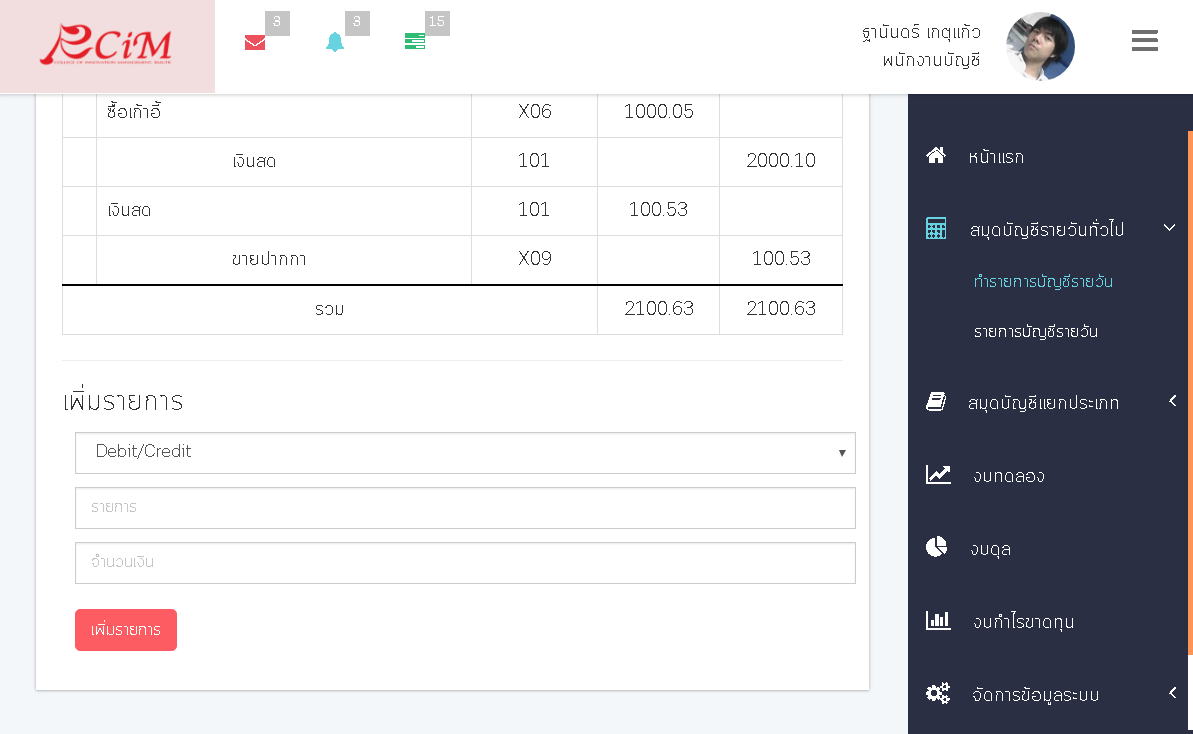
*หน้าแรก  
 |\_ สมุดบัญชีรายวันทั่วไป  
 |\_ ทำรายการบัญชีรายวัน  
 |\_ รายการบัญชีรายวัน*

*|\_ สมุดบัญชีแยกประเภท  
 |\_ งบทดลอง  
 |\_ งบดุล  
 |\_ งบกำไรขาดทุน  
 |\_ จัดการข้อมูลระบบ  
 |\_ จัดการข้อมูลผู้ใช้  
 |\_ ออกจากระบบ*

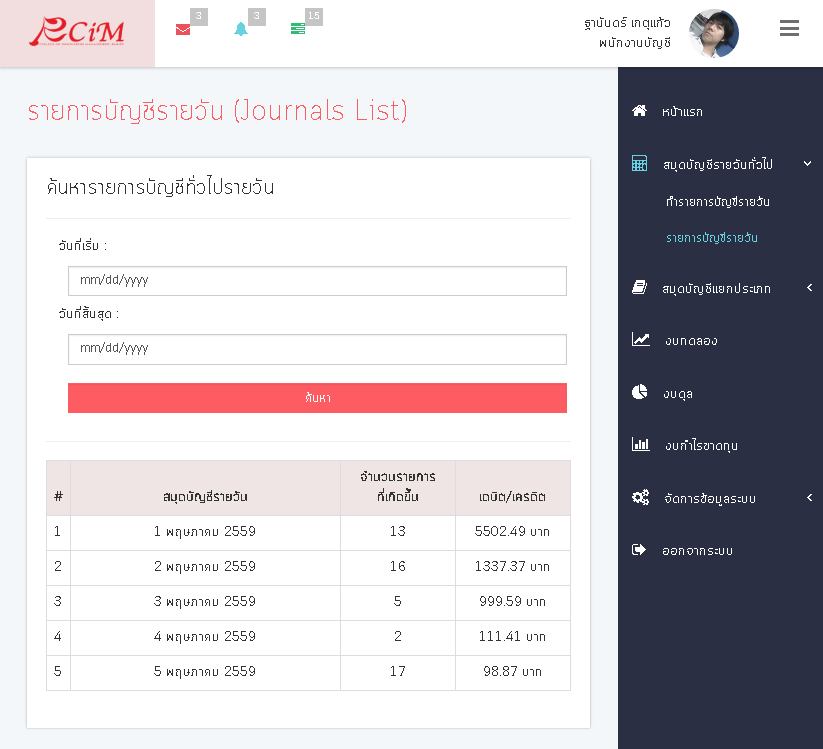
หน้าจอการทำบัญชีรายวัน



จะมีส่วนที่ไว้ในการเพิ่มรายการ ลงไปในตารางสมุดรายวัน

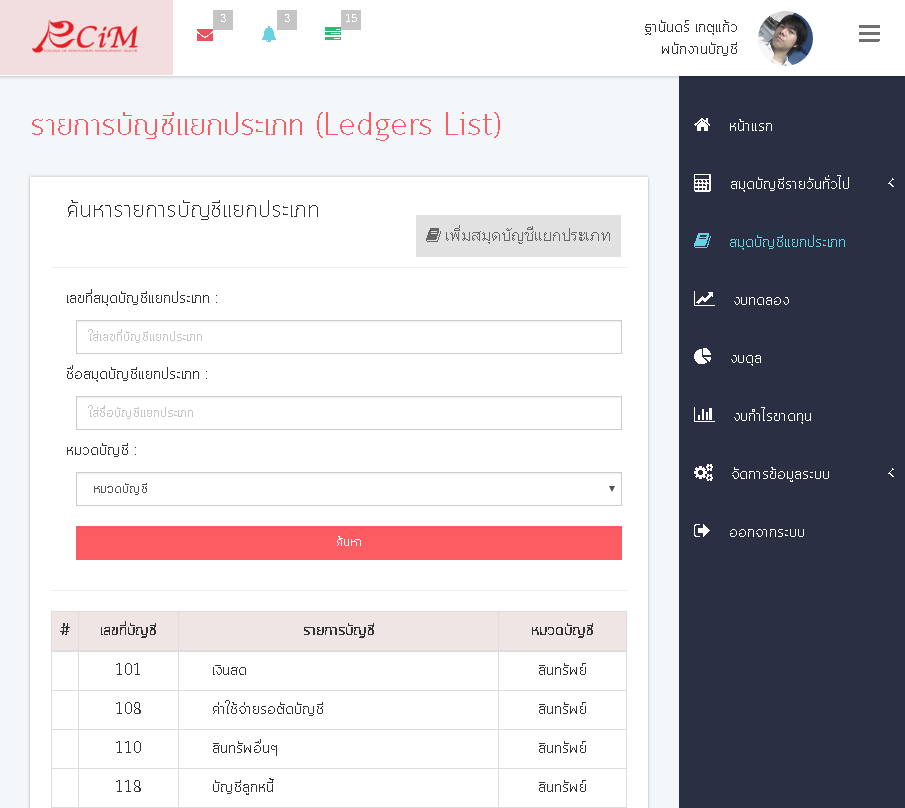


หน้าจอรายการบัญชีรายวัน

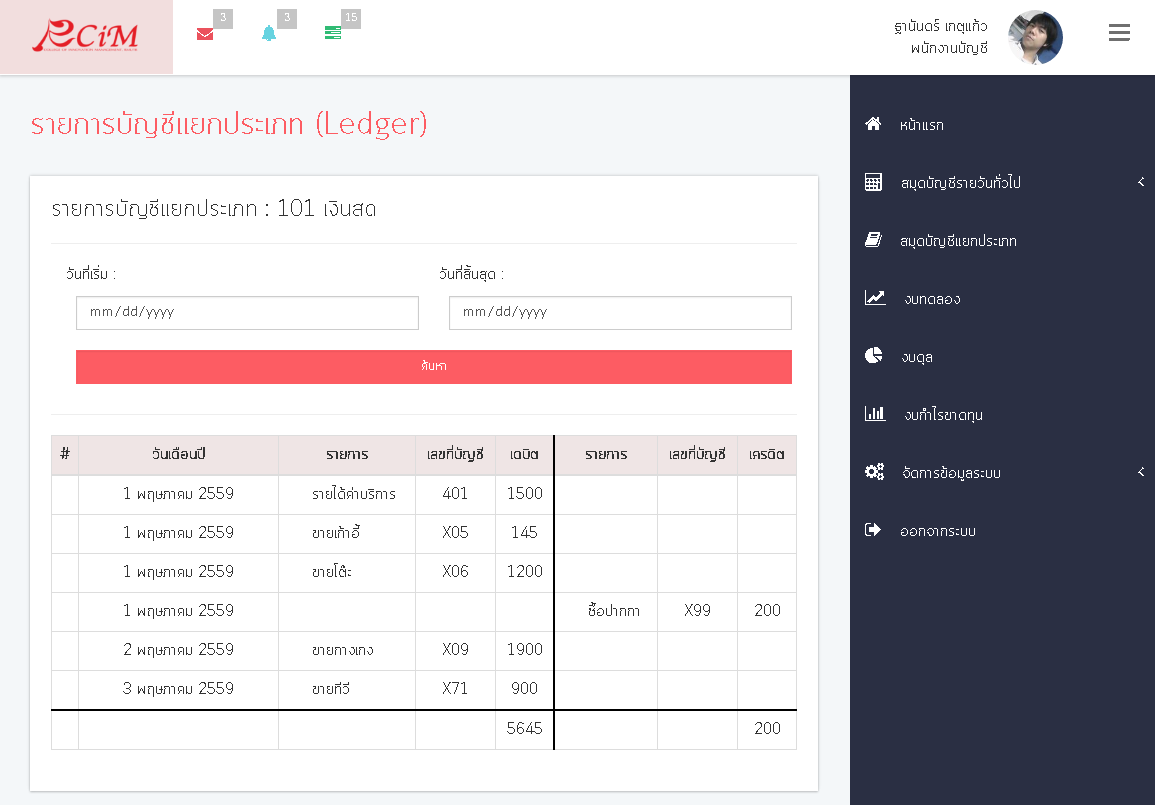


* สามารถกำหนดช่วงเวลาในการค้นหาสมุดบัญชีรายวันของวันที่ต้องการ
* เมื่อเลือกสมุดบัญชีรายวันที่ต้องการ โดยกดเข้าไปในแถวของตารางจะแสดงรายละเอียดของสมุดบัญชีรายวันของวันที่นั้นๆที่เราได้เลือก ( เหมือนหน้าจอทำบัญชีรายวัน )

หน้าจอรายการบัญชีแยกประเภท

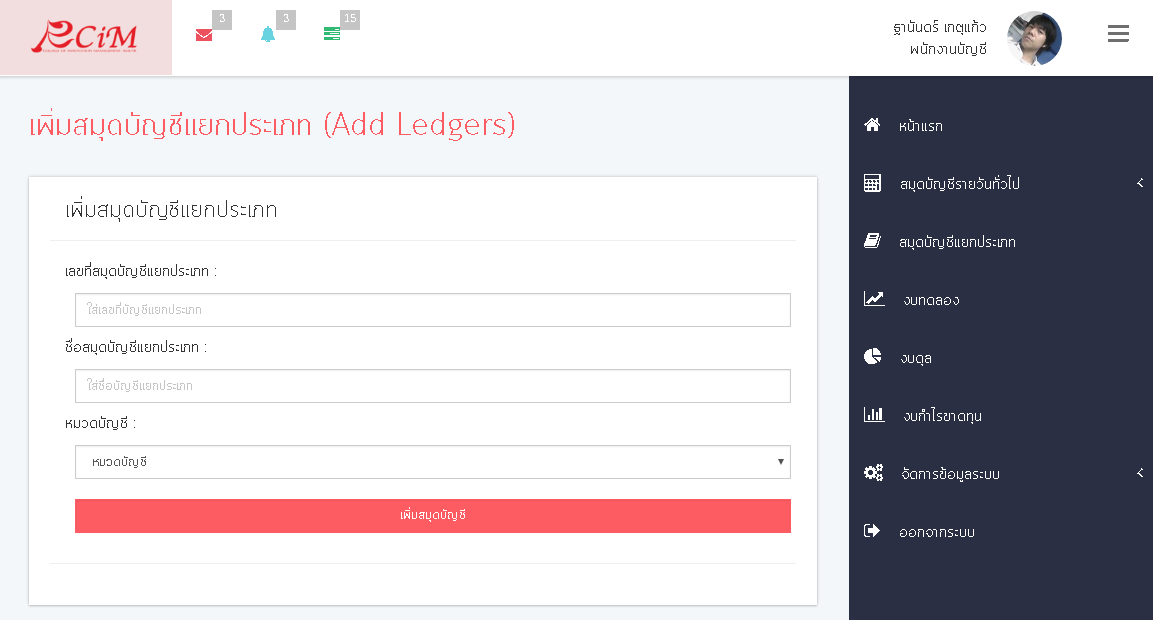


สามารถค้นหาบัญชีจาก เลขที่ ชื่อบัญชี หมวดบัญชี ได้เมื่อคลิกที่แถวของประเภทบัญชีที่ต้องการจะปรากฏบัญชีนั้นๆ

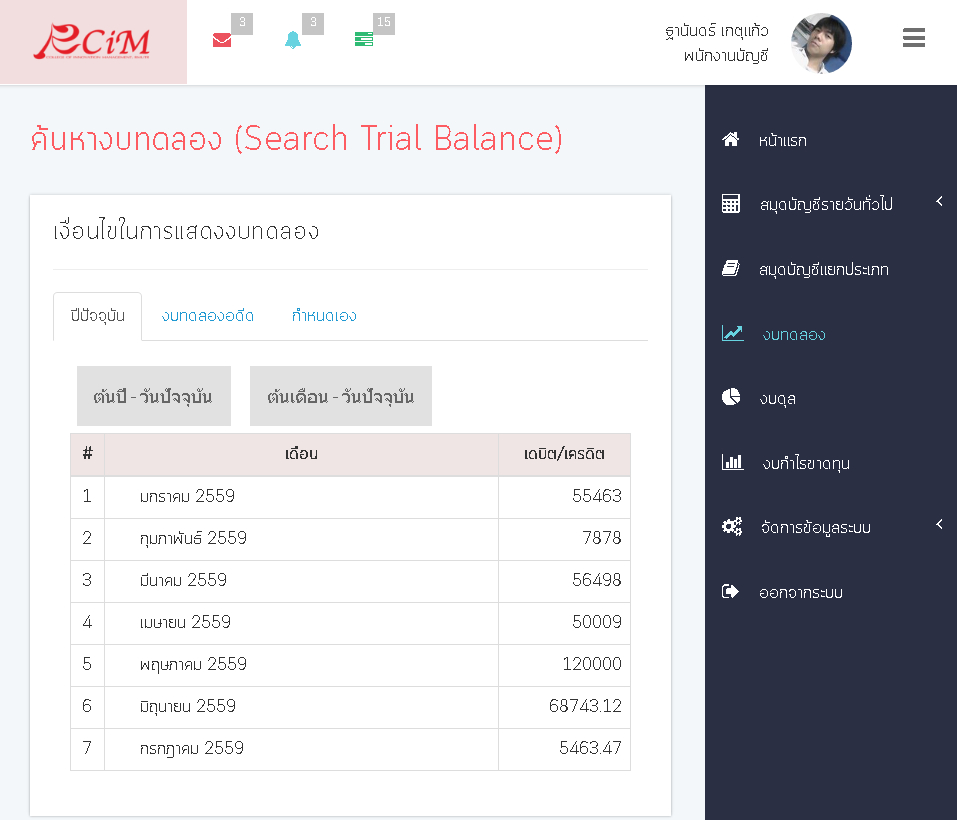


สามารถกรองข้อมูลที่ต้องการได้จากการกำหนดช่วงของเวลา

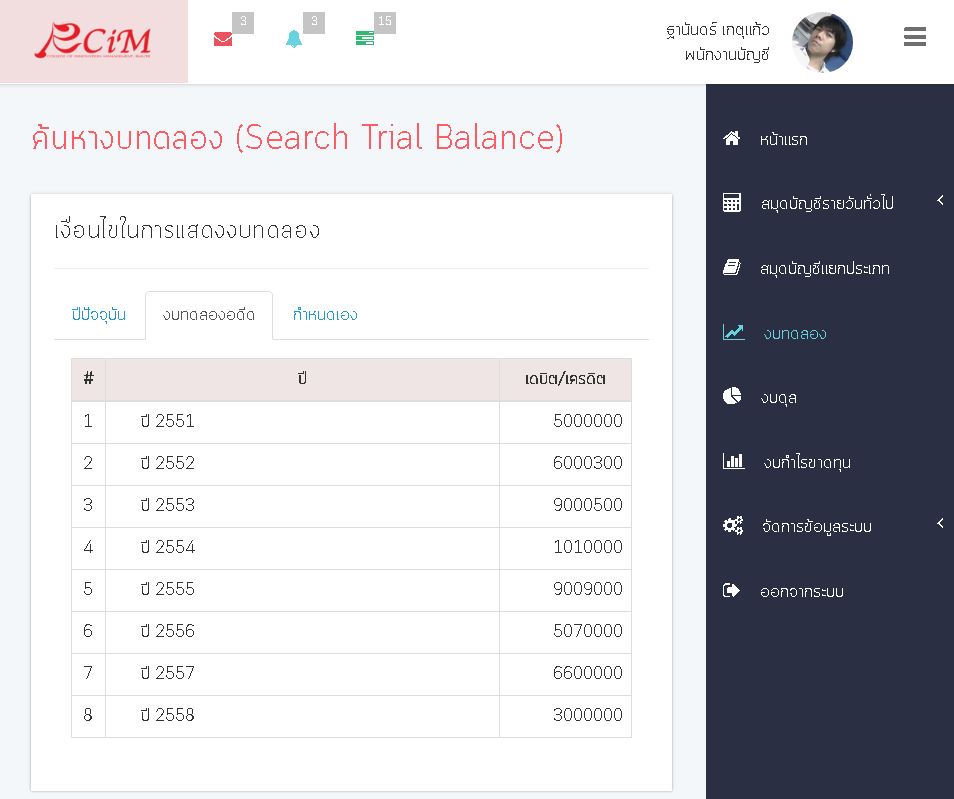
หน้าจอการเพิ่มสมุดบัญชีแยกประเภท



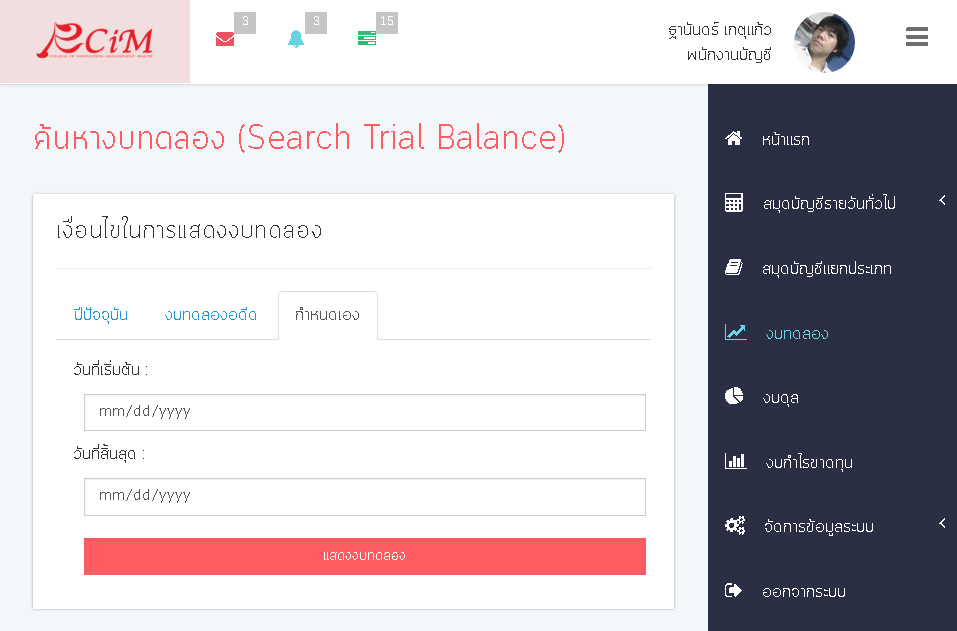
หน้าจอแสดงงบทดลอง(ค้นหา)



- เมื่อเลือกคลิกที่เดือนที่ต้องการจะแสดงงบทดลองของเดือนนั้นๆ  
- สามารถค้นหาตามเงื่อนไขต่างๆ โดยแบ่งออกเป็น 3 แทบ ได้แก่ การค้นหาตามปีปัจจุบัน งบทดลองในอดีต และรูปแบบเงื่อนไขที่กำหนดเอง

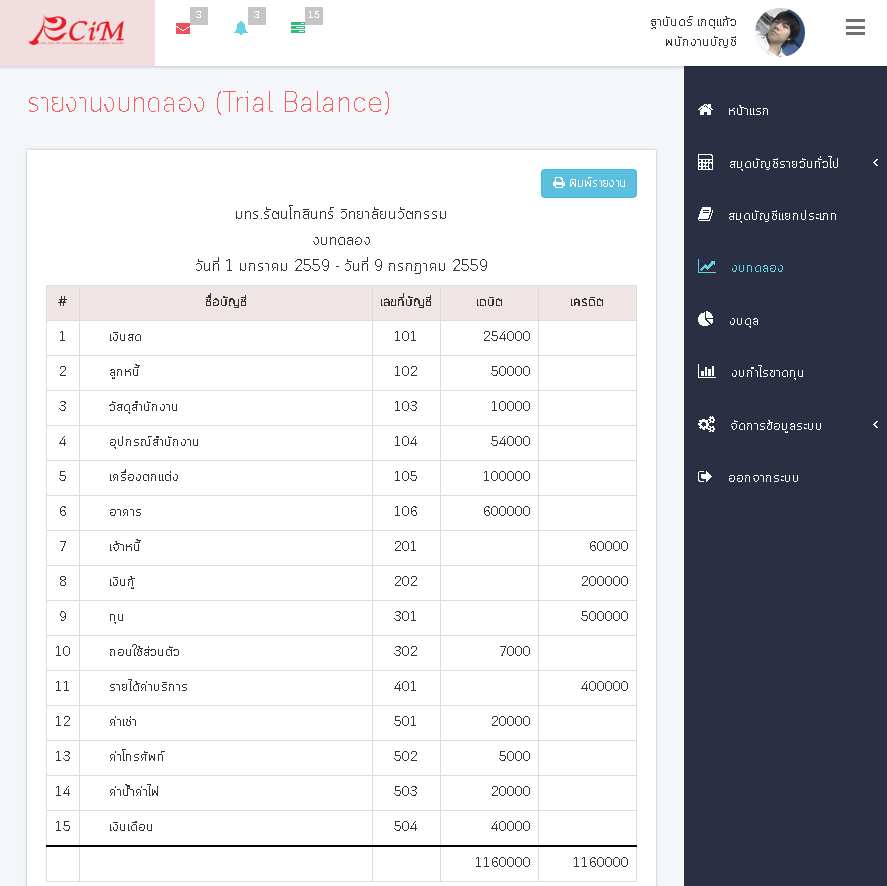


- เมื่อเลือกคลิกที่เดือนที่ต้องการจะแสดงงบดุลของปีนั้นๆ



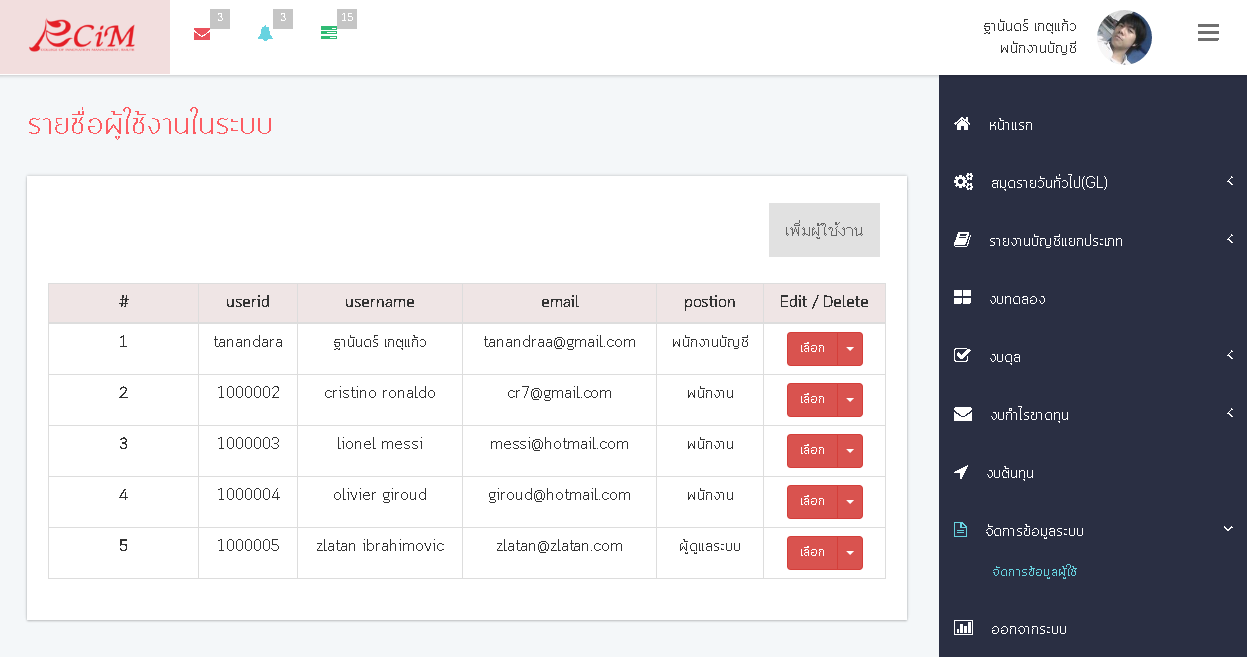
- ในแถบ กำหนดเอง สามารถกำหนดการแสดงงบทดลอง ตามวันที่กำหนดได้

หน้าจอแสดงงบทดลอง



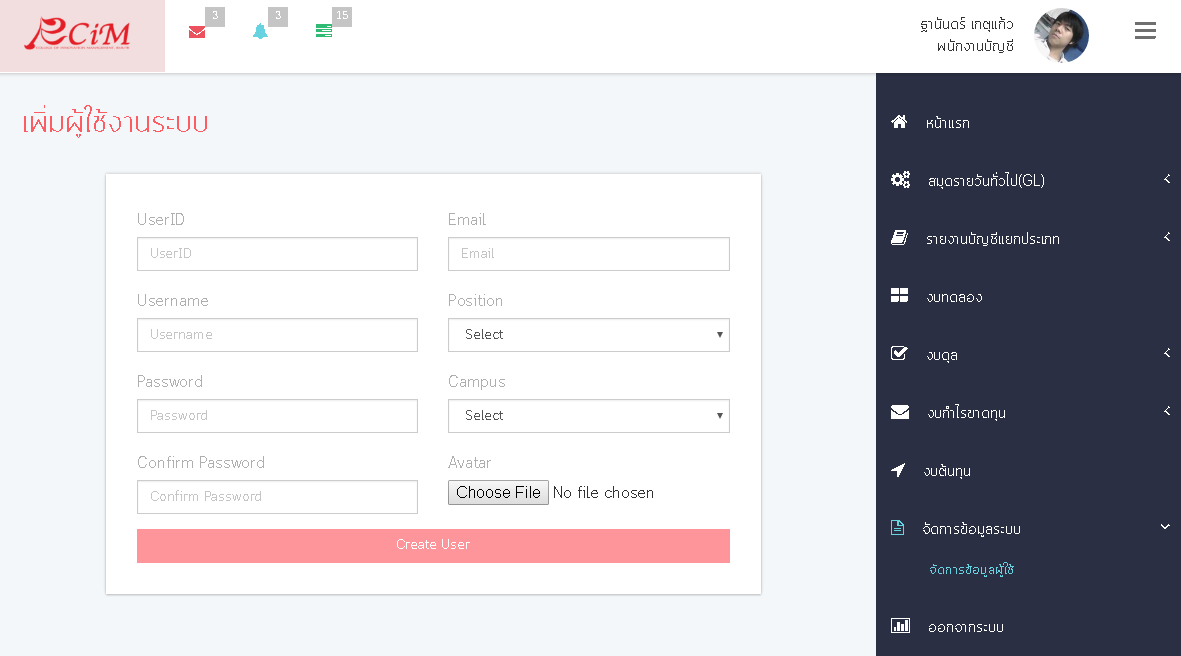
- แสดงรายการทั้งหมดของช่วงเวลาที่ได้ทำการเลือก  
 - แสดงยอดรวมของเดบิต เครดิต  
 - สามารถจัดพิมพ์รายงานได้

หน้าจอจัดการข้อมูลผู้ใช้งาน



- สามารถเลือก เพิ่ม ลบ แก้ไข ข้อมูลของผู้ใช้งานได้

หน้าจอเพิ่มผู้ใช้งาน



**เทคโนโลยีที่ใช้พัฒนาระบบ**

**Nginx**  
เอ็นจิ้นเอ็กถูกพัฒนาขึ้นและใช้งานครั้งแรกในปี 2002 มีหน้าที่เป็น เว็บเซิร์ฟเวอร์ ที่ใช้ทรัพยากรของเครื่องที่น้อยกว่า เช่น แรมและ ซีพียู ทำให้ เซิฟเวอร์ สามารถรองรับการทำงานได้มากยิ่งขึ้น เช่น การรองรับการเข้าใช้งานมากกว่า 10,000 การเชื่อมต่อพร้อมกัน การทำเซิร์ฟเวอร์เสมือน รองรับการทำงานของ TLS/SSL จึงเป็นเหตุผลที่เลือกใช้งานเอ็นจิ้นเอ็ก

**ExpressJS**  
ExpressJS เป็น Web Application Framework ที่ได้รับความนิยม ทำงานบน NodeJS ซึ่งใช้เป็น Back-End Web Framework ของฝั่งเซิร์ฟเวอร์ สามารถนำมาพัฒนาแอพพลิเคชั่นต่อไปได้ โดย ExpressJS นั้นจะใช้การ Routing (การกำหนดเส้นทางของระบบ) และ Middleware (การรับส่งข้อมูลของระบบ) สามารถเขียนได้ในรูปแบบ MVC ( Model View Controller ) และใช้ในการเชื่อมต่อกับฐานข้อมูล โดยในการพัฒนานั้นจะใช้ภาษา Javascript ในการเขียนโปรแกรม

**NodeJS**  
NodeJS คือแพลตฟอร์มที่สร้างโดย Javascript ทำให้สามารถใช้ภาษา Javascript ในการทำงานได้นอกเหนือจากการทำงานบนเว็บเบราเซอร์ สาเหตุที่เลือกใช้ NodeJS เนื่องจากต้องการใช้ภาษา Javascript ในการพัฒนาแอพพิลเคชั่น

**AngularJS**AngularJS เป็น Front-End Web Framework ที่พัฒนาโดย Google ซึ่งจะใช้ภาษา Javascript เป็นหลัก ทำให้ Web Developer สามารถเรียนรู้ เข้าถึงและจัดการส่วน Front End ได้ง่ายขึ้นทั้งในเรื่อง ส่วนจัดการรูปแบบของเว็บและการแสดงข้อมูล

**Twitter Bootstrap**Bootstrap เป็น Responsive Web Framwork ที่ช่วยในการให้เว็บไซต์รองรับการแสดงผลเว็บไซต์สำหรับหน้าจอที่มีขนาดแตกต่างกันให้อยู่ในรูปแบบที่เหมาะสม เช่น มือถือ สมาร์ทโฟน แท็บเล็ท

**MySQL**MySQL (มายเอสคิวแอล) เป็นระบบจัดการฐานข้อมูลเชิงสัมพันธ์ (Relational Database Management System) โดยใช้ภาษา SQL เป็นฐานข้อมูลที่มีความนิยมสูง โดยในการพัฒนานั้นจะใช้ MySQL ในการเก็บข้อมูลต่างๆ และนำมาประมวลผลโดยใช้คำสั่ง SQL ในการ Query ข้อมูล เพื่อให้ได้ข้อมูลที่ต้องการ

**ความปลอดภัยของระบบ**

* SSL  
  เป็นโปรโตคอลในการสื่อสารหรือส่งข้อมูลอย่างปลอดภัยบนเครือข่ายคอมพิวเตอร์และอินเตอร์เน็ต ที่ใช้กันส่วนใหญ่ได้แก่ การท่องเว็บไซต์ เช็คอีเมล์ ทำธุรกรรมออนไลน์ เป็นต้น ผู้ใช้ทั่วไปเข้าเว็บไซต์ที่มีการเข้ารหัสความปลอดภัยและมีใบรับรอง SSL Certificates จะสังเกตเห็น url ขึ้นต้นด้วย https:// และมีสัญลักษณ์รูปแม่กุญแจอยู่ ซึ่งจำเป็นจะต้องมีค่าใช้จ่าในการขอ Certification Authority (CA) จากผู้ให้บริการ แต่ในขณะนี้มีโครงการ Let’s Encrypt ที่เป็นโครงการที่ทำให้เราสามารถออก Certification Authority (CA) ได้ฟรีโดยไม่มีค่าใช้จ่าย
* IPS  
  Intrusion Prevention System (IPS) คือ ระบบที่คอยตรวจจับการบุกรุกของผู้ที่ไม่ประสงค์ดี โดยจะทำการวิเคราะห์ข้อมูลทั้งหมดที่ผ่านเข้าออกภายในเครือข่ายว่า มีลักษณะการทำงานที่เป็นความเสี่ยงที่ก่อให้เกิดความ เสียหายต่อระบบเครือข่ายหรือไม่ แล้วจึงจะทำการแจ้งเตือนให้ผู้ดูแลระบบทราบ เมื่อใดที่พบข้อมูลที่มีลักษณะการทำงานที่เป็นความเสี่ยงต่อระบบเครือข่ายก็จะทำการป้องกันข้อมูลดังกล่าวนั้น ไม่ให้เข้ามาภายในเครือข่ายได้
* Web Application Firewall

Web Application Firewall (WAF) เป็นอาจจะเป็นได้ทั้งซอฟต์แวร์ที่เอามาลงไว้ในเซิร์ฟเวอร์หรือเป็นพวกอุปกรณ์ ซึ่งจะทำหน้าทีตรวจจับ HTTP Traffic ถูกออกแบบมาเฉพาะเจาะจงในการป้องกัน การโจมตีที่เกิดขึ้นกับ Web Application และยังสามารถตรวจพบเหตุการณ์หรือสิ่งผิดปกติได้ในปริมาณและรายละเอียดที่มากกว่า IPS

* Server Hardening

การทำ Hardening คือการลดความเสี่ยงที่จะเกิดด้านความปลอดภัยให้ได้มากที่สุด เช่น แก้ใขจุดบกพร่องของระบบ แอพพลิเคชั่น หรือ ฐานข้อมูล กระบวนการเสริมความแข็งแรงด้านความปลอดภัยให้กับเครื่องเซิร์ฟเวอร์

* การ Backup ข้อมูล

การ Backup หรือการสำรองข้อมูลนั้นเป็นการคัดลอกข้อมูลเพื่อทำสำเนาเก็บไว้ เพื่อหลีกเลี่ยงความเสียหายที่จะเกิดขึ้น หากข้อมูลเกิดการเสียหายหรือสูญหาย โดยสามารถนำข้อมูลที่สำรองไว้มาใช้งานได้ทันที

**การทดสอบระบบ**

* Unit test - เป็นการทดสอบในระดับฟังค์ชั่นการทำงาน เพื่อเป็นการยืนยันการทำงานระดับย่อยที่สุดว่าทำงานได้ถูกต้อง เป็นการทดสอบโดยโปรแกรมเมอร์ผู้เขียนโค๊ด
* Acceptant test - เป็นการทดสอบโดยผู้ใช้ (End user) หรือโดยลูกค้า เพื่อดูว่าซอฟต์แวร์ทำงานถูกต้องตามต้องการหรือไม่
* Usability test - เป็นการทดสอบการใช้งานซอฟต์แวร์โดยผู้เชียวชาญส่วนติดต่อผู้ใช้ ว่าใช้งานได้ง่าย เข้าใจได้ง่าย หรือไม่

**ความต้องการของระบบ**

**Server**

* Operating System : Ubuntu Server 16.04 LTS
* Harddisk : 20 GB
* Memory : 1 GB
* CPU : 1 Core

**Client**

* Web browser ( Google Chrome , Firefox , Safari , Internet Explorer 11 )
* Internet