ระบบประมูลและขายสินค้าเพื่อการกุศล

AUCTION AND SELLING FOR CHARITY SYSTEM

ติณณ์ จิตต์จนะ

นลธวัช หนูมอ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2560

ระบบประมูลและขายสินค้าเพื่อการกุศล

AUCTION AND SELLING FOR CHARITY SYSTEM

ติณณ์ จิตต์จนะ

นลธวัช หนูมอ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2560

ปริญญานิพนธ์ปีการศึกษา 2560

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบประมูลและขายสินค้าเพื่อการกุศล

AUCTION AND SELLING FOR CHARITY SYSTEM

ผู้จัดทำ

1. นายติณณ์ จิตต์จนะ รหัสนักศึกษา 57010494
2. นายนลธวัช หนูมอ รหัสนักศึกษา 57010669

_____ อาจารย์ที่ปรึกษา

(อาจารย์ บัณฑิต พัสยา)

_____ อาจารย์ที่ปรึกษาร่วม

(ผศ. ดร.ชุติเมษฎ์ ศรีนิลทา)

ระบบประมูลและขายสินค้าเพื่อการกุศล

นายติณณ์	จิตต์จนะ	57010494
นายนลธวัช	หนูมอ	57010669
อ.บัณฑิต	พัสยา	อาจารย์ที่ปรึกษา
ผศ. ดร.ชุติเมษฎ์	ศรีนิลทา	อาจารย์ที่ปรึกษาร่วม

ปีการศึกษา 2560

บทคัดย่อ

ปัจจุบันจำนวนผู้ใช้งานอินเทอร์เน็ตมากขึ้นในทุกวัน ข้อดีของอินเทอร์เน็ตคือเข้าถึงได้ง่าย และสามารถกระจายข่าวสาร ข้อมูลต่างๆได้อย่างกว้างขวาง ซึ่งทำให้เกิดเครือข่ายสังคมออนไลน์ หรือ Social Network เช่น Facebook, Twitter เป็นต้น ที่มีผู้ใช้งานเป็นจำนวนมาก และได้มีการนำมาใช้ประโยชน์มากมาย อย่างหนึ่งคือการประกาศจำหน่ายสินค้า หรือจัดการประมูลขึ้นเพื่อนำเงินไปช่วยเหลือการกุศล แต่เครือข่ายสังคมนั้นมีขนาดใหญ่มากทำให้ข้อมูลต่างๆกระจัดกระจาย ถ้าผู้ใช้ต้องการช่วยเหลือในองค์กรการกุศลที่เจาะจงการค้นหาทำได้ยาก หรือต้องการนำสินค้ามาจำหน่ายแต่ตนเองไม่เป็นที่รู้จักอาจจำหน่ายสินค้าไม่ได้ และเมื่อช่วยเหลือองค์กรการกุศลไปแล้วการติดตามความคืบหน้าขององค์กรการกุศลก็อาจทำได้ยาก

เพื่อแก้ปัญหาข้างต้นจึงจัดทำเป็นระบบเพื่อรวบรวมสินค้าที่นำมาจำหน่ายหรือประมูล ที่ผู้ใช้สามารถดูรายการสินค้าได้ และนำสินค้ามาจำหน่ายหรือเปิดประมูลได้ รายได้จากสินค้าจะนำเงินไปช่วยเหลือองค์กรการกุศลต่างๆที่ผู้จำหน่ายเลือกไว้ ซึ่งการเพิ่มองค์กรการกุศลเข้าสู่ระบบจำเป็นต้องกรอกข้อมูลที่เป็นทางการเพื่อความน่าเชื่อถือ องค์กรการกุศลที่ถูกเพิ่มเข้ามาในระบบจะอยู่ในรายการที่ผู้ใช้สามารถดูรายละเอียด และความคืบหน้าได้

AUCTION AND SELLING FOR CHARITY SYSTEM

Mr. Tinn	Jittjana	57010494
Mr. Nontawat	Numor	57010669
Prof. Bundit	Pasaya	Advisor
Asst.Prof.Dr. Chutimet	Srinilta	Co-Advisor

Academic year 2560

Abstract

Nowadays, the number of Internet users increasing more and more every day. The benefits of it are easily accessible and broadcasting Information widely. Which gives rise to a Social Network such as Facebook, Twitter, etc. that have been used a lot and have implemented numerous uses. One is to announce the sale of goods or auction to help a charity, but the social network is very large making information scattered. It is difficult to search the charity if the user wants to help with specific charity or to sell their goods when user is not well-known person. Moreover, sometimes it is hard to track the progress of the charitable organization.

To solve the above problems, thus developing a system that is a hub for announcing sell or auction goods. Revenue from goods bring money to help charities that vendors chosen, and user can view information and progress of charities. Before the charitable organizations is added in the system, they must fill the form with official information showing their credibility.

สารบัญ

หน้า

สารบัญ

หน้า

สารบัญ

หน้า

สารบัญ

หน้า

สารบัญรูปภาพ

รูป หน้า

สารบัญตาราง

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

การบริจาค เป็นการให้โดยบุคคลหรือนิติบุคคล มักเพื่อวัตถุประสงค์การกุศล หรือเพื่อสงเคราะห์เหตุอย่างใดอย่างหนึ่ง การบริจาคมีได้หลายรูปแบบ รวมถึงการเสนอเงินสด, บริการ, สินค้าใหม่ หรือสินค้าใช้แล้ว เช่น เสื้อผ้า, อาหาร เป็นต้น และการจำหน่ายสินค้า หรือนำสินค้ามาประมูลและนำรายได้ไปช่วยเหลือองค์กรการกุศล

ปัจจุบันอินเทอร์เน็ตสามารถเข้าถึงได้ง่าย จึงมีการประมูล หรือจำหน่ายสินค้าผ่านอินเทอร์เน็ต ทำให้ผู้ซื้อหรือผู้เข้าร่วมประมูลใช้งานได้สะดวกขึ้น และผู้ขายหรือผู้นำสินค้ามาประมูลไม่จำเป็นต้องจำหน่ายที่ร้าน หรือจัดสถานที่สำหรับประมูล และสามารถประชาสัมพันธ์สินค้าของตนเองได้ แต่การทำระบบประมูลสินค้าผ่านอินเทอร์เน็ต หรือการทำร้านขายสินค้าออนไลน์ยุ่งยากสำหรับคนทั่วไป และถ้าหากผู้ซื้อหรือผู้เข้าร่วมประมูล ต้องการช่วยเหลือองค์กรการกุศลหนึ่ง การหาสินค้าที่ช่วยเหลือในอินเทอร์เน็ตอาจหาได้ยาก หรือต้องการทราบถึงความคืบหน้าขององค์กรการกุศลที่ตนเองได้ช่วยเหลืออาจหาข้อมูลได้ยาก และผู้ขายที่ยังไม่เป็นที่รู้จักอาจขาดความน่าเชื่อถือสำหรับผู้ซื้อได้

จากปัญหาข้างต้นจึงเกิดเป็นโครงงานทำเว็บไซต์รวบรวมการประมูลสินค้า และจำหน่ายสินค้าเพื่อช่วยเหลือองค์กรการกุศลต่างๆ โดยผู้ใช้สามารถนำสินค้ามาจำหน่ายหรือตั้งประมูล รวมถึงเลือกองค์กรการกุศลที่ต้องการนำเงินไปช่วยเหลือเองได้ โดยเว็บไซต์ได้มีการจัดหมวดหมู่สินค้า และรวบรวมสินค้าที่นำรายได้ไปช่วยเหลือองค์กรการกุศลเดียวกันเพื่อให้สะดวกต่อผู้ช่วยเหลือที่ต้องการสินค้า หรือช่วยเหลือองค์กรการกุศลที่ต้องการ และเพื่อแก้ไขปัญหาความน่าเชื่อถือ ผู้ซื้อสามารถดูข้อมูลและประวัติการจำหน่ายของผู้ขายได้ และดูรายละเอียด ความคืบหน้าขององค์กรการกุศลได้

1.2 เป้าหมายของโครงงาน

1) พัฒนาเว็บไซต์สำหรับประมูลและจำหน่ายสินค้า เพื่อนำรายได้จากการจำหน่ายไปช่วยเหลือองค์กรการกุศลได้
2) ผู้ใช้สามารถนำสินค้ามาจำหน่าย หรือประมูลผ่านเว็บไซต์ได้ด้วยตนเอง

1.3 วัตถุประสงค์ของโครงงาน

ศึกษา พัฒนาระบบประมูลและจำหน่ายสินค้าผ่านเว็บไซต์ที่ผู้ใช้สามารถนำสินค้ามาลงได้ด้วยตนเอง และสร้างความน่าเชื่อถือของผู้ขายให้กับผู้ซื้อได้ จากนั้นนำรายได้จากการจำหน่ายหรือประมูลไปช่วยเหลือองค์กรการกุศลต่างๆ ได้ถูกต้อง

1.4 สิ่งที่คาดว่าจะได้รับ

1) ผู้ใช้สามารถนำสินค้ามาจำหน่ายและประมูลผ่านเว็บไซต์ได้
2) ผู้ใช้หาสินค้าที่ต้องการ หรือสินค้าที่นำรายได้ไปช่วยองค์กรการกุศลที่ต้องการได้สะดวกและรวดเร็ว
3) ได้ความรู้เกี่ยวกับขั้นตอนการทำงานของระบบจำหน่าย และประมูลสินค้าออนไลน์
4) ได้รับความรู้เกี่ยวกับการติดต่อหน่วยงานสำหรับการรับบริจาค
5) ได้รับความรู้เกี่ยวกับการสร้างความน่าเชื่อถือให้แก่ผู้ขายและผู้ซื้อ

1.5 ขอบเขตของงาน

1) ระบบสามารถเพิ่มสินค้าที่นำมาจำหน่าย หรือประมูลได้ตามหมวดหมู่ที่กำหนด
2) เว็บไซต์สามารถใช้งานได้บนคอมพิวเตอร์ และโทรศัพท์มือถือ
3) สามารถชำระเงินด้วยการโอนเงินผ่านบัญชีธนาคาร และระบบพร้อมเพย์ของประเทศไทย

1.6 หลักการทำงานของระบบ

การทำงานของระบบจะแบ่งเป็น 2 ส่วนคือส่วนของผู้นำสินค้ามาลงในเว็บไซต์ และส่วนของผู้ใช้งานเว็บไซต์ทั่วไป

ในส่วนของผู้นำสินค้ามาลงในเว็บไซต์จะต้องเลือกว่าสินค้าที่เพิ่มเข้าระบบว่าจะจำหน่าย หรือประมูล หากเลือกการจำหน่ายจะต้องกรอกรายละเอียดของสินค้าที่นำมาลงเช่น ชื่อสินค้า จำนวนสินค้าที่มี ราคาที่ต้องการจำหน่าย เป็นต้นและเลือกองค์กรการกุศลที่จะนำรายได้จากการจำหน่ายไปช่วยเหลือ ผู้ขายสามารถจัดการสินค้าที่ตนเองนำมาจำหน่ายได้จากหน้าจัดการสินค้าของตนเอง หากผู้นำสินค้ามาลงเลือกการประมูลจะต้องกำหนดรายละเอียดสินค้าเหมือนการจำหน่าย และต้อง

กรอกข้อมูลเพิ่มได้แก่ระยะเวลาการประมูล ราคาเริ่มต้น ต่อมาจึงเลือกองค์กรการกุศลที่จะนำรายได้จากการประมูลไปช่วยเหลือ

ในส่วนของผู้ใช้งานเว็บไซต์สามารถเลือกได้ว่าต้องการซื้อสินค้า หรือประมูลสินค้า จากนั้นเลือกดูสินค้าจากหมวดหมู่ของสินค้า หรือจากสินค้าที่นำรายได้ไปช่วยเหลือองค์กรการกุศลเดียวกัน และทุกการซื้อขายที่สำเร็จจะมีการเก็บประวัติของผู้ซื้อและผู้ขายเพื่อความน่าเชื่อถือในการใช้งานครั้งต่อไป

1.7 ขั้นตอนการดำเนินงาน

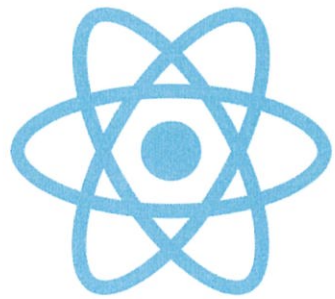
1) วิเคราะห์ปัญหาที่เกิดขึ้น และหาแนวทางแก้ไข
2) กำหนดวัตถุประสงค์และขอบเขตของงาน
3) ออกแบบการทำงานของระบบทั้งหมด
4) ศึกษาและวิเคราะห์หลักการ ทฤษฎีที่เกี่ยวข้องที่เหมาะสมกับงาน
5) พัฒนาระบบ
6) ทดสอบ แก้ไขปัญหาที่เกิดขึ้นจากระบบ
7) ประเมินประสิทธิภาพของงาน
8) ทดลองใช้ระบบจริง
9) ปรับปรุงระบบตามความเหมาะสม

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้องของส่วนติดต่อผู้ใช้งาน (Front-End)

2.1.1 React

React คือ JavaScript Library สร้างโดย Facebook สำหรับพัฒนาส่วนติดต่อกับผู้ใช้งาน เพื่อแสดง View การพัฒนาด้วย React จะแบ่งส่วนติดต่อผู้ใช้งานหนึ่งหน้าเป็นส่วนประกอบ (Component) ต่างๆ ในแต่ละส่วนประกอบจะมีสถานะ (State) ในตัวเองเมื่อสถานะมีการเปลี่ยนแปลงส่วนประกอบนั้นจะถูกแสดงผลใหม่ และสามารถติดต่อกันระหว่างส่วนประกอบได้ด้วยการส่ง props เมื่อนำส่วนประกอบมารวมกันจะได้หน้าเว็บ (Page) 1 หน้า และการเขียนด้วย React สามารถนำ JSX Syntax มาใช้ได้ คือการเขียนส่วนแสดงผลด้วย Syntax ที่คล้าย XML ใน ECMAScript



รูปที่ 2.1 สัญลักษณ์ React

```
class HelloWorld extends React.Component {
  render() {
    return (
      <div class="Content">
        Hello World
        <div class="Footer">
          This is Footer
        </div>
      </div>
    );
  }
}
```

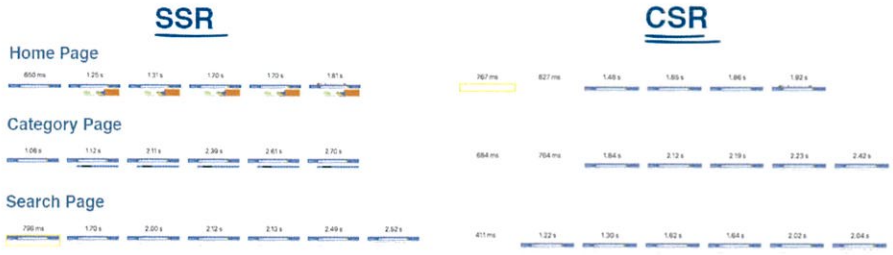
รูปที่ 2.2 การใช้งาน JSX Syntax

2.1.2 Next.js

Next.js คือ JavaScript Framework ที่นำมาใช้กับ React ทำให้ทำงานเป็น SSR (Server-side Rendering) ได้ และมี Client-Side Routing ของหน้าเว็บ (Page) ต่างๆ นอกจากนี้ Next.js มี HMR (Hot Module Replacement) ซึ่งมากับ Webpack ช่วยให้พัฒนาหน้าเว็บได้ง่ายขึ้น เพราะจะเห็นผลของการแก้ไขโค้ดทันทีไม่ต้อง Reload หน้าเว็บใหม่อีกครั้งเพื่อดูความแตกต่าง

2.1.2.1 Server-side Rendering

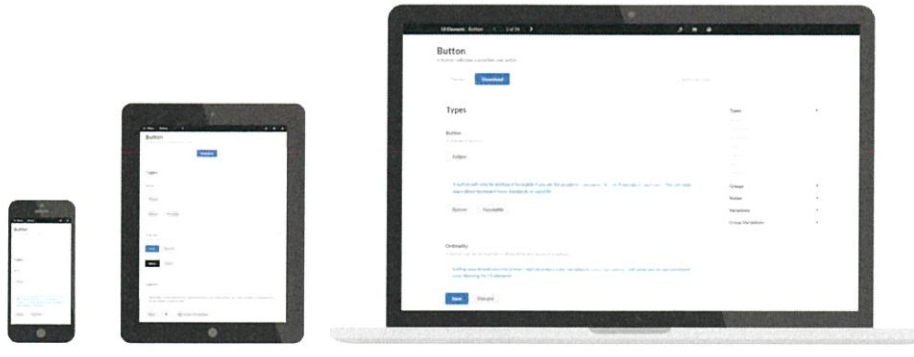
Server-side Rendering คือการที่ Client ส่งคำร้องขอหน้าเว็บไปที่ Server จากนั้น Server จะประมวลผลสิ่งที่ต้องแสดงผลเป็น HTML และส่งผลที่ได้ไปแสดงที่ Client บน DOM ได้ทันทีโดยไม่ต้องประมวลผลเพิ่มเติม ข้อดีของการทำ SSR คือช่วยเพิ่มประสิทธิภาพของการทำ SEO มากขึ้นเพราะสามารถเข้าถึงเนื้อหาของเว็บได้ทันทีเมื่อมีการร้องขอ และทำให้ผู้ใช้เห็นหน้าเว็บได้เร็วกว่า Client-side Rendering



รูปที่ 2.3 เปรียบเทียบ Server-side Rendering (SSR) และ Client-Side Rendering (CSR)

2.1.3 Semantic UI

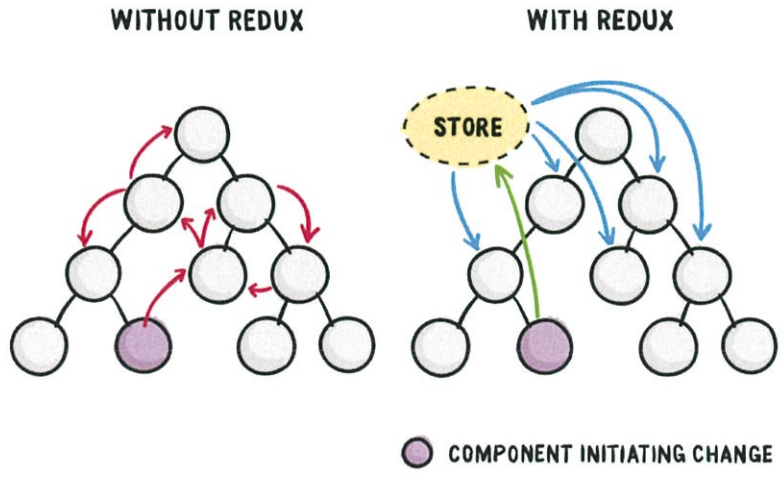
เป็น CSS Framework ที่มีส่วนประกอบของเว็บ (Component) พื้นฐานมาให้เช่น ปุ่ม ตาราง กล่องข้อความ เป็นต้นนำมาพัฒนาตรงตามที่ออกแบบไว้ และเป็นเครื่องมือจัดรูปแบบ (Layout) ของหน้าเว็บให้รองรับขนาดของหน้าจอที่ต่างกันได้เช่นหน้าเว็บที่แสดงบนคอมพิวเตอร์จะต่างกับโทรศัพท์ เนื่องจากการใช้งานของผู้ใช้



รูปที่ 2.4 การแสดงผล Semantic UI ในขนาดหน้าจอที่ต่างกัน

2.1.4 Redux

เป็นตัวจัดเก็บสถานะ (State Container) ที่ช่วยจัดการสถานะของเว็บทั้งเว็บ ที่ช่วยแก้ปัญหาเมื่อเว็บมีขนาดใหญ่ เช่นส่วนประกอบ 2 ส่วนที่อยู่ต่างหน้าเว็บกันที่มีสถานะที่ต้องใช้ร่วมกันการอัพเดทสถานะทั้ง 2 ที่ทำได้ยาก และถ้าเว็บมีโครงสร้างที่ใหญ่อาจทำให้สับสน และแก้ปัญหาที่เกิดขึ้นได้ยาก ซึ่ง Redux จะมาช่วยแก้ปัญหาข้างต้นได้



รูปที่ 2.5 เปรียบเทียบเมื่อนำ Redux มาใช้งาน

จากรูป 2.5 วงกลมสีเทาคือส่วนประกอบของเว็บ และลูกศรคือ Props ที่ส่งระหว่างส่วนประกอบของเว็บ หลักการทำงานของ Redux มี 3 ข้อได้แก่

1)Redux จะจัดเก็บสถานะรวมไว้ในที่เดียวคือ Store ในรูปแบบของ Object Tree

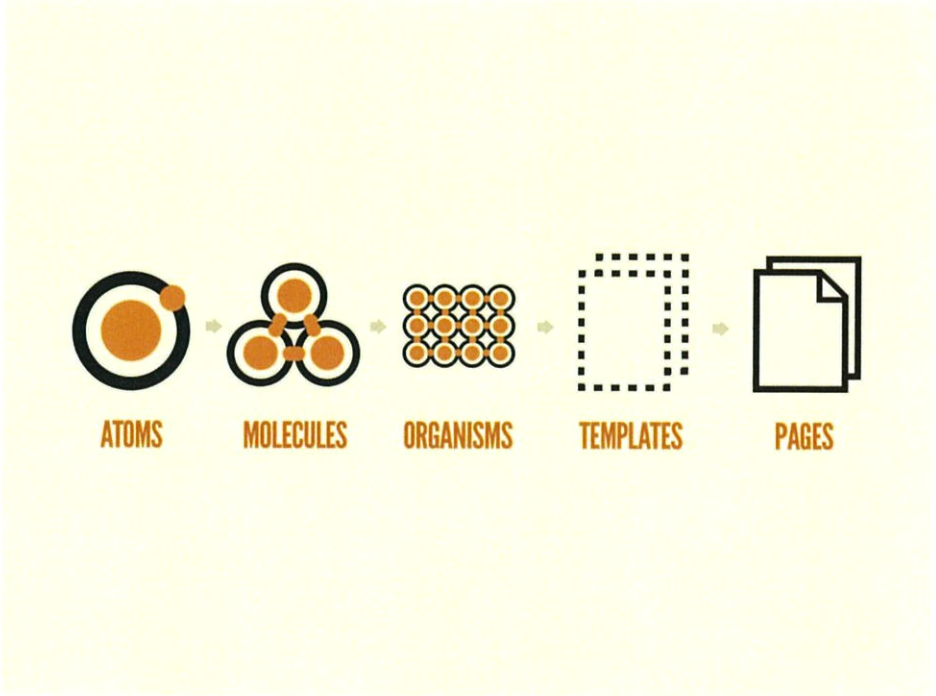
2)สถานะใน Store ไม่สามารถเปลี่ยนแปลงโดยตรงได้ (Read-Only) หากต้องการเปลี่ยนแปลงสถานะต้องสร้าง Action ขึ้นมาและส่งไปที่ Store เพื่อให้สถานะเปลี่ยนแปลง

3)เมื่อ Store ได้รับ Action ที่ส่งมา การเปลี่ยนสถานะจะต้องทำโดย Reducers ซึ่ง Reducer เป็นฟังก์ชันการทำงานที่รับข้อมูลได้แก่สถานะปัจจุบัน และ Action ที่ได้รับและเปลี่ยนแปลงสถานะใหม่ให้สอดคล้องกับข้อมูลที่ได้รับ

2.1.5 Atomic Design

Atomic Design คือแนวคิดในการออกแบบส่วนติดต่อผู้ใช้ โดยออกแบบส่วนต่างๆของเว็บจากหน่วยที่เล็กที่สุดคืออะตอม (Atom) เช่นกล่องรับข้อความ ปุ่ม เป็นต้น แล้วจึงนำอะตอมมาประกอบเป็นส่วนประกอบที่ใหญ่ขึ้น คือโมเลกุล (Molecule) เช่นนำกล่องข้อความมาประกอบกับตัวอักษรจะได้เป็นกล่องรับข้อความที่มีคำกำกับ 1 กล่อง ต่อมาจะนำโมเลกุลมาประกอบกันได้เป็นอวัยวะ (Organism) เช่น เมนูหลักของเว็บ (Navigation bar) ฟอร์มการสมัครสมาชิกเป็นต้น และ

สุดท้ายนำอวัยวะมาประกอบกันจะได้โครงร่างของเว็บ (Template) และเมื่อนำเนื้อหามาใส่จึงกลายเป็นหน้าเว็บที่สมบูรณ์ 1 หน้า



รูปที่ 2.6 หลักการประกอบของ Atomic Design

2.2 ทฤษฎีที่เกี่ยวข้องของส่วนการทำงานของระบบ (Back-End)

2.2.1 Django

Django คือ Web Framework ที่พัฒนาบนภาษา Python โดยถือเป็นภาษาระดับสูงสำหรับโปรแกรมเมอร์ Django และเป็น Open-sourceถูกออกแบบให้เหมาะสมกับงานที่ต้องการความรวดเร็วในการพัฒนา, ความเป็นระเบียบเรียบร้อยของงาน

รูปที่ 2.7 สัญลักษณ์ของ Django

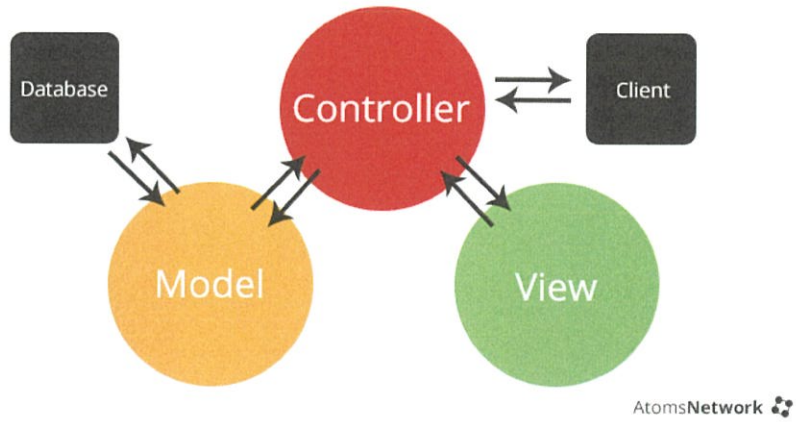
2.2.2 MVC

MVC ย่อมาจาก Model–view–controller คือ Software Architectural Pattern เพื่อแยกการทำงานของโปรแกรมออกเป็น 3 ส่วนหลักๆ สำหรับการแยกการทำงานเป็น 3 ส่วนเหล่านี้เพื่อประโยชน์ในการ Reuse โปรแกรมส่วนที่เคยเขียนไว้แล้ว และเพื่อให้การพัฒนาสามารถทำหลายๆ ส่วนพร้อมกันได้

1) Model ทำหน้าที่ติดต่อกับฐานข้อมูล (Database) โดยรับการรองขอข้อมูลเข้ามา และทำการติดต่อไปยังฐานข้อมูล พร้อมส่งผลลัพธ์กลับไป

2) View ทำหน้าที่สร้างหน้าเว็บไซต์ขึ้นมาเพื่อรอส่งกลับไปให้ Client โดยอาจจะทำงานร่วมกับ Model ที่ใช้ข้อมูลจากฐานข้อมูล

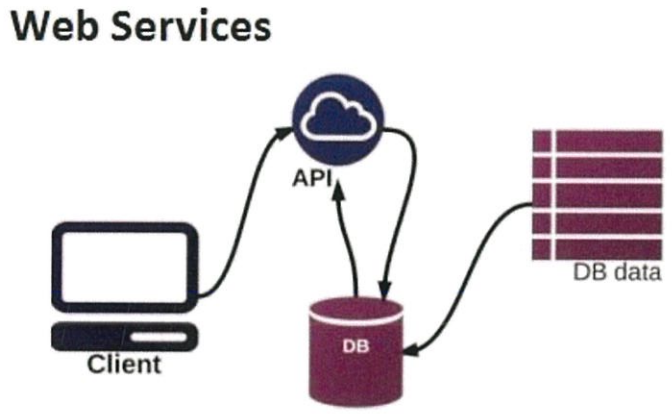
3) Controller ทำหน้าที่เป็นหน่วยประมวลผลกลางที่ติดต่อส่วนต่างๆ ของเว็บไซต์ เช่น ติดต่อไปยัง Model หรือติดต่อไปยัง View เป็นต้น โดยเปรียบเสมือน Gateway ที่คุยกับ Client



รูปที่ 2.8 หลักการทำงานของ Model-View-Controller

2.2.3 RESTful

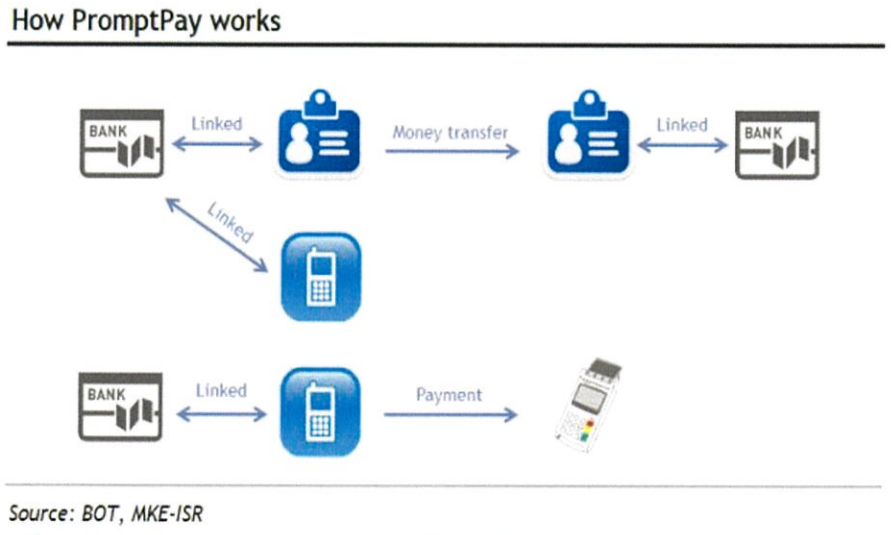
RESTful หรือ REST ย่อมาจาก Representational State Transfer คือ ลักษณะหนึ่งของ Web Service ใช้หลักการ Stateless คือไม่มี Session โดยการทำงานจะอาศัย URI/URL ของ Request และจะตอบกลับในรูปแบบต่างๆเช่น JSON เป็นต้นซึ่งการตอบกลับจะแตกต่างกันไปตาม HTTP Method ที่ได้รับได้แก่ GET POST PUT และ DELETE



รูปที่ 2.9 หลักการทำงานของ Web Service

2.2.4 การสร้าง QR Code ของระบบ PromptPay

การรับโอนเงินด้วยระบบพร้อมเพย์ QR Code ของประเทศไทยนั้น มีหลักการสร้างที่ได้มาตรฐานตาม EMVCo ของโลก ดังเอกสาร EMVCo QR Code Specification for Payment Systems: Merchant-Presented Mode สามารถรองรับการโอนเงินทั้งในรูปแบบ เบอร์โทรศัพท์มือถือ, รหัสประจำตัวประชาชน, เลขประจำตัวผู้เสียภาษี, e-Wallet ของประเทศไทย



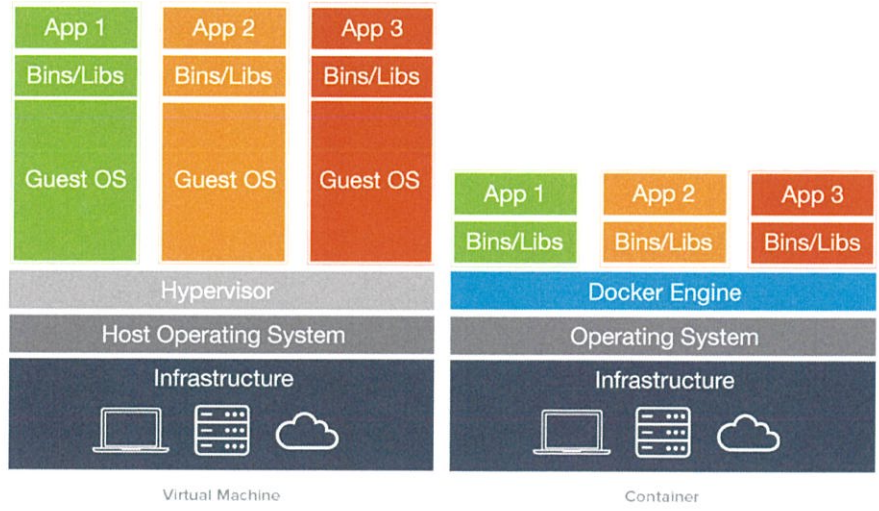
รูปที่ 2.10 หลักการทำงานของระบบพร้อมเพย์ในประเทศไทย

2.3 ทฤษฎีที่เกี่ยวข้องส่วนที่ติดต่อโครงสร้างของระบบ (Infrastructure)

2.3.1 Container

Container คือ Concept ของการสร้างสภาพแวดล้อมจำลองขึ้นมาสำหรับการทำงานใดๆ ของ Application โดย Container เมื่อนำไปสร้างบน OS หรือ Computer ใดๆ จะได้ข้อมูลเหมือนเดิม

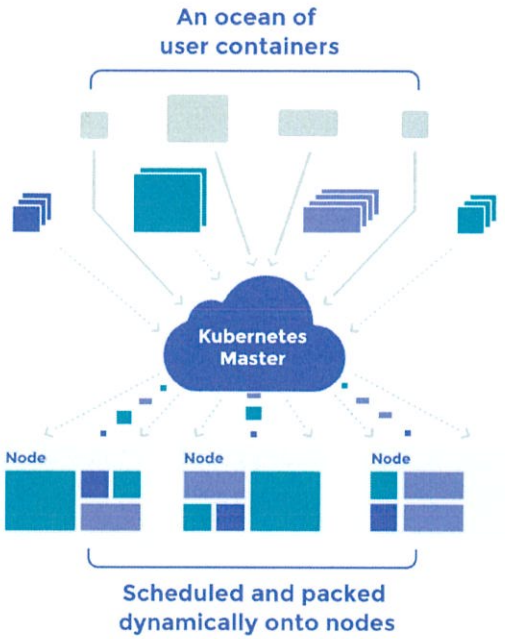
ทุกประการ โดย Container เองถูกทำงานบนระดับ OS ตั้งแต่แรก ไม่ต้องอาศัย Host-guest มาคั่นกลางดั่ง Virtual machine และ Software หลักที่นำมาช่วยจัดการส่วนนี้คือ Docker



รูปที่ 2.11 การเปรียบเทียบระหว่างการทำงานของ Virtual Machine และ Container

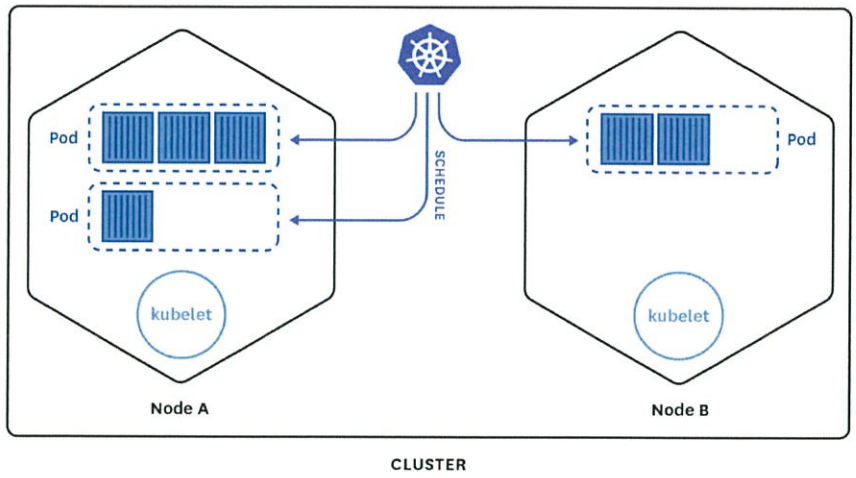
2.3.2 Kubernetes (K8s)

Kubernetes คือ Open-source ที่ถูกพัฒนาจากประสบการณ์กว่า 15 ปีของ Google เพื่อเป็นเครื่องมือในการจัดการระบบต่างๆ ตั้งแต่ Auto-Deployment, Scaling, จัดการ Application (Container) ให้มีความเรียบง่ายทั้งต่อการดูแล และการพัฒนาของผู้ใช้งาน โดยหลักการทำงานของ K8s นั้นใช้หลักการ Master-Worker(minion) โดย Master จะทำหน้าที่คอยควบคุมการทำงานของ Cluster



รูปที่ 2.12 ภาพรวมของหลักการ Kubernetes

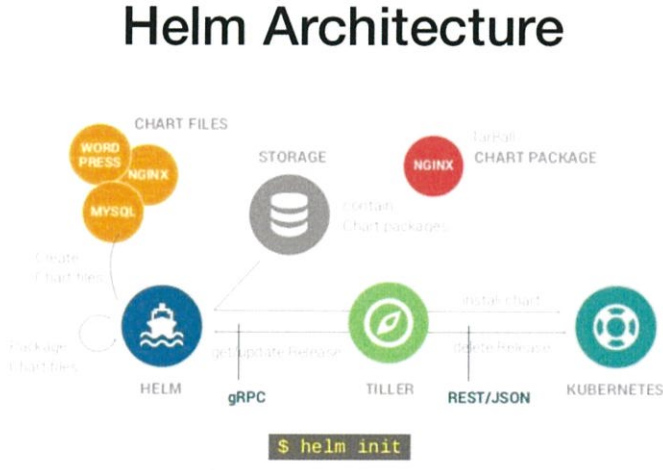
โดยใน Node หรือ Worker นั้นมีกลุ่มก้อนของ Application ที่ใช้งานถูก Run อยู่ เรียกว่า Pod โดยมี Master ที่คอยเป็น Scheduler คอยควบคุมการทำงานต่างๆ ที่ติดต่อผ่าน Kubelet ไปยัง Node อีกทอดหนึ่ง



รูปที่ 2.13 ตัวอย่างการทำงานของ Kubernetes ที่สั่งการทำงานไปยัง Pod ต่างๆ

2.3.3 Helm

Helm คือ เครื่องมือที่นำมาช่วยในการจัดการ Chart ของ Kubernetes ต่างๆ โดยความพิเศษของ Helm คือ สามารถสร้าง Pre-Configured เตรียมพร้อมไว้สำหรับการใช้งานต่างๆ ไว้ได้ หากงานมีความซ้ำซ้อนก็สามารถใช้ Chart จาก helm ก่อนหน้าได้ทันที หรืองานใดมีการเปลี่ยนแปลงเล็กน้อยก็สามารถทำ Dynamic Values ไว้ได้



รูปที่ 2.14 ภาพรวมของการทำงาน Helm ที่นำมาช่วยทำ Dynamic Value ให้ Kubernetes

บทที่ 3

การออกแบบและการพัฒนา

3.1 ภาพรวมของระบบ (Overview)

ระบบประมูลและขายสินค้าเพื่อการกุศลเป็นระบบที่ผู้ใช้สามารถเข้ามาซื้อ หรือประมูลสินค้าและนำรายได้ไปช่วยเหลือองค์กรการกุศลที่ผู้ขายกำหนด การเพิ่มองค์กรการกุศลจะต้องส่งเอกสารตามที่ระบบกำหนด และเมื่อผู้ดูแลระบบพิจารณาเอกสารที่ส่งมาจึงจะเพิ่มองค์กรการกุศลเข้าในระบบ เพื่อให้ผู้ใช้งานสามารถดูรายละเอียด หรือให้ผู้ขายสามารถกำหนดที่จะช่วยเหลือองค์กรการกุศลนั้นจากรายได้จากสินค้าได้

การชำระเงินผู้ชำระจะโอนเงินไปให้องค์กรการกุศลนี้โดยตรง และผู้ชำระจำเป็นจะต้องเพิ่มหลักฐานการโอนเข้ามาในระบบ ระบบจึงจะส่งหลักฐานให้ผู้ขายและองค์กรที่ได้รับเงินช่วยเหลือ ผู้ขายจึงจะส่งสินค้าให้กับผู้ซื้อ

รูปที่ 3.1 ภาพรวมของระบบ

3.2 ความต้องการของระบบ (System requirements)

ความต้องการของระบบจะแบ่งเป็น 2 ส่วนดังนี้

3.2.1 ความต้องการของระบบในการทำงาน (Functional Requirements)

การทำงานของระบบจะแบ่งตามประเภทของผู้ใช้งานได้ดังนี้

3.2.1.1 ผู้ซื้อ

ผู้ซื้อคือผู้ใช้ที่เข้าเว็บเพื่อมาซื้อสินค้า หรือเข้าร่วมการประมูลสินค้าเงินที่ผู้ใช้จ่ายจะนำไปให้องค์กรการกุศลที่ผู้ขายกำหนดไว้ ซึ่งระบบสามารถทำงานได้ดังนี้

1) ผู้ซื้อสามารถดูรายละเอียดของสินค้า และองค์กรการกุศลที่ช่วยเหลือได้
2) ผู้ซื้อสามารถคัดกรองสินค้าตามหมวดหมู่ที่กำหนดได้
3) ผู้ซื้อสามารถชำระเงินผ่านการโอนเงินด้วยเลขบัญชี หรือพร้อมเพย์ (PromptPay) ได้
4) ผู้ซื้อสามารถยืนยันการชำระเงินได้ด้วยการเพิ่มหลักฐานการโอนเงินเข้ามาในระบบได้
5) ผู้ซื้อสามารถเข้าร่วมการประมูลสินค้าได้
6) ผู้ซื้อสามารถดูรายละเอียดองค์กรการกุศลต่างๆได้
7) ผู้ซื้อสามารถแจ้งปัญหาผ่านเว็บได้

3.2.1.2 ผู้ขาย

ผู้ขายคือผู้ใช้ที่นำสินค้ามาขาย หรือเปิดการประมูลและกำหนดว่าจะนำรายได้จากสินค้าไปช่วยเหลือองค์กรการกุศลที่ต้องการ ระบบสามารถทำงานได้ดังนี้

1) ผู้ขายสามารถเลือกได้ว่าสินค้านั้นจะนำมาประมูล หรือจำหน่าย
2) ผู้ขายสามารถนำสินค้ามาจำหน่าย หรือเปิดประมูลตามหมวดหมู่ที่กำหนด
3) ผู้ขายสามารถกรอกรายละเอียดของสินค้าได้
4) ผู้ขายสามารถจำหน่าย หรือเปิดประมูลสินค้าได้มากกว่า 1 ชิ้น
5) ผู้ขายสามารถดูหลักฐานการโอนเงินของผู้ชำระเงินสินค้าของตนเองได้
6) ผู้ขายสามารถแจ้งปัญหาผ่านเว็บไซต์ได้

3.2.1.3 องค์กรการกุศล

1) องค์กรสามารถส่งเอกสารเพิ่มองค์กรเข้าระบบได้
2) องค์กรสามารถดูหลักฐานการโอนเงินของผู้ซื้อสินค้าที่นำเงินมาช่วยเหลือองค์กรตนเองได้

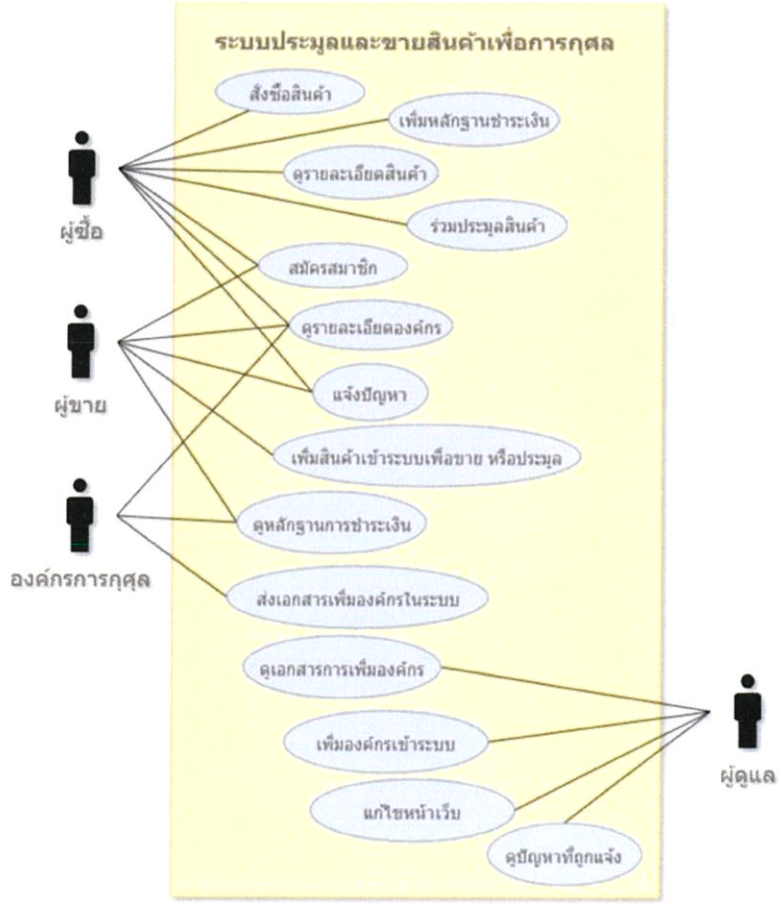
3.2.1.4 ผู้ดูแล

1) ผู้ดูแลสามารถดูเอกสารเพิ่มองค์กรการกุศลเข้าระบบได้
2) ผู้ดูแลสามารถเพิ่มองค์กรการกุศลเข้าระบบได้
3) ผู้ดูแลสามารถดูปัญหาที่แจ้งเข้ามาได้
4) ผู้ดูแลสามารถติดต่อกับผู้ใช้ผ่านอีเมล์ได้
5) ผู้ดูแลสามารถแก้ไขข้อมูลบางส่วนบนหน้าเว็บได้

3.2.2 ความต้องการของระบบด้านความสามารถ (Non-functional Requirements)

1) ผู้ขายสามารถจำหน่ายสินค้า หรือเปิดประมูลสินค้าได้มากที่สุด 10 ชิ้นพร้อมกัน
2) ผู้ใช้งานทุกคนจำเป็นจะต้องมีอีเมลในการสมัครสมาชิก

3.3 Use Case Diagram



รูปที่ 3.2 Use Case Diagram ของระบบ

3.3.1 อธิบาย Use Case Diagram

ขั้นตอนการสั่งซื้อสินค้า

Use Case :	สั่งซื้อสินค้า
Actor :	ผู้ซื้อ
Use case description :	ผู้ซื้อเพิ่มสินค้าในตะกร้า และสั่งซื้อสินค้าในตะกร้าได้
Precondition :	ผู้ซื้อต้องเข้าสู่ระบบ
Flow of events :	1) ผู้ซื้อเลือกดูสินค้า 2) เพิ่มสินค้าเข้าไปในตะกร้า 3) ดำเนินการสั่งซื้อสินค้าในตะกร้า 4) กรอกรายละเอียดที่อยู่การจัดส่งหากยังไม่เคยกรอก 5) ยืนยันการสั่งซื้อ
Postcondition :	เพิ่มคำสั่งซื้อในบัญชีผู้ซื้อ

ขั้นตอนเพิ่มหลักฐานชำระเงิน

Use Case :	**เพิ่มหลักฐานชำระเงิน**
Actor :	ผู้ซื้อ
Use case description :	ผู้ซื้อเพิ่มหลักฐานการชำระเงินให้ตรงตามคำสั่งซื้อ
Precondition :	ผู้ซื้อมีคำสั่งซื้อในบัญชีตนเอง
Flow of events :	1) ผู้ซื้อเลือกคำสั่งซื้อที่ต้องการเพิ่มหลักฐานการชำระเงิน 2) อัพโหลดหลักฐานการชำระเงินเป็นรูปภาพ 3) ผู้ซื้อยืนยันการเพิ่มหลักฐาน
Postcondition :	ระบบแจ้งผู้ขาย และองค์กรการกุศลที่เกี่ยวข้องกับคำสั่งซื้อนั้นๆ

ขั้นตอนดูรายละเอียดสินค้า

Use Case :	**ดูรายละเอียดสินค้า**
Actor :	ผู้ซื้อ
Use case description :	ผู้ซื้อดูรายละเอียดของสินค้า
Precondition :	-
Flow of events :	1) ผู้ซื้อเลือกหมวดหมู่ของสินค้า 2) ผู้ซื้อเลือกสินค้าที่ต้องการดูรายละเอียด
Postcondition :	ระบบแสดงรายละเอียดของสินค้า

ขั้นตอนการร่วมประมูลสินค้า

Use Case :	**ร่วมประมูลสินค้า**
Actor :	ผู้ซื้อ
Use case description :	ผู้ซื้อร่วมการประมูล โดยการเสนอราคาสินค้าที่มากกว่าราคาปัจจุบัน
Precondition :	ผู้ซื้อต้องเข้าสู่ระบบ และเลือกสินค้าที่ต้องการประมูล
Flow of events :	1) ผู้ซื้อกรอกราคาที่ต้องการเสนอ และเป็นราคาที่สูงกว่าราคาปัจจุบัน 2) ผู้ซื้อยืนยันการเสนอราคา
Postcondition :	เพิ่มรายการประมูลในบัญชีผู้ซื้อ

ขั้นตอนสมัครสมาชิก

Use Case :	**สมัครสมาชิก**
Actor :	1) ผู้ซื้อ 2) ผู้ขาย
Use case description :	สมัครสมาชิกเพื่อใช้งานระบบ
Precondition :	ผู้ที่สมัครต้องมีอีเมลที่ใช้งานได้
Flow of events :	1) กรอกรายละเอียดของตนเองให้ครบถ้วน 2) ยืนยันในอีเมลที่กรอกไป
Postcondition :	แสดงหน้าเข้าสู่ระบบ

ขั้นตอนการดูรายละเอียดองค์กรการกุศล

Use Case :	ดูรายละเอียดองค์กรการกุศล
Actor :	1) ผู้ซื้อ 2) ผู้ขาย 3) องค์กรการกุศล
Use case description :	ดูรายละเอียดต่างๆขององค์กรการกุศล
Precondition :	-
Flow of events :	1) เลือกหมวดหมู่ขององค์กรการกุศล 2) เลือกองค์กรการกุศลที่ต้องการดูรายละเอียด
Postcondition :	ระบบแสดงรายละเอียดขององค์กรการกุศล

ขั้นตอนการแจ้งปัญหา

Use Case :	**แจ้งปัญหา**
Actor :	1) ผู้ซื้อ 2) ผู้ขาย
Use case description :	แจ้งปัญหาที่เกิดขึ้นจากการใช้ระบบ
Precondition :	ต้องเข้าสู่ระบบเท่านั้น
Flow of events :	1) กรอกรายละเอียดของปัญหา 2) ยืนยันรายละเอียดปัญหาที่กรอก
Postcondition :	-

ขั้นตอนการเพิ่มสินค้าเข้าระบบเพื่อจำหน่าย หรือประมูล

Use Case :	**เพิ่มสินค้าเข้าระบบเพื่อจำหน่าย หรือประมูล**
Actor :	ผู้ขาย
Use case description :	ผู้ขายเพิ่มสินค้าเข้าระบบเพื่อนำมาขาย หรือประมูล
Precondition :	ผู้ขายต้องเข้าสู่ระบบเท่านั้น
Flow of events :	1) ผู้ขายเลือกว่าจะขาย หรือประมูลสินค้า 2) ผู้ขายเลือกหมวดหมู่สินค้าของตนเอง 3) กรอกรายละเอียดของสินค้า 4) ยืนยันการเพิ่มสินค้าเข้าระบบ
Postcondition :	เพิ่มสินค้าเข้าระบบและรายการสินค้าของผู้ขาย

ขั้นตอนการดูหลักฐานชำระเงิน

Use Case :	**ดูหลักฐานการชำระเงิน**
Actor :	1) ผู้ขาย 2) องค์กรการกุศล
Use case description :	ดูหลักฐานการชำระเงินตามคำสั่งซื้อที่เกี่ยวข้อง
Precondition :	ต้องเข้าสู่ระบบเท่านั้น
Flow of events :	1) เลือกรายการคำสั่งซื้อที่ตนเองเกี่ยวข้อง 2) เลือกดูหลักฐานการชำระเงิน
Postcondition :	ถ้ามีหลักฐานการชำระเงินในคำสั่งซื้อที่เลือกแล้วระบบจะแสดงหลักฐานการชำระเงิน แต่จะแสดงข้อความแจ้งให้ทราบหากไม่มีหลักฐานในคำสั่งซื้อนั้นๆ

ขั้นตอนการส่งเอกสารเพิ่มองค์กรเข้าระบบ

Use Case :	**ส่งเอกสารเพิ่มองค์กรเข้าระบบ**
Actor :	องค์กรการกุศล
Use case description :	ส่งเอกสารเพื่อเพิ่มองค์กรเข้าระบบให้ผู้ใช้สามารถช่วยเหลือได้
Precondition :	-
Flow of events :	1) กรอกรายละเอียดขององค์กรการกุศล 2) ยืนยันรายละเอียดต่างๆ
Postcondition :	-

ขั้นตอนการดูเอกสารการเพิ่มองค์กร

Use Case :	**ดูเอกสารการเพิ่มองค์กร**
Actor :	ผู้ดูแล
Use case description :	ดูเอกสารที่องค์กรการกุศลส่งเข้ามาเพื่อขอเพิ่มองค์กรในระบบ
Precondition :	ผู้ดูแลต้องเข้าสู่ระบบ
Flow of events :	1) เลือกองค์กรที่ต้องการดูรายละเอียดจากรายการองค์กรที่เพิ่มเอกสาร
Postcondition :	ระบบแสดงรายละเอียดเอกสารขององค์กร

ขั้นตอนการเพิ่มองค์กรเข้าระบบ

Use Case :	**เพิ่มองค์กรเข้าระบบ**
Actor :	ผู้ดูแล
Use case description :	ผู้ดูแลเพิ่มองค์กรการกุศลที่พิจารณาแล้วว่ามีความถูกต้องเข้าระบบ
Precondition :	ผู้ดูแลต้องเข้าสู่ระบบ
Flow of events :	1) เลือกองค์กรจากรายการองค์กรที่เพิ่มเอกสาร 2) เลือกเพิ่มองค์กรนั้นๆเข้าสู่ระบบ
Postcondition :	แจ้งเตือนองค์กรว่าได้เพิ่มในระบบแล้ว และส่งบัญชีเข้าใช้งานขององค์กร

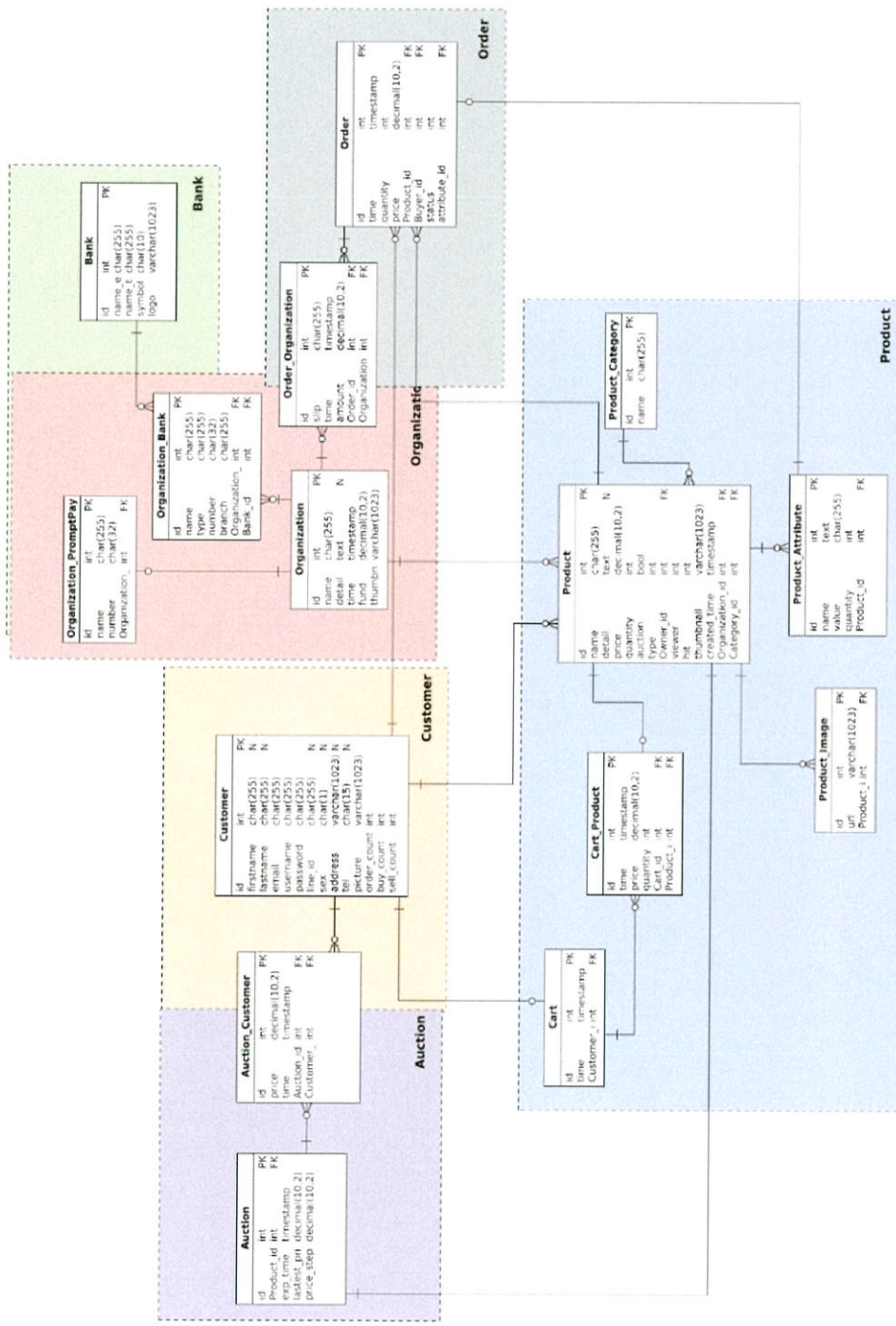
ขั้นตอนแก้ไขหน้าเว็บ

Use Case :	**แก้ไขหน้าเว็บ**
Actor :	ผู้ดูแล
Use case description :	แก้ไขข้อมูลบางส่วนของเว็บ
Precondition :	ผู้ดูแลต้องเข้าสู่ระบบ
Flow of events :	1) ผู้ดูแลเลือกส่วนที่ต้องการแก้ไข จากรายการส่วนของเว็บที่สามารถแก้ไขได้ 2) กรอกข้อมูลส่วนที่ต้องการแก้ไข 3) ยืนยันการแก้ไข
Postcondition :	-

ขั้นตอนการดูปัญหาที่ถูกแจ้ง

Use Case :	**ดูปัญหาที่ถูกแจ้ง**
Actor :	ผู้ดูแล
Use case description :	ดูปัญหาที่ผู้ใช้งานระบบแจ้งเข้ามาในระบบ
Precondition :	ผู้ดูแลต้องเข้าสู่ระบบ
Flow of events :	1) ผู้ดูแลเลือกปัญหาจากรายการของปัญหาที่ถูกแจ้งเข้ามา
Postcondition :	ระบบแสดงรายละเอียดของปัญหาที่ถูกแจ้ง และรายละเอียดการติดต่อผู้แจ้งปัญหา

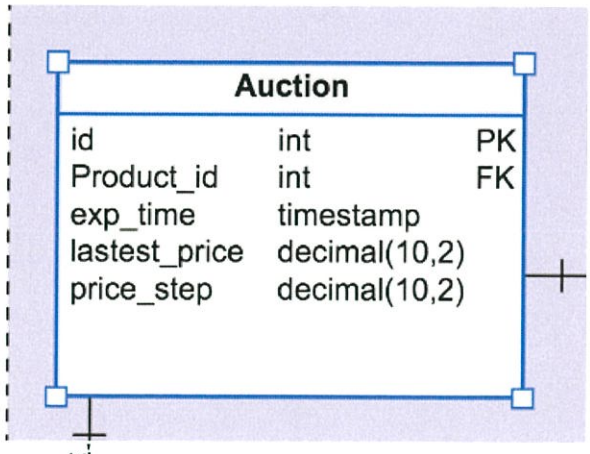
3.4 Entity Relationship diagram



รูปที่ 3.3 Entity Relationship diagram

การออกแบบฐานข้อมูลของระบบผู้พัฒนาได้ทำการออกแบบฐานข้อมูลโดยนำเสนอความสัมพันธ์ของตารางต่างๆ ที่มีอยู่ในระบบ โดยมุ่งเน้นทางภาพรวมของระบบพอสังเขป

3.4.1 ตาราง Auction

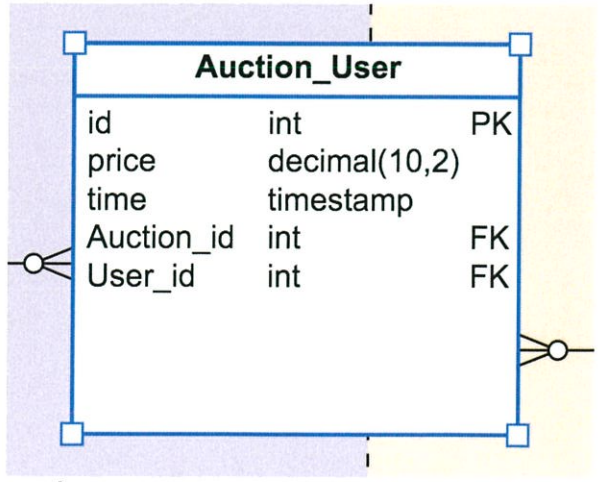


รูปที่ 3.4 ตาราง Auction

ตาราง Auction ใช้เก็บข้อมูลของการประมูลแต่ละครั้งที่เกิดขึ้น มีคุณสมบัติของข้อมูลที่เก็บในตารางดังนี้

1) id เป็น primary key ของตาราง ใช้เก็บรหัสของการประมูล
2) Product_id เป็น foreign key ใช้เก็บสินค้าของการประมูล
3) exp_time เก็บเวลาสิ้นสุดของการประมูล
4) lastest_price เก็บราคาล่าสุดของการประมูล
5) price_step เก็บจำนวนเงินขั้นต่ำของการประมูลแต่ละครั้ง

3.4.2 ตาราง Auction_User

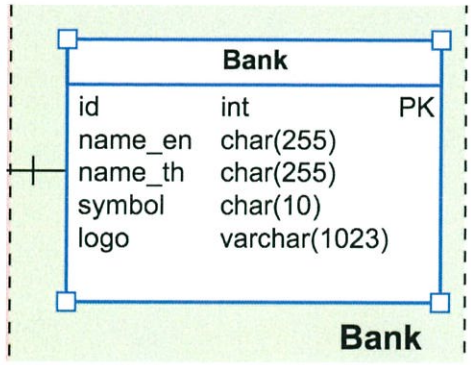


รูปที่ 3.5 ตาราง Auction_User

ตาราง Auction_User ใช้เก็บข้อมูลเกี่ยวกับการประมูลในแต่ละครั้ง มีคุณสมบัติของข้อมูลที่เก็บในตารางดังนี้

1) id เป็น primary key ของตาราง ใช้เก็บรหัสของการประมูลแต่ละครั้ง
2) price เก็บราคาของการประมูลแต่ละครั้ง
3) time เก็บเวลาที่เกิดการประมูลขึ้น
4) Auction_id เป็น foreign key ใช้เก็บการประมูล
5) User_id เป็น foreign key ใช้เก็บผู้ประมูลสินค้า

3.4.3 ตาราง Bank

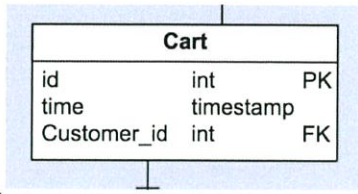


รูปที่ 3.6 ตาราง Bank

ตาราง Bank ใช้เก็บข้อมูลเกี่ยวกับธนาคารที่รับการโอนเงิน มีคุณสมบัติของข้อมูลที่เก็บในตารางดังนี้

1) id เป็น primary key ของตาราง ใช้เก็บรหัสของธนาคาร
2) name_en ใช้เก็บชื่อภาษาอังกฤษของธนาคาร
3) name_th ใช้เก็บชื่อภาษาไทยของธนาคาร
4) symbol ใช้เก็บชื่อย่อภาษาอังกฤษของธนาคาร
5) logo ใช้เก็บที่อยู่ไฟล์ของรูป logo ของธนาคาร

3.4.4 ตาราง Cart

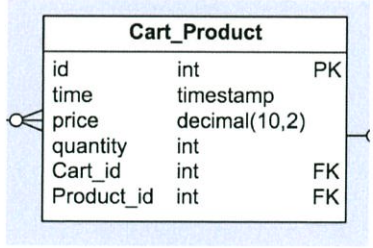


รูปที่ 3.7 ตาราง Cart

ตาราง Cart ใช้เก็บข้อมูลเกี่ยวกับตระกร้าสินค้า มีคุณสมบัติของข้อมูลที่เก็บในตารางดังนี้

1) id เป็น primary key ของตาราง ใช้เก็บรหัสตระกร้าสินค้า
2) time ใช้เก็บเวลาที่สร้างตระกร้าสินค้า
3) Customer_id เป็น foreign key ใช้เก็บรหัสของผู้เป็นเจ้าของตระกร้าสินค้า

3.4.5 ตาราง Cart_Product

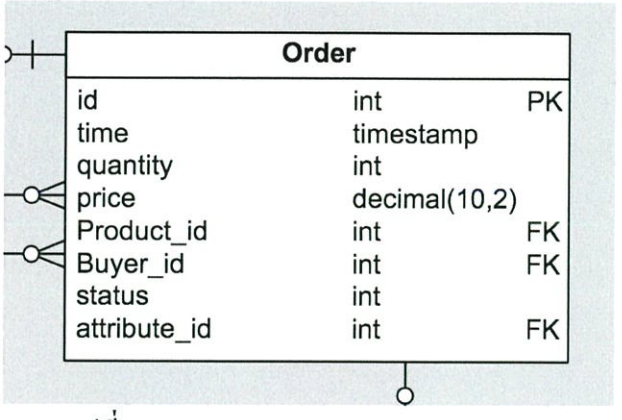


รูปที่ 3.8 ตาราง Cart_Product

ตาราง Cart_Product มีคุณสมบัติของข้อมูลที่เก็บในตารางดังนี้

1) id เป็น primary key ของตาราง ใช้เก็บรหัสของสินค้าที่อยู่ในตระกร้าสินค้า
2) time ใช้เก็บเวลาที่สร้างสินค้าชิ้นนี้ในตระกร้าสินค้า
3) price ใช้เก็บราคาของสินค้า
4) quantity ใช้เก็บจำนวนสินค้า
5) Cart_id เป็น foreign key ใช้เก็บรหัสของตระกร้าสินค้า
6) Product_id เป็น foreign key ใช้เก็บรหัสของสินค้า

3.4.6 ตาราง Order

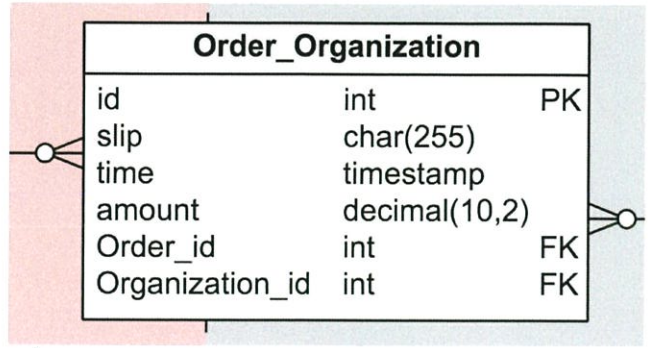


รูปที่ 3.9 ตาราง Order

ตาราง Order ใช้เก็บข้อมูลเกี่ยวกับคำสั่งซื้อสินค้า มีคุณสมบัติของข้อมูลที่เก็บในตารางดังนี้

1) id เป็น primary key ของตาราง ใช้เก็บรหัสของคำสั่งซื้อสินค้า
2) time ใช้เก็บเวลาของการสั่งซื้อสินค้า
3) quantity ใช้เก็บจำนวนสินค้า
4) price ใช้เก็บจำนวนเงินของการสั่งซื้อสินค้
5) Product_id เป็น foreign key ใช้เก็บสินค้าของการสั่งซื้อ
6) Buyer_id เป็น foreign key ใช้เก็บผู้ซื้อสินค้า
7) status ใช้เก็บค่าสถานะของการซื้อสินค้า
8) attribute_id เป็น foreign key ใช้เก็บคุณลักษณะของสินค้า

3.4.7 ตาราง Order_Organization

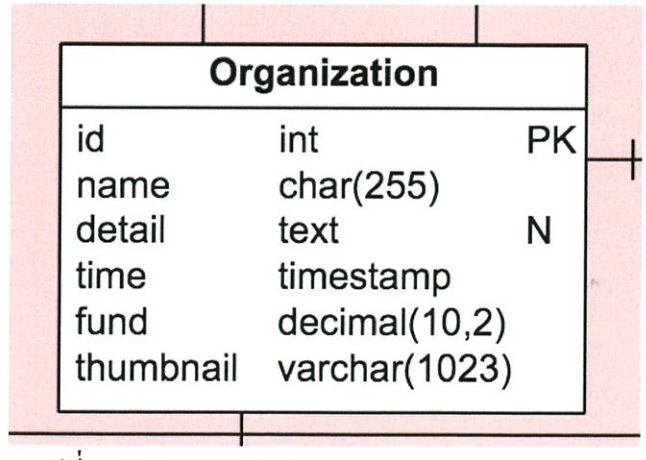


รูปที่ 3.10 ตาราง Order_Organization

ตาราง Order_Organization ใช้เก็บข้อมูลที่รายการสั่งซื้อสินค้าเชื่อมโยงไปถึงองค์กรที่รับบริจาคเงินของสินค้าชิ้นนั้นๆ มีคุณสมบัติของข้อมูลที่เก็บในตารางดังนี้

1) id เป็น primary key ของตาราง ใช้เก็บรหัสสินค้าที่เชื่อมโยงกับองค์กร
2) slip ใช้เก็บที่อยู่ไฟล์สลิปที่ผู้ซื้อบันทึกไว้
3) time ใช้เก็บเวลาที่สร้างรายการสินค้านี้
4) amount ใช้เก็บยอดเงินที่ใช้ในการสั่งซื้อสินค้า
5) Order_id เป็น foreign key ใช้เก็บรหัสสินค้า
6) Organization_id เป็น foreign key ใช้เก็บรหัสองค์กร

3.4.8 ตาราง Organization

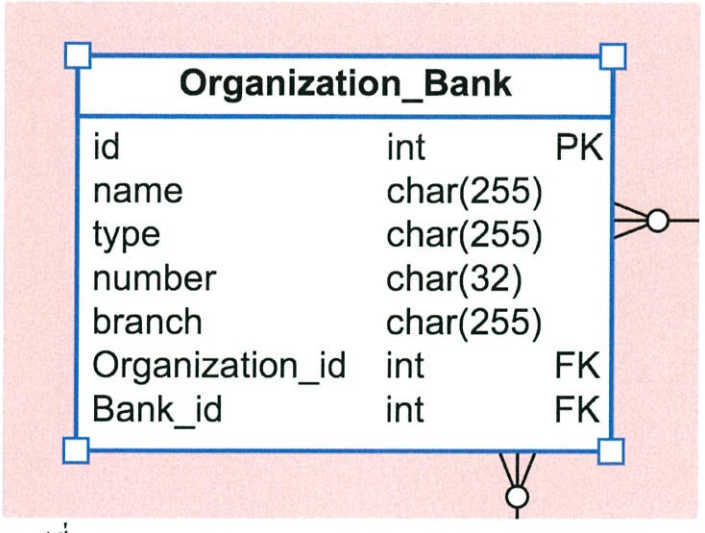


รูปที่ 3.11 ตาราง Organization

ตาราง Organization ใช้เก็บข้อมูลเกี่ยวกับองค์กรที่รับการบริจาค มีคุณสมบัติของข้อมูลที่เก็บในตารางดังนี้

1) id เป็น primary key ของตาราง ใช้เก็บรหัสองค์กร
2) name ใช้เก็บชื่อองค์กร
3) detail ใช้เก็บรายละเอียดขององค์กร
4) time ใช้เก็บเวลาที่องค์กรเข้าร่วมกับเว็บไซต์ของเรา
5) fund ใช้เก็บจำนวนเงินที่องค์กรได้รับการบริจาคแล้วทั้งหมด
6) thumbnail ใช้เก็บที่อยู่รูปภาพแรกขององค์กร

3.4.9 ตาราง Organization_Bank

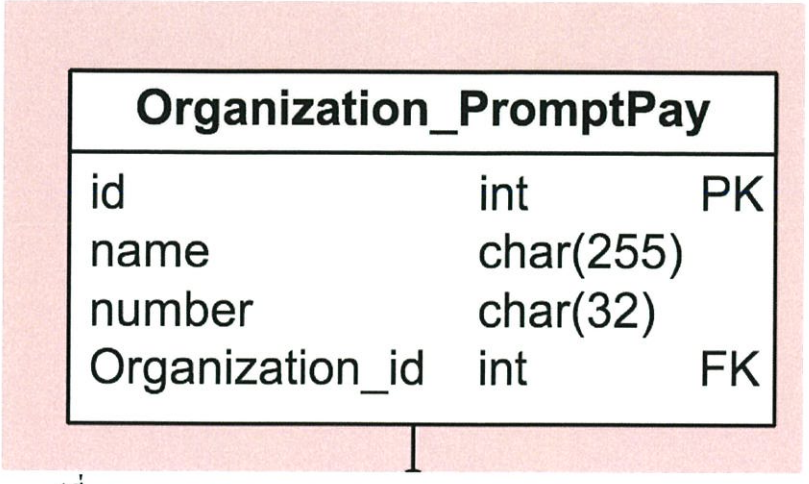


รูปที่ 3.12 ตาราง Organization_Bank

ตาราง Organization_Bank ใช้เก็บข้อมูลเกี่ยวกับบัญชีธนาคารขององค์กรที่รับการบริจาค มีคุณสมบัติของข้อมูลที่เก็บในตารางดังนี้

1) id เป็น primary key ของตารางใช้เก็บรหัสของบัญชีธนาคาร
2) name ชื่อบัญชีธนาคาร
3) type ใช้เก็บชนิดบัญชีธนาคาร
4) number ใช้เก็บเลขที่บัญชีธนาคาร
5) branch ใช้เก็บสาขาของบัญชีธนาคาร
6) Organizaiton_id เป็น foreign key ใช้เก็บรหัสองค์กรของเจ้าของบัญชี
7) Bank_id เป็น foreign key ใช้เก็บรหัสของธนาคาร

3.4.10 ตาราง Organization_PromptPay

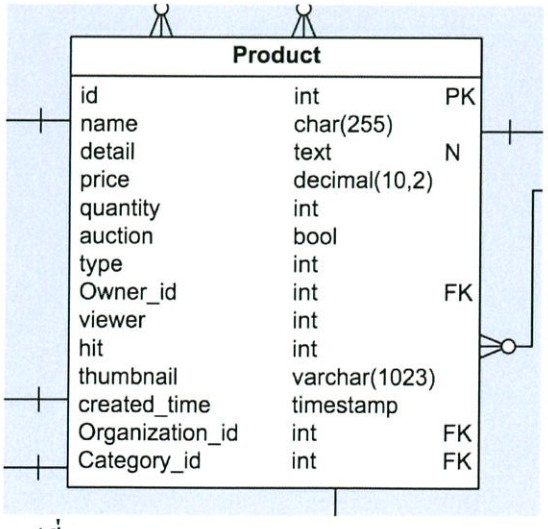


รูปที่ 3.13 ตาราง Organization_PromptPay

ตาราง Organization_PromptPay ใช้เก็บข้อมูลบัญชีพร้อมเพย์สำหรับองค์กร มีคุณสมบัติของข้อมูลที่เก็บในตารางดังนี้

1) id เป็น primary key ของตาราง ใช้เก็บรหัสของพร้อมเพย์สำหรับองค์กร
2) name ใช้เก็บชื่อบัญชีที่เชื่อมโยงพร้อมเพย์ขององค์กร
3) number ใช้เก็บรหัสพร้อมเพย์ขององค์กร
4) Ordganization_id เป็น foreign key ใช้เก็บรหัสขององค์กรที่รหัสพร้อมเพย์เชื่อมโยงอยู่

3.4.11 ตาราง Product

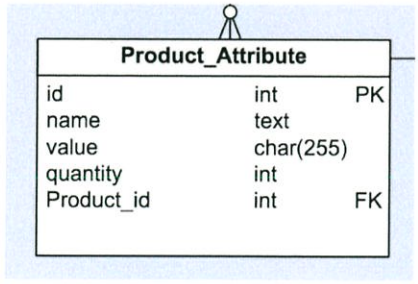


รูปที่ 3.14 ตาราง Product

ตาราง Product ใช้เก็บข้อมูลสินค้า มีคุณสมบัติของข้อมูลที่เก็บในตารางดังนี้

1) id เป็น primary key ของตาราง ใช้เก็บรหัสของสินค้า
2) name ใช้เก็บชื่อของสินค้า
3) detail ใช้เก็บรายละเอียดของสินค้า
4) price ใช้เก็บราคาของสินค้า
5) quantity ใช้เก็บจำนวนสินค้า
6) auction ใช้เก็บวิธีการขายสินค้า ว่าเป็นการขาย หรือประมูลสินค้า
7) type ใช้เก็บชนิดของสินค้า
8) Owner_id เป็น foreign key เพื่อระบุผู้ประกาศขายสินค้า
9) viewer ใช้เก็บจำนวนผู้เข้าชมสินค้า
10) hit ใช้เก็บจำนวนการเข้าชมสินค้า
11) thumbnail ใช้เก็บที่อยู่รูปภาพแรกของสินค้า
12) created_time ใช้เก็บเวลาที่สร้างสินค้าชิ้นนี้
13) Organization_id เป็น foreign key ใช้เก็บรหัสขององค์กรที่สินค้าจะบริจาคเงินให้
14) Category_id เป็น foreign key ใช้เก็บรหัสของหมวดหมู่สินค้า

3.4.12 ตาราง Product_attribute

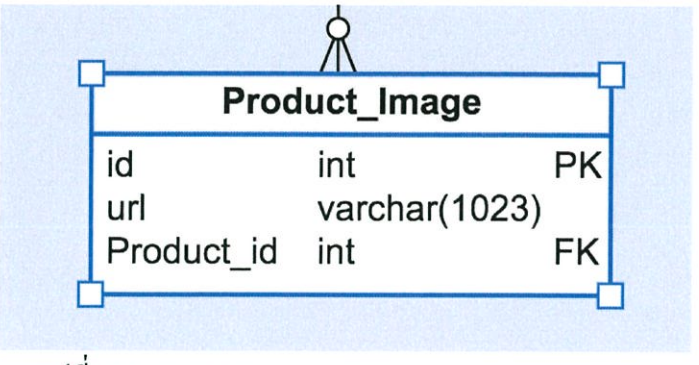


รูปที่ 3.15 ตาราง Product_attribute

ตาราง Product_attribute ใช้เก็บข้อมูลเกี่ยวกับรายละเอียดเฉพาะสินค้า มีคุณสมบัติของข้อมูลที่เก็บในตารางดังนี้

1) id เป็น primary key ของตาราง ใช้เก็บรหัสของรายละเอียดสินค้า
2) name ใช้เก็บชื่อของคุณลักษณะของสินค้า
3) value ใช้เก็บข้อมูลของคุณลักษณะของสินค้า
4) quantity ใช้เก็บปริมาณของสินค้าคงเหลือ
5) Product_id ใช้เก็บรหัสของสินค้า

3.4.13 ตาราง Product_Image

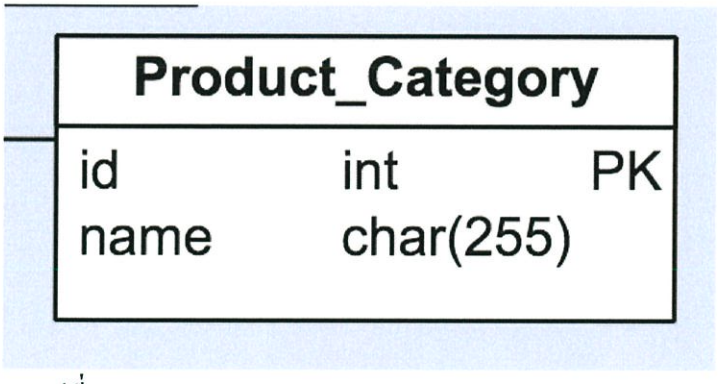


รูปที่ 3.16 ตาราง Product_Image

ตาราง Product_Image ใช้เก็บข้อมูลเกี่ยวกับรูปของสินค้า มีคุณสมบัติของข้อมูลที่เก็บในตารางดังนี้

1) id เป็น primary key ของตาราง ใช้เก็บรหัสของรูปสินค้า
2) url ใช้เก็บที่อยู่ไฟล์รูปของสินค้า
3) Product_id เป็น foreign key ที่ใช้ระบุถึงสินค้า

3.4.14 ตาราง Product_Category

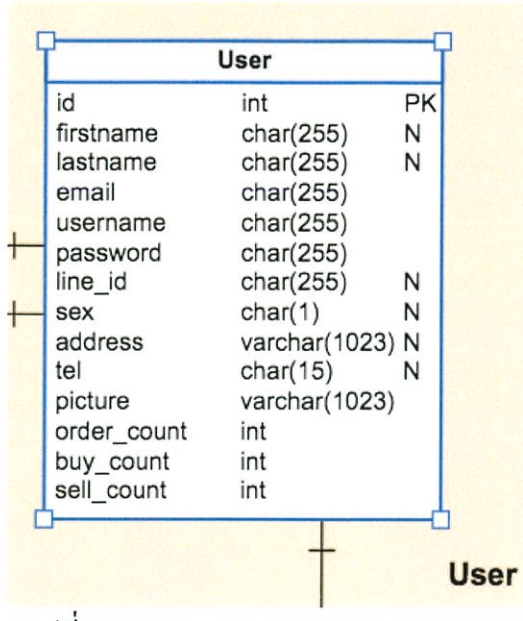


รูปที่ 3.17 ตาราง Product_Category

ตาราง Product_Category ใช้เก็บข้อมูลเกี่ยวกับหมวดหมู่ของสินค้า มีคุณสมบัติของข้อมูลที่เก็บในตารางดังนี้

1) id เป็น primary key ของตาราง ใช้เก็บรหัสของรูปสินค้า
2) name ใช้เก็บที่อยู่ไฟล์รูปของสินค้า

3.4.15 ตาราง Customer



รูปที่ 3.18 ตาราง User

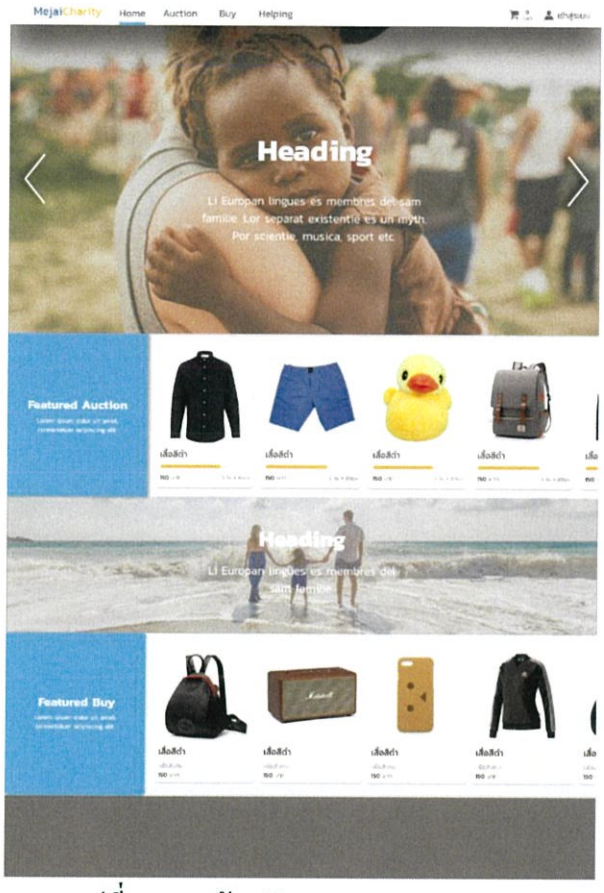
ตาราง User ใช้เก็บข้อมูลเกี่ยวกับผู้ใช้งาน มีคุณสมบัติของข้อมูลที่เก็บในตารางดังนี้

1) id เป็น primary key ของตาราง ใช้เก็บรหัสของผู้ใช้งาน
2) firstname ใช้เก็บชื่อจริงของผู้ใช้งาน
3) lastname ใช้เก็บนามสกุลของผู้ใช้งาน
4) email ใช้เก็บอีเมลของผู้ใช้งาน
5) username ใช้เก็บชื่อสำหรับการ login
6) password ใช้เก็บรหัสผ่านสำหรับการ login
7) line_id ใช้เก็บ line id ของผู้ใช้งาน
8) sex ใช้เก็บเพศของผู้ใช้งาน
9) address ใช้เก็บที่อยู่ของผู้ใช้งาน
10) tel ใช้เก็บเบอร์โทรศัพท์ของผู้ใช้งาน
11) picture ใช้รูปที่อยู่รูปถ่ายของผู้ใช้งาน
12) order_count ใช้เก็บจำนวนรายการสินค้าที่เสร็จสิ้นแล้ว
13) buy_count ใช้เก็บจำนวนครั้งที่ซื้อสินค้า
14) sell_count ใช้เก็บจำนวนครั้งที่ขายสินค้า

3.5 การออกแบบส่วนติดต่อผู้ใช้งาน (User Interface)

ส่วนติดต่อผู้ใช้งานจะแบ่งออกเป็นหน้าเว็บหน้าต่างๆ ตามการทำงานสามารถแบ่งหน้าได้ดังนี้

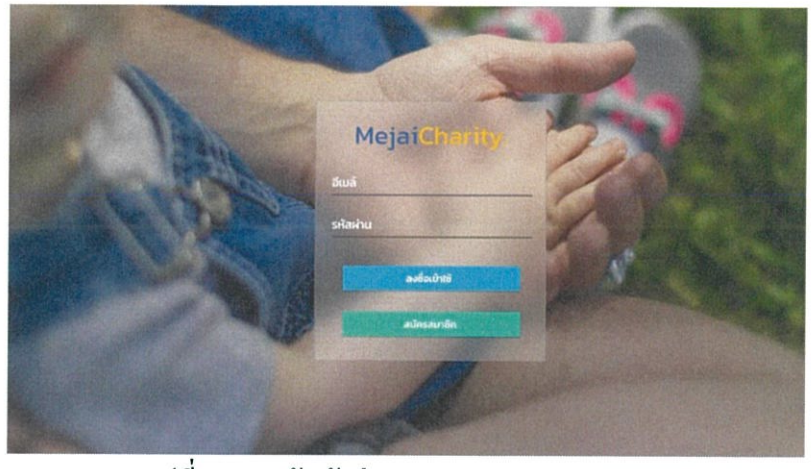
3.5.1 หน้าหลัก



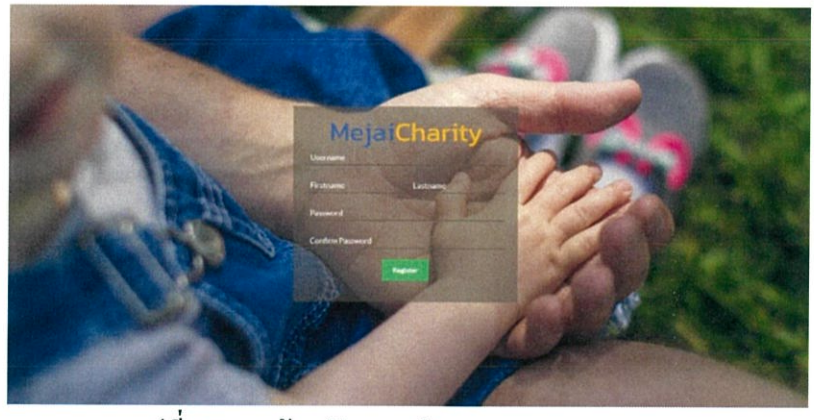
รูปที่ 3.19 หน้าหลัก

หน้าหลักจะเป็นหน้าเริ่มต้นเพื่อไปสู่หน้าต่างๆ รายละเอียดในหน้านี้จะมีรายการสินค้าและการประมูลที่เป็นที่นิยม สามารถเพิ่มได้โดยผู้ดูแล และข้อความของเว็บที่สามารถแก้ไขได้โดยผู้ดูแลเว็บ

3.5.2 หน้าเข้าสู่ระบบและสมัครสมาชิก



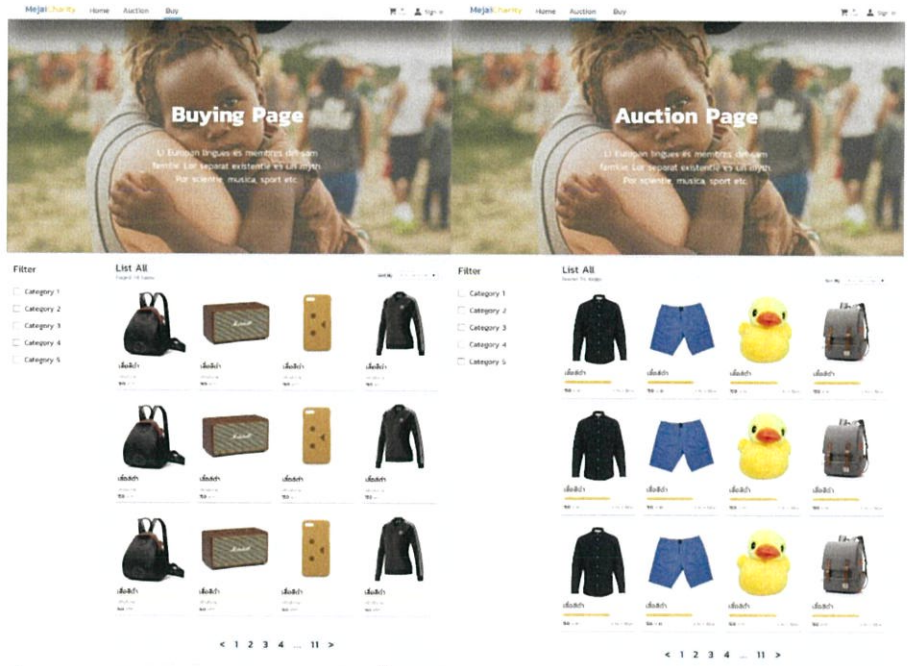
รูปที่ 3.20 หน้าเข้าสู่ระบบ



รูปที่ 3.21 หน้าสมัครสมาชิก

เมื่อกดเข้าสู่ระบบจากแถบด้านบนจะแสดงหน้าเข้าสู่ระบบก่อน หากผู้ใช้งานยังไม่ได้สมัครสมาชิกต้องกดปุ่มสมัครสมาชิกแล้วจึงแสดงฟอร์มการสมัคร หากสมัครเสร็จแล้วจะกลับมาที่หน้าเข้าสู่ระบบ ให้ผู้ใช้กรอกข้อมูลของตนเอง เมื่อเข้าสู่ระบบสำเร็จจะกลับไปที่หน้าหลัก

3.5.3 หน้าแสดงรายการสินค้า



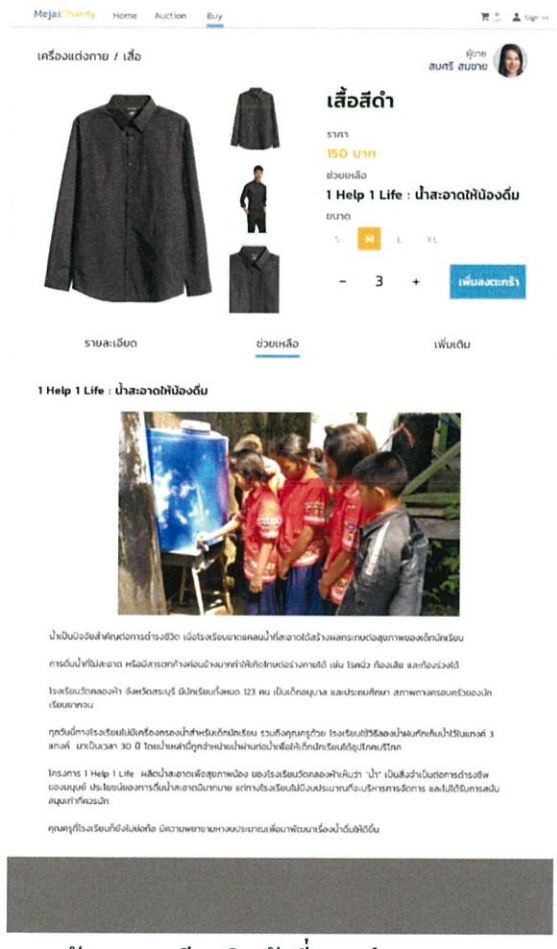
รูปที่ 3.22 หน้าแสดงรายการสินค้า ซื้อสินค้า (ซ้าย) ประมูล (ขวา)

หน้าแสดงรายการสินค้าจะมีสองประเภทคือสินค้าที่จำหน่าย และสินค้าที่เปิดประมูล โดยส่วนที่แตกต่างคือสินค้าที่ขายจะแสดงข้อมูลรูปภาพ ชื่อสินค้า ราคา และองค์กรการกุศลที่ช่วยเหลือ ส่วนในหน้าการประมูลจะแสดงรูปภาพ ชื่อสินค้า แถบสีเหลืองที่แทนเวลาที่จะหมดเวลาประมูล ราคาปัจจุบัน และเวลาที่เหลือจนกว่าจะสิ้นสุดการประมูล

3.5.4 หน้ารายละเอียดของสินค้า

หน้ารายละเอียดสินค้าจะมี 2 ประเภทคือ สินค้าที่จำหน่ายและสินค้าที่เปิดประมูล

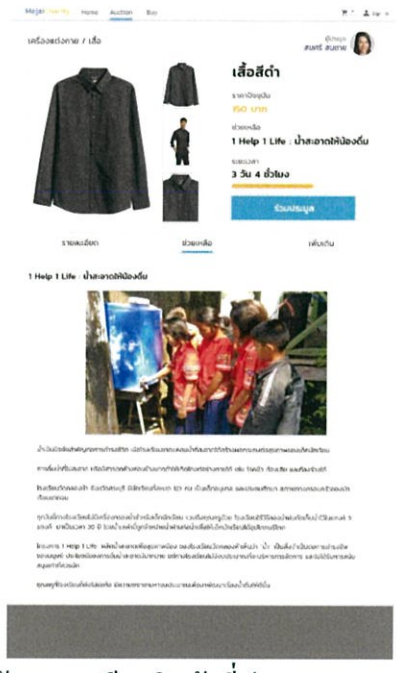
3.5.4.1 สินค้าที่จำหน่าย



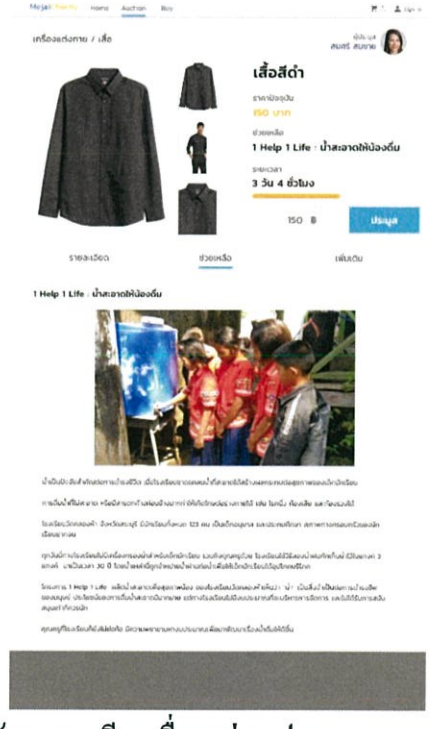
รูปที่ 3.23 หน้ารายละเอียดสินค้าที่จำหน่าย

สินค้าที่จำหน่ายจะแสดงข้อมูลรายละเอียดของสินค้า และสามารถเพิ่มจำนวนที่ต้องการเพิ่มลงตะกร้าได้

3.5.4.2 สินค้าที่ประมูล



รูปที่ 3.24 หน้ารายละเอียดสินค้าที่ประมูล

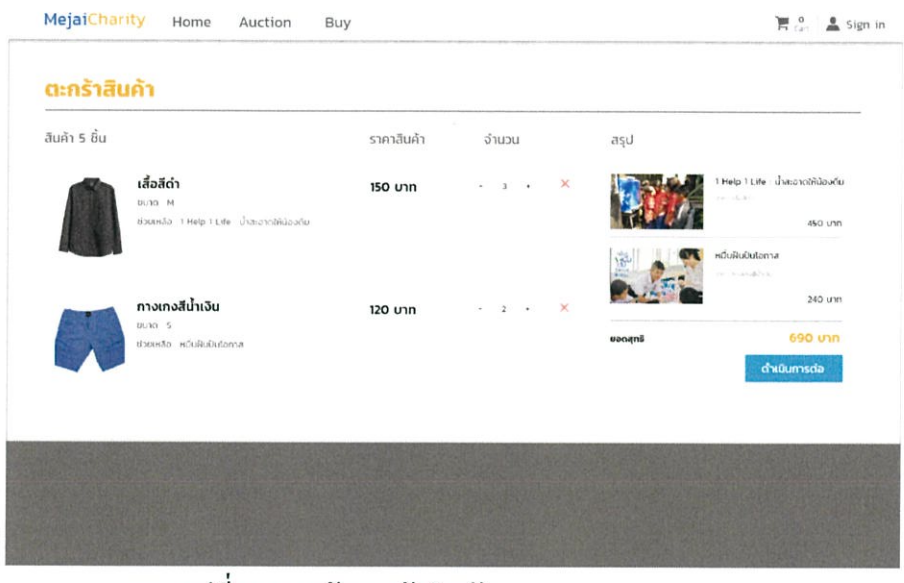


รูปที่ 3.25 หน้ารายละเอียดเมื่อกดร่วมประมูล

หน้ารายละเอียดสินค้าที่ประมูลจะแสดงรายละเอียดของสินค้า และรายละเอียดการประมูล ถ้าผู้ใช้ต้องการประมูลจะต้องกดที่ร่วมประมูล ระบบจะแสดงผลเป็นช่องให้กรอกราคา

และราคาที่กรอกต้องสูงกว่าราคาปัจจุบัน เมื่อกรอกเสร็จแล้วให้กดที่ประมูล ราคาปัจจุบันจะเป็นราคาที่ผู้ใช้กรอก

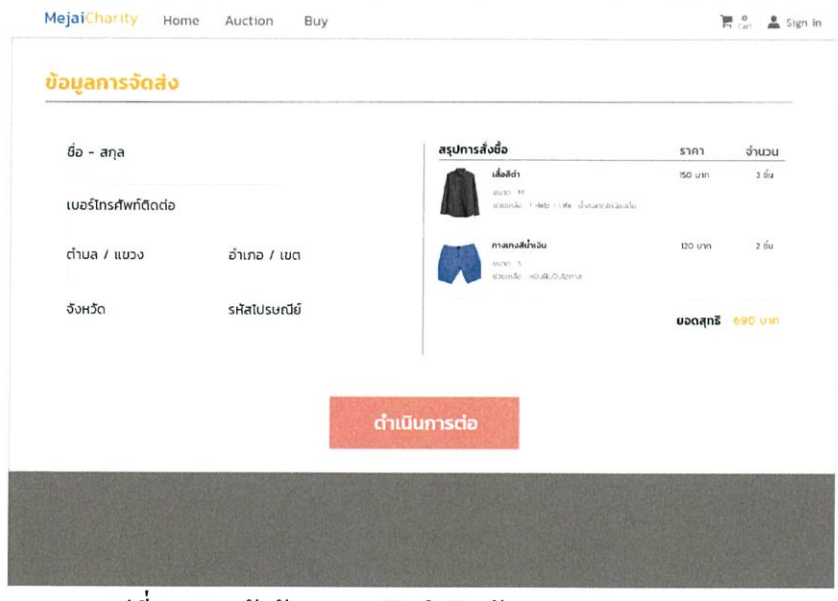
3.5.5 หน้าตะกร้าสินค้า



รูปที่ 3.26 หน้าตะกร้าสินค้า

หน้าตะกร้าสินค้าสามารถเข้าได้จากปุ่มตะกร้าสินค้าที่แถบด้านบน ในหน้านี้จะแสดงสินค้าที่อยู่ในตะกร้าของผู้ใช้ และสรุปจำนวนเงินที่ผู้ใช้ช่วยเหลือองค์กรการกุศล และปุ่มดำเนินการต่อเพื่อเข้าสู่หน้ากรอกข้อมูลการจัดส่งสินค้า

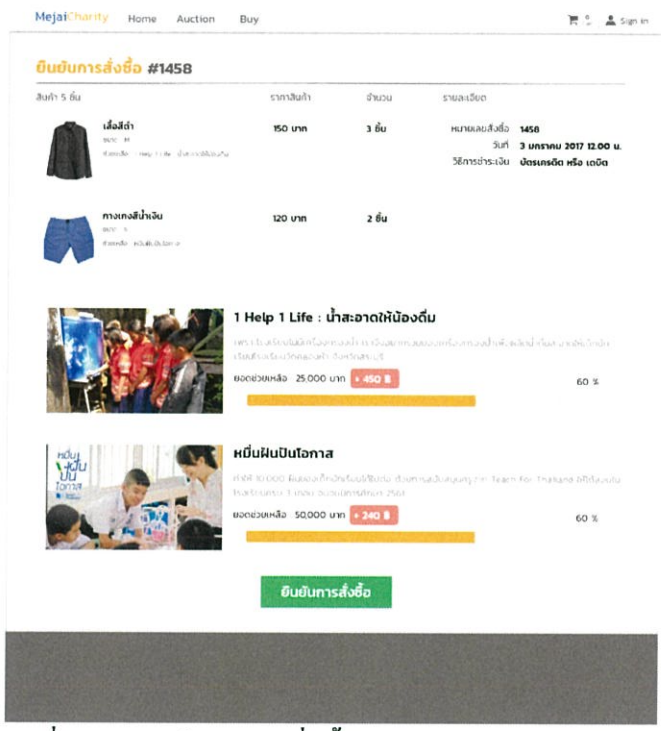
3.5.6 หน้าข้อมูลการจัดส่งสินค้า



รูปที่ 3.27 หน้าข้อมูลการจัดส่งสินค้า

หน้านี้จะมาจากหน้าตะกร้าสินค้าผู้ใช้จะต้องกรอกที่อยู่ที่ต้องการจัดส่งสินค้า และสรุปการสั่งซื้อด้านขวาเมื่อผู้ใช้กรอกข้อมูลแล้ว ให้กดดำเนินการต่อเพื่อยืนยันการสั่งซื้อสินค้า

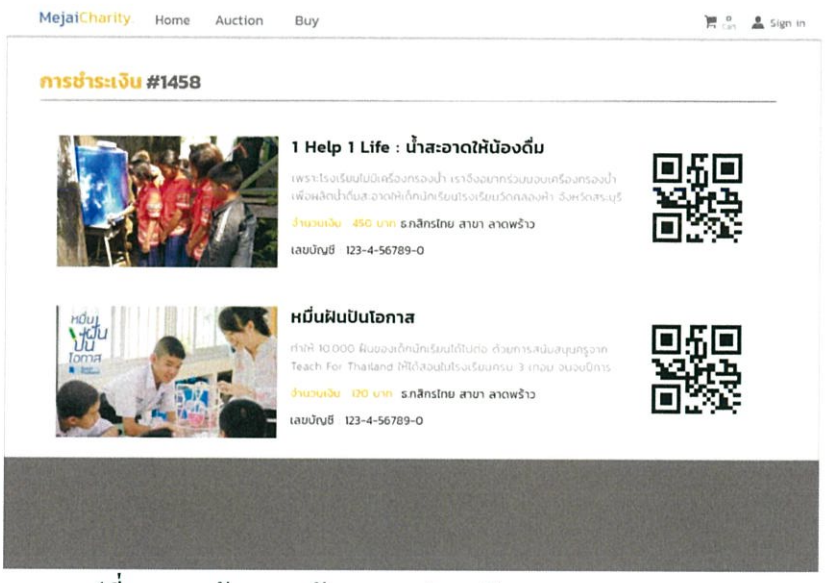
3.5.7 หน้ายืนยันการสั่งซื้อสินค้า



รูปที่ 3.28 หน้ายืนยันการสั่งซื้อสินค้า

หน้ายืนยันการสั่งซื้อจะมาจากหน้าข้อมูลการจัดส่ง หน้านี้จะแสดงข้อมูลการสั่งซื้อรายละเอียดองค์กรการกุศลที่ช่วยเหลือเมื่อผู้ใช้ตรวจสอบแล้วให้กดที่ยืนยันการสั่งซื้อจะไปหน้าแสดงข้อมูลการชำระเงิน

3.5.8 หน้าแสดงข้อมูลชำระเงิน



รูปที่ 3.29 หน้าแสดงข้อมูลการชำระเงิน

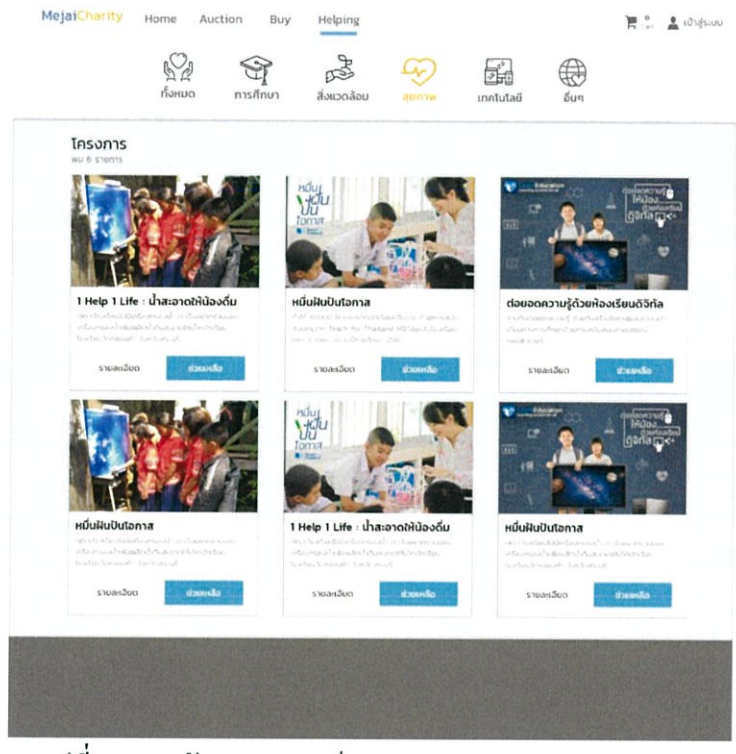
หน้านี้จะแสดงข้อมูลการชำระเงินเป็นเลขบัญชีขององค์กรการกุศล หรือชำระเงินผ่าน QR Code ผ่านระบบพร้อมเพย์ได้

3.5.9 หน้าข้อมูลของผู้ใช้

รูปที่ 3.30 หน้าข้อมูลผู้ใช้

หน้าข้อมูลผู้ใช้สามารถเลือกดูข้อมูลได้ดังนี้ ประวัติการซื้อสินค้า สินค้าที่ผู้ใช้จำหน่าย หรือเปิดประมูล

3.5.10 หน้ารายการองค์กรการกุศล



รูปที่ 3.31 หน้ารายการองค์กรการกุศล

หน้ารายการองค์กรการกุศลจะแสดงแถบด้านเป็นหมวดหมู่ขององค์กรและ จะแสดงรายการขององค์กรที่ตรงกับหมวดหมู่ด้านล่าง ผู้ใช้สามารถกดที่องค์กรกเพื่อดูรายละเอียดเพิ่มเติม และสามารถกดปุ่มช่วยเหลือเพื่อแสดงรายการสินค้าที่ช่วยเหลือได้

บทที่ 4

การทดลองและผลการทดลอง

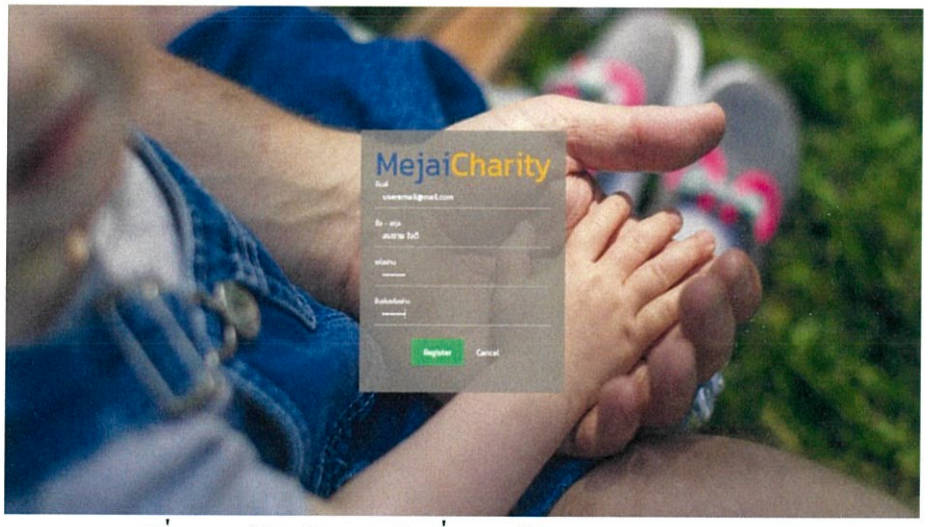
4.1 การทดลองเพิ่มสินค้าเข้าสู่ระบบเพื่อจำหน่าย

4.1.1 จุดประสงค์การทดลอง

การทดลองนี้จะสร้างบัญชีผู้ใช้งาน จากนั้นนำบัญชีที่สร้างเลือกองค์กรการกุศลที่ต้องการช่วยเหลือ และเพิ่มสินค้าเข้าระบบโดยกรอกรายละเอียดต่างๆ

4.1.2 วิธีการดำเนินการ

การเพิ่มสินค้าเข้าระบบจำเป็นต้องมีบัญชีผู้ใช้งาน จึงต้องสร้างบัญชีผู้ใช้งานโดยไปที่หน้าสมัครสมาชิก ระบบจะแสดงแบบฟอร์มให้กรอก



รูปที่ 4.1 หน้าสมัครสมาชิกที่กรอกข้อมูล

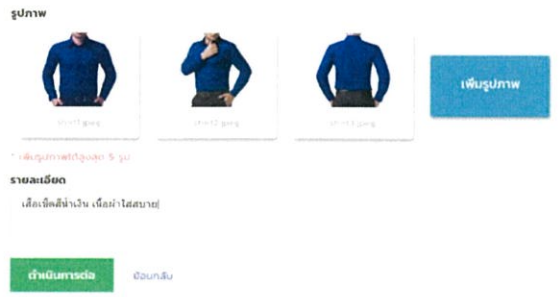
ต่อมาไปที่หน้าเข้าสู่ระบบ และนำบัญชีที่สร้างเข้าสู่ระบบ เมื่ออยู่ในระบบไปที่หน้าเลือกองค์กรการกุศล และเลือกช่วยเหลือองค์กร โดยกดที่ปุ่มช่วยเหลือ ระบบจะแสดงแบบฟอร์มให้กรอกรายละเอียดของสินค้า

รูปที่ 4.2 หน้าเข้าสู่ระบบ

รูปที่ 4.3 หน้าเลือกองค์กรการกุศลที่ต้องการช่วยเหลือ

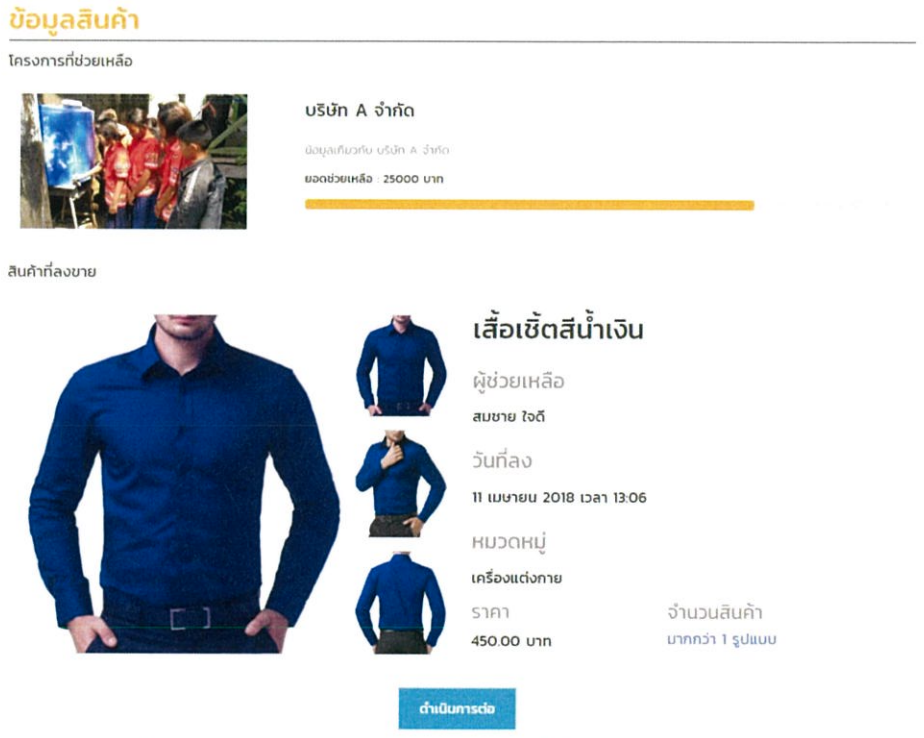
ในการทดลองนี้ สินค้าที่จะเพิ่มเป็นเสื้อเชิ้ตสีน้ำเงิน ราคา 450 บาท มีคุณสมบัติของสินค้าคือขนาดได้แก่ เล็ก กลาง ใหญ่ ให้ผู้ซื้อสามารถเลือกได้ และเพิ่มรูปภาพสำหรับสินค้านี้ 3 รูป

รูปที่ 4.4 แบบฟอร์มกรอกข้อมูลสินค้า



รูปที่ 4.5 แบบฟอร์มกรอกข้อมูลสินค้า (ต่อ)

เมื่อกดดำเนินการในแบบฟอร์ม สินค้าจะถูกเพิ่มเข้าระบบ และแสดงหน้ารายละเอียดของสินค้าที่เพิ่ม

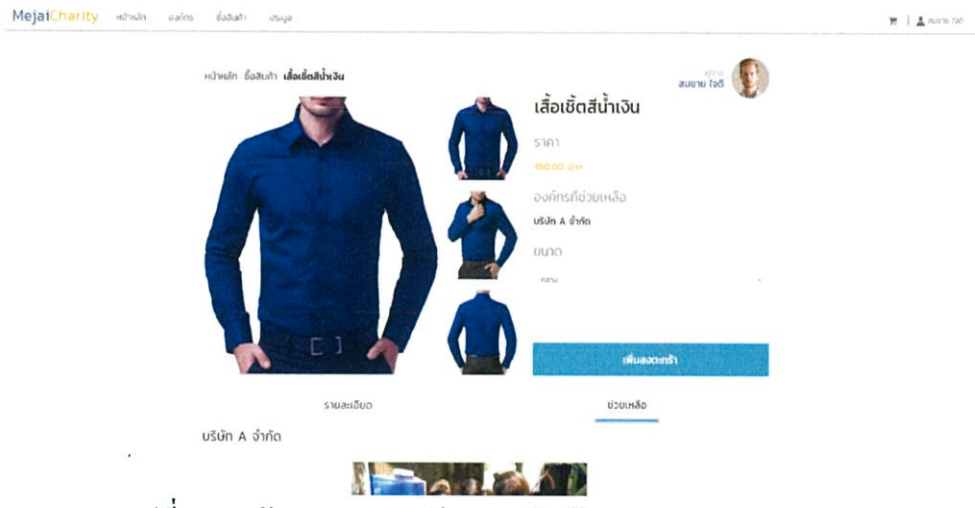


รูปที่ 4.6 หน้าแสดงรายละเอียดสินค้าที่เพิ่ม

4.1.3 ผลการทดลอง

เมื่อไปที่หน้าแสดงรายการสินค้าที่จำหน่ายจะพบสินค้าที่เพิ่ม และเมื่อเข้าไปหน้ารายละเอียดของสินค้าที่เพิ่ม ระบบแสดงรายละเอียดที่กรอกเข้าไปได้ถูกต้อง และสามารถเลือกคุณสมบัติต่างๆที่กำหนดไว้ได้

รูปที่ 4.7 หน้าแสดงรายการสินค้าที่จำหน่าย



รูปที่ 4.8 หน้าแสดงรายละเอียดของสินค้า

4.2 การทดลองซื้อสินค้าในระบบ

4.2.1 จุดประสงค์การทดลอง

การทดลองนี้จะใช้บัญชีผู้ใช้งานที่สร้างขึ้นจากการทดลองก่อนหน้า ซื้อสินค้าที่มีในระบบและทดลองเพิ่มหลักฐานการโอน

4.2.2 วิธีการดำเนินการ

ไปที่หน้ารายละเอียดสินค้าและเพิ่มสินค้าลงในตะกร้า เมื่อไปที่ตะกร้าสินค้าของผู้ใช้จะแสดงสินค้าที่เพิ่มเข้ามา จากนั้นกดดำเนินการต่อเพื่อเข้าสู่หน้ากรอกข้อมูลการจัดส่ง

รูปที่ 4.9 หน้าตะกร้าสินค้า

รูปที่ 4.10 หน้าแบบฟอร์มกรอกข้อมูลการจัดส่ง

เมื่อกรอกข้อมูลการจัดส่งและกดปุ่มดำเนินการต่อเพื่อยืนยันการซื้อสินค้า ในส่วนของผู้ซื้อระบบจะแสดงรายการสั่งซื้อในหน้าข้อมูลผู้ใช้งาน รายการสั่งซื้อที่ถูกสร้างขึ้นคือ หมายเลข 38 มีสถานะเป็น “รอเพิ่มหลักฐานการโอน” จากนั้นกดเข้าไปที่รายการสั่งซื้อและเพิ่มหลักฐานการโอนโดยอัพโหลดเป็นรูปภาพเข้าระบบ

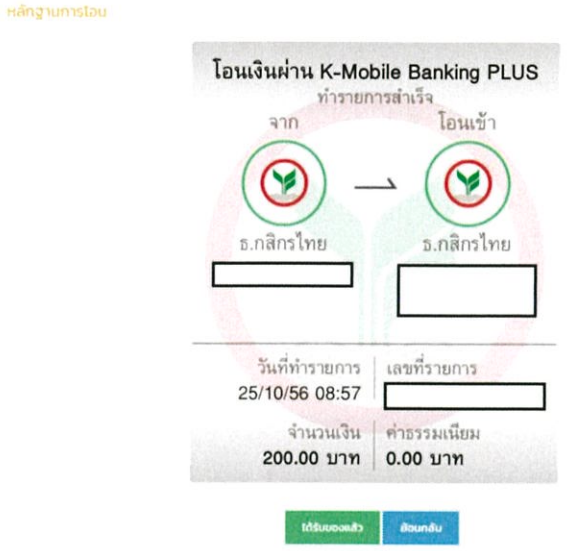
รูปที่ 4.11 หน้าข้อมูลผู้ใช้งานที่แสดงรายการสั่งซื้อ

4.2.3 ผลการทดลอง

เมื่อผู้ขายเข้าไปดูรายการสั่งซื้อหมายเลข 38 จะพบว่าสถานะของรายการสั่งซื้อเป็น “รอการส่งของ” และระบบแสดงหลักฐานการโอนที่เพิ่มได้ถูกต้อง

รูปที่ 4.12 หน้ารายการสั่งซื้อที่มีสถานะเป็น “รอการส่งของ”

ในส่วนของผู้ขายสามารถกปุ่ม ส่งของแล้ว ได้และสถานะของรายการสั่งซื้อจะเปลี่ยนเป็นส่งของแล้ว จากนั้นในส่วนของผู้ซื้อหน้ารายละเอียดรายการสั่งซื้อจะแสดงปุ่ม ได้รับของแล้ว ให้ผู้ซื้อกดเมื่อได้รับของที่สั่งแล้ว



รูปที่ 4.13 หน้ารายการสั่งซื้อที่แสดงปุ่ม "ได้รับของแล้ว"

4.3 การทดลองประมูลสินค้าในระบบ

4.3.1 จุดประสงค์การทดลอง

การทดลองนี้จะทดลองประมูลสินค้าในระบบ โดยประมูลราคาที่มากกว่าราคาปัจจุบัน รอให้เวลาการประมูลจบลง และชนะการประมูล

4.3.2 วิธีการดำเนินการ

เริ่มการทดลองด้วยการเข้าระบบบัญชีผู้ใช้งานที่มีอยู่ และเข้าสู่หน้ารายละเอียดสินค้าการประมูล โดยจะเข้าร่วมการประมูลเสื้อเชิ้ตสีน้ำเงิน ที่มีราคาปัจจุบันเป็น 20 บาท เมื่อกดปุ่ม ร่วมการประมูล ระบบจะแสดงกล่องให้กรอกราคาที่ต้องการ ในการทดลองนี้จะกรอก 250 บาท และกดที่ปุ่มประมูล ราคาปัจจุบันของระบบจะเปลี่ยนเป็นราคาที่กรอก และปุ่ม ร่วมประมูล จะเปลี่ยนเป็นราคาของคุณ รอจนกระทั่งหมดเวลาการประมูล

รูปที่ 4.14 หน้ารายละเอียดสินค้าการประมูล

รูปที่ 4.15 กล่องสำหรับกรอกราคาที่ต้องการประมูล

รูปที่ 4.16 ปุ่มเมื่อราคาที่กรอกเป็นราคาสูงสุด

เมื่อหมดเวลาการประมูลและเป็นผู้ชนะการประมูล สินค้าจะปรากฏในตะกร้าสินค้า ทำการกดปุ่มดำเนินการต่อ กรอกข้อมูลที่อยู่การจัดส่งและดำเนินการเหมือนการสั่งซื้อสินค้าจนเสร็จสิ้น

รูปที่ 4.17 หน้าตะกร้าสินค้าที่มีสินค้าประมูล

4.3.3 ผลการทดลอง

ในหน้าข้อมูลผู้ใช้ของผู้ซื้อและผู้ขาย จะปรากฏรายการสั่งซื้อเพิ่มขึ้นมา 1 รายการคือหมายเลข 39 มีสถานะรอเพิ่มหลักฐานการโอน และสามารถเพิ่มหลักฐานการโอนได้

รูปที่ 4.18 หน้ารายการคำสั่งซื้อ

4.4 การทดลองปรับโครงสร้างของฐานข้อมูลด้วย Django

4.4.1 จุดประสงค์การทดลอง

การทดลองนี้เพื่อทดสอบความถูกต้องของการทำงานด้วยระบบ Object-relational mapping ที่อยู่บน Django ในการเปลี่ยนแปลงโครงสร้างของฐานข้อมูล

4.4.2 วิธีดำเนินการ

เริ่มจากการตรวจสอบโครงสร้างฐานข้อมูล ณ ปัจจุบันของตาราง Order

#	Name	Type	Collation
1	id	int(11)	
2	time	timestamp	
3	quantity	int(11)	
4	price	decimal(10,2)	
5	status	int(11)	
6	product_id	int(11)	
7	buyer_id	int(11)	
8	slip	varchar(1023)	utf8_unicode_ci
9	address	varchar(1023)	utf8_unicode_ci
10	attribute_id	int(11)	

รูปที่ 4.19 โครงสร้างฐานข้อมูลของตาราง Order ก่อนการทดสอบ

จากรูปภาพข้างต้นมีทั้งหมด 10 field และทำการทดลองด้วยการเพิ่ม field "test" เป็นชนิด "int(11)" ด้วยระบบ Object-relational mapping ของ Django โดยรูปการทำงานนั้นจะถูกควบคุมด้วย Code ของระบบ

ทำการเพิ่ม Code ลงในระบบดังนี้

```
17          Product, on_delete=models.CASCADE, re
18      buyer = models.ForeignKey(
19          Customer, on_delete=models.CASCADE, r
20      )
21      attribute = models.ForeignKey(
22          ProductAttribute, on_delete=models.CA
+       test = models.IntegerField(default=0)
24
25      class Meta:
26          managed = True
27          db_table = 'Order'
28
```

รูปที่ 4.20 Code สำหรับการกำหนดรูปแบบโครงสร้างฐานข้อมูลใหม่

Code ที่เพิ่มเข้าสู่ระบบ Model ของ Django ดังบรรทัดที่ 23 หลังจากนั้นทำการสั่งให้ Django สร้าง Code ที่เชื่อมต่อฐานข้อมูลเพื่อทำการปรับโครงสร้างของฐานข้อมูลด้วยคำสั่งดังนี้

```
root@6b55c6841f90:/code# python manage.py makemigrations
Migrations for 'order':
  order/migrations/0014_order_test.py
    - Add field test to order
root@6b55c6841f90:/code#
```

รูปที่ 4.21 การรันคำสั่งเพื่อสร้างไฟล์สำหรับกำหนดโครงสร้างฐานข้อมูล

คำสั่งสำหรับการสั่ง Django ให้สร้าง คือ “python manage.py makemigrations” และ Django จะสร้างไฟล์สำหรับการสั่งการ Object-relational mapping ที่ชื่อว่า “0014_order_test.py” ออกมา ภายในไฟล์ดังกล่าวมีข้อมูลดังนี้

```
1  # -*- coding: utf-8 -*-
2  # Generated by Django 1.11 on 2018-04-12 06:56
3  from __future__ import unicode_literals
4
5  from django.db import migrations, models
6
7
8  class Migration(migrations.Migration):
9
10     dependencies = [
11         ('order', '0013_auto_20180412_0108'),
12     ]
13
14     operations = [
15         migrations.AddField(
16             model_name='order',
17             name='test',
18             field=models.IntegerField(default=0),
19         ),
20     ]
21
```

รูปที่ 4.22 Code ภายในไฟล์ที่ใช้สำหรับกำหนดโครงสร้างฐานข้อมูล

หลังจากนั้น ทำการสั่ง Django ให้ปรับโครงสร้างฐานข้อมูลด้วยคำสั่ง “python manage.py migrate” ดังภาพ

```
root@6b55c6841f90:/code# python manage.py makemigrations
Migrations for 'order':
  order/migrations/0014_order_test.py
    - Add field test to order
root@6b55c6841f90:/code# python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auction, auction_customer, auth, authtoken, bank, cart, car
t_product, contenttypes, customer, order, order_organization, organization, organization_b
ank, organization_promptpay, product, product_attribute, product_category, product_image,
sessions
Running migrations:
  Applying order.0014_order_test... OK
root@6b55c6841f90:/code#
```

รูปที่ 4.23 การรันคำสั่งเพื่อปรับโครงสร้างฐานข้อมูลใหม่

จากภาพ สังเกตได้ว่า Django จะทำการตรวจสอบหาจุดที่ต่างจากโครงสร้างของฐานข้อมูล และ Code สำหรับการปรับเปลี่ยน หากตรวจพบจะทำการปรับโครงสร้างของฐานข้อมูลให้ตรงกับ Code ที่ได้สร้างไว้

4.4.3 ผลการทดลอง

หลังจากทำการทดลองเสร็จสิ้นแล้ว ทำการตรวจสอบโครงสร้างของฐานข้อมูลภายในตาราง Order

#	Name	Type	Collation
1	id	int(11)	
2	time	timestamp	
3	quantity	int(11)	
4	price	decimal(10,2)	
5	status	int(11)	
6	product_id	int(11)	
7	buyer_id	int(11)	
8	slip	varchar(1023)	utf8_unicode_ci
9	address	varchar(1023)	utf8_unicode_ci
10	attribute_id	int(11)	
11	test	int(11)	

รูปที่ 4.24 โครงสร้างฐานข้อมูลของตาราง Order หลังการทดสอบ

โครงสร้างภายในฐานข้อมูลที่ field ที่ 11 เพิ่มขึ้นมาคือ “test” เป็นชนิด “int(11)”

4.5 การทดลองเรียกข้อมูลจากฐานข้อมูลด้วย Application programming interface

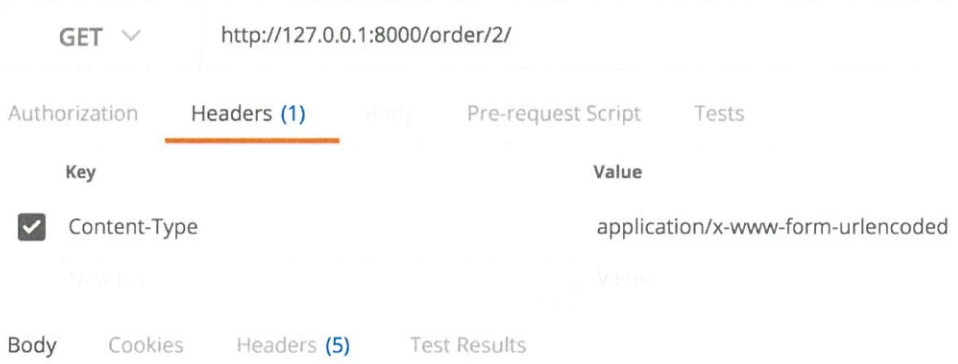
4.5.1 จุดประสงค์การทดลอง

การทดลองนี้เพื่อทดสอบการใช้ง่าย Application programming interface (API) ว่าสามารถส่ง Request ไปยัง API แล้วตอบกลับมาด้วยข้อมูลที่ถูกต้องจากฐานข้อมูลหรือไม่

4.5.2 วิธีดำเนินการ

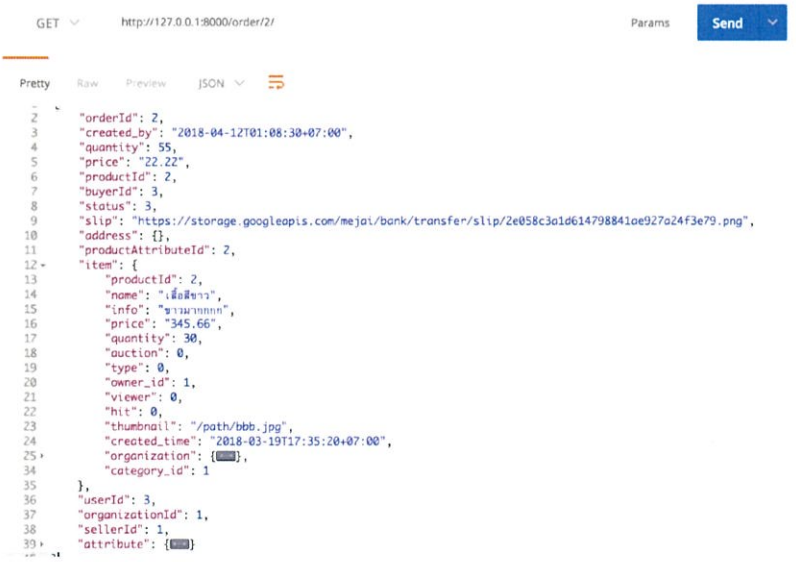
การทดลองนี้จะทดสอบด้วยการเรียกข้อมูลของ Order ลำดับที่ 2 จากระบบ Application programming interface

สร้าง Request ไปยัง endpoint ของการเรียกข้อมูลคือ GET {HOST}/order/{order_id}/ ดังภาพ



รูปที่ 4.25 ทดสอบการส่งคำร้องไปยัง Server เพื่อเรียกดูข้อมูลของ Order

หลังจากนั้นทำการเรียกข้อมูลไปยัง Server

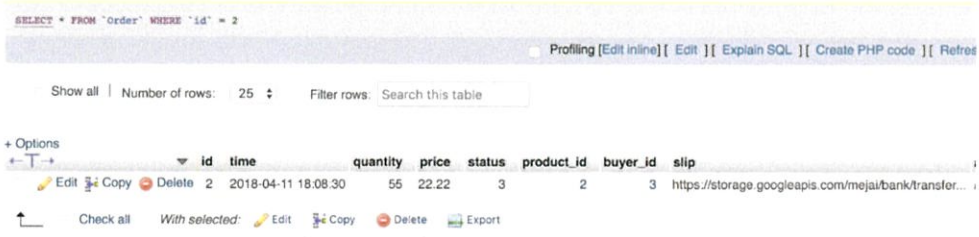


รูปที่ 4.26 ผลลัพธ์ที่ได้รับจาก Server ในการร้องขอข้อมูลของ Order

Server และระบบ Application programming interface จะตอบกลับมาด้วยข้อมูลของ Order ลำดับที่ 2 ของระบบด้วยรูปแบบ JSON

4.5.3 ผลการทดลอง

ตรวจสอบความถูกต้องของข้อมูลกับฐานข้อมูล โดยสามารถค้นหาข้อมูลของ Order ลำดับที่ 2 ด้วยคำสั่ง SQL ดังนี้ “SELECT * FROM `Order` WHERE `id` = 2” และได้รับภาพลัพธ์ดังภาพ ข้อมูลที่ได้รับถูกต้องตรงกับการเรียกผ่านระบบ Application programming interface



รูปที่ 4.27 ผลลัพธ์จากการรันคำสั่ง SQL เพื่อค้นหา Row ข้อมูลของ Order

4.6 การทดลองทดสอบระบบรักษาเสถียรภาพของ Infrastructure ด้วย Kubernetes

4.6.1 จุดประสงค์การทดลอง

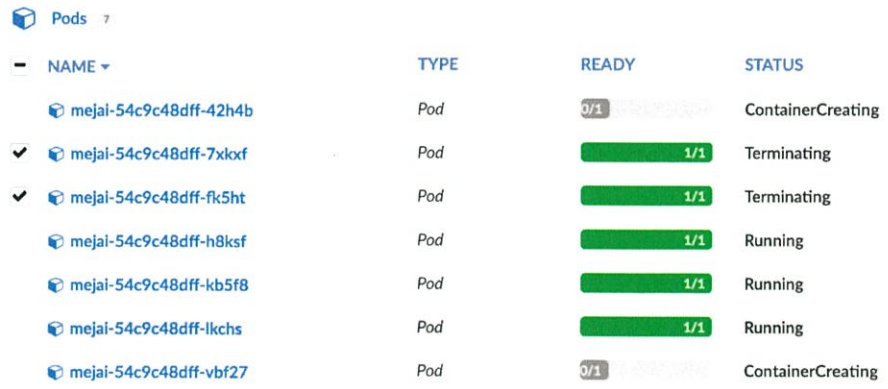
การทดลองนี้เพื่อทดสอบความถูกต้องของการทำงานของระบบ Kubernetes ว่าสามารถรักษาเสถียรภาพของ Infrastructure

4.6.2 วิธีดำเนินการ

ตรวจสอบจำนวน Replica ภายในระบบ

รูปที่ 4.28 Dashboard สำหรับตรวจสอบการทำงานของ Replica ในระบบ

จากการตรวจสอบพบจำนวน Replica ทั้งหมด 5 pods หากมีการเปลี่ยนแปลง หรือมีปัญหาเกิดขึ้นกับระบบ เช่น มี 2 pods ที่ทำงานผิดปกติ หรือหายไปจากระบบ ตัว Kubernetes ต้องทำการสร้าง pod อื่นขึ้นมาทดแทนทันที ทำการทดลองลบ pod 2 ตัวแรกจากระบบ

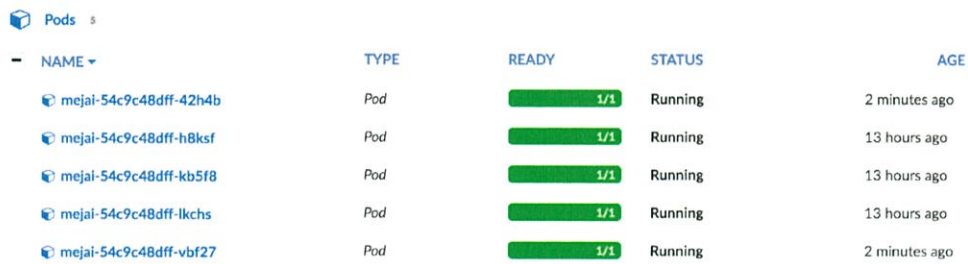


รูปที่ 4.29 ทดสอบลบ Pod ภายในระบบ

จากการทดสอบลบ pod 2 ตัวจากระบบ จะสังเกตได้ว่าจะมี 2 pods เกิดขึ้นใหม่เพื่อทดแทนทันที เป็นการทำงานและควบคุมจาก Kubernetes

4.6.3 ผลการทดลอง

จากการทดสอบลบ pod 2 ตัวจากระบบ ทำให้เกิด 2 pods ใหม่ขึ้นมา โดยสังเกตได้จาก AGE ที่มีอายุเพียง 2 นาที และ pod ดังกล่าวเมื่อเริ่มระบบพร้อมสำหรับการใช้งานแล้วนั้น จะทำการเชื่อมต่อกลับเข้ามาในระบบของ Kubernetes ทันที และนำ 2 pods ที่มีปัญหา (ที่ทดลองทำการลบออก) ออกจากระบบด้วย



รูปที่ 4.30 ผลลัพธ์หลังจากการทดสอบลบ Pods

บทที่ 5
บทสรุปและข้อสรุป

5.1 บทสรุป

ระบบประมูลและขายสินค้าเพื่อการกุศลเป็นระบบที่พัฒนาในรูปแบบเว็บไซต์ ที่แก้ปัญหาการประมูลหรือจำหน่ายสินค้าออนไลน์และต้องการนำรายได้ไปช่วยเหลือองค์กรการกุศล นอกจากนี้ระบบช่วยสร้างความเชื่อใจระหว่างผู้ซื้อและผู้จำหน่ายได้ โดยหลังจากการสั่งซื้อ หรือชนะการประมูลของสินค้าหนึ่ง จะต้องมีการยืนยันขั้นตอนกับระบบด้วยเช่น การเพิ่มหลักฐานการโอนของผู้ซื้อ การแจ้งกับระบบว่าผู้จำหน่ายจัดส่งสินค้าให้กับผู้ซื้อแล้ว เป็นต้น และผู้ซื้อสามารถดูประวัติการขายของผู้จำหน่ายได้

ในการพัฒนาระบบได้ออกแบบส่วนติดต่อผู้ใช้งานและโครงสร้างของระบบ จากนั้นพัฒนาตามที่ออกแบบไว้โดยใช้ React ในส่วนติดต่อผู้ใช้งาน โดยมี Redux ที่ช่วยจัดการสถานะต่างๆของส่วนติดต่อผู้ใช้งาน ส่วนโครงสร้างของระบบนำ Django มาใช้ในการติดต่อฐานข้อมูล ประมวลผลข้อมูลต่างๆ และติดต่อส่วนติดต่อผู้ใช้งานในรูปแบบ RESTful API

ระบบโครงสร้างของระบบนั้นใช้ Kubernetes ในควบคุมการทำงานของระบบ และดูแลระบบด้วยตนเอง โดยระบบสามารถรักษาเสถียรภาพของการทำงานไว้ได้ตลอดเวลาโดยผู้สร้างระบบทำการกำหนดกฎต่างๆ ไว้ตั้งแต่ต้น และ Kubernetes จะดูแลระบบและคงสภาพนั้นไว้ตลอดเวลาที่กำหนด หากมีความผิดพลาด หรือการเปลี่ยนแปลงที่ไม่คาดฝันเกิดขึ้น ตัวระบบจะสามารถรักษาตัวเอง เพื่อให้คงสภาพเดิมของระบบไว้ได้

5.2 ปัญหาอุปสรรคและแนวทางแก้ไข

1) ส่วนติดต่อผู้ใช้งานไม่สามารถทำงานได้ หรือแสดงข้อมูลผิดพลาด แก้ไขโดยปรับโครงสร้างการเก็บสถานะในส่วนติดต่อผู้ใช้งาน และกำหนดค่าเริ่มต้นของสถานะ

2) เมื่อข้อมูลมีความยาวมากขึ้นเช่น ชื่อของสินค้า ราคาที่มีเลขหลายหลัก เป็นต้นทำให้ส่วนติดต่อผู้ใช้งานบางส่วนแสดงผลผิดพลาด ทำให้ข้อมูลตกหล่น แก้ไขโดยปรับเปลี่ยนโครงสร้างของส่วนติดต่อผู้ใช้งานบางส่วน ให้รองรับความยาวของข้อมูลที่มากขึ้น และคงการทำงานเดิมเอาไว้

3) การโอนเงินที่ต่างธนาคารทำให้เกิดความยุ่งยาก และอาจมีค่าใช้จ่ายเพิ่มเติม แก้ไขด้วยการนำระบบพร้อมเพย์มาใช้ และมี QR Code ทำให้ผู้ใช้สามารถชำระเงินบนแอพพลิเคชันต่างๆ ของธนาคารได้

4) ระบบของ Django นั้นเขียนบนพื้นฐานของภาษา Python จึงมี library ที่หลากหลาย และอาจจะมีการทำงานที่ทับซ้อนกัน รวมไปถึงความปลอดภัยที่ต้องเข้าใจถึงหลักการทำงานของ library นั้นๆ

5) การใช้ Kubernetes นั้น สถานะของ API ในระบบยังคงเป็น เวอร์ชั่น Beta จึงต้องมั่นคอยตรวจสอบการอัพเดตเวอร์ชันใหม่ๆ และทดสอบการทำงานกับระบบเดิมของเราเอง

6) การใช้ Helm เพื่อช่วยสร้าง Chart สำหรับ Kubernetes นั้น ต้องเขียนให้ถูกต้องชัดเจนเพื่อลดความผิดพลาดที่จะเกิดขึ้นในอนาคตเพราะมีโอกาสผิดพลาดได้สูงจากการสเกลระบบขึ้นในระดับที่ใหญ่มากขึ้น

7) ด้วยระบบ Kubernetes ที่ต้องใช้ container ในการติดตั้ง ต้องเตรียม Environment ให้เรียบร้อย ซึ่งมีความแตกต่างจาก Environment ที่ใช้ในการพัฒนา แก้ไขด้วยการใช้ ENV ภายใน Helm เข้ามาแก้ไขได้

8) ปัญหาการจัดเก็บรูปภาพที่ผู้ใช้งานอัพโหลด เพราะไม่มีพื้นที่ที่เป็น Persistent Storage สำหรับ Kubernetes เพราะไม่ได้ออกแบบมาให้เก็บรูปภาพได้ จึงแก้ไขด้วยการใช้ Storage ของ Google ในการจัดเก็บรูปภาพต่างๆ

10) ปัญหาความปลอดภัยของรูปภาพที่ผู้ใช้งานอัพโหลดเข้าระบบ อาจจะทำให้คนทั่วไปสามารถเข้าถึงไปด้วยการสุ่มสร้าง URL ขึ้นมา แก้ไขโดยใช้วิธีการสร้าง hash ที่ random มาสำหรับการแก้ไขชื่อ เพื่อป้องกันการเข้าถึงแบบสุ่ม

11) ระบบเพิ่มข้อมูลพร้อมกับการอัพโหลดรูปภาพ ซึ่งการปรับเปลี่ยนจากส่วนติดต่อผู้ใช้งานทำได้ยาก จึงแก้ไขด้วยการแยกระบบระหว่างการ เพิ่ม/แก้ไข ข้อมูล กับระบบอัพโหลดรูปภาพออกจากกัน

12) ปัญหาการที่ Pods ในระบบ Kubernetes มีหลายจำนวน และกระจายกันไปตาม Node แต่ละตัว แต่ทางผู้พัฒนาต้องการเพียง 1 Endpoint สำหรับการติดต่อ Server จึงต้องทำการสร้าง Load Balance ผ่าน Helm ออกไปยังระบบเพื่อใช้งาน Endpoint เดียว

5.3 แนวทางการพัฒนาต่อ

ส่วนติดต่อผู้ใช้งานจะปรับการออกแบบให้ผู้ใช้งานง่ายขึ้น และเกิดความน่าเชื่อถือสำหรับผู้ใช้งานครั้งแรกและพัฒนาหน้าเว็บให้มีประสิทธิภาพสามารถใช้งานได้บนขนาดหน้าจอที่หลากหลายโดยปรับรูปแบบของหน้าเว็บให้เหมาะสมกับขนาดจอที่ผู้ใช้งานกำลังใช้งาน

ส่วนการทำงานของ Kubernetes นั้นจะพัฒนาให้มีความสัมพันธ์กับ Helm มากขึ้น และการจัดเก็บ Images ของ Docker จะต้องทำให้เป็นระบบ, มีขอบเขตชัดเจน และเรียบง่ายต่อความเข้าใจในการศึกษาของผู้ที่สนใจในการพัฒนาต่อจากนั้น

ภาคผนวก ก

การสร้างและใช้งาน React

การสร้างส่วนประกอบหนึ่ง หรือ Component ของ React จะสร้างไฟล์ JavaScript จำนวน 1 ไฟล์ต่อ 1 Component

```
import React, { Component } from 'react'
class HelloWorld extends Component {
        render() {
                return (
                        <div>Hello World</div>
                )
        }
}
export default HelloWorld
```

โปรแกรมที่ 1การสร้างส่วนประกอบพื้นฐาน

จากโค๊ดข้างต้นจะได้ Component ที่แสดงผลคำว่า Hello World โดยจะต้องนำคลาส React และ Component จากไฟล์ react มาใช้ และประกาศชื่อ Component เป็น HelloWorld ที่ถูก export default ไว้เพื่อให้สามารถนำไปใช้ได้ รายละเอียดต่างๆของ Component จะต้องใส่ไว้ในฟังก์ชันในรูปแบบ JSX

ภาคผนวก ข

การสร้างและใช้งาน Redux

การใช้งาน Redux ซึ่งเป็นตัวจัดการสถานะของส่วนติดต่อผู้ใช้งาน จะเริ่มสร้างจาก Action ที่จะเกิดเมื่อต้องการเปลี่ยนสถานะ

```
export const Hello = () => {
   return {
      type: SAY_HELLO',
      text: 'Hello World'
  }
}
```

โปรแกรมที่ 2 การกำหนด Action

จากโค๊ดข้างต้นเป็นการสร้างชนิดของ Action ขึ้นมาโดยกำหนดที่ type และสามารถใส่ข้อมูลได้ในลักษณะของ JavaScript Object ซึ่ง Action ข้างต้นจะมีข้อมูลเป็น text ที่มีค่า Hello World ต่อมาจึงสร้าง Reducer ที่ทำหน้าที่เปลี่ยนสถานะตาม type ของ Action ที่เกิดขึ้น

```
const textReducer = ( state = {}, action ) => {
   switch (action.type) {
         case 'SAY_HELLO':
                return {
                sayText: action.text
         }
  }
}
```

โปรแกรมที่ 3 การสร้าง Reducer

ภาคผนวก ค

การสร้างและใช้งาน Django

การกำหนดโครงสร้างของฐานข้อมูลสำหรับ Django นั้น สามารถกระทำผ่าน Library ที่มีชื่อว่า models ที่ถูกบรรจุอยู่ใน django.db หลังจากที่กำหนดโครงสร้างเสร็จแล้ว Django จะสร้างไฟล์สำหรับการ Config ฐานข้อมูลให้อัตโนมัติ

```
from __future__ import unicode_literals
from django.db import models
class Bank(models.Model):
  name_en = models.CharField(max_length=255)
  name_th = models.CharField(max_length=255)
  symbol = models.CharField(max_length=10)
  logo = models.CharField(max_length=1023)
  class Meta:
    managed = True
    db_table = 'Bank'
```

โปรแกรมที่ 4 ไฟล์สำหรับ Config ฐานข้อมูล

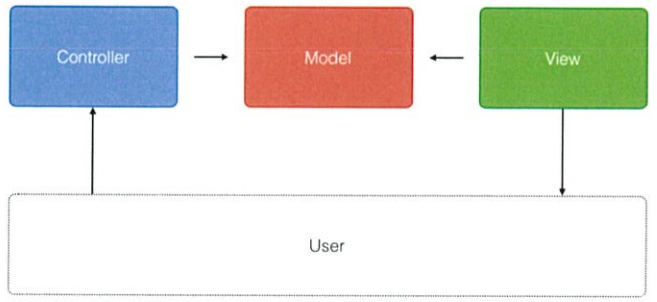
จากโค้ดข้างต้นนั้น เป็นการกำหนดฐานข้อมูลที่มีชื่อ Table ว่า "Bank" โดยกำหนดให้มี Attribute ดังนี้ name_en, name_th, symbol และ logo โดย Django จะสร้างไฟล์สำหรับการ Migration ไปยัง Database ให้ตรงตามที่กำหนดไว้

ภาคผนวก ฆ

การสร้างและใช้งาน MVC

เนื่องจาก MVC นั้นเป็นการแยกส่วนของโปรแกรมของเราออกเป็น Model, View และ Controller

1) Model - ในที่นี้ทำหน้าที่เป็นส่วนของการติดต่อกับฐานข้อมูล
2) View - เป็นส่วนในการเตรียมข้อมูลสำหรับการแสดงผลของ API
3) Controller - ทำหน้าที่ควบคุมการทำงานต่างๆ รวมไปถึงการทำงานของ API ด้วย



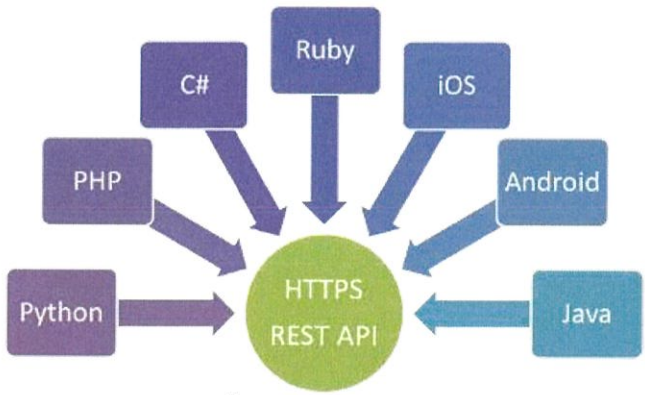
โครงสร้าง MVC

จากภาพข้างต้น MVC นั้นเป็นการออกแบบโปรแกรมให้เป็นส่วนย่อยทั้งหมด 3 ส่วน และผู้ใช้งานจะทำการติดต่อกับ Controller และ View ส่วนระบบฐานข้อมูลนั้นจะถูกติดต่อจากส่วนของ Model อีกที

ภาคผนวก ง

การสร้างและใช้งาน RESTFul

RESTFul หรือ REST นั้นย่อมาจาก Representational state transfer ซึ่งคือการสร้าง Webservice ที่เป็นตัวกลางสำหรับการติดต่อ และลดปัญหาความแตกต่างของแต่ละภาษาที่ใช้พัฒนา ทางผู้จัดทำ Mejai นั้น เล็งเห็นประโยชน์ของการทำ RESTFul ในอนาคตว่า หากผู้ใดต้องการพัฒนาต่อด้วยภาษาอื่นๆ นอกเหนือจากที่ Mejai ได้พัฒนามานั้น ก็สามารถจะทำต่อได้ง่าย



การติดต่อไปที่ RESTful API

โดยการใช้งานนั้นทางผู้จัดทำเลือกมาตราฐานที่มีชื่อว่า JSON (JavaScript Object Notation) ซึ่งถือเป็นรูปแบบที่ใช้อย่างแพร่หลาย ยกตัวอย่างข้อมูลของ Organization list ภายในระบบ Mejai ดังนี้

```
{
  "data": [
    {
      "id": 2,
      "category": 1,
      "contribution": 250000,
      "name": "1 Help 1 Life : น้ำสะอาดให้น้องดื่ม",
      "thumbnail": "path",
      "description": "เพราะโรงเรียนไม่มีเครื่องกรองน้ำ เราจึงอยากร่วมมอบเครื่องกรองน้ำเพื่อผลิตน้ำดื่มสะอาดให้เด็กนักเรียนโรงเรียนวัดคลองห้า จังหวัดสระบุรี"
    },
    {
      "id": 4,
      "category": 3,
      "contribution": 250000,
      "name": "1 Help 1 Life : น้ำสะอาดให้น้องดื่ม",
      "thumbnail": "path",
      "description": "เพราะโรงเรียนไม่มีเครื่องกรองน้ำ เราจึงอยากร่วมมอบเครื่องกรองน้ำเพื่อผลิตน้ำดื่มสะอาดให้เด็กนักเรียนโรงเรียนวัดคลองห้า จังหวัดสระบุรี"
    }
  ]
}
```


โปรแกรมที่ 5 ข้อมูลที่ได้จาก RESTful

ภาคผนวก จ

การสร้างและใช้งาน Kubernetes

Kubernetes นั้น คือเครื่องมือที่นำมาช่วยในการ Deploy และดูแลเรื่อง Scale container ใน Mejai โดย Kubernetes จะทำหน้าที่ดูแลความเสถียรของระบบให้ตลอดเวลา ยกตัวอย่างโค้ดสำหรับการตั้งค่าให้ Kubernetes โดยทำงานผ่าน Helm อีกชั้น ดังนี้

```
nameOverride: mejai
serviceNameSuffix: service
replicaCount: 5
image:
 repository: mrnonz/mejai
 tag: latest
 pullPolicy: Always
containerPorts: 80
environments:
 - name: MYSQL_ROOT_PASSWORD
   secret:
     name: mejai
     key: MYSQL_ROOT_PASSWORD
 - name: MYSQL_PASSWORD
   secret:
     name: mejai
     key: MYSQL_PASSWORD
 - name: MYSQL_DATABASE
   value: mejai
 - name: MYSQL_HOST
   value: 10.148.0.5
 - name: MYSQL_USER
   value: root
```

โปรแกรมที่ 6 การตั้งค่า Kubernetes ผ่าน Helm

```
service:
 type: LoadBalancer
 ports:
  - name: mejai-http
    port: 80
    targetPort: 80
    protocol: TCP
livenessProbe:
initialDelaySeconds: 180
 periodSeconds: 15
 timeoutSeconds: 5
 failureThreshold: 6
 successThreshold: 1
readinessProbe:
 initialDelaySeconds: 30
 periodSeconds: 10
 timeoutSeconds: 5
 failureThreshold: 6
 successThreshold: 1
```

โปรแกรมที่ 6 การตั้งค่า Kubernetes ผ่าน Helm (ต่อ)

จากโค้ดข้างต้นเป็นการกำหนดให้ Mejai Service นั้นเปิดพอร์ต 80 สำหรับการเข้าใช้งานไว้ และกำหนดให้คงไว้ซึ่งจำนวน Pod ณ เวลาใดๆ เท่ากับ 5 Pods รวมไปถึงการกำหนดตัวแปร Environment ต่างๆ ให้สำหรับ Container ใน Pod ด้วย

บรรณานุกรม

Facebook. 2560. A JavaScript library for building user interfaces. [Online].
Available : https://reactjs.org/

Chai Phonbopit. 2558. React คืออะไร ? + เริ่มต้นเขียน React.[Online].
Available : https://devahoy.com/posts/getting-started-with-reactjs/

Arunoda Susiripala. Learning Next.js.[Online].
Available : https://learnnextjs.com/

Alex Grigoryan. 2560. The Benefits of Server Side Rendering Over Client Side Rendering.[Online].Available : https://medium.com/walmartlabs/the-benefits-of-server-side-rendering-over-client-side-rendering-5d07ff2cefe8

Juan Vega. 2560. Client-side vs. server-side rendering: why it's not all black and white.[Online].
Available : https://medium.freecodecamp.org/what-exactly-is-client-side-rendering-and-hows-it-different-from-server-side-rendering-bd5c786b340d

Alex. 2559. The 3 Principles of Redux.[Online].Available : https://medium.com/@ee7klt/the-3-principles-of-redux-c5367ea40fdc

Nuttavut Thongjor. 2559. จัดการ data flow ใน React.js อย่างมีประสิทธิภาพด้วย Redux.[Online].
Available : https://www.babelcoder.com/blog/posts/react-redux-isomorphic-day3-introduction-to-redux

Brad Frost. 2556. Atomic Design.[Online].
Available : http://bradfrost.com/blog/post/atomic-web-design/

Tom Christie. 2560. Django REST framework. [Online].
Available : http://www.django-rest-framework.org

Django Software Foundation. 2560. The web framework for perfectionists with deadlines. [Online]. Available : https://www.djangoproject.com

ProgrammingHelp.com. 2556. Fundamentals of an MVC Framework. [Online]. Available : http://www.programminghelp.com/mvc/fundamentals-mvc-framework/

RESTful. 2557 RESTful หรือ REST คือ. [Online] Available: https://saixiii.com/what-is-restful/

RESTful. 2557. An Introduction to RESTful APIs. [Online] Available: https://dzone.com/articles/an-introduction-to-restful-apis

CHARLTON MEDIA GROUP. 2560. Banks' profits to take a hit as Thailand goes cashless. [Online].Available : http://asianbankingandfinance.net/cards-payments/news/banks'-profits-take-hit-thailand-goes-cashless

Shashank Sahni. 2559. ThousandEyes Embraces Docker to Deploy Enterprise & Cloud Agents. [Online].Available : https://blog.thousandeyes.com/thousandeyes-docker-enterprise-agents/

The Kubernetes Authors. 2560. Production-Grade Container Orchestration. [Online]. Available: https://kubernetes.io

Rimantas Mocevicius. 2559. Kubernetes Overview, Part One. [Online]. Available: https://deis.com/blog/2016/kubernetes-overview-pt-1/

Jean-Mathieu Saponaro. 2559. Monitoring in the Kubernetes era. [Online]. Available: https://www.datadoghq.com/blog/monitoring-kubernetes-era/

Alexei Ledenev. 2560. Helm - Application deployment management for Kubernetes. [Online]. Available: https://www.slideshare.net/alexLM/helm-application-deployment-management-for-kubernetes

Nontawat Numor. 2560. Generate PromptPay QR Code. [Online].

Available: https://wi.th/promptpay