รายงานสหกิจศึกษาฉบับสมบูรณ์

การจัดการเวิร์ดเพสโดยอัตโนมัติผ่านเครื่องมือไอฟินิกซ์
WordPress Automation Management via iPhoenix Tool

นายมณฑล แสงสินทรัพย์

ภาควิชาวิศวกรรมคอมพิวเตอร์ สาขาวิชาวิศวกรรมสารสนเทศ
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2560

รายงานสหกิจศึกษาฉบับสมบูรณ์

การจัดการเวิร์ดเพสโดยอัตโนมัติผ่านเครื่องมือไอฟินิกซ์

WordPress Automation Management via iPhoenix Tool

นายมณฑล แสงสินทรัพย์

ภาควิชาวิศวกรรมคอมพิวเตอร์ สาขาวิชาวิศวกรรมสารสนเทศ

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2560

ชื่อโครงงานสหกิจศึกษา การจัดการเวิร์ดเพสโดยอัตโนมัติผ่านเครื่องมือไอฟินิกซ์

ชื่อ-สกุล นักศึกษา นายมณฑล แสงสินทรัพย์

คณะ วิศวกรรมศาสตร์ **ภาควิชา** วิศวกรรมคอมพิวเตอร์

สาขาวิชา วิศวกรรมสารสนเทศ

ชื่อ-สกุล อาจารย์นิเทศ ผศ.มยุรี เลิศเวชกุล

ชื่อ-สกุล ผู้นิเทศ นายภวินท์ คงคาสวรรค์

ชื่อสถาณประกอบการ บริษัท พรอนโต้ กรุ๊ป จำกัด

บทคัดย่อ

โครงงานฉบับนี้นำเสนอการจัดการเวิร์ดเพส (WordPress) ภายในองค์กรโดยการใช้เครื่องมือไอฟินิกซ์ (iPhoenix) เพื่อจัดการการสร้างเว็บไซต์แบบมัลติไซต์, การติดตั้งอุปกรณ์เสริม (Plugins), การติดตั้งธีม (Themes), การอัพเกรดหรือดาวน์เกรดเวิร์ดเพสคอร์ (WordPress Core), การทำโดเมนแมปปิ้ง (Domain mapping), การทำใบรับรองความปลอดภัย (SSL Certificate) และการทำรีไดเร็กชั่น (Redirection) ซึ่งจะทำทุกอย่างที่กล่าวมาข้างต้นอย่างอัตโนมัติและอย่างสะดวกผ่านอินเตอร์เฟส (Interface) ที่ใช้งานง่าย อีกทั้งผู้ใช้งานไม่จำเป็นต้องมีความรู้ทางเทคนิคก็สามารถจัดการเวิร์ดเพสได้จำนวนมากภายในเวลาไม่นาน

คำสำคัญ: เครื่องมือไอฟินิกซ์, การสร้างเว็บไซต์, การติดตั้งอุปกรณ์เสริม, การติดตั้งธีม, การอัพเกรดหรือดาวน์เกรดเวิร์ดเพสคอร์, การทำโดเมนแมปปิ้ง, การทำใบรับรองความปลอดภัย, การทำรีไดเร็กชั่น

Cooperative Title:	WordPress automation management by iPhoenix tool
Student intern name:	Monthon Sangsinsup
Faculty: Bachelor of Engineering	**Department:** Computer Engineering
	Program: Information Engineering
Advisor name:	Asst.Prof. Mayuree Lertwatechakul
Mentor name:	Phawin Khongkhasawan
Company:	Pronto Group Ltd.

ABSTRACT

This project presents WordPress management inside the organization by an iPhoenix tool to manage the multisite (Network) creation, WordPress plugins, WordPress themes, WordPress core, Domain mapping, SSL certificate and Redirection. All of these works automatically and easier by using via an interface. Even though users do not have technical knowledge, they can manage a lot of WordPress in a very short time.

Keyword: iPhoenix, Multisite creation, WordPress plugins, WordPress themes, WordPress core, Domain mapping, SSL certificate, Redirection

กิตติกรรมประกาศ

การจัดทำโครงงานนี้เป็นโครงงานของบริษัท พรอนโต กรุ๊ป จำกัด โดยทีม Pronto Tools เป็นผู้พัฒนาและนำเทคโนโลยีใหม่ๆ เข้ามาใช้ ทำให้ช่วยเพิ่มประสิทธิภาพการทำงานภายในองค์กรให้ดียิ่งขึ้น โดยผลงานดังกล่าวสำเร็จลุล่วงไปได้ด้วยดี เนื่องจากได้รับคำแนะนำ คำปรึกษาและการติดตามความคืบหน้า ผู้จัดทำจึงขอขอบคุณทุกท่านที่ได้ให้ความช่วยเหลือตลอดระยะเวลาที่ได้มีโอกาสเข้าไปดำเนินงานโครงการสหกิจศึกษา และเรียนรู้ประสบการณ์การทำงานต่างๆ ตลอดจนสิ้นสุดโครงการ ตั้งแต่วันที่ 1 มิถุนายน 2560 จนถึง 28 ธันวาคม 2560

โครงงานนี้ไม่อาจสำเร็จลุล่วงได้ หากขาดความกรุณาของพนักงานในทีม Pronto Tools และพนักงานที่ปรึกษา นายภวินท์ คงคาสวรรค์ ที่คอยให้แนวคิด คำแนะนำ คำปรึกษาต่างๆ ตลอดจนการดูแลเอาใจใส่ ช่วยเหลือและยังมอบประสบการณ์อันมีค่าตลอดระยะเวลาที่ผ่านมา

ขอขอบพระคุณผศ.มยุรี เลิศเวชกุล อาจารย์นิเทศโครงการสหกิจศึกษา ที่คอยให้คำแนะนำและชี้แนะแนวทางในการทำโครงงาน ตลอดจนการติดตามความคืบหน้าของผลงาน

สุดท้ายนี้ขอขอบคุณครอบครัว และเพื่อนนักศึกษา ที่คอยช่วยเหลือ เป็นกำลังใจ และเป็นแรงผลักดันในการผ่านอุปสรรคต่างๆ รวมถึงผู้มีพระคุณทุกท่านที่มิได้กล่าวนามไว้ ณ ที่นี้เป็นส่วนหนึ่งของความสำเร็จทั้งหมด จึงขอขอบคุณไว้ ณ โอกาสนี้

มณฑล แสงสินทรัพย์

สารบัญ

หน้า

สารบัญตาราง

สารบัญรูปภาพ

บทที่ 1

บทนำ

1.1 ข้อมูลสถานประกอบการที่เข้าร่วมปฏิบัติงานสหกิจศึกษา

พรอนโต้ กรุ๊ป จำกัด เป็นบริษัทที่รับทำ Internet Presence Management หรือดูแลตัวตนของบริษัทต่างๆ บน อินเตอร์เน็ท ตั้งแต่ออกแบบ สร้าง และดูแลเว็บไซต์ครบวงจร

1.1.1 Pronto marketing (ปี พ.ศ.2551)

คุณ Derek Brown เคยทำงานที่ Microsoft 13ปี ในหลายตำแหน่ง คุณ Derek เคยเป็น Director of product management for small business server. เมื่อคุณ Derek ได้ก่อตั้งบริษัท Pronto Marketing กับ Cory ซึ่งเป็นลูกชาย ได้เล็งเห็น ปัญหาว่าบริษัท IT ขนาดเล็กมีปัญหาเรื่องการทำ Online Marketing เนื่องจากมีข้อจำกัดในเรื่อง เวลาและความรู้ในด้านนี้ Pronto Marketing จึงมาช่วยรับผิดชอบในการทำกิจกรรมต่างๆ ที่เกี่ยวข้องกับ Business marketing เช่น Posting blog, Sending out email Newsletter, Optimizing SEO, Adding new landing page or form ทั้งหมดนี้เป็นการบริการที่มีค่าใช้จ่ายรายเดือนแบบสมเหตุสมผล

1.1.2 Pronto Tools (ปี พ.ศ.2555)

จากทีม Research & Development (R&D) ของบริษัท Pronto Marketing ที่มี Project ต่างๆที่กำลังจะขยายและเพิ่มมากขึ้น จึงได้สร้างบริษัทใหม่ที่มีเป้าหมายในการพัฒนา Software เป็น หลักในชื่อ “Pronto Tools” ซึ่งมุ่งเน้นการพัฒนา Software ใหม่ๆ สำหรับการ support ภายใน บริษัท และสำหรับขายลูกค้าภายนอก

1.1.3 ปัจจุบัน

ปัจจุบัน Pronto Marketing ได้ให้บริการลูกค้ากว่า 1,500 เว็บไซต์ ใน 13 ประเทศทั่วโลก ส่วน Pronto Tools ยังคงมีส่วนเป็น R&D ของ Pronto Marketing เช่นเดิม

1.2 ความเป็นมาและความสำคัญ

การเข้าร่วมโครงการสหกิจศึกษากับทางบริษัท พรอนโต กรุ๊ป จำกัด ในทีมพรอนโตทูลส์ (Pronto Tools) และได้รับมอบหมายให้เข้าร่วมพัฒนาโครงงานในส่วนของไอฟินิกซ์ (iPhoenix) ซึ่งเป็นเครื่องมือที่ช่วยจัดการเวิร์ดเพส (WordPress) ที่ใช้จริงภายในบริษัท ก่อนจะริเริ่มพัฒนาไอฟินิกซ์ขึ้นนั้น การจัดการเวิร์ดเพสซึ่งก็คือการดูแลเว็บไซต์ของลูกค้าจะทำโดยตรงผ่านการใช้เวิร์ดเพสซึ่งแต่ละแผนกที่รับผิดชอบในส่วนต่างๆ ก็ต้องมีความรู้ของเวิร์ดเพสในส่วนนั้นๆ หลังจากบริษัทเริ่มขยายตัวและลูกค้าก็เพิ่มมากขึ้น ทำให้การจัดการเว็บไซต์ของลูกค้านั้นใช้เวลานานมากขึ้น เช่นการสร้างเว็บไซต์ให้ลูกค้าจำเป็นต้องให้ทีมอินฟราสตักเจอ (Infrastructure) ช่วยในเรื่องเซ็ตอัพเซอร์เวอร์และรายละเอียดหยิบย่อยที่พนักงานทั่วไปไม่สามารถทำได้ กว่าจะสร้างได้หนึ่งไซต์ใช้เวลาถึงวัน หรือการติดตั้งอุปกรณ์เสริม (Plugin) ตัวเดียวกันให้แก่เว็บไซต์ของลูกค้าจำนวนมากๆ นั้นจะต้องติดตั้งทีละเว็บไซต์จนครบความต้องการ การแก้ปัญหาเบื้องต้นคือการเพิ่มจำนวนพนักงานมากขึ้นตามความต้องการของตัวงาน แต่ลูกค้ามีแนวโน้มที่มากขึ้น ทางบริษัทจะมีแต่เสียมูลค่าไปกับการจ้างพนักงานมากขึ้น อีกทั้งการทำงานเดิมซ้ำๆ หลายๆครั้ง อาจเกิดข้อผิดพลาดระหว่างทางได้ และพนักงานใหม่ยังต้องใช้เวลาเรียนรู้ และประสบการณ์ที่เพียงพอที่จะทำให้งานนั้นมีประสิทธิภาพ

จากปัญหาข้างต้น ทีมพรอนโตทูลส์จึงคิดโครงการไอฟินิกซ์ขึ้นเป็นเครื่องมือที่ช่วยอำนวยความสะดวกแก่การจัดการเว็บไซต์ของลูกค้า ให้พนักงานในบริษัทได้ใช้งานเพื่อเพิ่มประสิทธิภาพ สะดวกและรวดเร็ว โดยไอฟินิกซ์นั้นสามารถจัดการเว็บไซต์จำนวนมากได้ภายในเวลาไม่นาน เช่นการสร้างเว็บไซต์ทำได้เพียงกดปุ่มเดียวผ่านไอฟินิกซ์ ซึ่งไม่ต้องพึ่งพาทีมอินฟราสตรักเจอ การทำรีไดเร็กชั่นซึ่งทีมเอสอีโอ (SEO) ต้องจัดการรูล (Rule) เพื่อทำให้เว็บไซต์ยังคงอันดับบนโลกอินเตอร์เน็ตโดยใช้ไอฟินิกซ์นำเข้ารูลที่ลูกค้าให้มาจำนวนมากๆ ใส่เข้าไปในเว็บไซต์ของลูกค้านั้นๆ ซึ่งไม่จำเป็นต้องเพิ่มที่ละรูลเหมือนแต่ก่อน ทั้งหมดนี้เป็นการลดเวลาอย่างมาก ลดความผิดพลาดในการทำซ้ำๆ อีกทั้งใช้งานง่ายผู้ใช้ไม่ต้องเรียนรู้มากนัก ซึ่งเพิ่มประสิทธิภาพให้แก่พนักงานในบริษัท

เพื่อเพิ่มประสิทธภาพให้แก่ระบบไอฟินิกซ์ได้มีการนำเทคโนโลยีแอร์โฟล (Airflow) มาช่วยในการมอนิเตอร์โดยนักพัฒนาสามารถดูได้ว่าฟังก์ชั่นต่างๆของไอฟินิกซ์สถานะเป็นอย่างไร ทำได้ราบลื่นหรือไม่ หรือเกิดข้อผิดพลาดก็มีการแจ้งเตือนเพื่อตรวจสอบข้อผิดพลาดได้อย่างง่ายได้ เพื่อการแก้ปัญหาที่รวดเร็ว ลดการกระทบการทำงานของพนักงานในบริษัทระหว่างแก้ไขปัญหานั้นๆ ได้เป็นอย่างดี

1.3 วัตถุประสงค์ของโครงงาน

1.3.1 เพื่อจัดการเว็บไซต์จำนวนมากในเวลาอันสั้น

1.3.2 เพื่อความสะดวกสบายแก่พนักงานที่เกี่ยวข้อง

1.3.3 เพื่อลดภาระของพนักงานและใช้เวลาอย่างคุ้มค่า

1.3.4 เพื่อลดงานที่ซ้ำซ้อนและจัดการอย่างแม่นยำ

1.3.5 เพื่อลดค่าใช้จ่ายในบริษัท แก้ปัญหาการลงทุนกับทรัพยากรที่ไม่จำเป็น

1.4 ขอบเขตของโครงงาน

ขอบเขตงานตลอดระยะเวลาที่เข้าร่วมโครงการสหกิจศึกษาที่ได้รับมอบหมายให้รับผิดชอบจากบริษัทแบ่งออกเป็น 2 ส่วนดังนี้

1.4.1 iPhoenix server คือส่วนจัดการเวิร์ดเพสหลักประกอบไปด้วย 2 ส่วนคือ

WordPress Multisite management (Network management) ระบบจัดการเวิร์ดเพสแบบมัลติไซต์ ประกอบไปด้วยฟังก์ชั่นย่อยดังนี้

- Network creation ฟังก์ชันสร้างเวิร์ดเพสแบบมัลติไซต์
- WP Core management ฟังก์ชันจัดการเวิร์ดเพสคอร์
- Themes management ฟังก์ชันจัดการธีม
- Plugins management ฟังก์ชันจัดการปลั๊กอิน

Domain management ระบบจัดการโดเมนซึ่งประกอบด้วยฟังก์ชันย่อยดังนี้

- Domain mapping ฟังก์ชันแมปปิ้งโดเมน
- SSL Certificate ฟังก์ชันจัดการใบรับรองความปลอดภัย
- Redirection ฟังก์ชันจัดการรีไดเร็กชั่น

1.4.2 iPhoenix worker คือส่วนจัดการเข้าคิว (Queue) ตารางเวลา (Schedule) สังเกตการณ์ (Monitoring) และส่งคำสั่งต่างๆของ iPhoenix server ไปยังระบบที่จัดการอยู่เบื้องหลังต่อไปโดยใช้เทคโนโลยี Airflow เข้ามาช่วยจัดการดังกล่าว

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1.5.1 ประโยชน์ต่อบริษัท

- ประหยัดค่าใช้จ่ายเนื่องจากลดการว่าจ้างพนักงานเพิ่ม
- ประหยัดเวลาในการจัดการเว็บไซต์ของลูกค้าเนื่องจากใช้เครื่องมือเข้าช่วย
- จัดการเว็บไซต์ของลูกค้าได้มีประสิทธิภาพมากขึ้น ลดความผิดพลาดจากการทำงานซ้ำๆ

1.5.2 ประโยชน์ต่อพนักงานที่เกี่ยวข้อง

- จัดการเว็บไซต์ของลูกค้าได้ในปริมาณมากขึ้น
- พนักงานสามารถบริหารงานได้มากขึ้น
- สะดวกต่อการทำงานเนื่องจากเครื่องมือที่ใช้มีอินเตอร์เฟสที่ใช้งานง่าย

1.5.3 ประโยชน์ต่อผู้วิจัย

- เรียนรู้วัฒนธรรม Agile ของบริษัทและได้ใช้ Scrum ในการดำเนินการโครงงาน
- เรียนรู้ภาษา Python และ Django framework
- เรียนรู้คำสั่ง Unix และระบบ infrastructure
- เรียนรู้ WordPress
- สามารถนำความรู้ที่ได้รับ ไปประยุกต์ใช้ และเตรียมความพร้อมด้านต่างๆ ในอนาคต

1.6 ขั้นตอนการดำเนินการ

ในการดำเนินการโครงงานนี้ ได้นำวัฒนธรรม Agile และกรอบการทำงานแบบ Scrum มาใช้ในการดำเนินการ โดยลักษณะการพัฒนาโครงงานจะดำเนินไปทีละ 2 สัปดาห์หรือเรียกว่า 1 Sprint

แต่ละ Sprint จะประกอบไปด้วยการดำเนินการดังนี้

Planning เป็นการวางแผนถึงสิ่งที่จะทำภายใน 1 Sprint จะมีสิ่งที่ต้องดำเนินการอะไรบ้าง และอย่างไรบ้าง มีการประเมินความระดับความยากของงานร่วมกันในทีมเพื่อให้เสร็จลุล่วงภายใน Sprint นี้

Developing เป็นช่วงในการดำเนินการพัฒนาโครงงานตลอดใน Sprint

Refinement ในช่วงกลาง Sprint หรือวันแรกของสัปดาห์ที่สองจะเป็นการมาคุยถึงความคลีบหน้าของงานใน Sprint และประเมินงานว่าดำเนินการไปได้ด้วยดีหรือไม่ และพูดคุยเตรียมการถึง Sprint หน้าอย่างคร่าวๆ เพื่อลดเวลาในการ Planning ครั้งต่อไป

Sprint Review วันสุดท้ายของ Sprint จะรวมตัวเพื่อคุยสรุปการทำงานใน Sprint ทั้งหมดเพื่อปรับปรุงแก้ไขและ พัฒนาตัวงานในครั้งต่อๆ ไป

Retrospective หลังจาก Sprint Review ในวันเดียวกันจะมีการพูดคุยกันที่เน้นไปทางความรู้สึกนึกคิดภายใน Sprint เพื่อแรกเปลี่ยนความคิด และปรับปรุงแก้ไข และพัฒนาตัวเองให้ดียิ่งขึ้น

ขั้นตอนดังกล่าวจะเปลี่ยนวนไปเรื่อยๆ เป็น Sprint จนจบโครงการ ประกอบด้วยรายละเอียดและระยะเวลาการดำเนินการดังนี้

ตารางที่ 1 ตารางแสดงขั้นตอนการดำเนินการ

กิจกรรม	ระยะเวลา (เดือน)				
	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.
1. Sprint Planning (วางแผนแต่ละ Sprint)					
2. Developing (พัฒนาฟังก์ชั่นต่างๆ)					
3. Refinement (ตรวจความคลีบหน้า)	◄	—	—	—	►
4. Sprint Review (สรุปงานในแต่ละ Sprint)					
5. Retrospective					
6. ศึกษาเพิ่มเติม (Python, Django, Unix, Airflow, WordPress, Unit Test, etc.)	◄	—	—	►	
7. พัฒนาฟังก์ชัน Domain mapping, SSL Certificate, Redirection	◄	►			
8. พัฒนาระบบ Network management		◄	—	►	
9. พัฒนาระบบ WP Core management			◄	—	►
10. พัฒนาระบบ Theme and Plugin management				◄	►
11. แก้ไขข้อผิดพลาดและปรับปรุงระบบ	◄	—	—	—	►
12. จัดทำเอกสารประกอบโครงงาน			◄	—	►

1.7 อุปกรณ์หรือเทคโนโลยีที่ใช้ในการพัฒนา

1.7.1 ฮาร์ดแวร์ (Hardware)

- โน๊ตบุ๊ค (MacBook Air)
- อินเทอร์เน็ต
- เซิร์ฟเวอร์ (Server)

1.7.2 ซอฟท์แวร์ (Software)

- Django Framework
- Apache Airflow
- Docker
- Nginx
- Unit Testing Framework

1.7.3 ภาษาที่ใช้พัฒนา

- Python
- HTML/CSS/Javascript
- Unix Command
- YAML

บทที่ 2

แนวคิด ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

โครงงานชิ้นนี้เป็นโครงงานที่เกี่ยวข้องกับการทำเครื่องมือเพื่ออำนวยความสะดวกในการจัดการเว็บไซต์ ซึ่งระบบหลังบ้านนั้นต้องใช้ความรู้ทางทฤษฎีที่เกี่ยวข้องหลายอย่าง ไม่ว่าจะเป็นทฤษฎีที่เกี่ยวข้องด้านภาษาคอมพิวเตอร์และทฤษฎีที่เกี่ยวข้องทางเทคโนโลยีที่ใช้

2.1 ทฤษฎีที่เกี่ยวข้องทางด้านภาษาคอมพิวเตอร์

2.1.1 ภาษาไพทอน (Python)

ไพทอน เป็นภาษาระดับสูงแบบอินเตอร์พรีเตอร์ (interpreter) คือจะประมวลผลไปทีละบรรทัดและปฏิบัติตามคำสั่งที่ได้รับ ถูกพัฒนาโดย Guido van Rossum ในปี ค.ศ.1989 ซึ่งเป็นภาษาที่ถูกพัฒนาโดยไม่ยึดติดกับแพลตฟอร์ม กล่าวคือสามารถรันภาษา Python ได้ทั้งบนระบบ Unix, Linux , Windows NT, Windows 2000, Windows XP หรือแม้แต่ระบบ FreeBSD

โค้ดของไพทอนถูกสร้างขึ้นมาจากภาษาซี ออกเวอร์ชั่น 0.9.0 เป็นเวอร์ชั่นแรกในปี 1990 และเวอร์ชั่นล่าสุด 3.6 ในปลายปี 2016

ภาษาไพทอนนั้นมีจุดเด่นดังนี้

- เป็นภาษาแบบอินเตอร์พรีเตอร์ (interpreter)
- สนับสนุนแนวคิดออปเจกต์โอเรียนเทดหรือ OOP (Object Oriented Programming)
- โค้ดที่เขียนด้วย Python สามารถนำไปรันบนระบบปฏิบัติการได้หลากหลาย
- เป็น Open Source สามารถนำไปใช้งานได้ทั้งทางการค้าและภาคการศึกษา
- เป็นภาษาที่เรียนรู้ได้ง่ายและเขียนได้อย่างรวดเร็ว

- สามารถใช้พัฒนาเว็บและโปรแกรมได้
- มีฟังก์ชันสนับสนุนฐานข้อมูล เช่น MySQL, Sybase, Oracle , Informix, ODBC และอื่นๆ
- เป็น Dynamic typing คือ สามารถเปลี่ยนชนิดข้อมูลได้ง่ายและสะดวก
- มี Build-in Object Types คือ โครงสร้างของข้อมูลที่สามารถใช้ได้ใน Python ประกอบด้วย ลิสต์, ดิกชันนารี, สตริง ที่ง่ายต่อการใช้งานและมีประสิทธิภาพสูง
- มีเครื่องมือต่างๆ มากมาย เช่น การประมวลผลเท็กซ์ไฟล์ การเรียงข้อมูล การเชื่อต่อสตริง การตรวจสอบเงื่อนไขของข้อความ การแทนคำ เป็นต้น
- จัดการหน่วยความจำอย่างอัตโนมัติ สามารถจักการพื้นที่หน่วยความจำที่ไม่ต่อเนื่องให้ทำงานได้อย่างมีประสิทธิภาพ
- อนุญาตให้ฝังชุดคำสั่งของ Python เอาไว้ภายในโค๊ดภาษา C/C++ ได้
- มีมอดูลสนับสนุนเกี่ยวกับเน็ตเวิร์ก Regular Expression, xml, GUI
- สามารถประมาลผลทางด้านวิทยาศาสตร์ และวิศวกรรมศาสตร์ได้อย่างมีประสิทธิภาพ
- มีไลบรารีรองรับจำนวนมากมาย
- มีการพัฒนาอย่างต่อเนื่อง

<u>ไพทอนพื้นฐาน</u>

1. ชนิดของข้อมูลไพทอน

ชนิดข้อมูลที่ถูกใช้งานบ่อยมีดังนี้

- ข้อมูลชนิดตัวเลข หมายถึง ข้อมูลที่เป็นชุดของตัวเลข ซึ่งประกอบด้วย ตัวเลขจำนวนเต็ม ตัวเลขที่มีจุดทศนิยม รวมทั้งตัวเลขที่มีค่าเป็นบวกหรือเป็นลบด้วย
  ชนิดข้อมูลแบบตัวเลขก็จะมีให้เลือกใช้ 4 ชนิดด้วยกัน ดังนี้
  - Integer
  - Long Integer
  - Float
  - Complex Number
- ชนิดของข้อมูลแบบอักขระ หมายถึง ชนิดข้อมูลที่เป็นอักขระเพียงหนึ่งตัวเท่านั้น โดยตัวอักขระนี้จะอยู่ในเครื่องหมาย apostrophes (' ') อาจจะเป็นตัวอักษรภาษาอังกฤษตัว

เล็ก หรือตัวใหญ่ สัญลักษณ์พิเศษต่างๆ หรือตัวเลข โดยที่ตัวเลขเมื่ออยู่ในเครื่องหมาย apostrophes (' ') นั้น จะไม่สามารถนำไปคำนวณเหมือนกับข้อมูลชนิดตัวเลขได้ เช่น 'A', 'B', 'a', 'b', '9', '#'

- ชนิดข้อมูลแบบข้อความ หมายถึง ชนิดข้อมูลที่เป็นอักขระตั้งแต่หนึ่งตัวอักขระขึ้นไป ซึ่งจะเรียงต่อกัน โดยจะอยู่ภายใต้เครื่องหมาย Apostrophes (' ') แต่จะมีอักขระเรียงกันยาวไม่เกิน 255 อักขระ เช่น 'Hello', '0123456789'
- ชนิดของข้อมูลแบบตรรกศาสตร์ หมายถึง ชนิดของข้อมูลที่ให้ผลลัพธ์ที่ได้จากการตัดสินใจจากเงื่อนไขหรือนิพจน์ โดยผลของข้อมูลชนิดนี้จะมีเพียงค่าจริงและค่าเท็จเท่านั้น

2. ตัวดำเนินการ (Operator) สำหรับเปรียบเทียบความสัมพันธ์ในไพทอน

ตัวดำเนินการ (Operator) คือกลุ่มที่เครื่องหมายหรือสัญลักษณ์จะเป็นตัวทำหน้าที่รวมค่าเปรียบเทียบค่า โดยจะกระทำกับตัวแปรต่างๆ สำหรับตัวดำเนินการที่ใช้ในการเปรียบเทียบความสัมพันธ์ในภาษาไพธอน นั้น มีดังนี้

ตารางที่ 2 ตารางแสดงตัวดำเนินการและความหมาย

สัญลักษณ์	ความหมาย
==	เท่ากับ
!=	ไม่เท่ากับ
>	มากกว่า
<	น้อยกว่า
>=	มากกว่าหรือเท่ากับ
<=	น้อยกว่าหรือเท่ากับ
Is	เป็นชนิดเดียวกันหรือไม่
is not	ไม่เป็นชนิดเดียวกัน
in	เป็นสมาชิกหรือไม่
not in	ไม่เป็นสมาชิก

สำหรับตัวดำเนินการที่ใช้เปรียบเทียบ (Relation Operator) จะนำข้อมูลสองค่ามาเปรียบเทียบกัน โดยข้อมูลทั้งสองค่าที่จะนำมาเปรียบเทียบกันนั้น จะต้องเป็นข้อมูลประเภทเดียวกัน โดยผลลัพธ์ที่ได้จะเป็นจริง หรือ เท็จ

3. การประกาศตัวแปรและคำสงวนของไพทอน

ตัวแปร (Variable) เป็นการกำหนดชนิดข้อมูลของตัวแปร สำหรับใช้ในการเก็บหน่วยความจำ ตัวแปรจะมีชื่อ (identifier) สำหรับใช้ในการอ้างอิงถึงข้อมูลของมัน

ในการเขียนโปรแกรมคอมพิวเตอร์นั้น ตัวแปรของคอมพิวเตอร์จะแตกต่างจากตัวแปรทางคณิตศาสตร์ เพราะค่าของตัวแปรทางคอมพิวเตอร์นั้น ไม่จำเป็นต้องประกอบไปด้วยสูตรหรือสมการที่สมบูรณ์เหมือนกับในทางคณิตศาสตร์ และตัวแปรในทางคอมพิวเตอร์นั้น ตัวแปรอาจจะมีการทำงานซ้ำๆ เช่น การกำหนดค่าในที่หนึ่ง สามารถนำค่านั้นๆไปใช้อีกที่หนึ่งในโปรแกรม และนอกจากนี้ยังสามารถกำหนดค่าใหม่ให้กับตัวแปรได้ตลอดเวลา

การประกาศตัวแปรของไพทอน แต่การประกาศตัวแปรของภาษาไพทอน นั้น ก็มีเงื่อนไขที่ต้องคำนึงตามกฎการตั้งชื่อตัวแปร ดังนี้

1) ต้องขึ้นต้นด้วยตัวอักษร ห้ามใช้ตัวเลขหรือสัญลักษณ์ใดๆ
2) ห้ามมีช่องว่างหรือเว้นวรรค
3) ห้ามใช้เครื่องหมายต่อไปนี้ในการประกาศตัวแปร !...@...#...$...%...^...&...*...(...)...-...=...+...~
4) ห้ามตั้งชื่อตัวแปรซ้ำกับคำสงวน
5) ควรตั้งชื่อตัวแปรที่สื่อความหมายให้ชัดเจน เพื่อผู้อื่นตีความหมายได้เข้าใจ แต่ถ้ามีความหมายยาวมาก ให้ย่อ เช่น student_name ควรใช้ st_name เป็นต้น
6) ตัวแปรที่มีตัวพิมพ์ใหญ่และตัวพิมพ์เล็กผสมกันจะมีความหมายต่างกับตัวพิมพ์เล็กเพียงอย่างเดียว เช่น St_Id แตกต่างจากตัวแปร st_id เป็นต้น

คำสงวน คือชื่อหรือคำที่ภาษาไพทอนใช้เฉพาะเพื่อเป็นคำสั่ง หรือมีไว้เพื่อเขียนเป็นโครงสร้างของตัวภาษาเอง โดยคำสงวนของไพทอนจะมีทั้งหมด 31 คำ ดังนี้

ตารางที่ 3 ตารางแสดงคำสงวนของไพทอน

and	del	from	not	while	as	elif	for
if	else	global	or	with	assert	pass	lambda
yield	except	break	print	import	exec	class	try
raise	in	is	return	def	continue	finally	

4. การแสดงผลผ่านหน้าจอของไพทอน

การแสดงผลผ่านหน้าจอของไพทอนเป็นการให้โค้ดติดต่อกับผู้ใช้ และส่วนติดต่อกับผู้ใช้จะต้องมีทั้งการรับค่าและการแสดงค่า

โดยการแสดงผลทางหน้าจอนั้น จะใช้คำสั่ง *print*

```
print (5 + 6)
>>> 11
print ("Hello Kmitl")
```

ภาพที่ 1 แสดงตัวอย่างการใช้คำสั่ง print แสดงผลทางหน้าจอ

ตัวอย่างก่อนหน้าเป็นการแสดงค่าอย่างง่าย ยังมีตัวอย่างการแสดงค่าแบบอื่น เช่น การการแสดงผลข้อความที่มีเครื่องหมาย ' (quote) หรือ " (double quote) ร่วมด้วย การแสดงผลข้อมูลหลายบรรทัด และการแสดงผลแบบข้อความเชื่อมข้อความ

```
print ("I'm a student")
>>> I'm a student
print ('I said "I love Kmitl"')
>>> I said "I love Kmitl"
print ('for what it\'s worth')
```

ภาพที่ 2 แสดงตัวอย่างการแสดงผลกับข้อความที่มี quote หรือ double quote

```
lyric = """For what it's worth, I'm sorry for the hurt
I'll be the first to say, "I made my own mistakes"
For what it's worth, I know it's just a word and words betray
But sometimes we lose our way
For what it's worth"""

print lyric

>>> For what it's worth, I'm sorry for the hurt
```

ภาพที่ 3 แสดงตัวอย่างการแสดงผลกับข้อความหลายบรรทัด

```
a = "I"
b = "love"
c = "You"
```

ภาพที่ 4 แสดงตัวอย่างการแสดงผลข้อความแบบเชื่อมข้อความ

5. การรับค่าจากแป้นพิมพ์ในไพทอน

การรับค่าจากแป้นพิมพ์ในไพทอนนั้นสามารถรับค่าได้โดยใช้ฟังก์ชั่น *input()* โดยหากกำหนดให้รับค่า input ใส่เข้ากับตัวแปรหนึ่ง ตัวแปรนั้นจะมีชนิดเป็นสตริง

```
name = input("Enter your name : ")

>>> Enter your name : Monthon
```

ภาพที่ 5 แสดงตัวอย่างการรับค่าจากแป้นพิมพ์

6. ตัวแปรชนิดลิส (List) ในไพทอน

ตัวแปรลิสในไพทอนเป็นตัวแปรแบบ compound data type คือ 1 ตัวแปร สามารถเก็บค่าได้มากกว่าหนึ่งตัวเพื่อการใช้งานที่หลากหลาย การกำหนดค่าสมาชิกต่างๆ ของตัวแปรลิสจะต้องอยู่ในเครื่องหมาย *[...]*

```
list = ['abc', 1, 'Kmitl', 23.4]

list_2 = ['Boy', 'Girl']

print (list)

>>> ['abc', 1 , 'Kmitl', 23.4]

print (list[0])

>>> abc

print (list[1:3])

>>> [1 , 'Kmitl']

print (list[2:])

>>> [1 , 'Kmitl', 23.4]

print (list  2 * 2)
```

ภาพที่ 6 แสดงตัวอย่างการเก็บค่าใส่ตัวแปรชนิดลิส

7. ตัวแปรชนิดทูเปิล (Tuple) ในไพทอน

ทูเปิลเป็นตัวแปรที่เก็บข้อมูลแบบลำดับ (sequence data type) คล้ายๆกับลิสแต่ทูเปิลเมื่อสร้างขึ้นแล้วเราไม่สามารถเพิ่มค่าให้ตัวแปรได้อีก โดยทูเปิลจะเก็บค่าของสมาชิกแต่ละตัวไว้ใน *(...,...,...)*

```
tuples = ('abc', 1, 'Kmitl', 23.4)

print (tuples)

>>> ('abc', 1 ,  'Kmitl', 23.4)

print (tuples[0])

>>> abc
```

ภาพที่ 7 แสดงตัวอย่างการเก็บค่าใส่ตัวแปรชนิดทูเปิล

8. ตัวแปรชนิดดิกชันนารี (Dictionary) ในไพทอน

ดิกชันนารีในไพทอนเรียกได้ว่าเป็น hash table type โดยจะมี key และ value คู่กันไปเสมอ key สามารถ เอา data type อะไรของ ไพทอนมากำหนดก็ได้ แต่ควรใช้เพียงแค่ string และ numbers จะดีกว่าตัวแปรดิกชันนารี ถ้ากำหนดค่าให้อยู่ในเครื่องหมาย *{...}* (curly braces) และ key จะถูกกำหนดอยู่ในเครื่องหมาย *[ ]* (square braces)

```
dict = {'a': 'This is a', 1: 'This is one'}

print (dict)

>>> {1: 'This is one', 'a': 'This is a'}

print (dict['a'])

>>> This is a
```

ภาพที่ 8 แสดงตัวอย่างการเก็บค่าใส่ตัวแปรชนิดดิกชันนารี

9. เงื่อนไขในไพทอน

ในไพทอนหากต้องการเขียนเงื่อนไขเพื่อควบคุมการทำงานจะใช้คำสั่ง *if* เหมือนภาษาทั่วไป แต่มีรูปแบบที่ต่างออกไป

รูปแบบการเขียน if Statements

If Condition:

Statements

...

...

นอกจากนี้ยังมีรูปแบบอื่นๆ เช่น else, elif

elif Condition:

Statements

```
var = 1

if var == 1:

   print ('this is one')
```

ภาพที่ 9 แสดงตัวอย่างการเขียน if condition

```
var = 2

if var == 1:

   print ('this is one')

elif var == 2:

   print ('this is two')

else:

   print ('nothing')
```

ภาพที่ 10 แสดงตัวอย่างการเขียน if/elif/else condition

for loop ในไพทอนนั้นเป็นการทำงานซ้ำๆ จนกว่าจะเจอเงื่อนไขที่เป็นเท็จ โดยจะกำหนดช่วงการทำงาน *for* ใช้คู่กับ *in* เมื่อ in คือ ตรรกศาสตร์ที่ต้องการเปรียบเทียบ

รูปแบบของ for loop

for iterating_var in sequence:

statements(s)

```
list = [2, 'a', 'Kmitl']

for l in list:

    print (l)

ผลที่ได้คือ:

>>> 2
```

ภาพที่ 11 แสดงตัวอย่างการเขียน for loop

11. การเขียนฟังก์ชัน (function) ในไพทอน

การเขียนฟังก์ชั่นในไพทอนมีสองแบบคือ ฟังก์ชั่นที่นักพัฒนาเขียนขึ้น กับฟังก์ชั่นที่มีให้ใช้อยู่แล้ว โดยฟังก์ชั่นที่เขียนเองนั้นจะใช้ *def* เป็นไวยกรณ์และตามด้วยชื่อของฟังก์ชั่นที่ต้องการ

รูปแบบของการสร้างก์ชั่นเพื่อใช้งาน

def ชื่อฟังก์ชั่น():

...

```
def myfunc():

    print("Hello my function")



myfunc()
```

ภาพที่ 12 แสดงตัวอย่างการเขียนฟังก์ชั่น

2.1.1.คำสังยูนิกซ์ (Unix Command)

ระบบปฏิบัติการยูนิกซ์ (Unix Operating System)

เป็นระบบปฏิบัติการที่เคยพัฒนาในห้องแล็บเบล (Bell) เป็นเทคโนโลยีแบบเปิด (open system) ซึ่งเป็นแนวคิดที่ผู้ใช้ไม่ต้องผูกติดกับระบบใดระบบหนึ่งหรืออุปกรณ์ยี่ห้อเดียวกัน สร้างขึ้นเพื่อใช้กับเครื่องมินิคอมพิวเตอร์และเมนเฟรม ใช้ในการควบคุมการทำงานของศูนย์คอมพิวเตอร์ที่มีการเชื่อมลูกข่ายคอมพิวเตอร์ หรืออุปกรณ์ต่อพ่วงเป็นจำนวนมาก ดังนั้นยูนิกซ์จึงมักใช้ในระบบ

เครือข่ายคอมพิวเตอร์ที่มีขนาดใหญ่ และมีการเชื่อมต่อเครือข่ายระยะไกลต่อมาได้มีการพัฒนาให้สามารถนำยูนิกซ์มาใช้กับเครื่องคอมพิวเตอร์ได้

คำสั่งยูนิกซ์เบื้องต้น

ตารางที่ 4 แสดงคำสั่งยูนิกซ์เบื้องต้น

ลำดับ	คำสั่ง	ความหมาย
1.	ls	แสดงรายชื่อไฟล์และไดเร็คทอรี่
2.	cd	ไปยังไดเร็คทอรี่ที่ต้องการ
3.	cp	คัดลอกไฟล์หรือไดเร็คทอรี่
4.	mv	ย้ายหรือเปลี่ยนชื่อไฟล์หรือไดเร็คทอรี่
5.	rm	ลบไฟล์
6.	pwd	แสดงที่อยู่ (path) ปัจจุบัน
7.	touch	สร้างไฟล์ใหม่
8.	mkdir	สร้างไดเร็คทอรี่ใหม่
9.	rmdir	ลบไดเร็คทอรี่ที่ว่างเปล่า
10.	cat	ดูเนื้อหาของไฟล์
11.	wc	นับจำนวนบรรทัด คำ และขนาดไฟล์
12.	chown	เปลี่ยนเจ้าของไฟล์ (ownership)
13.	chmod	เปลี่ยนค่าของความมีสิทธิ์ในการเข้าถึงและใช้งานไฟล์
14.	grep	ค้นหาข้อความในระดับบรรทัด
15.	df	แสดงขนาดดิสก์ที่เหลืออยู่
16.	zip	บีบอัดไฟล์
17.	unzip	แตกไฟล์จากการ zip
18.	ps	แสดงการประมวลผลทั้งหมด
19.	kill	ยกเลิกการประมวลผลนั้นๆ
20.	history	ดูประวัติการใช้งานคำสั่งต่างๆ ที่ผ่านมา

2.1.2 ภาษา YAML

YAML หรือ YML เป็นภาษา Markup ชนิดหนึ่ง ซึ่งหน้าที่คล้าย JSON หรือ XML คือ แลกเปลี่ยนข้อมูลกันระหว่างโปรแกรมมิ่งภาษาต่างๆ แต่ต่างที่เป็นภาษาที่สั้นและเข้าใจง่ายกว่า โดยตัวภาษาเองนั้นจะไม่มี tag ใดๆ มาเกี่ยวข้อง ใช้การเว้นวรรคในการแยกข้อมูลแทน และไฟล์ที่เก็บจะมีนามสกุลไฟล์เป็น .yml

```
PageType:
    HomePage:
        Fields:
            BannerImage: {CMSField: Image, UploadFolder: homepage-images, Tab:
Banner}
            BannerCaption: {CMSField: Text, Tab: Banner}
        Components:
```

ภาพที่ 13 แสดงตัวอย่างการเขียน YAML

2.2 ทฤษฎีที่เกี่ยวข้องทางเทคโนโลยีที่ใช้

2.2.1 Django Framework

Django Framework เป็นกรอบการทำงานสำหรับการพัฒนาเว็บไซต์ด้วยไพทอน โดยมีเป้าหมายในการสร้างเว็บไซต์ที่ทำงานร่วมกับฐานข้อมูล (database) และมีความซับซ้อนให้ง่ายขึ้น รวมไปถึงการ render ข้อมูลออกมาให้ฝั่ง front-end แสดงผลข้อมูลเหล่านั้นได้

ในการสร้างโปรเจค Django ขึ้นมานั้นจะต้องติดตั้งดังนี้

- ติดตั้งไพทอน
- ติดตั้ง pip3 (package manager ที่ใช้สำหรับติดตั้งแพ็คเกจเสริมให้กับ python3)
- ติดตั้ง Django

ตัวอย่างการเริ่ม Django project บนระบบ Unix (Ubuntu)

1. ไปยัง folder ที่ต้องการจะเก็บโปรเจค จากนั้นเริ่มสร้างโปรเจคชื่อ “hello_kmitl” ด้วยคำสั่ง *$ django-admin startproject hello_kmitl*

ภาพที่ 14 แสดงโครงสร้างไฟล์เมื่อสร้างโปรเจค django

ภายในโฟเดอร์ที่เราสร้างโปรเจคขึ้น (hello_kmitl) จะมีไฟล์ต่างๆ ดังนี้

- manage.py คือไฟล์สคริปสำหรับรันคำสั่งต่างๆ ที่เกี่ยวข้องกับ django
- __init__.py คือไฟล์เปล่าไม่มีข้อมูลภายในไฟล์ แต่มีหน้าที่เพื่อให้ภาษาไพทอนรู้ว่าโฟลเดอร์ที่อยู่นี้เป็นโฟลเดอร์ที่ใช้เก็บ python package
- settings.py คือไฟล์ที่ใช้เก็บ configuration ทั้งหมดของ project เอาไว้ เช่น การตั้งค่า Database, Timezone, Logging เป็นต้น
- urls.py คือไฟล์ที่ใช้เก็บการ routing ของ HTTP request
- wsgi.py คือไฟล์ที่ใช้เก็บข้อมูลของ Django project ของเรา ใช้สำหรับการ deploy project เมื่อต้องการเชื่อมต่อกับ Web Server

2. ทดลองรัน Django project ที่ port 8000 ด้วยคำสั่ง *python3 manage.py runserver 0.0.0.0:8000* และทดลองเข้าเว็บ browser ด้วย URL: http://localhost:8000/

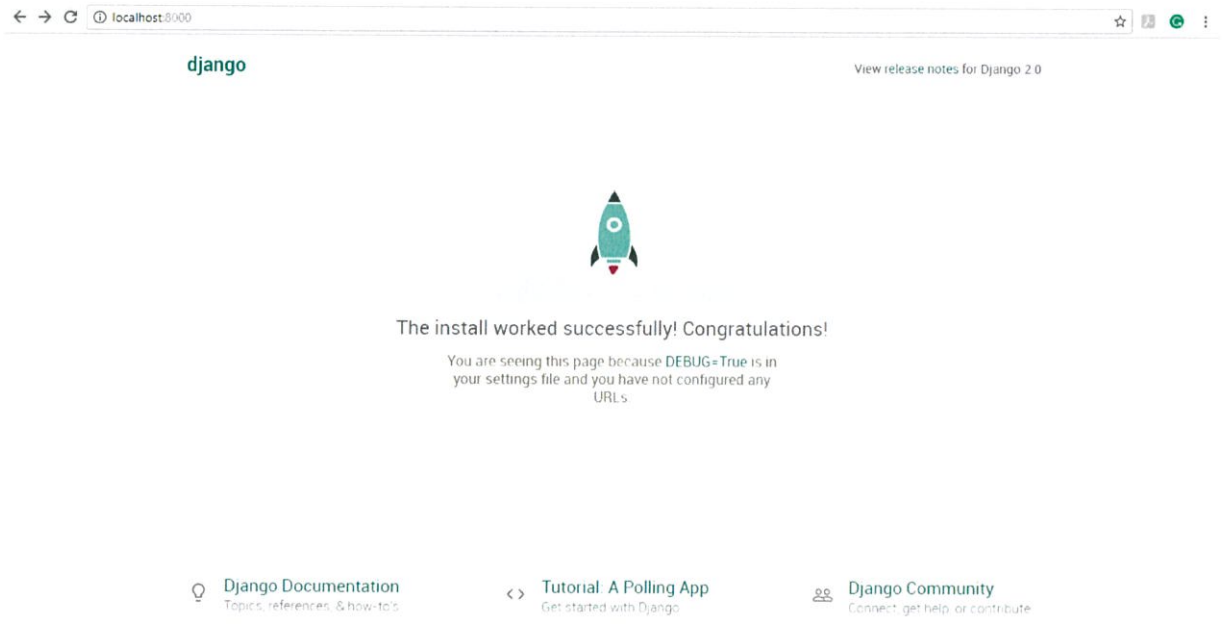
```
barenz@DESKTOP-CAJ32KP:~/project_barenz/start_django/hello_kmitl$ python3 manage.py runserver 0.0.0.0:8000
Performing system checks...

System check identified no issues (0 silenced).

You have 14 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s
 auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.

January 18, 2018 - 17:14:53
Django version 2.0.1, using settings 'hello_kmitl.settings'
Starting development server at http://0.0.0.0:8000/
Quit the server with CONTROL-C.
[18/Jan/2018 17:15:27] "GET / HTTP/1.1" 200 16559
[18/Jan/2018 17:15:27] "GET /static/admin/css/fonts.css HTTP/1.1" 200 423
[18/Jan/2018 17:15:27] "GET /static/admin/fonts/Roboto-Bold-webfont.woff HTTP/1.1" 200 82564
[18/Jan/2018 17:15:28] "GET /static/admin/fonts/Roboto-Light-webfont.woff HTTP/1.1" 200 81348
[18/Jan/2018 17:15:28] "GET /static/admin/fonts/Roboto-Regular-webfont.woff HTTP/1.1" 200 80304
```

ภาพที่ 15 แสดงการรัน django project ครั้งแรก



ภาพที่ 16 แสดงผลหน้าเว็บบราวเซอร์เมื่อรัน django project

สร้าง django application โดยจะตั้งชื่อว่า show_hello_message ด้วยคำสั่ง *python3 manage.py startapp show_hello_message* จากนั้นเข้าไปที่โฟลเดอร์ show_hello_message

ภาพที่ 17 แสดงโครงสร้างภายในแอพพลิเคชั่น show_hello_message

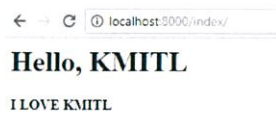
3. ทดลองแสดงข้อความ "Hello KMITL" กับ "I LOVE KMITL" บนหน้าเว็บบราวเซอร์โดยเริ่มที่ไฟล์ *view.py* เป็นไฟล์ที่ใช้เก็บ logic ของ web application คอยควบคุมสิ่งที่รับมาจาก client web browser (HTTP parameters ต่าง ๆ) และควบคุมสิ่งที่จะตอบกลับไปยัง client web browser ทดลองเปิดไฟล์ view โดยพิมพ์ vim view.py (หรือใช้ editor) และเพิ่มคำสั่งต่อไปนี้ลงไป

ภาพที่ 18 แสดงโค้ดที่จะแสดงข้อความในไฟล์ view.py

4. จากนั้นแก้ไขไฟล์ urls.py ให้เป็นดังรูป

ภาพที่ 19 แสดงโค้ดในไฟล์ urls.py

5. ลองรัน django อีกครั้งและไปที่บราวเซอร์เข้า url: *http://localhost:8000/index/*



ภาพที่ 20 แสดงเว็บบราวเซอร์ที่แสดงข้อความ "Hello, KMITL", "I LOVE KMITL"

2.2.2 Apache Airflow

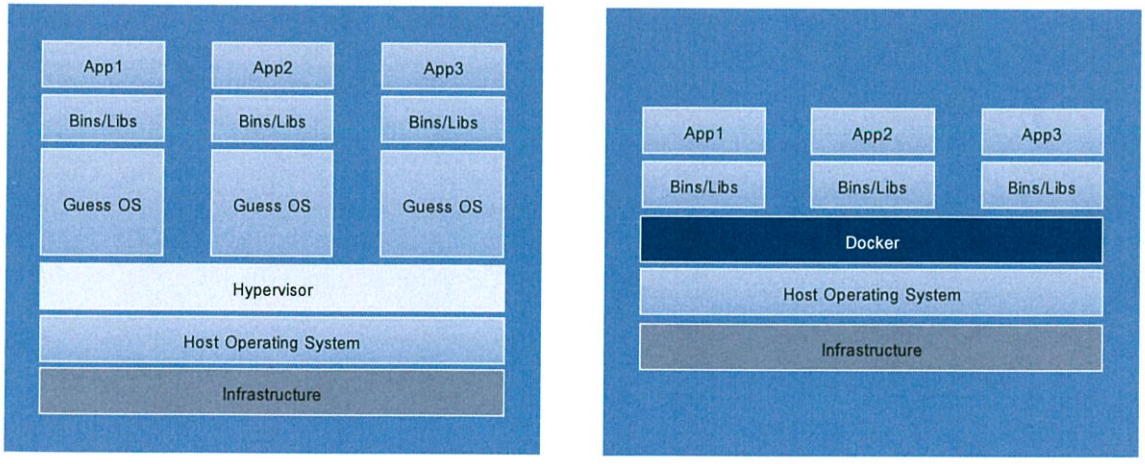
เป็นระบบที่ทำ workflow management ที่ช่วยเรื่อง scheduling and monitoring งานต่างๆ และอีกทั้งยังได้ถูกออกแบบมาใช้ทำ ETL ได้เป็นอย่างดี ระบบนี้เริ่มต้นพัฒนาโดย AirBnB ในปี 2014 และได้เข้าร่วม Apache Incubator Projects ในปี 2016

ข้อดีของการใช้ Airflow เช่น

- การสร้าง workflow คือการสร้าง Directed Acyclic Graph (DAG)
- วิธีการสร้าง DAG เราเขียนด้วยภาษา Python (configuration as code!)
- โปรเจคเป็น active development มี community ที่ใหญ่
- หน้า UI สวยงาม

2.2.3 Docker

Docker คือ ซอฟแวร์คอนเทรนเนอร์ที่มีการทำงานในลักษณะจำลองสภาพแวดล้อมขึ้นมาบนเครื่อง server เพื่อใช้ในการ run service ที่ต้องการ มีการทำงานคล้ายคลึงกับ Virtual Machine แต่ข้อแตกต่างที่ชัดเจนคือ Virtual Machine ที่รู้จักกันก่อนหน้านี้นั้น เป็นการจำลองทั้ง OS เพื่อใช้งานและหากต้องการใช้งาน service ใดๆ จึงทำการติดตั้งเพิ่มเติมบน OS นั้นๆ แต่สำหรับ docker นั้นจะใช้ container ในการจำลองสภาพแวดล้อมขึ้นมา เพื่อใช้งานสำหรับ 1 service ที่ต้องการใช้งานเท่านั้น โดยไม่ต้องมีส่วนของ OS เข้าไปเกี่ยวข้องเหมือน Virtual Machines อื่นๆ



ภาพที่ 21 แสดงสถาปัตยกรรมของ docker เทียบกับ virtual machine

Docker image

Docker image เป็นเหมือนตัวต้นแบบของ container ซึ่งภายในจะประกอบด้วยแอพลิเคชั่นต่างๆ ที่มีการติดตั้งไว้เพื่อใช้งานสำหรับเซอร์วิสนั้นๆ รวมทั้งมีการตั้งค่าต่างๆ ไว้เรียบร้อยแล้ว จากนั้นก็นำมาสร้างเป็น docker image บน registry เพื่อนำไปใช้งาน ทั้งนี้ผู้ใช้งานยังสามารถสร้าง docker image สำหรับใช้งานเองได้อีกด้วย

Docker container

Docker container สามารถมองได้เสมือนกล่อง ซึ่งนำ docker image มาติดตั้ง เพื่อให้สามารถใช้งานเซอร์วิสที่ต้องการจาก image นั้นๆ ได้ โดยใน container แต่ละตัวจะมีการใช้งาน RAM, CPU, ไฟล์ การตั้งค่า ต่างๆ เป็นของแต่ละ container เอง และยังสามารถสั่ง start, stop ได้ที่ container นั้นๆ อีกด้วย

ข้อดีของการใช้ Docker เมื่อเทียบกับการใช้ Virtual Machine (VMs)

- ไม่ต้องเสียเวลาในสร้าง OS ใหม่ และการ config ของแต่ละ OS
- เบาและเร็วกว่ามาก ไม่ว่าจะเป็น start stop และ restart เพราะใช้ OS, CPU และ RAM ร่วมกันกับ Host OS
- Docker สามารถรัน container ได้มากกว่า VMs ในเครื่องที่มีทรัพยากรที่เท่ากัน
- มีระบบ Registry ทำให้สามารถเคลื่อนย้าย หรือติดตั้ง container ได้สะดวก และรวดเร็วกว่ามาก
- Containers รันอยู่บน Docker Engine ทำให้ไม่ต้องสนใจ Infrastructure หรือ Host OS ว่าจะเป็นอย่างไร ทำให้หมดปัญหาเช่น เครื่อง Develop สามารถรันได้ แต่เครื่อง Production มันรันไม่ได้บ้าง หรือเครื่อง Develop ของแต่ละคนติดตั้งเครื่องมือคนละเวอร์ชันกัน ก็เพียง build container เป็น image แล้วส่งในคนในทีมใช้ เพียงเท่านี้ปัญหาก็หมดไป

2.2.4 Nginx

Nginx (อ่านว่า เอ็นจิ้นเอ็ก) เป็นเว็บเซิร์ฟเวอร์ประเภทหนึ่งเหมือนกับ Apache หรือว่าตัว IIS ซึ่งโดยจุดเด่นของ Nginx คือ มีประสิทธิภาพมากกว่า Apache ด้วยการที่ใช้ทรัพยากรของเครื่องที่น้อยกว่า เช่น RAM (แรม) และ CPU (ซีพียู) ทำให้เซิฟเวอร์นั้นสามารถทำงานได้มากยิ่งขึ้น แต่เนื่องมาจากว่าการตั้งค่า (configuration) ที่ค่อนข้างจะยุ่งยากรวมไปถึงการใช้งานบางอย่างที่ไม่ได้รองรับเหมือนกันกับ

Apache ซึ่งมีจำนวนผู้ใช้มาก ทำให้ตัว Nginx ถูกใช้งานเพียงบางอย่าง เช่น การทำเว็บไซต์เกี่ยวกับดาวน์โหลด การทำเว็บไซต์เกี่ยวกับพวกสตรีมมิ่ง การทำเว็บไซต์อัปโหลด ซึ่งจะมีพื้นที่สามารถรองรับจำนวนของผู้ใช้ได้มากกว่านั่นเอง

ข้อดีของ Nginx

- รองรับมารตฐานในด้านความปลอดภัย HTTP/2
- รับรองการทำงาน HTTP ได้อย่างครบถ้วน
- ไฟล์ที่เป็น static จะประมวลผลได้เร็วกว่า Apache
- ทำงานแบบ Asynchronous โดยไม่มีการหยุดรออะไรทั้งนั้น แยกๆ กันไปทำงานทันที จึงใช้ทรัพยาน้อยกว่า ทำงานได้เร็วกว่า รองรับจำนวนผู้ใช้งานได้มากกว่า Apache

ข้อเสียของ Nginx

- การตั้งค่าที่ค่อนข้างจะยุ่งยากกว่า Apache เนื่องจากการออกแบบที่ค่อนข้างต้องการประสิทธิภาพที่สูงทำให้ต้องตัดการประมวลผลที่เป็นด้วยตัวเองออกไป แล้วไปให้โปรเซสอื่นหรือระบบอื่นจัดการประมวลผลแบบ Dynamic ให้แทนเช่น FastCGI, SCGI, uWSGI, memcache
- การบำรุงรักษายากกว่า Apache เนื่องจาก Nginx ได้มีการออกแบบให้เป็นโมดูลเช่นกันแต่ไม่ได้ยืดหยุนมาก ถ้าจะต้องการเพิ่มหรือแก้ไขโมดูลต่างๆ จะไม่ค่อยสะดวก
- การเอาไปทำงานได้หลายแพลตฟอร์ม ยังพอร์ตไปไม่ครบนัก ติดตั้งไม่ง่ายนัก การทำงานร่วมกับองค์ประกอบอื่นยังต้องตั้งค่าอีกเยอะ ต่างกับ Apache ที่พอร์ตไปทุกที่ได้ง่ายกว่า

2.2.5 Python Unit Testing

Unit test คือการเขียนเทส ในส่วนเล็กของโปรแกรมเพื่อแสดงว่าสามารถทำงานได้อย่างที่ควรและต้องทำง่ายได้ง่ายและเร็ว เขียนสั้น กระชับ และเขียนเฉพาะส่วนที่โปรแกรมเมอร์เห็นว่าจำเป็นต้องเทส ซึ่งเทสชุดนี้โปรแกรมเมอร์เป็นคนใช้เท่านั้น คนอื่นจะไม่ได้ใช้และจะไม่เข้าใจอีกด้วย เพียงแต่ถ้าเทสสามารถทำงานได้อย่างที่มันควรทำได้ ในส่วนของ tester และคนอื่นๆ เช่น ผู้ใช้งาน จะได้รับผลประโยชน์ตรงที่เจอบั๊กน้อยลง Python Unit Testing คือการเขียนเทสโดยใช้ภาษาไพธอนซึ่งมีความง่ายและสะดวก

บทที่ 3

การติดตั้งเครื่องมือและการนำไปใช้

การดำเนินการในโครงงานการจัดการ WordPress ด้วยเครื่องมือ iPhoenix ประกอบไปด้วยขั้นตอนดังนี้

1. การติดตั้งเครื่องมือ
2. การนำเครื่องมือไปใช้

3.1 การติดตั้งเครื่องมือ

3.1.1 ติดตั้ง Docker

ไฟล์ต่างๆทั้งหมดของเครื่องมือ iPhoenix ถูกรันบน Docker อีกที ดังนั้นจึงต้องติดตั้ง Docker ไว้บนเครื่องของเราก่อน ซึ่งข้อดีของการใช้ Docker นั้นคือสามารถรัน iPhoenix ขึ้นที่ platform แบบไหนก็ได้เช่น Mac os, Windows, Linux เป็นต้น ซึ่งเพียงการติดตั้ง Docker เท่านั้น ที่แตกต่างตามแต่ละ platform เครื่องมืออื่นๆ ติดตั้งบน Docker เท่านั้น

ในที่นี้จะทำการติดตั้ง Docker บนระบบปฏิบัติการ Linux ซึ่งทำตามขั้นตอนดังนี้

- ตั้งค่า repository ของ Docker
  1. อัพเดตแพคเกต *apt*

```
$ sudo apt-get update
```

ภาพที่ 22 แสดงการอัพเดตแพคเกต apt

ติดตั้งแพคเกตเพื่อให้ *apt* ใช้ repository ผ่าน HTTPS

```
$ sudo apt-get install \
    apt-transport-https \
    ca-certificates \
    curl \
    software-properties-common
```

ภาพที่ 23 แสดงการติดตั้งแพคเกตต่างๆ สำหรับใช้ repository ผ่าน HTTPS

2. เพิ่มคีย์ GPG ของ Docker

```
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

ภาพที่ 24 แสดงการเพิ่มคีย์ GPG

3. ตั้งค่าให้ repository มีความเสถียร

```
$ sudo add-apt-repository \
  "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) \
  stable"
```

ภาพที่ 25 แสดงการเพิ่มความเสถียรให้กับ repository

- ติดตั้ง Docker

1. อัพเดตแพคเกต apt

```
$ sudo apt-get update
```

ภาพที่ 26 แสดงการอัพเดตแพคเกต apt

2. ติดตัง Docker เวอร์ชันล่าสุด

```
$ sudo apt-get install docker-ce
```

ภาพที่ 27 แสดงการติดตั้ง Docker เวอร์ชั่นล่าสุด

3. ทดสอบเพื่อยืนยันความถูกต้องของการติดตั้ง Docker

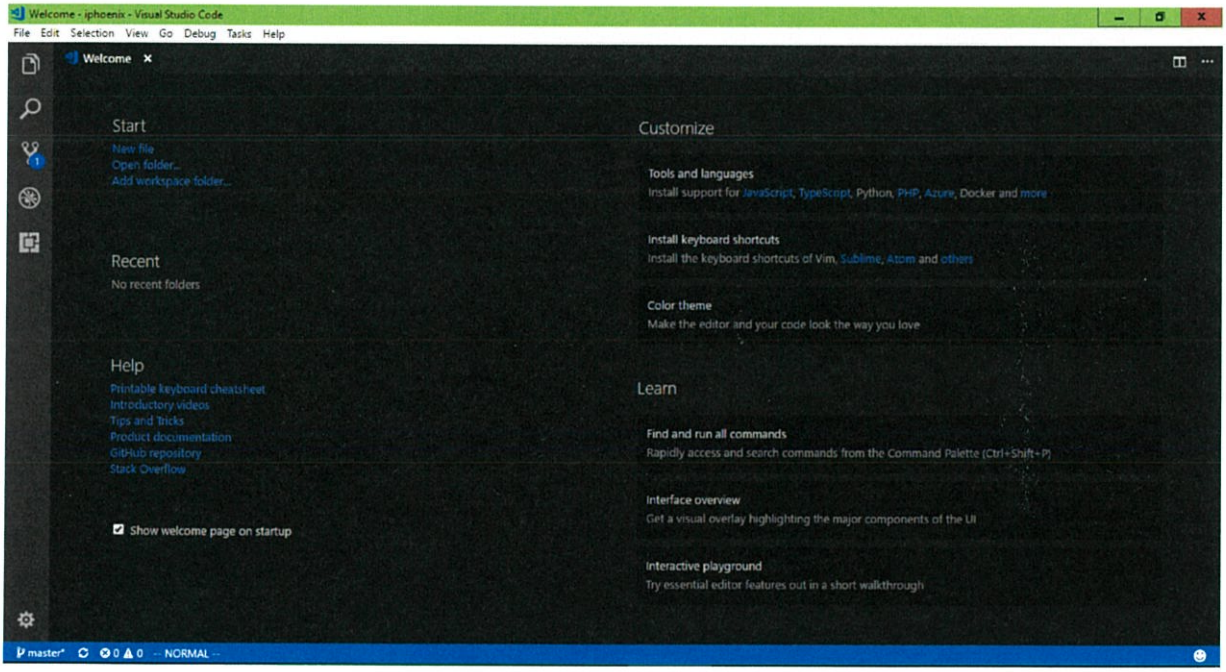
```
$ sudo docker run hello-world
```

ภาพที่ 28 แสดงการทดสอบเพื่อยืนความถูกต้องของการติดตั้ง Docker

3.1.2 ติดตั้งเครื่องมือ Visual Studio Code

Visual Studio Code หรือชื่อสั้นๆ คือ VS Code คือ Editor ที่ใช้ในการพัฒนางานต่างๆ ซึ่งในที่นี้ใช้เป็นตัวเขียนโค้ดหลักของทั้งโปรเจค iPhoenix นี้

การดาวน์โหลด VS Code เพียงแค่เซิร์ท Google คำว่า *Download VS Code* ก็เจอทันที หลังจากการดาวน์โหลดมาให้ทำการเปิดขึ้น หน้าตาของ VS Code จะเป็นดังนี้



ภาพที่ 29 แสดงหน้าต่างของ Visual Studio Code

3.1.3 ติดตั้งระบบ iPhoenix

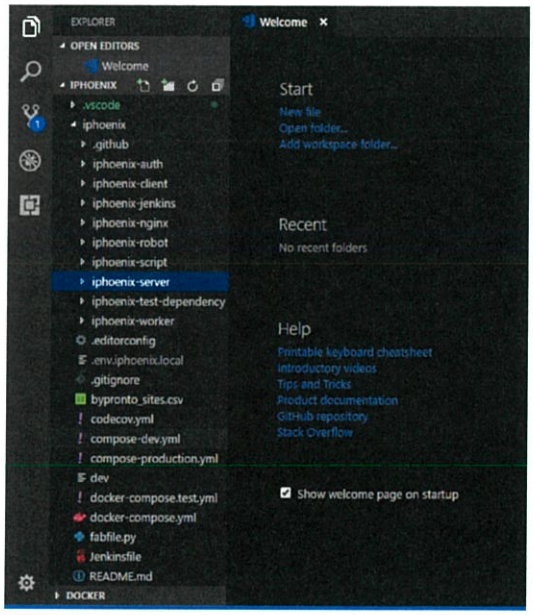
เนื่องจากทางบริษัทได้วางโครงสร้างของระบบ iPhoenix เอาไว้เรียบร้อยแล้ว และได้ทำการเก็บไฟล์ต่างไว้บน GitHub ซึ่งเป็นแบบส่วนบุคคล คือเป็นของบริษัทเท่านั้นหากไม่ได้บัญชีของทางบริษัทก็ไม่สามารถนำมาใช้ได้

การนำไฟล์โปรเจคลงมาไว้บนเครื่องของเราสามารถทำได้โดยใช้คำสั่งของ git ในการโคลนลงมา ซึ่งในที่นี้ไม่ขอเปิดเผยชื่อ repository ของทางบริษัท

```
$ git clone [repository of GitHub]
```

ภาพที่ 30 แสดงคำสั่งการโคลนโปรเจคจาก repository ของ GitHub

เมื่อรันคำสั่งเสร็จก็จะได้ไดเร็กทอรีของโปรเจค iPhoenix มาทั้งหมดซึ่งจะเก็บไฟล์ต่างๆ ตามโครงสร้างที่ทางบริษัทได้ออกแบบเอาไว้

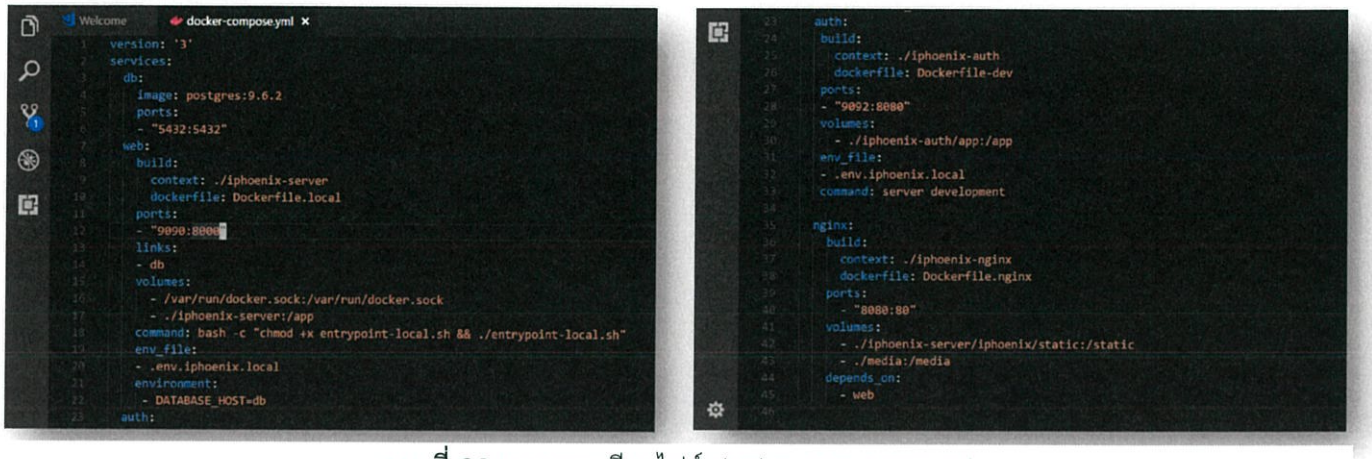


ภาพที่ 31 แสดงโครงสร้างของระบบ iPhoenix จากโปรแกรม Visual Studio Code

โครงสร้างของ iPhoenix หลักๆ ที่สำคัญเช่น ไฟล์ *docker-compose.yml*, โฟลเดอร์ *iphoenix-server, iphoenix-worker* เป็นต้น

ขั้นตอนการวางโครงสร้างต่างๆ ของ iPhoenix

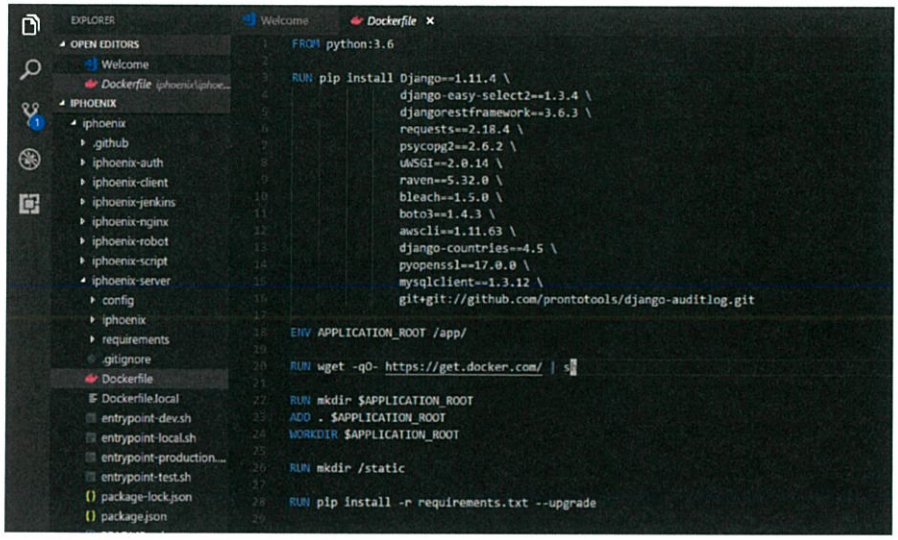
1. เขียนไฟล์ docker-compose.yml ขึ้นมา ซึ่งคือ ไฟล์ที่ใช้จัดการ docker container ทีละหลายๆตัว ซึ่ง container ของ iphoenix นั้น จะเก็บเซอร์วิสต่างๆ คือ database, web, authentication และ nginx



ภาพที่ 32 แสดงการเขียนไฟล์ docker-compose.yml

จากในไฟล์นั้นเซอร์วิส *db* คือ database ของ iPhoenix, *web* คือ หน้าอินเตอร์เฟสที่ให้ผู้ใช้ใช้งาน, *auth* คือ authentication เป็นการกำหนดสิทธิ์ในการใช้ iPhoenix และ *nginx* คือเว็บเซอร์เวอร์ที่ใช้ในการรัน *web* อีกที ซึ่งในแต่ละเซอร์วิสนั้นจะมีการ build จากไฟล์ dockerfile คือ ไฟล์ที่ใช้เตรียมลงแพคเกตต่างๆที่จะใช้ในแต่ละ container และในส่วน *volumes* ของเซอร์วิส *web* นั้นคือการเชื่อมไฟล์หรือโฟลเดอร์ของ iPhoenix ที่อยู่บนเครื่องเราเข้ากับ container ของเซอร์วิส *web* เฉพาะส่วนของโค้ดที่ใช้แสดงอินเตอร์เฟส

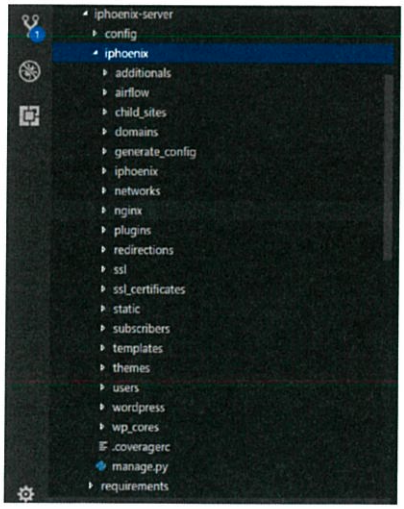
2. เขียนไฟล์ Dockerfile ของแต่ละเซอร์วิสเพื่อจะให้ environment ของแต่ละเซอร์วิสเป็นไปตามที่ต้องการ ในที่นี้จะดูในส่วนของเซอร์วิส *web* เพราะเป็นตัวหลักของอินเตอร์เฟส iPhoenix ส่วนเซอร์วิส *db* นั้นทำการ build โดยใช้ image ของ postgres



ภาพที่ 33 แสดงการเขียนไฟล์ Dockerfile ของเซอร์วิส web

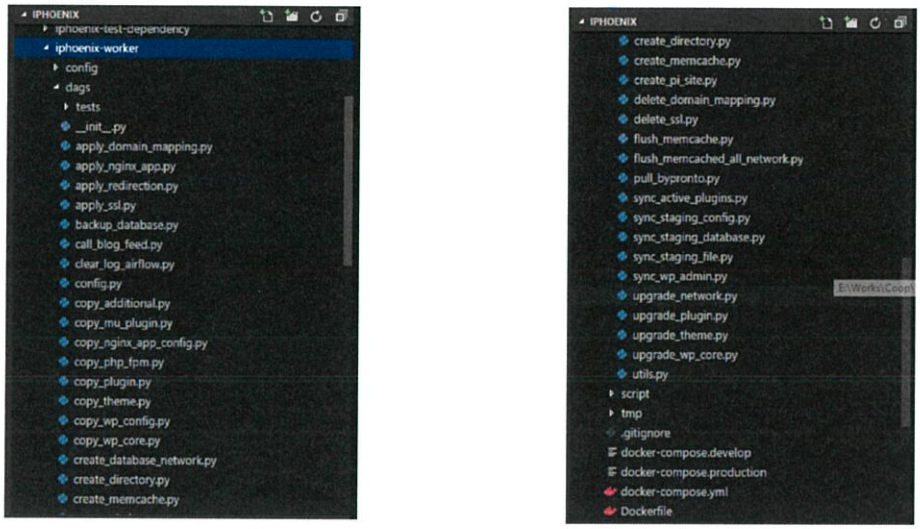
ซึ่งแพคเกจที่สำคัญๆ เช่น Django และ Python เนื่องจากอินเตอร์เฟสของ iPhoenix ถูกเขียนภาษาไพทอนและใช้กรอบการทำงาน Django และแพคเกตอื่นๆ ที่เป็นพื้นฐานจะถูกเก็บแยกไฟล์ไว้ใน requirements.txt เพื่อการลงซ้ำจะได้มาเรียกที่ไฟล์นี้เสมอ

3. ที่โฟลเดอร์ *iphoenix-server* จะเก็บฟังก์ชั่นต่างๆ ของ iPhoenix ไว้ทั้งหมด และใช้ Django ในการพัฒนาโครงสร้างจึงมีลักษณเดียวกับการรันโปรเจค Django ขึ้น ฟังก์ชั่นที่ทำการพัฒนาขึ้นเช่น จัดการ theme, plugin, domain mapping, redirection เป็นต้น



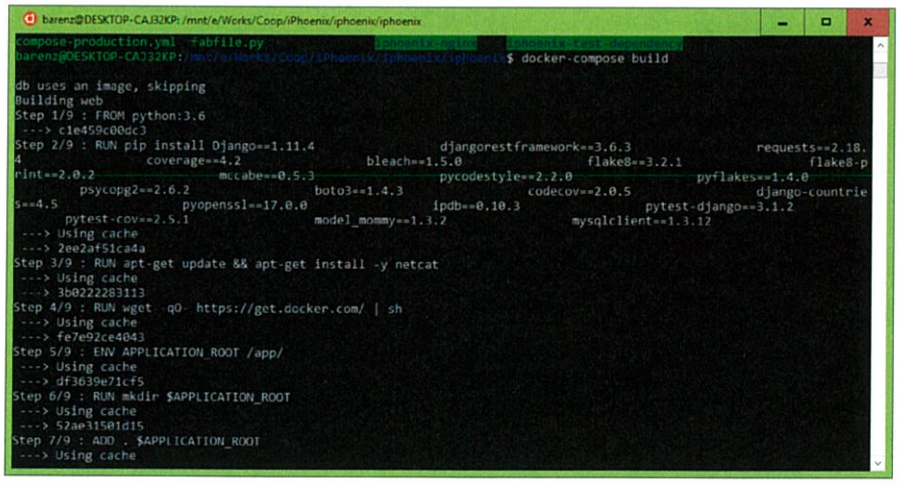
ภาพที่ 34 แสดงโครงสร้างของ iphoenix-server

4. ที่โฟลเดอร์ *iphoenix-worker* จะเก็บเครื่องมือที่มีชื่อว่า Airflow ซึ่งช่วยในการจัดการคิวในการทำงานของฟังก์ชั่นต่างๆ ของ *iphoenix-server* การติดต่อไปยังเซอเวอร์นอก และมีอินเตอร์เฟสสำหรับการมอนิเตอร์ฟังก์ชั่นเหล่านั้น เพื่อให้เป็นไปอย่างราบลื่น



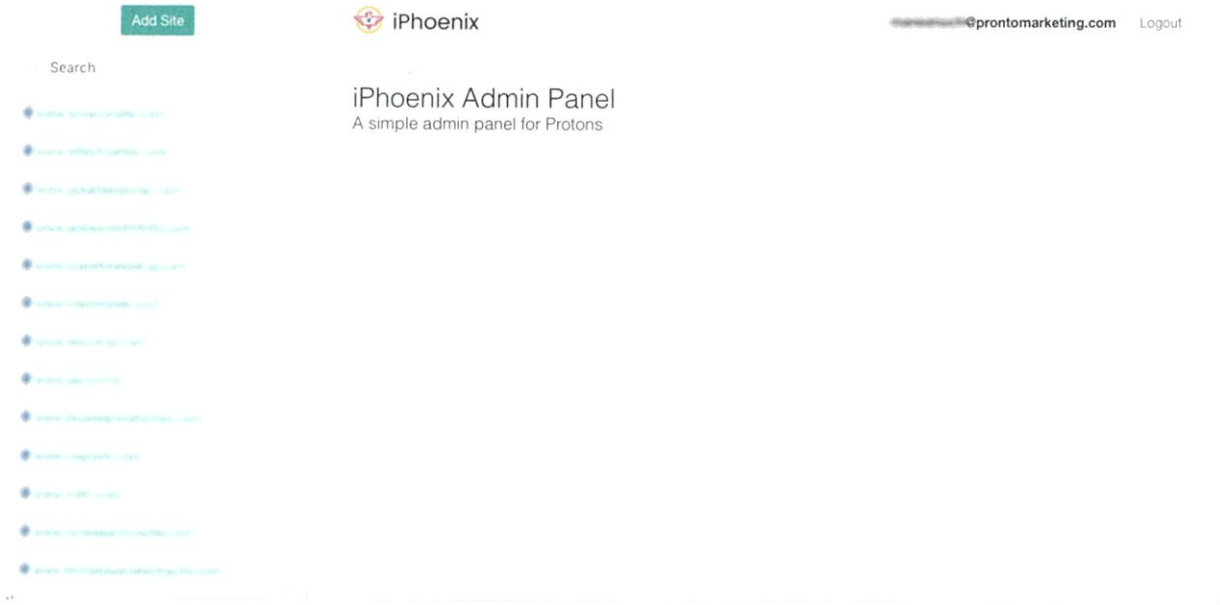
ภาพที่ 35 แสดงโครงสร้างของ iphoenix-worker

5. ทำการ build ไฟล์ docker-compose เพื่อติดตั้งแพคเกตต่างๆ ให้กับเซอร์วิสด้วยคำสั่ง *docker-compose build*



ภาพที่ 36 แสดงการ build แพคเกตต่างๆ ใน docker container ของเซอร์วิส web

6. หลังจาก build เสร็จเรียบร้อยต่อมาทำการรันเซอร์วิส web ขึ้นเพื่อแสดงหน้าอินเตอร์เฟสของ iPhoenix ด้วยคำสั่ง *docker-compose up -d*
7. หลังพิมพ์คำสั่งเรียบร้อย สามารถเช็คได้ว่ามี container เซอร์วิสไหนทำงานอยู่บ้าง ต่อจากนั้นให้เข้าเว็บบราวเซอร์ พิมพ์ URL ชื่อว่า localhost:9000 จะแสดงอินเตอร์เฟสหน้าแรกขึ้นมา



ภาพที่ 37 แสดงหน้าอินเตอร์เฟสของ iPhoenix

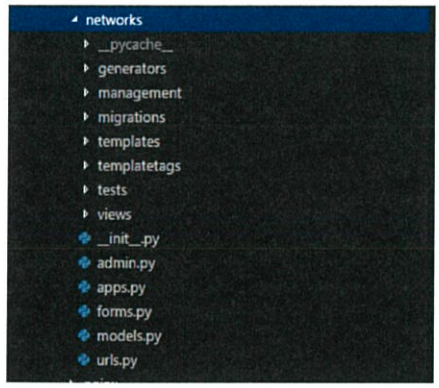
3.1.4 ติดตั้งฟังก์ชั่นต่างๆ ของ iPhoenix

3.1.4.1 WordPress Multisite Management (Networks Management)

เป็นฟังก์ชั่นใช้ในการจัดการเวิร์ดเพสแบบมัลติไซต์ WordPress Multisite หรือในคำศัพท์ของ WordPress เรียกว่า Network ซึ่งเมื่อสร้างขึ้น จะประกอบไปด้วย 2 Site ในส่วนของ Site แรกจะเป็นส่วนที่ลูกค้าใช้งานจริง ส่วนของ Site ที่สองเป็นส่วนที่ใช้ในการพัฒนาหรือปรับปรุงแก้ไขเว็บไซต์ที่ลูกค้าใช้งานจริง (Site แรก) ทำให้สามารถพัฒนาเว็บไซต์ในระหว่างที่ลูกค้าประกอบธุรกิจได้โดยไม่ต้องปิดเว็บไซต์ชั่วคราว

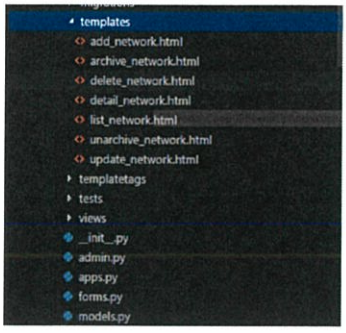
ขั้นตอนการพัฒนาฟังก์ชั่น Networks Management

1. เนื่องจากพัฒนาฟังก์ชั่นด้วย Django โครงสร้างสำคัญจะประกอบไปด้วยไฟล์ในโฟลเดอร์ *views, templates* และไฟล์ *admin.py, apps.py, forms.py, models.py, urls.py*

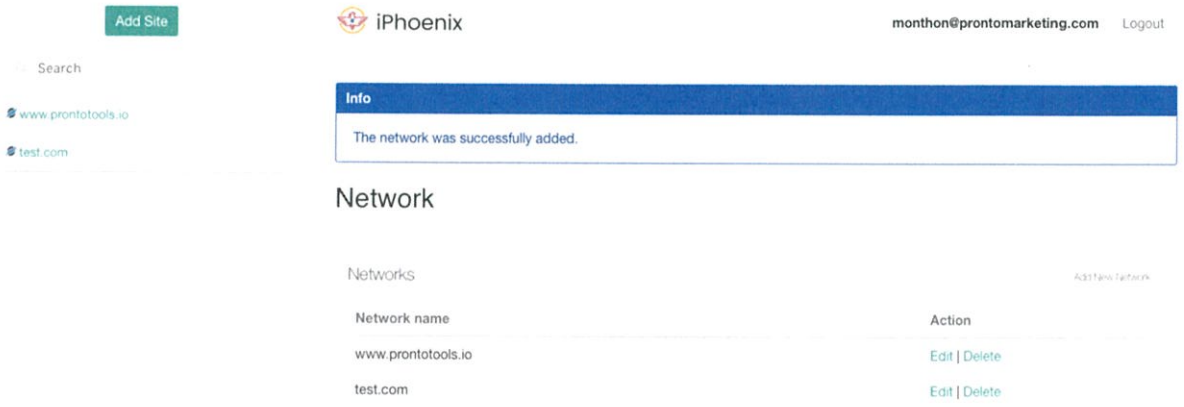


ภาพที่ 38 แสดงโครงสร้างของฟังก์ชั่น Networks Management

2. ในโฟลเดอร์ *templates* จะเก็บไฟล์ html ซึ่งเป็นโค้ดที่ใช้แสดงหน้าอินเตอร์เฟสของฟังก์ชั่น ซึ่งจะประกอบไปด้วยฟังก์ชั่นย่อยคือ หน้าแสดงรายชื่อเว็บไซต์ที่เป็นแบบมัลติไซต์ หน้าเพิ่ม แก้ไข ลบไซต์ และหน้ารายละเอียดของไซต์

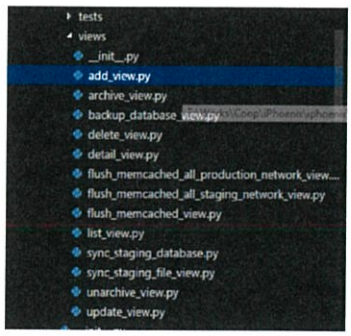


ภาพที่ 39 แสดงโครงสร้างของโฟลเดอร์ templates



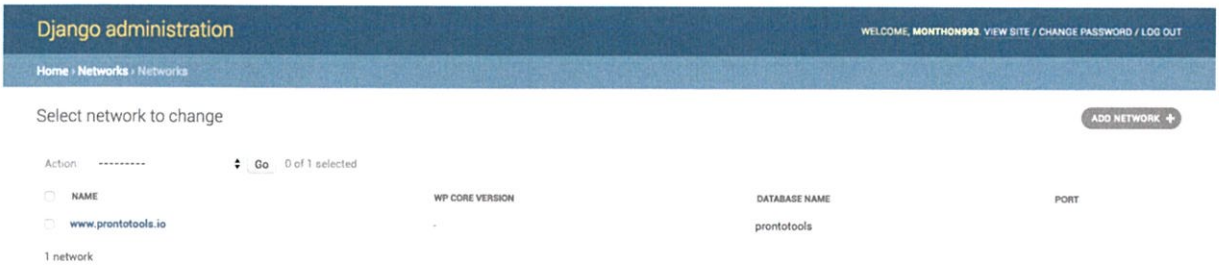
ภาพที่ 40 แสดงรายชื่อไซต์

3. โฟลเดอร์ *views* ประกอบไปด้วยไฟล์ *view.py* ต่างๆ ซึ่ง view นั้นคือ controller เขียนขึ้นเพื่อสั่งให้ทำงานตามที่เราต้องการเช่น การเพิ่ม ลบ แก้ไขไซต์ และเขียนขึ้นเพื่อให้ประมวลผลหน้า template จากไฟล์ html มาเป็นหน้าอินเตอร์เฟส



ภาพที่ 41 แสดงโครงสร้างของ views

4. ไฟล์ *admin.py* เนื่องจาก Django มีระบบแอดมินในตัวจึงสามารถสร้างหน้าจัดการในส่วนของระบบหลังบ้านได้ ไฟล์นี้จึงสร้างขึ้นเพื่อจัดการไซต์ในส่วนของสิทธ์แอดมินคือจัดการได้ทุกอยากหากระบบหน้าบ้านมีปัญหา

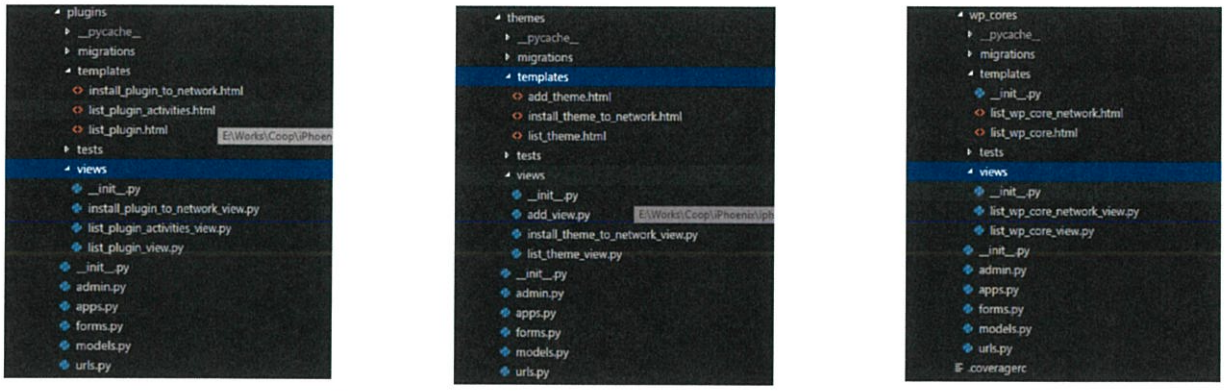


ภาพที่ 42 แสดงอินเตอร์เฟสระบบแอดมินของฟังชั่นการจัดการไซต์

5. ไฟล์ forms.py เป็นไฟล์ที่สร้างขึ้นเพื่อจัดการกับฟอร์มต่างๆ เช่นในฟอร์มของการเพิ่มไซต์ เงื่อนไขต่างๆในการที่จะกรอกฟอร์มจะเขียนที่ไฟล์นี้
6. ไฟล์ models.py เป็นไฟล์ที่เก็บฟิลด์ต่างๆ ในฐานข้อมูล ในฟังก์ชั่นการจัดการไซต์นี้จะมีฟิลด์ name, wp_core_version, database_name และ archived อีกทั้งเขียน override ฟังก์ชั่น save() เพื่อเงื่อนไขบางอย่างก่อนที่จะบันทึกข้อมูลลงฐานข้อมูลได้เช่น ในการเพิ่มไซต์แบบมัลติไซต์ (Network) จะเขียนให้สร้างไซต์ย่อยๆ อีกสองไซต์ด้วย ใน iPhoenix เรียกไซต์ที่เกิดขึ้นมาตอนสร้าง network ว่า Child site
7. ไฟล์ urls.py เป็นไฟล์ที่เก็บ url ของฟังก์ชั่นในแต่ละส่วนของ view ซึ่งจะทำให้เชื่อมต่อไปยัง view ต่างๆ ได้ผ่าน url ที่เขียนขึ้นในไฟล์นี้

3.1.4.2 ฟังก์ชั่น Plugin, Theme and WordPress Core Management

ทั้งสามฟังก์ชั่นนี้โครงสร้างที่เขียนขึ้นมีลักษณคล้ายกัน เนื่องจากการจัดการที่เหมือนกัน

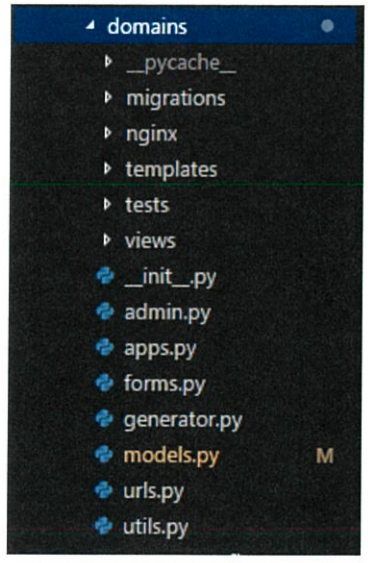


ภาพที่ 43 แสดงโครงสร้างของ plugin, theme และ wp core

3.1.4.3 ฟังก์ชั่น Domain mapping

เป็นฟังก์ชั่นที่จัดการเรื่อง Domain mapping ของเวิร์ดเพรส ทำให้สามารถใช้ URL เดิมได้ในกรณีที่จด Domain ใหม่ขึ้นมา หรือถ้าหากซื้อเป็น Multi-Domain เพื่อทำลูกเล่นในการเซิร์ชที่หลากหลายหรือป้องกันผู้ใช้สะกดคำผิดเมื่อกรอก URL ค้นหาเว็บไซต์ ก็สามารถทำ Mapping มายัง Domain หลักได้

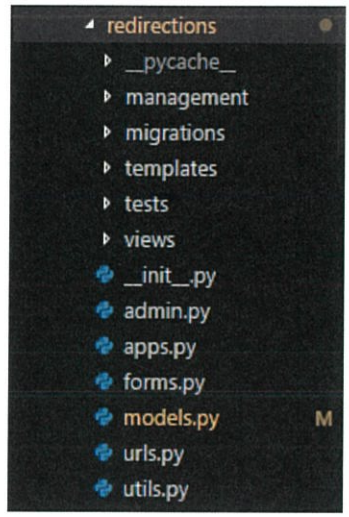
โครงสร้างของโฟลเดอร์จะคล้ายกับโครงสร้างของ networks ซึ่ง domain นั้นเป็นส่วนย่อยของ childsites และ childsites เป็นส่วนย่อยของ networks อีกทีหนึ่ง



ภาพที่ 44 แสดงโครงสร้างของ domains

3.1.4.4 ฟังก์ชั่น Redirection

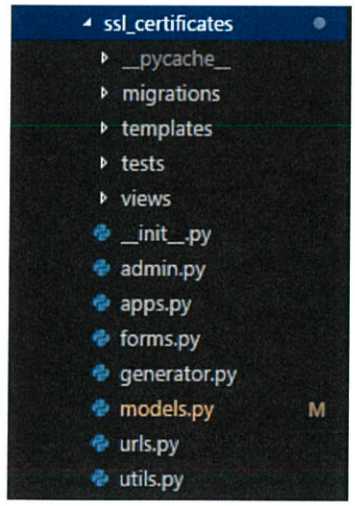
เป็นฟังก์ชั่นที่จัดการกับ sub-directory ที่อาจจะไม่ได้ใช้แล้วแต่ต้องการให้เข้าผ่าน URL เดิมได้หรือทำเพื่อเทคนิคการ query หลีกเลี่ยงการพิมพ์ผิดให้ redirect ไปยัง URL ที่เราต้องการภายใต้ domain นั้น



ภาพที่ 45 แสดงโครงสร้างของ redirections

3.1.4.5 ฟังก์ชั่น SSL Certificate

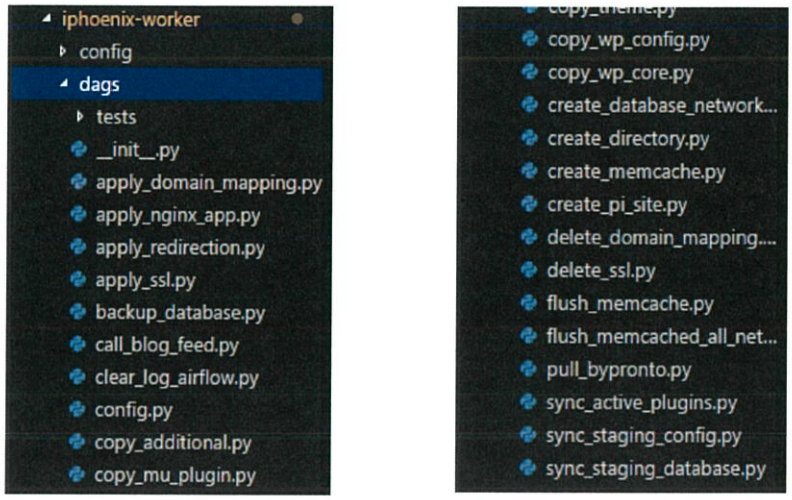
เป็นฟังก์ชั่นที่จัดการเรื่อง security ต่างๆ ให้กับ domain นั้นๆ



ภาพที่ 46 แสดงโครงสร้างของ SSL Certificate

3.1.5 ติดตั้ง Airflow

Airflow จะอยู่ในโปรเจคเดียวกับ iPhoenix แต่แยกเป็นอีกโฟลเดอร์หนึ่งซึ่งอยู่ในโฟล์เดอร์ iphoenix-worker

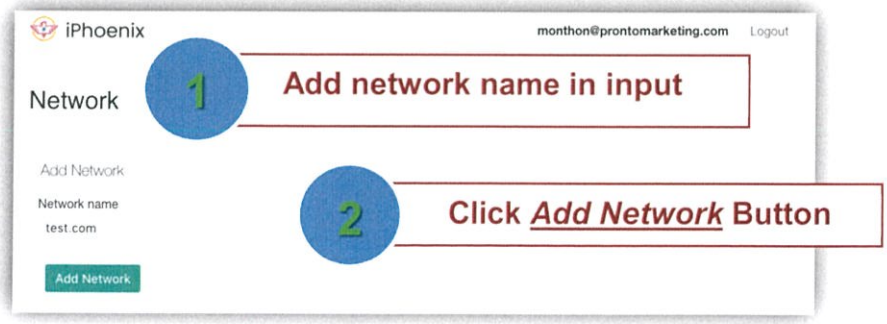


ภาพที่ 47 แสดงโครงสร้างของ Airflow

3.2 การนำเครื่องมือไปใช้

3.2.1 Networks Management

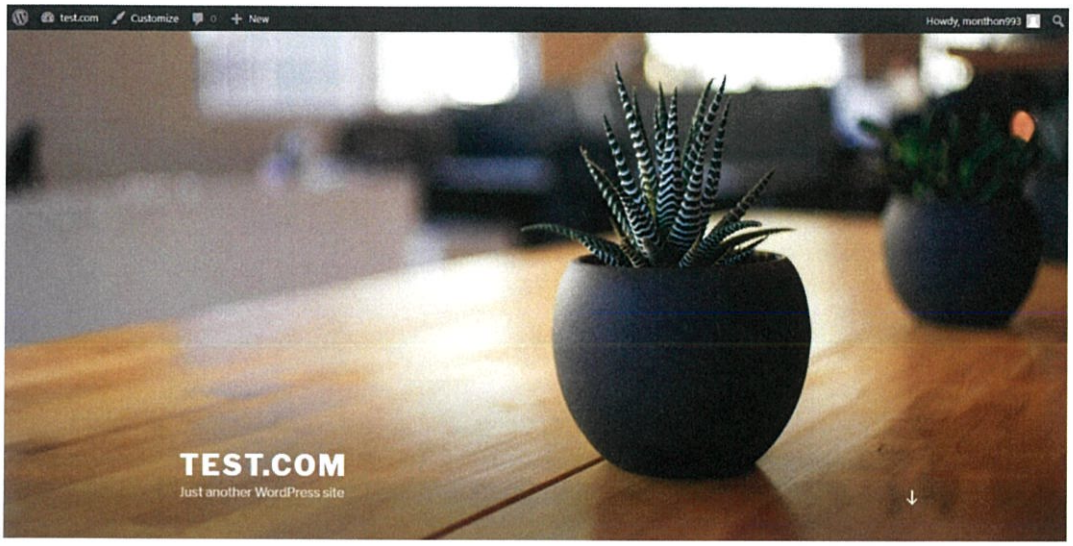
ขั้นตอนการสร้าง network ตัวอย่างเช่น สร้าง network test.com



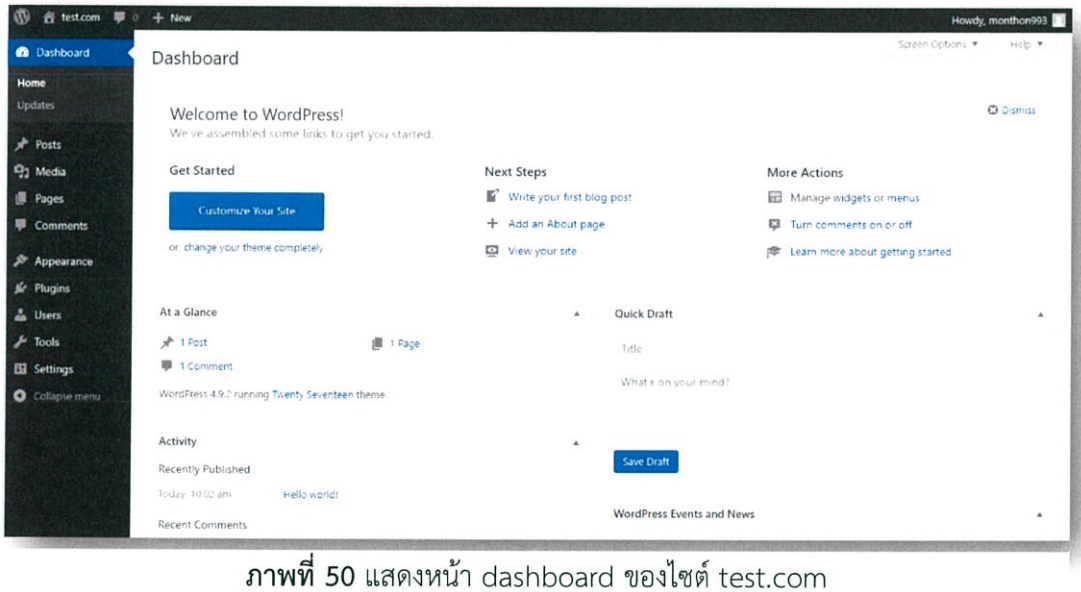
ภาพที่ 48 แสดงขั้นตอนการสร้าง network

1. ใส่ชื่อ network ที่ต้องการในช่องอินพุต ในที่นี้ใส่ test.com
2. กดปุ่มสีเขียวที่ชื่อว่า “Add Network”

เพียงเท่านี้ก็จะได้ไซต์เปล่าๆ ขึ้นมาหนึ่งไซต์



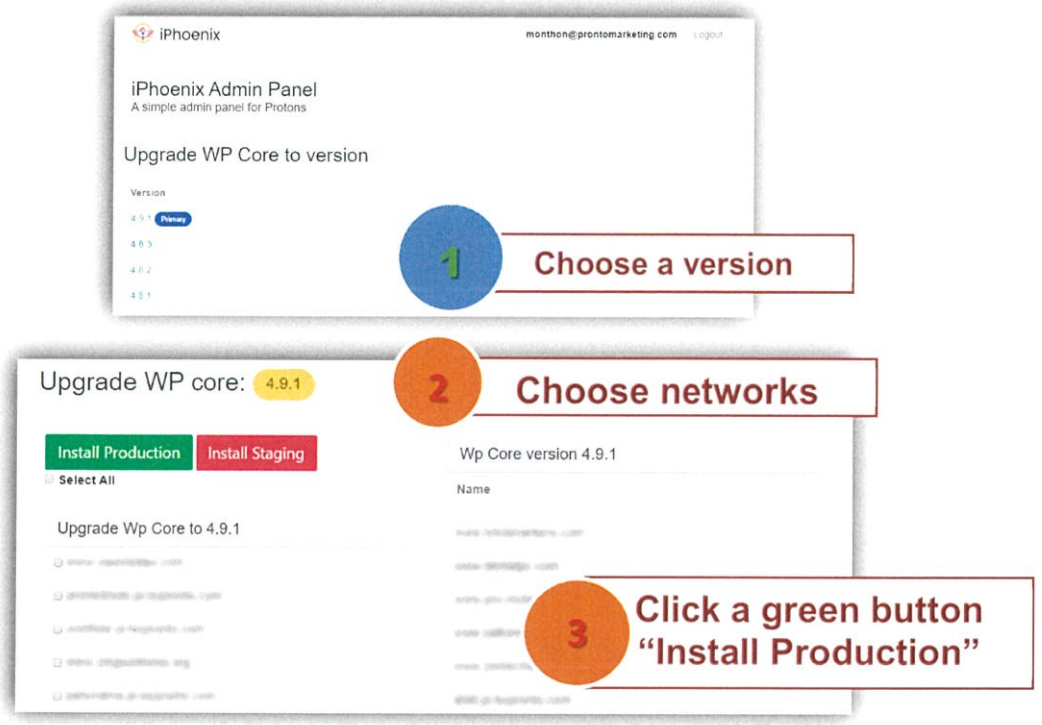
ภาพที่ 49 แสดงหน้าไซต์ของ test.com



ภาพที่ 50 แสดงหน้า dashboard ของไซต์ test.com

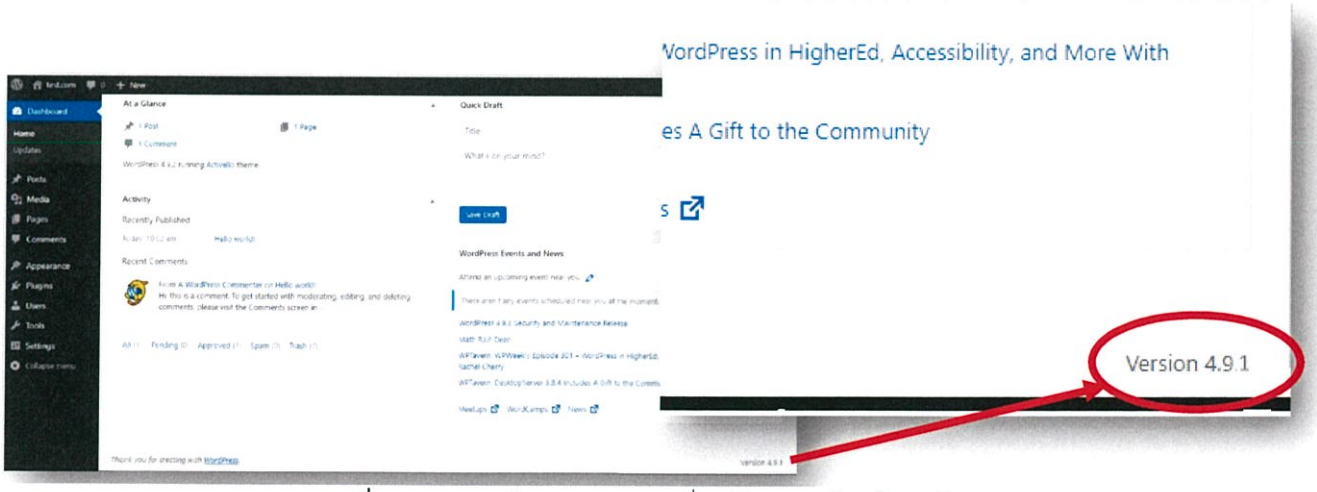
3.2.2 WordPress Core Management

การอัพเกรด WordPress core เป็นเวอร์ชั่นใหม่หรือเก่าลง สามารถทำได้ดังนี้ เช่น ต้องการจะอัพเกรดเป็นเวอร์ชั่น 4.9.1



ภาพที่ 51 แสดงขั้นตอนการอัพเกรด WordPress Core

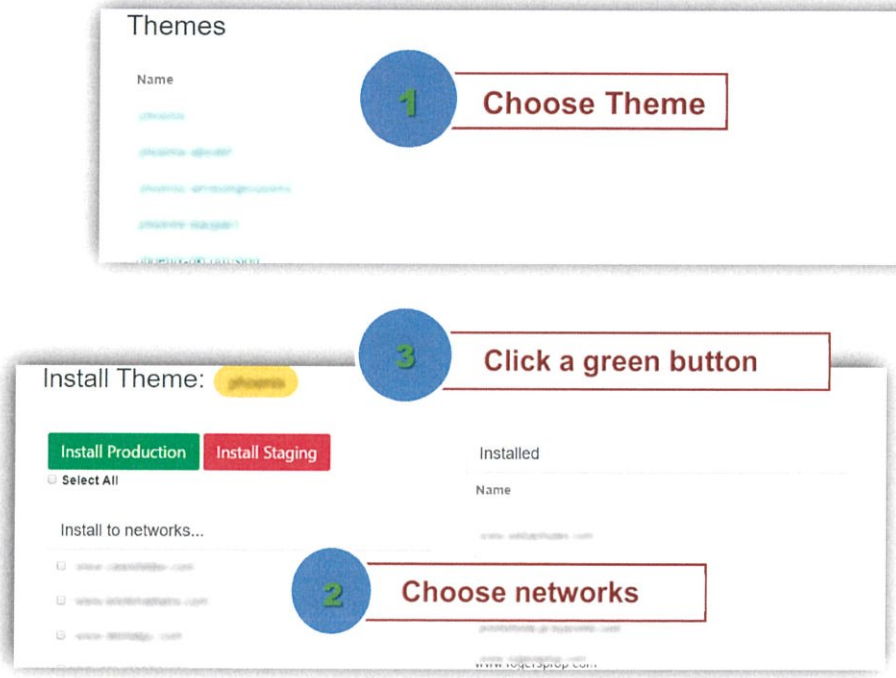
1. เลือกเวอร์ชั่นที่ต้องการ ในที่นี้เลือก 4.9.1
2. เลือก network ของ WordPress ที่ต้องการจะอัพเกรด
3. คลิกปุ่มสีเขียว "Install Production" เพื่ออัพเกรดเวอร์ชั่น



ภาพที่ 52 แสดงหน้า dashboard ที่ถูกอัพเกรดเรียบร้อยแล้ว

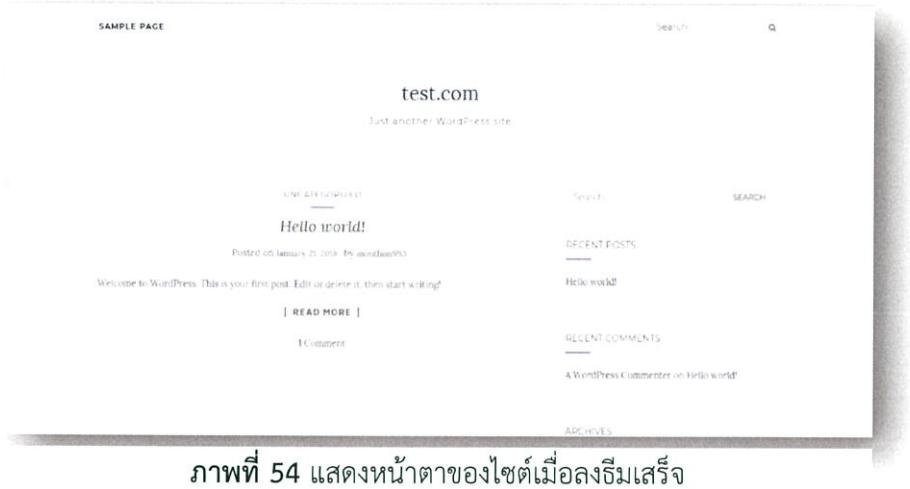
3.2.3 Theme Management

ในการจัดการธีมนั้นจะมีการ install, update และ delete ธีมในตัวอย่างจะแสดงการ install ธีมให้กับ network test.com



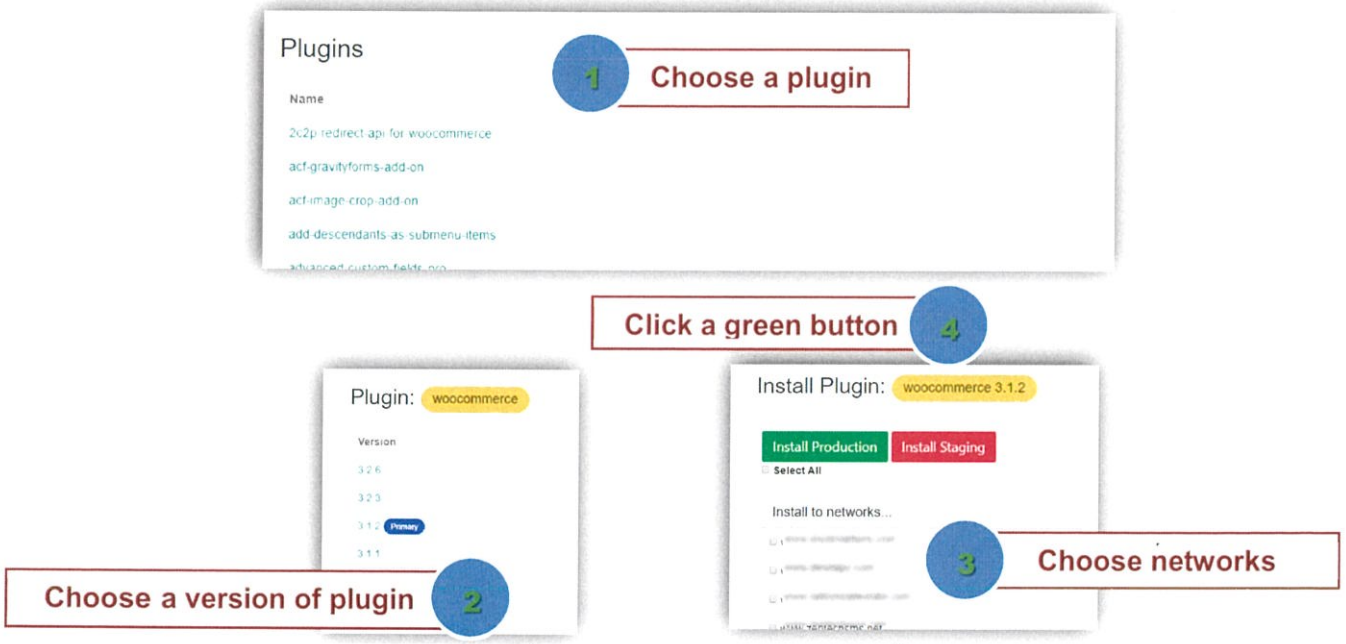
ภาพที่ 53 แสดงขั้นตอนการ install ธีม

1. เลือกธีมที่ต้องการ
2. เลือก network ที่ต้องการจะลงธีม
3. กดปุ่มสีเขียวเพื่อ install ธีมให้กับ network



ภาพที่ 54 แสดงหน้าตาของไซต์เมื่อลงธีมเสร็จ

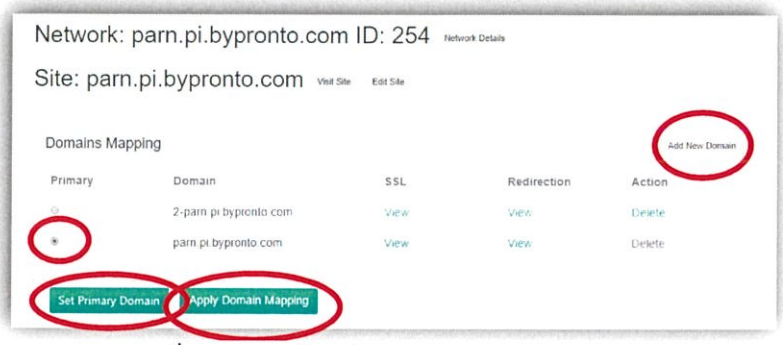
3.2.4 Plugin Management



ภาพที่ 55 แสดงตัวอย่างการ install ปลั๊กอิน

1. เลือกปลั๊กอินที่ต้องการ
2. เลือกเวอร์ชั่นของปลั๊กอิน
3. เลือก network ที่ต้องการจะ install ปลั๊กอิน
4. คลิกที่ปุ่มสีเขียว

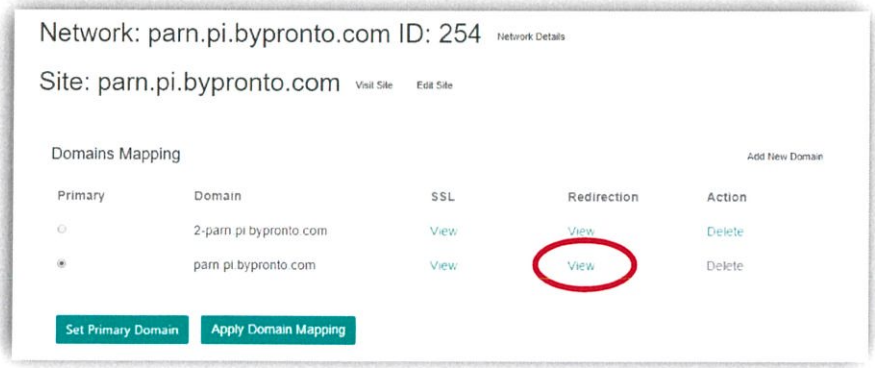
3.2.5 Domain Mapping Management



ภาพที่ 56 แสดงตัวอย่างการทำ domain mapping

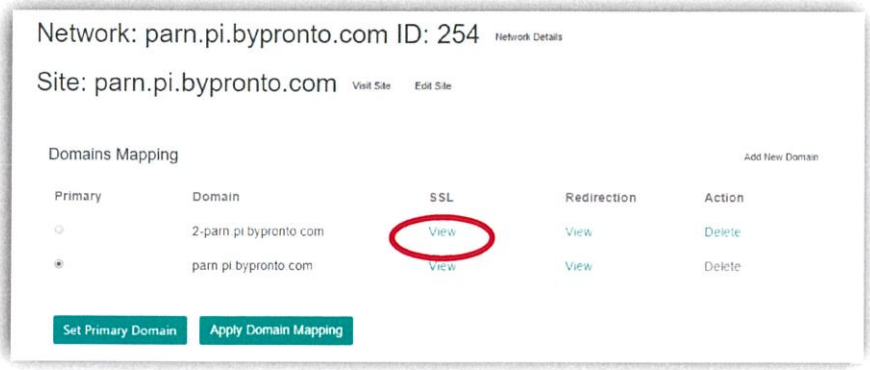
1. เพิ่มโดเมนใหม่ที่ต้องการจะ mapping โดยการกดปุ่ม “Add New Domain”
2. เลือก primary ให้กับโดเมนที่ต้องการจะให้โดเมนอื่นแมพเข้ามา
3. กดปุ่ม “Set Primary Domain”
4. กดปุ่ม “Apply Domain Mapping”

3.2.6 Redirection



ภาพที่ 57 แสดงตัวอย่างการทำ Redirection

3.2.7 SSL Certificate



ภาพที่ 58 แสดงตัวอย่างการทำ SSL Certificate

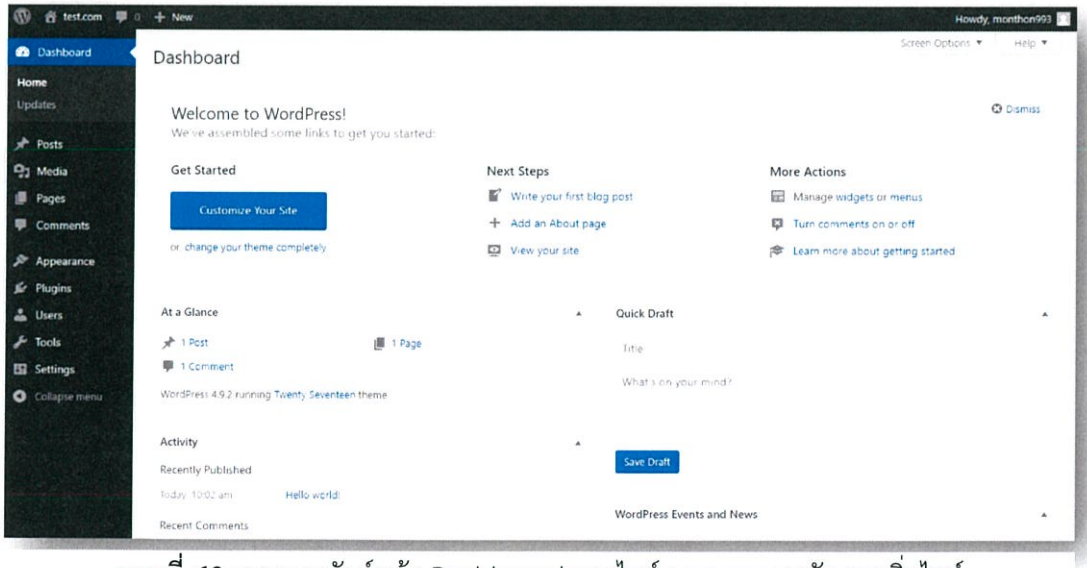
บทที่ 4

ผลการวิจัย

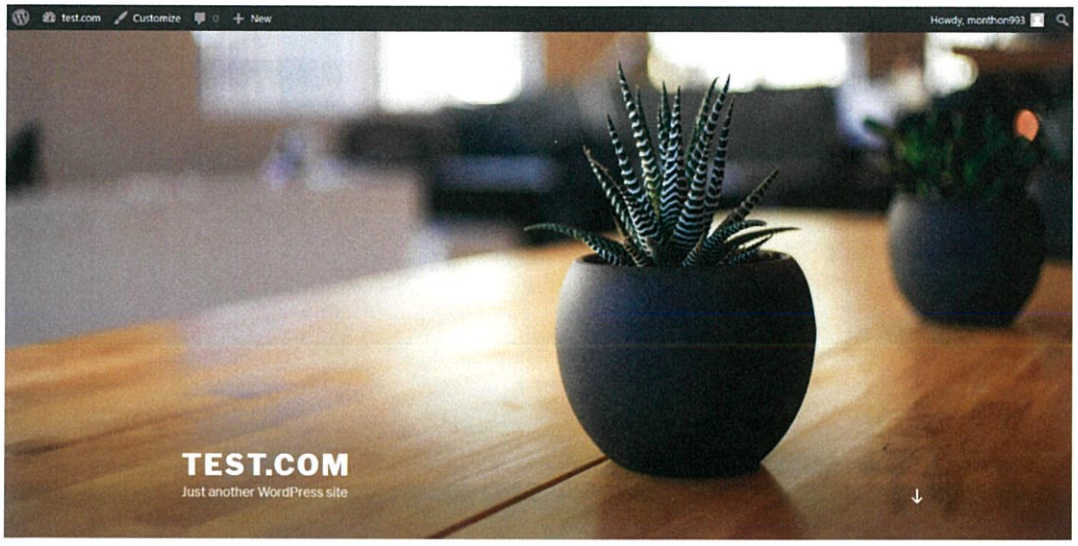
4.1 การเพิ่มไซต์ลูกค้าด้วยฟังก์ชั่นการจัดการเวิร์ดเพสมัลติไซต์

ภาพที่ 59 แสดงตัวอย่างการเพิ่มไซต์แบบมัลติไซต์

ผลลัพธ์หลังจากการเพิ่มไซต์ ระบบ back-end จะดำเนินการและได้ไซต์เวิร์ดเพสขึ้นมาในลักษณะดังนี้

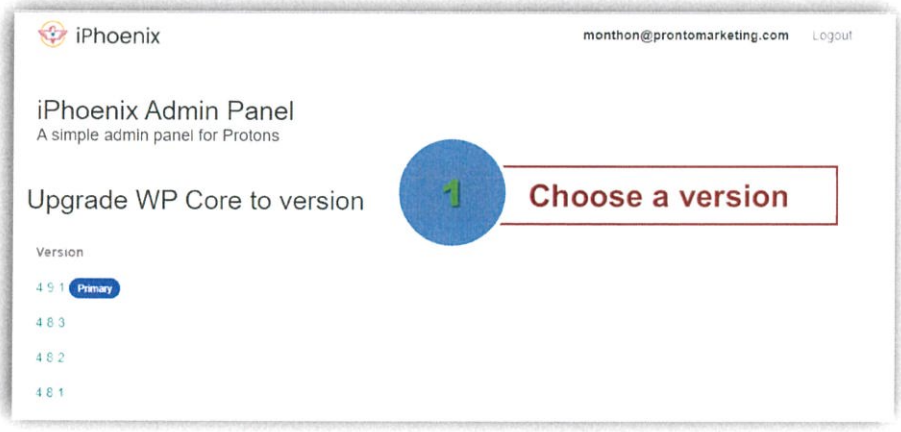


ภาพที่ 60 แสดงผลลัพธ์หน้า Dashboard ของไซต์ test.com หลังจากเพิ่มไซต์



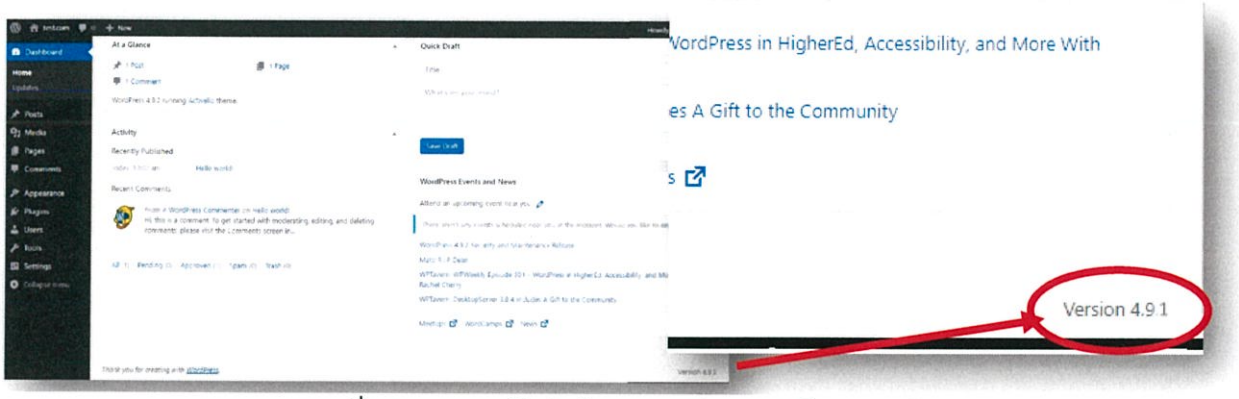
ภาพที่ 61 แสดงผลลัพธ์หน้า Home Page ของไซต์ test.com หลังจากเพิ่มไซต์

4.2 การอัพเกรดเวิร์ดเพสด้วยฟังก์ชั่นการจัดการเวิร์ดเพสคอร์



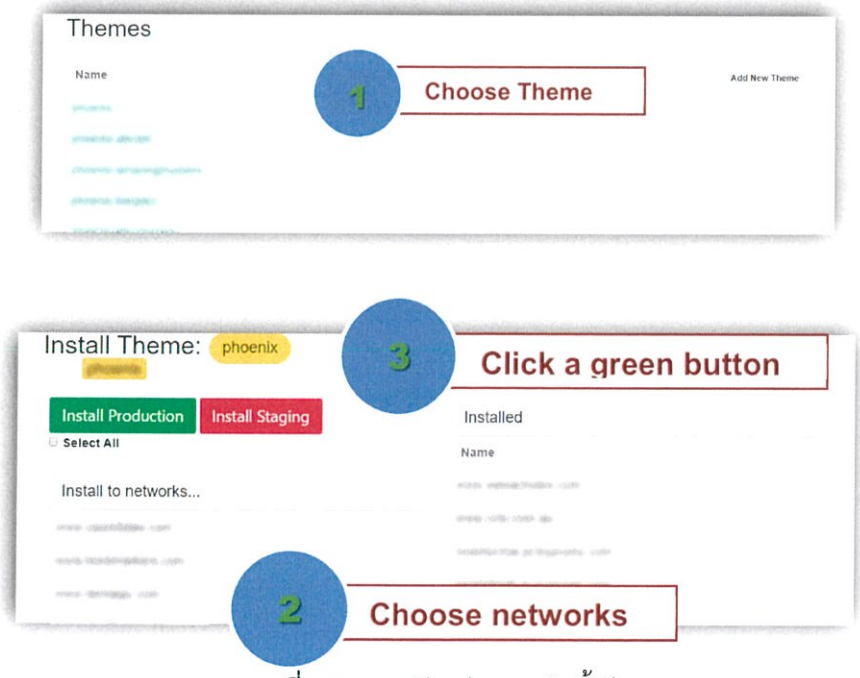
ภาพที่ 62 แสดงตัวอย่างการอัพเกรดเวิร์ดเพสคอร์

ผลลัพธ์หลังจากการเพิ่มไซต์ ระบบ back-end จะดำเนินการและได้ไซต์เวิร์ดเพสขึ้นมาในลักษณะดังนี้



ภาพที่ 63 แสดงผลลัพธ์หลังจากการอัพเกรดเวิร์ดเพสคอร์

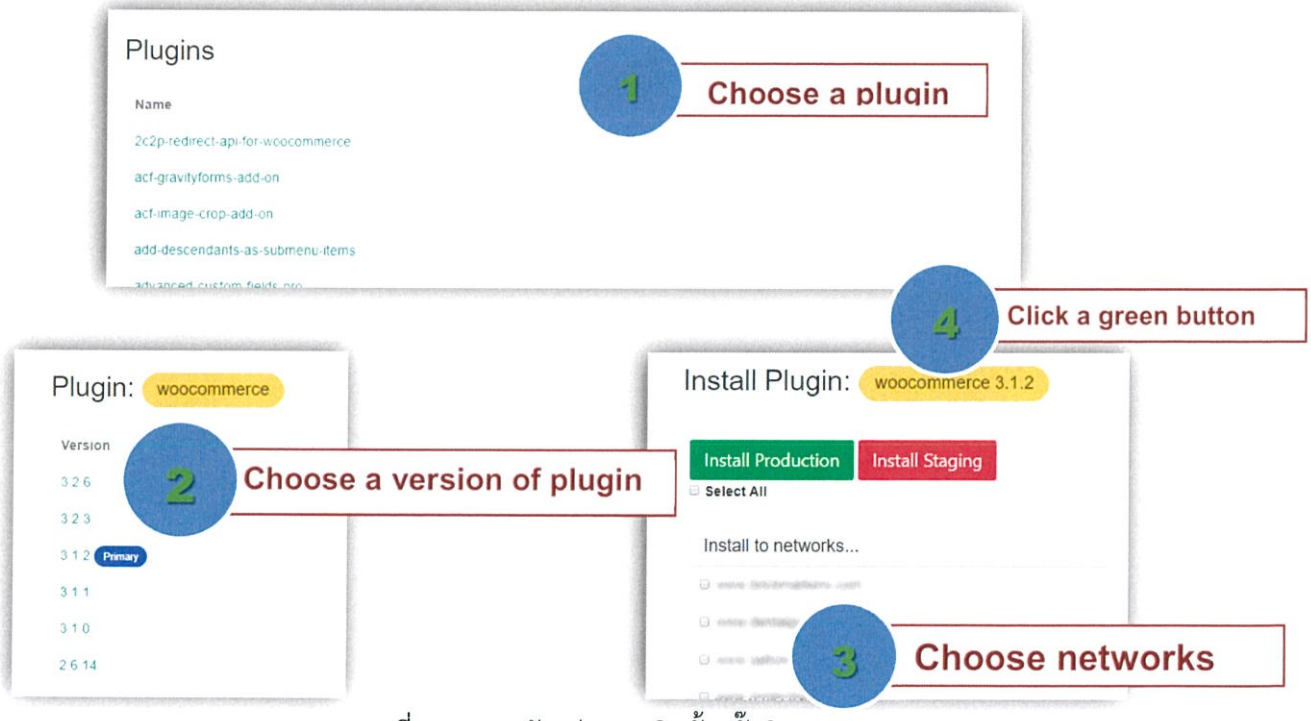
4.3 การติดตั้งธีมด้วยฟังก์ชั่นการจัดการธีม



ภาพที่ 64 แสดงตัวอย่างการติดตั้งธีม

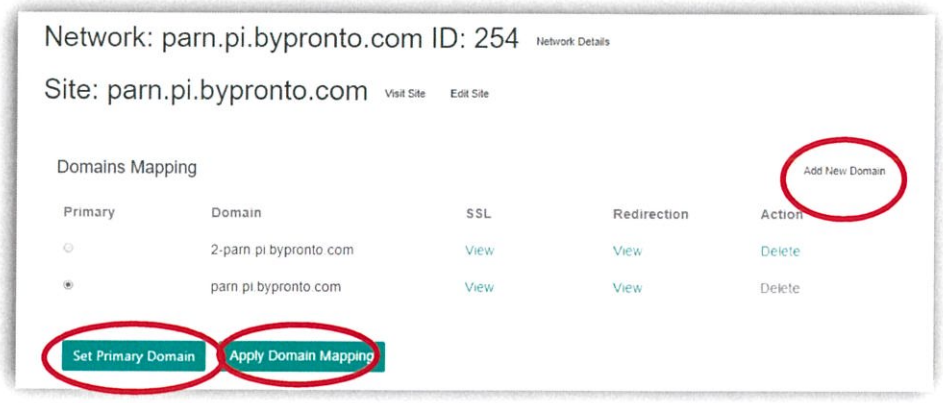
ภาพที่ 65 แสดงผลลัพธ์หลังจากการติดตั้งธีมให้กับไซต์ test.com

4.4 การติดตั้งปลั๊กอินด้วยฟังก์ชั่นการจัดการปลั๊กอิน



ภาพที่ 66 แสดงตัวอย่างการติดตั้งปลั๊กอิน

4.5 การจัดการโดเมนแมพปิ้ง

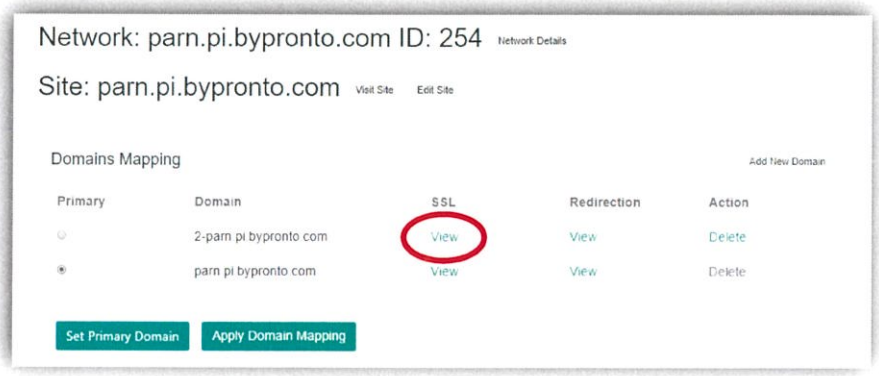


ภาพที่ 67 แสดงตัวอย่างการจัดการโดเมนแมพปิ้ง

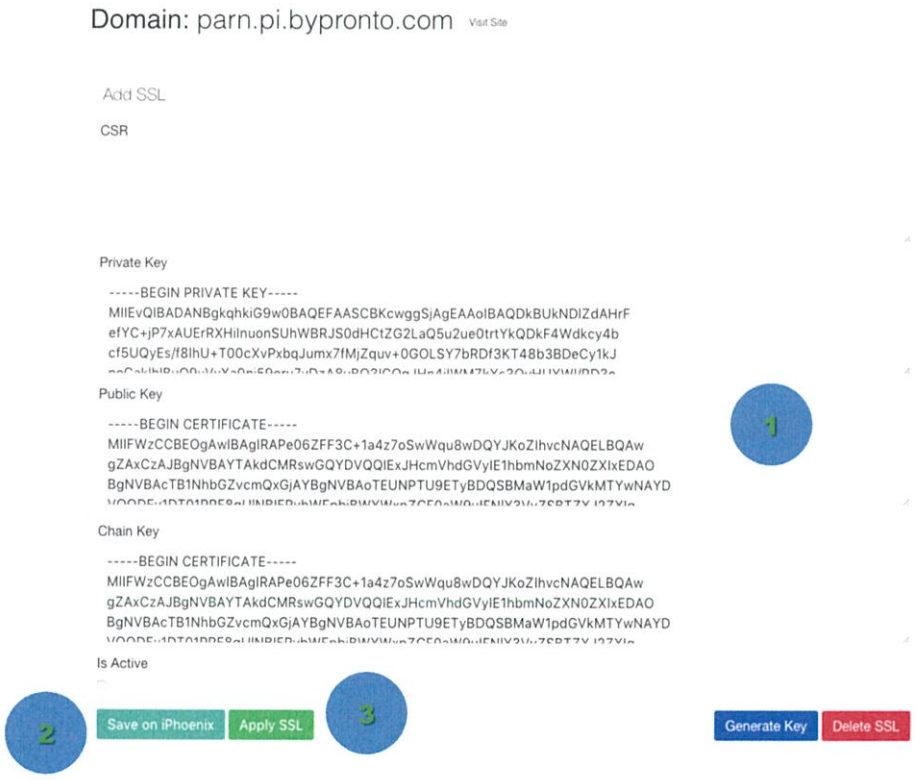
ตัวอย่างเช่น test.wordpress.com แมพไปเป็น www.test.com สามารถทำ Domain Mapping ง่ายๆ ตามขั้นตอนดังนี้

1. กดปุ่ม Add New Domain เพื่อสร้าง Domain จาก domain name ที่ซื้อมา (ใน iPhoenix จะมี domain ที่เป็น default อยู่แล้ว 1 domain)
2. เลือก Primary ที่ domain ที่เราต้องการให้เป็น domain หลัก เพื่อให้ domain อื่น map ไปหา
3. กดปุ่ม Set Primary Domain เพื่อให้ server ไปจัดการเซ็ต domain นั้นๆ ให้เป็น Primary ใน WordPress
4. กดปุ่ม Apply Domain Mapping เพื่อเซ็ตให้ทุก domain ทำการ map ไปยัง primary domain

4.6 การจัดการ SSL Certificate

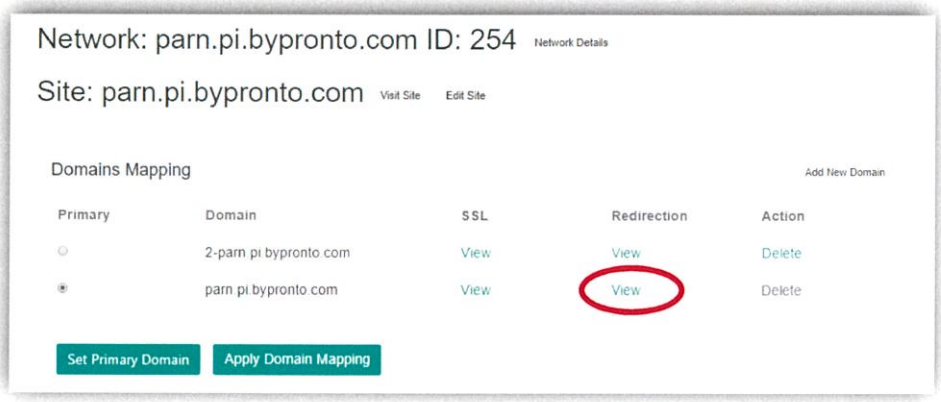


ภาพที่ 68 แสดงตัวอย่างการจัดการ SSL Certificate

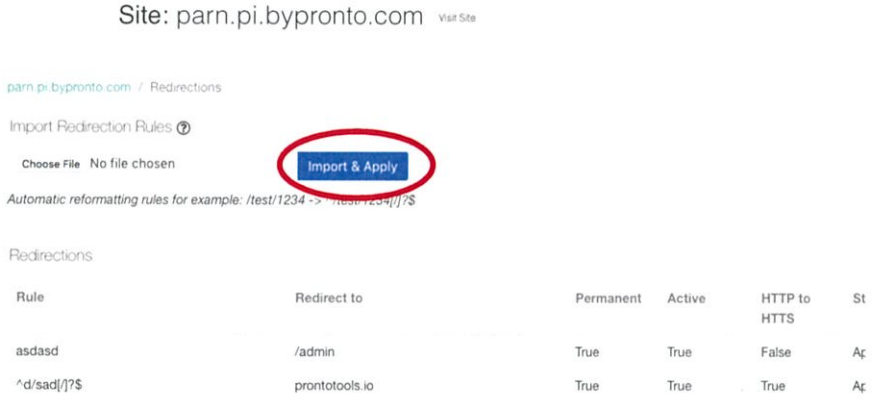


1. เพิ่มคีย์ของลูกค้าให้ถูกแต่ละช่อง
2. กดปุ่ม *Save on iPhoenix* เพื่อบันทึกข้อมูลลง iPhoenix
3. กดปุ่ม *Apply SSL* เพื่อให้คีย์ต่างๆ เพิ่มเข้าไปในไซต์ที่เราต้องการ

4.7 การจัดการรีไดเร็กชั่น



ภาพที่ 69 แสดงตัวอย่างการจัดการรีไดเร็กชั่น



เพียงทำการกดปุ่ม *Import & Apply* แล้วเลือกไฟล์ .csv ระบบก็จะทำการเพิ่ม rule เข้า iPhoenix และส่งไปยังไซต์ที่เราต้องการ

บทที่ 5

สรุปผลการวิจัย

5.1 สรุปผลการดำเนินงาน

การจัดการเวิร์ดเพสโดยอัตโนมัติผ่านเครื่องมือไอฟินิกซ์ ช่วยให้พนักงานที่ทำงานในส่วนต่างๆ ของบริษัททำงานได้ปริมาณมากและมีประสิทธิภาพ สะดวกและรวดเร็ว ไม่ต้องทำงานเดิมๆ ซ้ำๆ ลดความผิดพลาดที่จะเกิดได้

ฟังก์ชั่นการจัดการเวิร์ดเพสแบบมัลติไซต์ ช่วยให้พนักงานที่รับผิดชอบในการสร้าง แก้ไขหรือลบไซต์ลูกค้าสามารถดำเนินการได้ง่าย โดยการสร้างนั้นเพียงแค่เพิ่มชื่อไซต์ของลูกค้าผ่านหน้าอินเตอร์เฟสเพิ่มไซต์ของ iPhoenix และกดปุ่มเพิ่ม ระบบทางฝั่ง back-end จะทำงานให้ทั้งหมดเพื่อดำเนินการสร้างไซต์ขึ้นมา ซึ่งหากไม่มีฟังก์ชั่นนี้ การสร้างไซต์นั้นเป็นไปได้ยากลำบากเพราะต้องเพิ่งพาทีมอินฟราสตรักเจอ มาช่วยจัดการและใช้เวลาเป็นวัน แต่ฟังก์ชั่นนี้ใช้เวลาไม่นานเกิน 10 นาที

ฟังก์ชั่นการจัดการเวิร์ดเพสคอร์ ธีมและปลั๊กอิน ช่วยให้พนักงานที่รับผิดชอบดูแลจัดการไซต์ลูกค้านั้นทำงานได้ง่ายขึ้น ในการอัพเกรดเวอร์ชั่นของเวิร์ดเพสคอร์ การติดตั้งธีมใหม่ๆ หรือการติดตั้งปลั๊กอินให้กับไซต์ลูกค้า สามารถทำได้ทีละหลายๆ ไซต์ จำนวนมากได้ และรวดเร็ว ไม่ต้องมาติดตั้งซ้ำๆ หากหลายไซต์ต้องการธีมหรือปลั๊กอินที่เหมือนกัน ลดการทำงานซ้ำ และลดความผิดพลาด

ฟังก์ชั่นการจัดการรีไดเร็กชั่น ช่วยให้พนักงานที่รับผิดชอบในส่วนนี้ จัดการกับ rule ที่มากๆ ได้อย่างง่ายดาย สามารถเพิ่ม rule ให้กับไซต์ลูกค้าทีละจำนวนมากๆ ได้ เพียงอิมพอร์ตมากจากไฟล์ .csv ที่ลูกค้ากำหนดมาให้ช่วยลดเวลาเป็นอย่างมาก และฟังก์ชั่นนี้ช่วยตรวจสอบว่า rule ที่เพิ่มมามีความถูกต้องหรือไม่ ซึ่งลดความผิดพลาดได้เป็นอย่างมากเมื่อเทียบกับการเพิ่มเอง

เอกสารอ้างอิง

[1] "learn python" [ออนไลน์] เข้าถึงได้จาก https://www.codecademy.com/learn/learn-python (วันที่สืบค้นข้อมูล 2 มิถุนายน 2560)

[2] "บทเรียน Python สอนการเขียน ภาษา Python เบื้องต้น" [ออนไลน์] เข้าถึงได้จาก http://www.mindphp.com/บทเรียนออนไลน์/83-python.html (วันที่สืบค้นข้อมูล 2 มิถุนายน 2560)

[3] "Getting started with Django" [ออนไลน์] เข้าถึงได้จาก https://www.djangoproject.com/start/ (วันที่สืบค้นข้อมูล 10 มิถุนายน 2560)

[4] "Django documentation" [ออนไลน์] เข้าถึงได้จาก https://docs.djangoproject.com/en/2.0/ (วันที่สืบค้นข้อมูล 10 มิถุนายน 2560)

[5] "Get Docker CE for Ubuntu" [ออนไลน์] เข้าถึงได้จาก https://docs.docker.com/install/linux/docker-ce/ubuntu/ (วันที่สืบค้นข้อมูล 12 มิถุนายน 2560)

[6] "ทำความรู้จัก Docker และการใช้งานบน CentOS 7" [ออนไลน์] เข้าถึงได้จาก https://www.hostpacific.com/using-docker-on-centos7/ (วันที่สืบค้นข้อมูล 12 มิถุนายน 2560)

[7] "Web server เว็บเซิร์ฟเวอร์ nginx เอนจิ้นเอ็ก คืออะไร" [ออนไลน์] เข้าถึงได้จาก http://www.mindphp.com/คู่มือ/73-คืออะไร/3535-nginx-คืออะไร.html (วันที่สืบค้นข้อมูล 14 มิถุนายน 2560)

[8] "Quickstart: Compose and WordPress" https://docs.docker.com/compose/wordpress/ (วันที่สืบค้นข้อมูล 15 มิถุนายน 2560)

[9] "Apache Airflow (incubating) Documentation" [ออนไลน์] เข้าถึงได้จาก https://airflow.apache.org/ (วันที่สืบค้นข้อมูล 18 มิถุนายน 2560)

[10] "Airflow at Pronto" [ออนไลน์] เข้าถึงได้จาก https://www.prontotools.io/airflow-pronto/ (วันที่สืบค้นข้อมูล 18 มิถุนายน 2560)

[11] "unittest — Unit testing framework" [ออนไลน์] เข้าถึงได้จาก https://docs.python.org/3/library/unittest.html (วันที่สืบค้นข้อมูล 19 มิถุนายน 2560)

[12] "มาเขียน Unit Test ใน Python กัน" [ออนไลน์] เข้าถึงได้จาก https://thaiprogrammer.org/มาเขียน-unit-test-ใน-python-กัน/ (วันที่สืบค้นข้อมูล 25 มิถุนายน 2560)

[13] "YML วายเอ็มแอล คือ .YML ด็อทวายเอ็มแอล คือ นามสกุลของไฟล์นามสกุล" [ออนไลน์] เข้าถึงได้จาก http://www.mindphp.com/คู่มือ/73-คืออะไร/3492-yml-คือ.html (วันที่สืบค้นข้อมูล 26 มิถุนายน 2560)

[14] “Unix / Linux - Getting Started” [ออนไลน์] เข้าถึงได้จาก https://www.tutorialspoint.com/unix/unix-getting-started.htm (วันที่สืบค้นข้อมูล 29 มิถุนายน 2560)

[15] “ยูนิกซ์” [ออนไลน์] เข้าถึงได้จาก https://th.wikipedia.org/wiki/ยูนิกซ์ (วันที่สืบค้นข้อมูล 29 มิถุนายน 2560)