คู่มือการพัฒนาเว็บแอปพลิเคชั่นด้วยภาษาจาวา บนแพลตฟอร์มกูเกิ้ลแอปเอ็นจิ้น

JAVA WEB APPLICATION DEVELOPMENT MANUAL BASED ON GOOGLE APP ENGINE PLATFORM

นางสาวอังคณา โรจน์อัมพร รหัส 51362237

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร
ปีการศึกษา 2555

ใบรับรองปริญญานิพนธ์

ชื่อหัวข้อโครงงาน คู่มือการพัฒนาเว็บแอปพลิเคชั่นด้วยภาษาจาวา บนแพลตฟอร์มกูเกิ้ลแอปเอ็นจิ้น

ผู้ดำเนินโครงงาน นางสาวอังคณา โรจน์อัมพร รหัส 51362237

ที่ปรึกษาโครงงาน ดร.พงศ์พันธ์ กิจสนาโยธิน

สาขาวิชา วิศวกรรมคอมพิวเตอร์

ภาควิชา วิศวกรรมไฟฟ้าและคอมพิวเตอร์

ปีการศึกษา 2555

.....

คณะวิศวกรรมศาสตร์มหาวิทยาลัยนเรศวร อนุมัติให้ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตร์บัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์

.....ที่ปรึกษาโครงงาน
(ดร.พงศ์พันธ์ กิจสนาโยธิน)

.....กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.พนมขวัญ ริยะมงคล)

.....กรรมการ
(นางสาวจิราพร พุกสุข)

.....กรรมการ
(นายภาณุพงศ์ สอนคม)

ชื่อหัวข้อโครงงาน	คู่มือการพัฒนาเว็บแอปพลิเคชั่นด้วยภาษาจาวา บนแพลตฟอร์มกูเกิ้ลแอปเอ็นจิ้น
ผู้ดำเนินโครงงาน	นางสาวอังคณา โรจน์อัมพร รหัส 51362237
ที่ปรึกษาโครงงาน	ดร.พงศ์พันธ์ กิจสนาโยธิน
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2555

.....

บทคัดย่อ

โครงงานนี้มีวัตถุประสงค์เพื่อศึกษาคุณลักษณะต่างๆของกูเกิ้ลแอปเอ็นจิ้นซึ่งเป็นหนึ่งในเทคโนโลยีการประมวลผลแบบคลาวด์คอมพิวติ้งที่ให้บริการในรูปแบบแพลตฟอร์มสำหรับพัฒนาและเก็บแอปพลิเคชั่น พร้อมกับการนำคุณลักษณะเหล่านั้นมาใช้เพื่อพัฒนาจาวาเว็บแอปพลิเคชั่นและนำความรู้ที่ได้รับจากการศึกษาจัดทำเป็นหนังสือคู่มือการใช้งาน โดยมีเว็บแอปพลิเคชั่นที่พัฒนาขึ้นมาเป็นตัวอย่างประกอบการอธิบาย

โครงงานนี้ได้แบ่งการดำเนินการออกเป็น 2 ส่วน ส่วนแรกคือส่วนของการศึกษาและใช้งานคุณลักษณะต่างๆของกูเกิ้ลแอปเอ็นจิ้นสำหรับการพัฒนาเว็บแอปพลิเคชั่น โดยส่วนนี้จะทำการศึกษาเฉพาะคุณลักษณะที่รองรับการพัฒนาเว็บแอปพลิเคชั่นด้วยภาษาจาวาเท่านั้นและในการพัฒนาเว็บแอปพลิเคชั่นจะใช้ชุดพัฒนาของกูเกิ้ลแอปเอ็นจิ้นสำหรับภาษาจาวาเป็นตัวช่วยในการสร้างสภาพแวดล้อมเสมือนเพื่อจำลองการทำงานของแอปพลิเคชั่นในคอมพิวเตอร์ก่อนที่จะทำการอัปโหลดและเผยแพร่ไปยังกูเกิ้ลแอปเอ็นจิ้นเพื่อใช้งานจริงผ่านทางอินเทอร์เน็ต ส่วนต่อมาเป็นการจัดทำหนังสือคู่มือจากความรู้ในส่วนแรก โดยเนื้อหาของหนังสือจะมีการอธิบายคุณลักษณะหลัก 3 ประการ คือ คุณลักษณะที่ใช้ในการพัฒนาเว็บแอปพลิเคชั่น คุณลักษณะที่ใช้ในการกำหนดค่าเว็บแอปพลิเคชั่นและคุณลักษณะที่ใช้ในการดูแลจัดการเว็บแอปพลิเคชั่น นอกเหนือจากนี้ในหนังสือยังมีการเปรียบเทียบข้อดีข้อเสียในการเลือกใช้งานแพลตฟอร์มของกูเกิ้ลแอปเอ็นจิ้นเทียบกับแพลตฟอร์มอื่น และมีตัวอย่างการพัฒนาเว็บแอปพลิเคชั่นประกอบด้วย

Project title	Java Web Application Development Manual Based On Google App Engine Platform
Name	Angkana Rojamporn ID. 51362237
Project advisor	Phongphun Kijsanayothin, Ph.D.
Major	Computer Engineering
Department	Electrical and Computer Engineering
Academic year	2012

Abstract

The goal of this project is to understand the features of Google App Engine (Platform as a service provided by Google), and also to use Google App Engine to develop web application. The output of this project is the manual, which contains step-by-step for developing web application in order to explain the use of each features provided by Google App Engine API.

The project can be divided into two parts. The first part is to understand and to use the features for developing web application based on java programming language. The second part is a preparation of the manual, which describe three features: (1) development, (2) configuration, and (3) management. Moreover, the manual also includes a comparison of Google App Engine with other platforms.

กิตติกรรมประกาศ

โครงงานฉบับนี้นี้สำเร็จลุล่วงไปได้ด้วยความช่วยเหลืออย่างดียิ่งจากผู้มีอุปการะคุณหลายท่าน โดยเฉพาะอย่างยิ่งขอขอบคุณ ดร.พงศ์พันธ์ กิจสนาโยธิน อาจารย์ที่ปรึกษาโครงงานที่ได้ให้คำแนะนำและข้อคิดเห็นต่างๆของโครงงานมาโดยตลอด

ขอขอบคุณ ผู้ช่วยศาสตราจารย์ ดร.พนมขวัญ ริยะมงคล อาจารย์ภาณุพงศ์ สอนคม และอาจารย์จิราพร พุกสุขที่กรุณาสละเวลามาเป็นอาจารย์กรรมการสอบโครงงาน พร้อมทั้งให้คำแนะนำที่เป็นประโยชน์

ขอขอบคุณคณาจารย์ทุกท่านที่ได้ประสิทธิประสาทวิชาความรู้ต่างๆให้ ตลอดระยะเวลา 4 ปี เป็นความรู้ที่สามารถนำมาใช้ในการทำโครงงานนี้ และนำไปใช้ได้ต่อไปในอนาคต

สุดท้ายนี้เหนือสิ่งอื่นใด ผู้จัดทำโครงงานขอขอบพระคุณ คุณพ่อ คุณแม่ ผู้มอบความรัก ความเมตตาและเป็นกำลังใจให้เสมอมา ญาติพี่น้อง และเพื่อนๆที่คอยให้คำปรึกษา แนะนำ และติชม ผู้ดำเนินโครงงาน รวมถึงคณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร ที่ได้อนุมัติเงินค่าใช้จ่ายในการทำโครงงานนี้ด้วย

ผู้จัดทำโครงงานขอกราบขอบพระคุณเป็นอย่างสูง มา ณ โอกาสนี้

นางสาวอังคณา โรจน์อัมพร

สารบัญ

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญรูป

รูปที่ **หน้า**

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

สารบัญตาราง

ตารางที่ **หน้า**

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญ

ปัจจุบันเทคโนโลยีด้านคลาวด์คอมพิวติ้ง (Cloud Computing) เป็นเรื่องที่ใกล้ตัวและได้รับการกล่าวถึงอยู่บ่อยครั้ง ส่วนหนึ่งเป็นเพราะแนวโน้มในการประยุกต์นำคลาวด์คอมพิวติ้งมาใช้ประโยชน์ในด้านต่างๆทั้งในประเทศและต่างประเทศนั้นเริ่มมีให้เห็นเป็นรูปธรรมมากขึ้น

Google App Engine เป็นหนึ่งในเทคโนโลยีด้านคลาวด์คอมพิวติ้งที่ให้บริการในรูปแบบของแพลตฟอร์มสำหรับการพัฒนาและเก็บแอปพลิเคชั่น (Platform as a Service) ซึ่งทางกูเกิ้ลพัฒนาขึ้นมาสำหรับให้บริการพื้นที่ในการเก็บและพัฒนาเว็บแอปพลิเคชั่น มีจุดเด่นอยู่ที่การใช้งานง่ายแต่มีความยืดหยุ่นของระบบสูง จึงช่วยทำให้การพัฒนาเว็บแอปพลิเคชั่นเป็นเรื่องง่ายและช่วยให้นักพัฒนาสามารถปรับปรุงและพัฒนาขยายระบบต่อได้อย่างไม่จำกัด อีกทั้งในเรื่องค่าใช้จ่ายยังเป็นการคิดค่าใช้จ่ายแบบเป็นธรรม คือ จ่ายเท่าปริมาณที่ใช้ไปโดยเริ่มจากการใช้งานฟรีในระบบโควตาก่อน และหากต้องการการใช้งานที่มากขึ้นก็อาจจะต้องเสียค่าใช้จ่ายเพิ่มขึ้นเล็กน้อย

โครงงานนี้จึงได้มุ่งเน้นไปที่การศึกษาและใช้งานคุณลักษณะต่างๆของ Google App Engine สำหรับการพัฒนาเว็บแอปพลิเคชั่น พร้อมทั้งทำการพัฒนาเว็บแอปพลิเคชั่นขึ้นมาโดยนำคุณลักษณะต่างๆที่ได้ศึกษามาใช้งานแล้วจึงนำความรู้ที่ได้มาจัดทำเป็นหนังสือคู่มือ เพื่ออธิบายการใช้งานคุณลักษณะต่างๆ โดยคาดว่าจะก่อให้เกิดประโยชน์แก่ผู้อ่าน ทั้งในด้านความรู้และความเข้าใจในการใช้งาน Google App Engine รวมทั้งเกิดประโยชน์ในการช่วยพัฒนาทักษะในการพัฒนาเว็บแอปพลิเคชั่นโปรแกรมสำหรับใช้งานบนแพลตฟอร์มของ Google App Engine ต่อไปในอนาคต

1.2 วัตถุประสงค์

1. เพื่อศึกษาคุณลักษณะต่างๆของ Google App Engine สำหรับการพัฒนาเว็บแอปพลิเคชั่นด้วยภาษาจาวา
2. เพื่อทำการพัฒนาเว็บแอปพลิเคชั่นด้วยภาษาจาวาสำหรับทำงานบน Google App Engine
3. เพื่อศึกษาข้อดีและข้อเสียของ Google App Engine เทียบกับแพลตฟอร์มอื่น
4. เพื่อจัดทำหนังสือคู่มือการใช้งาน Google App Engine เบื้องต้น
5. สามารถนำความรู้ที่ได้ไปประยุกต์ใช้ในอนาคต

1.3 ขอบเขตของโครงงาน

1. เข้าใจถึงองค์ประกอบและการใช้งานคุณลักษณะส่วนใหญ่ของ Google App Engine
2. สามารถสร้าง ใช้งาน และจัดการเว็บแอปพลิเคชั่นบน Google App Engine platform ได้
3. เข้าใจถึงข้อดีและข้อเสียของ Google App Engine เทียบกับแพลตฟอร์มอื่น
4. สามารถนำองค์ความรู้ที่ได้จากการศึกษามาจัดทำเป็นหนังสือคู่มือการใช้งานได้

1.4 ขั้นตอนการดำเนินงาน

1. รวบรวมข้อมูลที่เกี่ยวข้องเพื่อใช้ในการศึกษาคุณลักษณะต่างๆของ Google App Engine
2. ศึกษาวิธีการในการพัฒนาแอปพลิเคชั่น ด้วยภาษาจาวาสำหรับ Google App Engine
3. ศึกษาวิธีการอิมพลีเม้นท์โค้ดโปรแกรมในภาษาจาวาสำหรับใช้งาน Google App Engine
4. ออกแบบและพัฒนาเว็บแอปพลิเคชั่นสำหรับใช้เป็นตัวอย่างประกอบการอธิบายในหนังสือ
5. นำองค์ความรู้ที่ได้จากการศึกษาจัดทำเป็นหนังสือคู่มือการใช้งาน Google App Engine
6. ตรวจสอบและแก้ไขข้อบกพร่องของโปรแกรม และหนังสือคู่มือการใช้งาน
7. สรุปและจัดทำรูปเล่มโครงงาน

1.5 แผนการดำเนินงาน

ตารางที่ 1.1 ตารางแสดงระยะเวลาแผนการดำเนินงาน

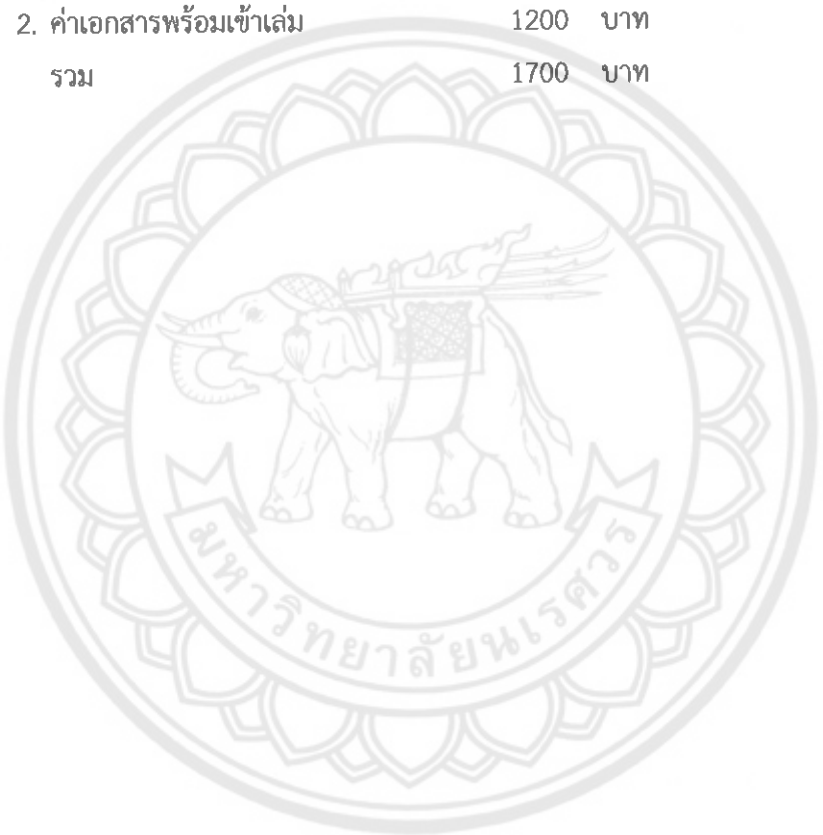
กิจกรรม	ปี 2555							ปี 2556			
	ก.ค.	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.	พ.ค.
1. รวบรวมข้อมูลที่เกี่ยวข้องเพื่อใช้ในการศึกษาคุณลักษณะต่างๆของ Google App Engine											
2. ศึกษาวิธีการในการพัฒนาแอปพลิเคชั่น ด้วยภาษาจาวาสำหรับ Google App Engine											
3. ศึกษาวิธีการอิมพลีเม้นท์โค้ดโปรแกรมในภาษาจาวาสำหรับใช้งาน Google App Engine											
4. ออกแบบและพัฒนาเว็บแอปพลิเคชั่นสำหรับใช้เป็นตัวอย่างประกอบการอธิบายในหนังสือ											
5. นำองค์ความรู้ที่ได้จากการศึกษาจัดทำเป็นหนังสือคู่มือการใช้งาน Google App Engine											
6. ตรวจสอบและแก้ไขข้อบกพร่องของโปรแกรม และหนังสือคู่มือการใช้งาน											
7. สรุปและจัดทำรูปเล่มโครงงาน											

1.6 ผลที่คาดว่าจะได้รับ

1. มีความเข้าใจในคุณลักษณะต่างๆของ Google App Engine
2. สามารถพัฒนาเว็บแอปพลิเคชั่นด้วยภาษาจาวาสำหรับทำงานบน Google App Engine ได้
3. มีความเข้าใจถึงข้อดีและข้อเสียของการใช้งานแพลตฟอร์ม Google App Engine
4. สามารถนำองค์ความรู้จากการศึกษามาจัดทำเป็นหนังสือคู่มือได้

1.7 งบประมาณของโครงการ

1. ค่าวัสดุที่ใช้ในการดำเนินโครงงาน	500	บาท
2. ค่าเอกสารพร้อมเข้าเล่ม	1200	บาท
รวม	1700	บาท



บทที่ 2
หลักการและทฤษฎีเบื้องต้น

ก่อนที่จะสามารถเข้าใจคุณลักษณะต่างๆของ Google App Engine นั้นจำเป็นต้องมีความรู้พื้นฐานในหลายๆส่วนประกอบกัน ไม่ว่าจะเป็นความรู้พื้นฐานในเรื่องของเทคโนโลยีคลาวด์คอมพิวติ้ง หรือความรู้พื้นฐานเกี่ยวกับเว็บแอปพลิเคชั่น รวมไปถึงการใช้งานเทคโนโลยีการประมวลผลทางฝั่งเซิร์ฟเวอร์ เช่น Java Servlet และ JSP เป็นต้น

เพื่อให้สามารถเข้าใจถึงคุณลักษณะต่างๆของ Google App Engine ได้โดยง่าย และเพื่อให้สามารถพัฒนาเว็บแอพพลิเคชันด้วยภาษาจาวาสำหรับทำงานบน Google App Engine ได้นั้น จำเป็นต้องมีความรู้และทราบหลักการที่เกี่ยวข้องเสียก่อน ซึ่งหลักการและทฤษฎีที่สำคัญทั้งหมดได้อธิบายไว้ดังต่อไปนี้

2.1 Cloud Computing [1]

Cloud Computing (การประมวลผลแบบกลุ่มเมฆ) เป็นลักษณะการให้บริการแบบหนึ่งแก่ผู้ใช้งานผ่านทางอินเตอร์เน็ท โดยผู้ให้บริการจะให้ทรัพยากรส่วนหนึ่งแก่ผู้ใช้บริการในการใช้งานด้านต่างๆ มีแนวคิดมาจากการดึงสมรรถนะและประสิทธิภาพในการประมวลผลของคอมพิวเตอร์หลายๆ เครื่องที่อยู่ต่างสถานที่ มาใช้ในการทำงานประสานกัน ซึ่งเป็นลักษณะที่พัฒนาต่อจากแนวความคิดและการให้บริการของ Virtualization ซึ่งเป็นการจำลองเครื่องเสมือนด้วยซอฟแวร์และ Web Services ซึ่งเป็นซอฟต์แวร์ที่ช่วยในการแลกเปลี่ยนข้อมูลระหว่างคอมพิวเตอร์ผ่านระบบอินเตอร์เน็ต โดยผู้ใช้งานไม่จำเป็นต้องมีความรู้ในเชิงเทคนิคสำหรับพื้นฐานการทำงานนั้นแต่อย่างใด

จุดเด่นของ Cloud Computing อยู่ที่เทคโนโลยีเวอร์ช่วลไลเซชั่น (Virtualization) ซึ่งทำหน้าที่เป็นตัวกลางในการสั่งการ บริหารจัดการ และจัดสรรทรัพยากรของเครื่องคอมพิวเตอร์เหล่านั้น ทั้งหน่วยความจำ ฮาร์ดดิสก์ รวมไปถึงซีพียู เพื่อกระจายงานการประมวลผลให้ พอทำงานเสร็จจึงส่งงานกลับมาให้ทางส่วนที่เป็นเทคโนโลยีเวอร์ช่วลไลเซชั่นรวบรวมงานที่ทำการประมวลผลแล้วอีกทีหนึ่ง ซอฟต์แวร์แอพพลิเคชั่นต่างๆ จึงไม่ต้องรับภาระในเรื่องการเลือกสั่งว่าจะให้เครื่องคอมพิวเตอร์เครื่องใดช่วยประมวลผลให้ และในมุมมองของเทคโนโลยีเวอร์ช่วลไลเซชั่นนั้นถือว่าสามารถคุมฮาร์ดแวร์ได้ทุกประเภท

ผู้ให้บริการ Cloud Computing ส่วนใหญ่จะให้บริการในลักษณะของเว็บแอปพลิเคชั่น โดยให้ผู้ใช้ทำงานผ่านเว็บบราวเซอร์ ขณะที่ซอฟต์แวร์และข้อมูลทั้งหมดถูกเก็บไว้บนเซิร์ฟเวอร์ของผู้

ให้บริการ ตัวอย่างของ Cloud Computing ที่เป็นที่รู้จัก เช่น YouTube หรือระบบเครือข่ายสังคมออนไลน์ต่างๆ รวมไปถึง Google App Engine เป็นต้น

2.1.1 โครงสร้างของการประมวลผลแบบกลุ่มเมฆ [2]

โครงสร้างของการประมวลผลแบบก้อนเมฆมีอยู่ด้วยกันหลายส่วน สามารถแบ่งออกเป็นส่วนหลักได้ 6 ส่วน ดังนี้

1. กลุ่มเมฆเซิร์ฟเวอร์ (Cloud Server) เป็นเซิร์ฟเวอร์จำนวนมหาศาลนับแสนเครื่องที่ตั้งอยู่ในที่เดียวกันและต่อเชื่อมเข้าหากันด้วยเครือข่ายเป็นระบบกริดในระบบนี้จะใช้ซอฟต์แวร์เวอร์ช่วล-ไลเซชั่นในการทำงานเพื่อให้โปรแกรมประยุกต์ทำงานขึ้นกับระบบน้อยที่สุด

2. ส่วนติดต่อกับผู้ใช้ (User interaction interface) ซึ่งเป็นส่วนที่ติดต่อกับผู้ใช้นี้จะทำหน้าที่รับคำขอบริการ (Request) จากผู้ใช้ในรูปแบบของเว็บโปรโตคอล

3. ส่วนจัดเก็บรายการบริการ (Services Catalog) ทำหน้าที่เก็บและบริหารรายการของบริการ ผู้ใช้สามารถค้นดูบริการที่มีจากที่นี่

4. ส่วนบริหารงาน (system management) ทำหน้าที่กำหนดทรัพยากรที่เหมาะสมเมื่อผู้ใช้เรียกใช้บริการ เมื่อมีการขอใช้บริการ ข้อมูลการขอ (Request) จะถูกส่งผ่านให้ส่วนนี้

5. ส่วนจัดหาทรัพยากร (provisioning services) ส่วนบริหารงานจะติดต่อกับส่วนนี้เพื่อจองทรัพยากรจากกลุ่มเมฆและเรียกใช้โปรแกรมประยุกต์แบบเวบที่เหมาะสมให้ เมื่อโปรแกรมประยุกต์ทำงานแล้วก็จะส่งผลที่ได้ให้ผู้ใช้ที่เรียกใช้บริการต่อไป

6. ส่วนตรวจสอบข้อมูลการใช้งาน (Monitoring and Metering) เป็นส่วนที่ใช้เก็บข้อมูลการใช้งานเพื่อใช้ในการเก็บค่าบริการหรือเก็บข้อมูลสถิติเพื่อปรับปรุงระบบต่อไป

2.1.2 รูปแบบของการให้บริการบนระบบ Cloud Computing [3]

การให้บริการบนระบบ Cloud Computing นั้นสามารถแบ่งออกตามรูปแบบของชั้นของการให้บริการได้ ดังนี้

1. การให้บริการซอฟต์แวร์หรือ Software as a Service (SaaS) เป็นการให้บริการการประมวลผลแอปพลิเคชันที่แม่ข่ายของผู้ให้บริการและเปิดให้การบริการทางด้านซอฟแวร์ต่างๆ

2. การให้บริการแพลทฟอร์มหรือ Platform as a Service (PaaS) เป็นการให้บริการการประมวลผล ซึ่งมีระบบปฏิบัติการและการสนับสนุนเว็บแอปพลิเคชันเข้ามาร่วมด้วย

3. การให้บริการโครงสร้างพื้นฐานหรือ Infrastructure as a Service (IaaS) เป็นการให้บริการเฉพาะโครงสร้างพื้นฐาน มีประโยชน์ในการประมวลผลทรัพยากรจำนวนมาก

4. บริการระบบจัดเก็บข้อมูล หรือ data Storage as a Service (dSaaS) ทำให้สามารถรองรับระบบการจัดเก็บข้อมูลที่มีขนาดใหญ่ไม่จำกัด รองรับการสืบค้นและการจัดการข้อมูลขั้นสูง

5. บริการร่วมรวมลำดับความเชื่อมโยงหรือ Composite Service (CaaS) เป็นการให้บริการในส่วนซึ่งทำหน้าที่รวมโปรแกรมประยุกต์หรือจัดลำดับการเชื่อมโยงแบบ Workflow ข้ามเครือข่าย รวมถึงการทำหน้าที่ในการจัดการความปลอดภัยด้วย

2.1.3 ประโยชน์ของ Cloud Computing [4]

ระบบประมวลผลแบบกลุ่มเมฆนั้นถือได้ว่ามีประโยชน์ต่อผู้ใช้มากมายหลายประการและสามารถจำแนกออกเป็น 4 เรื่องหลักได้ ดังนี้

1. ทำให้การสร้างระบบคอมพิวเตอร์สามารถตั้งอยู่ที่ใดก็ได้ เพราะการที่มีระบบการประมวลผลแบบกลุ่มเมฆเกิดขึ้น ทำให้ไม่จำเป็นต้องทราบถึงสถานที่ตั้งของระบบคอมพิวเตอร์ ระบบจึงสามารถตั้งอยู่ที่ใดก็ได้ เพียงแต่ผู้ใช้ต้องมีเครือข่ายที่มีแบนด์วิทธ์เพียงพอในการเข้าถึงเท่านั้น ทำให้สามารถย้ายระบบคอมพิวเตอร์ไปก่อสร้างศูนย์ข้อมูลที่มีค่าใช้จ่ายต่ำได้ เช่น บริเวณที่มีที่ดินราคาถูกอยู่ใกล้โรงไฟฟ้าหรือแหล่งพลังงานราคาถูก เป็นต้น

2. ทำให้เกิดการใช้งานคอมพิวเตอร์อย่างมีประสิทธิภาพ เนื่องจากระบบคลาวด์คอมพิวติ้งมีการแบ่งสรรการใช้งานระหว่างผู้ใช้จำนวนมหาศาล ทำให้สามารถออกแบบระบบที่ไม่ต้องเผื่อการใช้งานที่รับงานหนักไว้มากนัก และยังสามารถเพิ่มจำนวนเครื่องได้ง่ายเมื่อมีความต้องการสูงขึ้น

3. เมื่อเกิดการปรับโครงสร้างระบบและเครือข่าย ผู้ใช้ได้จะรับผลกระทบน้อยมาก เนื่องจากระบบประมวลผลแบบกลุ่มเมฆนั้น มีการการแยกการบำรุงรักษาโครงสร้างด้านระบบและเครือข่าย ออกจากการบำรุงรักษาระบบโปรแกรมประยุกต์ ทั้งนี้เป็นผลจากแนวคิดเวอร์ชวลไลเซชั่นและไดนามิคโปรวิชั่นนิ่งที่คุมการจัดสรรทรัพยากรไอทีตามความต้องการ ที่สำคัญการแยกส่วนของการติดต่อผู้ใช้ออกด้วยเทคโนโลยีเว็บ 2.0 ทำให้ผู้ใช้รับผลกระทบจากการปรับโครงสร้างระบบน้อยมาก

4. ความมีสเถียรภาพสูงขึ้น เนื่องจากระบบประมวลผลแบบกลุ่มเมฆทำให้เกิดการแยกระหว่างการโปรแกรมและระบบที่ใช้รันงาน จึงสามารถเคลื่อนย้ายโปรแกรมไปบนระบบได้ ผลที่ตามมาคือ สามารถที่จะรันโปรแกรมได้หลายชุดและสามารถย้ายโปรแกรมออกจากระบบคอมพิวเตอร์มที่เกิดปัญหาได้ง่าย ทำให้เกิดความมีเสถียรภาพสูงขึ้น

5. การประหยัดค่าใช้จ่ายในการซื้อเครื่องเซิร์ฟเวอร์ เนื่องมาจากแนวคิดในข้อที่ 4 ทำให้เกิดแนวคิดที่ตามมาคือ แนวคิดในการเช่าซื้อพลังในการประมวลผล เนื่องจากงานการประมวลผลที่หนักๆจะไม่เกิดขึ้นบ่อยนัก หากเทียบความคุ้มค่าระหว่างการเช่าซื้อกับการซื้อเซิฟเวอร์มาใช้เอง จะเห็นได้ว่าการมีเซิร์ฟเวอร์ไว้เองนั้นจะทำให้เกิดการสิ้นเปลืองงบประมาณ เพราะนานๆทีจึงจะได้ใช้งานเซิฟเวอร์อย่างเต็มความสามารถ อีกทั้งการบำรุงรักษา ซ่อมแซม รวมทั้งพัฒนาปรับเปลี่ยนขยายขีดความสามารถของ

เครื่องเซิฟเวอร์ ทั้งการเพิ่มหน่วยความจำ ขยายซีพียู ไปจนถึงปรับเปลี่ยนซอฟแวร์ใหม่ๆ ก็ทำให้เกิดค่าใช้จ่ายต่อเนื่องตามมามากมาย ฉะนั้นการเช่าซื้อระบบเพื่อมาทำงานที่เกิดไม่บ่อยนัก หรืองานหนักที่นานๆจะเกิดขึ้นสักครั้งหนึ่ง จึงเป็นหนทางที่ช่วยในการประหยัดงบประมาณได้มากทีเดียว

2.1.4 เว็บแอปพลิเคชั่น (Web Application) [5]

ผู้คนส่วนมากมักจะคุ้นเคยกับ Desktop Application ซึ่งก็คือแอปพลิเคชั่นหรือโปรแกรมคอมพิวเตอร์ที่ติดตั้งไว้สำหรับใช้งานบนเครื่องคอมพิวเตอร์ส่วนบุคคล เช่น โปรแกรมจำพวก Microsoft Office ที่ใช้พิมพ์เอกสารหรือโปรแกรมมัลติมีเดียอย่าง Winamp เป็นต้น โดยโปรแกรมเหล่านี้จำเป็นต้องได้รับการติดตั้งลงบนเครื่องคอมพิวเตอร์ของก่อนการใช้งานและในเวลาใช้งานนั้นจะสามารถจะใช้งานได้เพียงทีละคนเท่านั้น

หากเป็นพนักงานทำงานอยู่ตามบริษัทต่างๆ ก็อาจจะคุ้นเคยกับโปรแกรมที่บริษัทใช้ เช่น ERP, MRP หรือโปรแกรมห้องสมุด ซึ่งโปรแกรมพวกนี้มักจะเป็นโปรแกรมแบบ Client – Server คือ โปรแกรมที่ใช้งานโดยคนหลายคนพร้อมกัน มีการเก็บข้อมูลไว้ที่ฐานข้อมูลส่วนกลาง ทำให้ทุกคนใช้ข้อมูลเดียวกันร่วมกันได้ โปรแกรมดังกล่าวจะถูกแบ่งออกเป็นสองส่วน คือ ส่วนที่ติดตั้งไว้ที่ Server และส่วนที่ติดตั้งไว้ในคอมพิวเตอร์ของผู้ใช้ หรือที่เรียกว่า Client ซึ่งทั้งสองส่วนนี้จะทำงานร่วมกันโดยโปรแกรมบน Server จะรับหน้าที่ในการทำงานหลักที่จำเป็นต้องใช้พลังในการประมวลผลสูง เช่น การคำนวน การค้นหาข้อมูล การเก็บข้อมูล เป็นต้น ส่วนโปรแกรมที่ติดตั้งที่คอมพิวเตอร์ หรือที่เรียกว่า Client นั้นจะทำหน้าที่นำเสนอข้อมูล และรับข้อมูลจากผู้ใช้ หรือที่เรียกว่าเป็น User Interface นั่นเอง

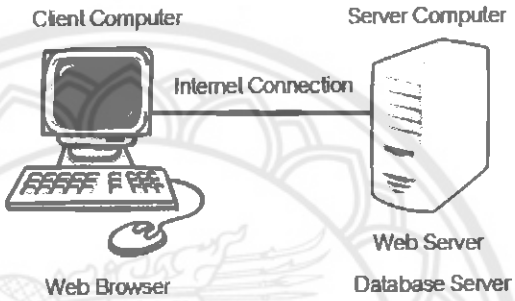
จากที่กล่าวมาข้างต้นนั้น จะเห็นว่าโปรแกรมแบบนี้ซับซ้อนและดูแลยาก เพราะเมื่อใดที่ทำการอัปเกรดโปรแกรมที่ Server เมื่อนั้นก็จำเป็นที่จะต้องทำการอัปเกรดโปรแกรมในส่วนของ Client ด้วยเช่นกัน ซึ่งจะเห็นได้ว่าเป็นเรื่องที่ยุ่งยากและดูแลจัดการได้ลำบากเพราะ Client นั้นมีหลายเครื่อง ทำให้ยากที่จะทำการอัปเกรดได้ครบ

ในระยะหลังมานี้ จึงมีโปรแกรมอีกประเภทหนึ่งได้รับความนิยมมากขึ้น และเป็นที่รู้จักกันอย่างแพร่หลาย นั่นก็คือ เว็บแอปพลิเคชั่น (Web Application) ซึ่งอาจเรียกได้ว่าเป็นแอปพลิเคชั่นที่สามารถใช้งานได้ผ่านเว็บบราวเซอร์ (Web Browser) นั่นเอง โดยที่เว็บแอปพลิเคชั่นนั้นเป็นโปรแกรมที่ติดตั้งที่ฝั่งเซิร์ฟเวอร์ (Server) และสามารถใช้งานแทนโปรแกรมทั้งแบบ Desktop และแบบ Client - Server ได้เป็นอย่างดี เช่น Google Application ซึ่งนำมาใช้แทน Microsoft Office โดยมีทั้ง Word Processor ที่ใช้แทน Microsoft Word และ Spread Sheet ที่ใช้แทน Excel ด้วย อีกทั้งในปัจจุบันนี้โปรแกรมแบบ Client-Server หลายตัวเช่น SAP, Lotus Notes ฯลฯ ก็กำลังแปลงตัวเป็นเว็บแอปพลิเคชั่นเพื่อตอบสนองความต้องการของผู้ใช้เช่นกัน เพราะเว็บแอปพลิเคชั่นนั้นมีข้อดีคือการไม่จำเป็นต้องใช้ Client Program ทำให้ไม่ต้องทำการอัปเกรด Client Program บ่อยๆและสามารถ

ใช้ผ่านการเชื่อมต่ออินเตอร์เน็ต (Internet Connection) ที่มีความเร็วต่ำกว่าได้ทำให้สามารถใช้โปรแกรมได้จากทุกพื้นที่บนโลกได้นั่นเอง

ส่วนประกอบของเว็บพลิเคชั่น [6]

ส่วนประกอบพื้นฐานของเว็บแอปพลิเคชั่นนั้นแบ่งตามลักษณะการทำงานออกได้เป็น 2 ส่วน คือ ส่วนของไคลเอนต์ (Client) และส่วนของเซิฟเวอร์ (Server) โดยทั้ง 2 ส่วนนี้จะเชื่อมต่อกันผ่านทางเครือข่ายอินเตอร์เน็ต (หรืออินทราเน็ต) แสดงดังรูป 2.1



รูปที่ 2.1 แสดงส่วนประกอบของเว็บแอปพลิเคชั่น

การเข้าถึงเว็บแอปพลิเคชั่นจะใช้เว็บบราวเซอร์โดยรันผ่านคอมพิวเตอร์ฝั่ง Client ซึ่งบราวเซอร์จะทำการเปลี่ยนโค้ด HTML ให้เป็นหน้าตาของจอภาพตามโค้ดที่เขียนไว้ ซึ่งโปรแกรมเว็บบราวเซอร์ที่เป็นที่นิยมใช้งานก็ได้แก่ Internet Explorer และ Google Chrome ในขณะที่โปรแกรมที่นิยมใช้งานเพื่อรันเว็บแอปพลิเคชั่นทางฝั่งของ server หรือที่เรียกกันว่า Web Server นั้นก็ได้แก่ Apache และ IIS เป็นต้น

โปรโตคอล HTTP

ในการติดต่อแลกเปลี่ยนข้อมูลระหว่างเซิฟเวอร์ (Server) และไคลเอนต์ (Client) ของเวิร์ลไวด์เว็บ (WWW) จำเป็นต้องใช้โปรโตคอลในการสื่อสารระหว่างกัน ซึ่งโปรโตคอลหลักที่ใช้นั้นก็คือ โปรโตคอล HTTP (Hypertext Transfer Protocol) โปรโตคอล HTTP เป็นโปรโตคอลที่มีการทำงานอยู่ในชั้น Application ของโปรโตคอล TCP/IP โดยจะเป็นการทำงานในลักษณะของสถาปัตยกรรม Client-Server กล่าวคือเมื่อ Client (คือเว็บบราวเซอร์) ต้องการติดต่อหรือขอข้อมูลจากเซิร์ฟเวอร์ (คือ Web Server) จะมีการส่งคำร้องขอหรือที่เรียกว่า Request ไปยังเซิร์ฟเวอร์ เมื่อเซิร์ฟเวอร์ได้รับการร้องขอ (Request) จะทำการประมวลผลแล้วส่งการตอบสนองหรือที่เรียกว่า Response กลับไป

ยังเครื่อง Client ที่เรียกเข้ามา โดยการติดต่อกันระหว่าง เว็บบราวเซอร์และ Web Server นี้จะทำงานบนโปรโตคอล HTTP ซึ่งโปรโตคอล HTTP จะมีการทำงานในลักษณะ Stateless คือไม่จดจำสถานะของการติดต่อที่มีเข้ามา โดยทางด้าน Server จะไม่ดูแลความถูกต้องของข้อมูลที่ส่งกลับไปหา Client และไม่จดจำว่าการร้องขอ (Request) ต่างๆที่เข้ามานั้นเป็นของ Client ใด ค่า Response ที่ Server ส่งกลับไปให้ส่วนใหญ่จะอยู่ในลักษณะไฟล์Text หรือ Binary ทั้งนี้ทั้งนั้นขึ้นอยู่กับลักษณะของการร้องขอ (Request) ด้วย ซึ่งการร้องขอ (Request) ก็มีหลายรูปแบบด้วยกันสังเกตได้จากเอกสาร HTML ในส่วนของ Tag Form หรือสังเกตได้จาก URL ของเว็บบราวเซอร์ โดยรูปแบบที่นิยมใช้งานจะมีอยู่ 2 เมธอดด้วยกันคือ

1. เมธอด GET

เป็นการร้องขอ (Request) ที่มีการใช้งานบ่อยครั้งที่สุด ซึ่งเป็นลักษณะการร้องขอจาก Client ที่ต้องการข้อมูลแบบ Static เช่น เอกสาร HTML, ไฟล์รูปภาพ เป็นต้น โดยเมธอดนี้จะมีลักษณะการส่งการร้องขอจาก Client ดังนี้

```
http://www.test.com/test.jsp?id=001&name=boy
```

รูปที่ 2.2 แสดงตัวอย่างเมธอด GET

จากตัวอย่าง หลังเครื่องหมาย ? จะเป็นการส่งพารามิเตอร์ "id" ที่มีค่าเท่ากับ "001" และพารามิเตอร์ "name" ที่มีค่าเท่ากับ "boy" ไปให้กับไฟล์ test.jsp ใน เว็บบราวเซอร์ เพื่อนำค่าต่างๆ เหล่านี้ส่งให้กับโปรแกรมที่ทำงานอยู่เบื้องหลัง เช่น CGI,ASP,PHP หรือ JSP นำไปประมวลผล และส่งค่ากลับมาในรูปแบบของเอกสาร HTML ให้กับ Client ต่อไป

2. เมธอด POST

เป็น Request ที่มีไว้เพื่อให้ Client สามาระเข้าถึงข้อมูลในลักษณะ Dynamic ทางฝั่ง Server ได้โดยการส่งในลักษณะนี้สามารถส่งข้อมูลได้จำนวนมาก ในลักษณะที่เป็น Multi-Part Message เช่น การ Upload Text ไฟล์ หรือไฟล์รูปภาพไปยัง Server การส่งแบบ Post มีลักษณะการส่งดังนี้

```
<form method= "POST" action= "test.jsp">
```

รูปที่ 2.3 แสดงตัวอย่างการใช้งานเมธอด POST

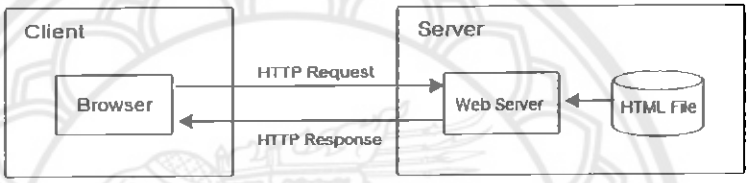
จากตัวอย่างเป็นการส่งข้อมูลแบบ Post โดยการกำหนดค่าพารามิเตอร์ method ให้เท่ากับ POST และเป็นการส่งข้อมูลให้กับไฟล์ test.jsp ผ่านทางพารามิเตอร์ action

สรุปข้อแตกต่างระหว่างเมธอด GET และ เมธอด POST

ข้อแตกต่างของการส่งข้อมูลด้วยเมธอด Post และเมธอด Get คือการส่งข้อมูลแบบ Post จะไม่แสดงข้อมูลที่ส่งไปและส่งข้อมูลได้มากกว่า ส่วนการส่งข้อมูลแบบ Get จะแสดงข้อมูลที่ส่งและถูกจำกัดจำนวนของข้อมูลที่ส่ง

หลักการทำงานของเว็บแอปพลิเคชั่นแบบ Static และ Dynamic

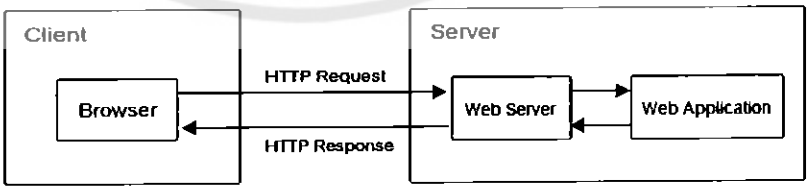
เว็บแอปพลิเคชั่นแบบ Static เป็นการทำงานของเว็บแอปพลิเคชั่นที่ไม่มีการเปลี่ยนแปลงข้อมูล มีขั้นตอนการทำงานดังรูป 2.2



รูปที่ 2.4 ขั้นตอนการทำงานของเว็ปแอปพลิเคชั่นแบบ Static

จากรูปที่ 2.2 เริ่มแรกผู้ใช้งาน (Client) จะทำการร้องขอ (HTTP request) เพจที่ต้องการผ่านทางบราวเซอร์ไปยัง web Server เมื่อ web server ได้รับการร้องขอ เซิร์ฟเวอร์จะส่งไฟล์ HTML ที่เก็บไว้กลับไปให้กับบราวเซอร์ จากนั้นบราวเซอร์จะนำผลตอบสนองการร้องขอหรือผลลัพธ์ (HTTP response) ออกมาแสดงในรูปแบบของเอกสาร HTML

1.เว็บแอปพลิเคชั่นแบบ Dynamic เป็นการทำงานของเว็บแปพลิเคชั่นที่มีการเปลี่ยนแปลงของข้อมูล มีขั้นตอนการทำงานดังรูป 2.3



รูปที่ 2.5 ขั้นตอนการทำงานของเว็บแอปพลิเคชั่นแบบ Dynamic

จากรูปที่ 2.3 เริ่มแรกถ้าผู้ใช้งานมีการส่งข้อมูลเข้าไปในเว็บเพจ หรือคลิกปุ่มต่างๆ บราวเซอร์จะทำการส่งคำร้อง (Request) ไปยังเซิร์ฟเวอร์ เมื่อเว็บเซิร์ฟเวอร์ได้รับการร้องขอ จะทำการตรวจสอบว่าคำร้องขอนั้นเป็นคำร้องขอแบบ dynamic ใช่หรือไม่ ถ้าใช่จะส่งการทำงานต่อไปยังเว็บแอปพลิเคชั่น เมื่อเว็บแอปพลิเคชั่นได้รับการร้องขอจะทำการประมวลผลข้อมูลที่ผู้ใช้งานร้องขอมาและเปลงเป็นเอกสาร HTML จากนั้นจะส่งเอกสารที่ได้กลับไปยัง Web Server ซึ่งจะส่งการตอบสนอง (HTTP request) กลับไปยังบราวเซอร์เพื่อแสดงผลออกมาในรูปของเอกสาร HTML เป็นลำดับสุดท้าย

2.1.5 Services หรือ API [6]

เอพีไอหรือส่วนต่อประสานโปรแกรมประยุกต์ (Application programming interface) หมายถึงวิธีการที่ระบบปฏิบัติการ ไลบรารี หรือบริการอื่นๆที่เปิดให้โปรแกรมคอมพิวเตอร์สามารถติดต่อเรียกใช้งานได้ โดยแบ่งเป็น เอพีไอที่ขึ้นกับภาษา (language-dependent API) คือเอพีไอที่สามารถทำการเรียกใช้จากโปรแกรมที่เขียนขึ้นด้วยภาษาเพียงภาษาใดภาษาหนึ่งและเอพีไอไม่ขึ้นกับภาษา (language-independent API) คือเอพีไอที่สามารถเรียกได้จากโปรแกรมหลายๆภาษา

2.2 ภาษา JAVA [7]

ภาษาจาวา (Java programming language) เป็นภาษาโปรแกรมเชิงวัตถุ (Object Oriented-Programming) พัฒนาโดยเจมส์ กอสลิงและวิศวกรคนอื่นๆที่ซันไมโครซิสเต็มส์ในปีพ.ศ. 2534 (ค.ศ. 1991) โดยเป็นส่วนหนึ่งของ โครงการกรีน (the Green Project) และสำเร็จออกสู่สาธารณะในปี พ.ศ. 2538 (ค.ศ. 1995) ซึ่งภาษานี้มีจุดประสงค์เพื่อใช้แทนภาษาซีพลัสพลัส (C++) โดยรูปแบบที่เพิ่มเติมขึ้นคล้ายกับภาษาอ็อบเจกต์ทีฟซี (Objective-C) แต่เดิมภาษานี้เรียกว่า ภาษาโอ๊ก (Oak) ซึ่งตั้งชื่อตามต้นโอ๊กใกล้ที่ทำงานของ เจมส์ กอสลิง แต่ว่ามีปัญหาทางลิขสิทธิ์ จึงเปลี่ยนไปใช้ชื่อ "จาวา" ซึ่งเป็นชื่อกาแฟแทน

แม้ว่าจะมีชื่อคล้ายกัน แต่ภาษาจาวาไม่มีความเกี่ยวข้องใด ๆ กับภาษาจาวาสคริปต์ (JavaScript) ปัจจุบันมาตรฐานของภาษาจาวาดูแลโดย Java Community Process ซึ่งเป็นกระบวนการอย่างเป็นทางการ ที่อนุญาตให้ผู้ที่สนใจเข้าร่วมกำหนดความสามารถในจาวาแพลตฟอร์มได้

ภาษาจาวา และจาวาแพลตฟอร์มนั้นไม่เหมือนกัน แต่เนื่องจากชื่อที่เหมือนกัน และการเรียกขานที่มักจะพูดถึงพร้อมกันบ่อยๆ ทำให้คนทั่วไป มักสับสนว่า ภาษาจาวา และจาวาแพลตฟอร์มเป็นสิ่งเดียวกัน แต่ในความเป็นจริงนั้นทั้งสองสิ่งแม้จะทำงานเสริมกัน แต่ก็เป็นสิ่งที่แยกออกจากกัน โดยภาษาจาวานั้น คือภาษาสำหรับใช้เขียนโปรแกรมภาษาหนึ่ง ดังที่ได้อธิบายไปข้างต้น ส่วนจาวาแพลตฟอร์มนั้น คือ สภาพแวดล้อมสำหรับการใช้งานโปรแกรมจาวา โดยมีองค์ประกอบหลัก

คือ จาวาเวอร์ชวลแมทชีน (Java virtual machine) และ ไลบรารีมาตรฐานจาวา (Java standard library) โปรแกรมที่ทำงานบนจาวาแพลตฟอร์มนั้น ไม่จำเป็นจะต้องสร้างด้วยภาษาจาวา เช่น อาจจะใช้ภาษาไพทอน (Python) หรือ ภาษาอื่นๆ ก็ได้ ส่วนภาษาจาวานั้น ก็สามารถนำไปใช้พัฒนาโปรแกรมสำหรับแพลตฟอร์มอื่นได้เช่นเดียวกัน เช่น คอมไพเลอร์ gcj สามารถคอมไพล์โปรแกรมที่เขียนด้วยภาษาจาวา ให้ทำงานได้โดยไม่ต้องใช้จาวาเวอร์ชวลแมชีน

2.3 SDK [8]

SDK ย่อมาจาก Software Development Kit (SDK) คือ เครื่องมือใช้พัฒนาโปรแกรม ประกอบด้วยตัวแปลภาษา (คอมไพลเลอร์), ตัวเชื่อม (ลิงเกอร์), และตัวแก้ไข (ดีบักเกอร์) หรืออาจมีโปรแกรมมาตรฐาน (ไลบรารี่) กับเอกสารรวมอยู่ด้วย นั่นคือเป็นชุดโปรแกรมที่เป็นเครื่องมือสำหรับนักพัฒนาโปรแกรมบนแพลตฟอร์มนั้นๆนั่นเอง

2.4 Servlet [9]

Servlet เป็นแอปพลิเคชั่นที่ทำงานทางฝั่งเซิร์ฟเวอร์ (Server Side Application) มีรูปแบบการทำงานคล้ายๆกับภาษา CGI มีความสามารถในการจัดการกับเว็บแอปพลิเคชั่นแบบ Dynamic Content และถูกสร้างขึ้นจากภาษา Java ส่งผลให้ Servlet ยังคงมีคุณสมบัติของ Object Oriented อยู่ โดย Servlet ที่สร้างขึ้นมานั้นจะทำงานอยู่ใน Servlet Engine (หรืออาจเรียกว่า Servlet Container ก็ได้) ใน Servlet Engine หนึ่งๆ อาจประกอบไปด้วยหลายๆ Servlet เช่น Servlet ที่ทำหน้าที่ในการเก็บข้อมูลสมาชิกหรือ Servlet ที่ทำหน้าที่ในการตรวจสอบการ Login เป็นต้น

```
package myapp.helloworld;

import java.io.IOException;[]

@SuppressWarnings("serial")
public class HelloWorldServlet extends HttpServlet {
    public void doGet(HttpServletRequest req, HttpServletResponse resp)
            throws IOException {
        resp.setContentType("text/plain");
        resp.getWriter().println("Hello, world");
    }
}
```

รูปที่ 2.6 แสดงตัวอย่างการใช้งาน Java Servelt

2.5 JSP (Java Server Page) [10]

Java Server Page หรือเรียกสั้นว่า JSP เป็นเทคโนโลยีที่ทำงานบนฝั่งเซิร์ฟเวอร์ (Server Side Script) มีความสามารถในการจัดการกับเว็บแอปพลิเคชั่นแบบ Dynamic Content โดย JSP ถูกพัฒนามาจาก Servlet เพื่อแก้ไขปัญหาหนึ่งที่เกิดขึ้นกับ Servlet คือเนื่องมาจากการที่ Servlet เป็นการผสมข้อมูลในส่วนของ Business Logic (ข้อมูลทางตรรก เช่น JavaBean,Database) กับ Presentation Layer (ข้อมูลในส่วนของการแสดงผล) รวมเข้าด้วยกันและยังเปรียบเสมือน Java File ที่มีการฝังแท็ก HTML ลงไป ทำให้ผู้ที่จะพัฒนาจำเป็นต้องมีความรู้ในด้านภาษาจาวามากพอสมควรและการแก้ไขในส่วนของหน้าตาที่ใช้แสดงผลนั้นค่อนข้างทำได้ยาก แต่ JSP นั้นจะมีการแยกในส่วนของ Business Logic กับ Presentation Layer ออกจากกันและ JSP ยังเปรียบเสมือน HTML-Page ที่มีการฝัง Java Code ลงไป ทำให้การเขียนโปรแกรมมีประสิทธิภาพมากขึ้น โดยการแยกหน้าที่ของผู้พัฒนาตามความถนัด เช่น หากถนัดเขียนโค้ดก็ทำงานในส่วนของ Business Logic แต่หากถนัดที่จะออกแบบก็ให้ทำงานในส่วนของ Presentation Layer เป็นต้น

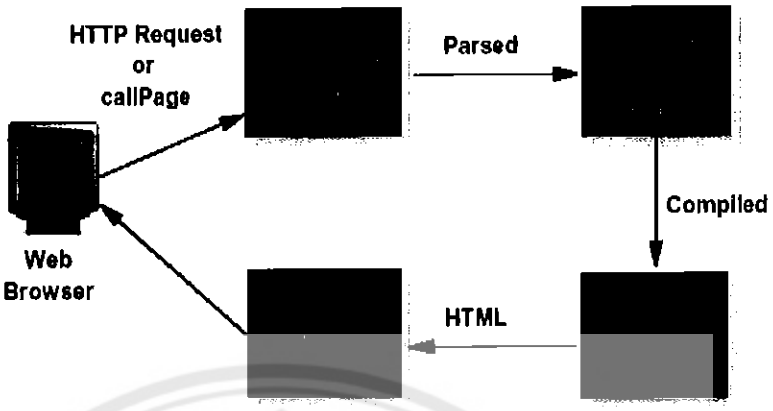
```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Welcome your first java ee</title>
</head>
<body>
<h1>Hello World</h1>
<h2>and this is a message: ${message}</h2>
</body>
</html>
```

รูปที่ 2.7 แสดงการตัวอย่างการใช้งานไฟล์ JSP

2.6 การทำงานของเว็บแอปพลิเคชั่นโดยใช้ Servlet และ JSP

เว็บแอปพลิเคชั่นนั้นจะใช้ทั้ง Servlet และ JSP ในการทำงานร่วมกัน เพื่อตอบสนองกับ Request ที่เข้ามา โดยหน้าที่ของ Servlet ก็คือการจัดการกับ request และ response ต่างๆ และส่งค่าเหล่านั้นไปแสดงผลใน JSP

ตัวอย่างเช่น เมื่อมีการร้องขอ หรือเกิด Request จาก Client (หรือ Browser) เข้ามาที่ JSP บนฝั่ง Server เมื่อ Server ได้รับ request จะทำการประมวลผล JSP ให้เป็น Servlet ก่อนแล้วจึงส่ง Response กลับไปให้ Client ในรูปแบบของ HTML ดังรูปที่ 2.6



รูปที่ 2.8 แสดงการทำงานร่วมกันระหว่าง JSP และ Servlet ของเว็บแอปพลิเคชั่น [11]

2.7 Google App Engine [12]

Google App Engine (มักถูกเรียกว่า App Engine และใช้ตัวย่อว่า GAE) คือ แพลตฟอร์มการพัฒนาและการให้บริการพื้นที่แอปพลิเคชันของ Google ที่เปิดให้ผู้พัฒนาเว็บแอปพลิเคชั่นสามารถเขียนโปรแกรมเข้าไปเชื่อมต้องกับโครงสร้างข้อมูลของ Google ได้มากขึ้น โดยผู้พัฒนานั้นไม่จำเป็นต้องทำการการติดตั้งหรือตั้งค่า web server เอง แต่จะต้องสมัครเข้าใช้งานแล้วทำการอัปโหลด source code ไปยัง Google App Engine เพื่อให้ได้เว็บแอปพลิเคชั่นที่จะอิงกับสถาปัตยกรรมของ Google ไม่ว่าจะเป็นระบบฐานข้อมูลหรือโครงสร้างพื้นฐานอื่นๆ ทั้งยังอาศัยพลังประมวลผลของ Google ที่รวดเร็วกว่าในการประมวลผลเว็บแอปพลิเคชั่นนั้นๆ ที่สำคัญ Google App Engine ยังช่วยตัดค่าใช้จ่ายและความซับซ้อนในการติดตั้งและจัดการกับเซิร์ฟเวอร์ของตัวเองออกไปอีกด้วย

ปัจจุบัน Google App Engine รองรับได้ 3 ภาษา คือ Java, Python และ Go (อาจรวมไปถึงภาษา JVM เช่น Groovy, JRuby, Scala, Clojure, Jpython และ PHP ผ่านทาง special version ของ Quercus) คาดว่าในอนาคตจะรองรับภาษาต่างๆเพิ่มขึ้นอีก ส่วนการใช้งานหากเป็นแบบฟรีจะมีพื้นที่เก็บข้อมูลให้ 500 MB มี bandwidth และ CPU ที่รองรับได้การเข้าชมได้ 5 ล้านครั้งต่อเดือน ซึ่งถือว่าเพียงพอต่อการใช้งานโดยทั่วๆไป

API หลักที่ App Engine มีให้นักพัฒนาสามารถใช้งานได้ ประกอบไปด้วย User API สำหรับใช้ในการระบุตัวตนของผู้ใช้, Memcache สำหรับช่วยลดเวลาในการแสดงหน้าเพจ, Mail API สำหรับการส่งอีเมลล์ผ่านทางแอปพลิเคชั่น, Image API ที่ช่วยในการประมวลผลรูปภาพ และ XMPP API สำหรับการส่ง instance message เป็นต้น

บทที่ 3
วิธีการดำเนินโครงงาน

ในการจัดทำโครงงานคู่มือการพัฒนาเว็บแอปพลิเคชั่นด้วยกูเกิ้ลแอปเอ็นจิ้นนั้น ได้แบ่งการดำเนินการออกเป็น 3 ขั้นตอน ดังนี้

1. ศึกษาคุณลักษณะที่สำคัญของกูเกิ้ลแอปเอ็นจิ้นที่จำเป็นต่อการใช้งาน
2. ออกแบบและพัฒนาเว็บแอปพลิเคชั่นเพื่อนำคุณลักษณะต่างๆของกูเกิ้ลแอปเอ็นจิ้นมาใช้งาน
3. จัดทำเนื้อหาของแต่ละคุณลักษณะในรูปแบบหนังสือ โดยมีเว็บแอปพลิเคชั่นที่ได้ออกแบบไว้เป็นตัวอย่างประกอบ

3.1 ศึกษาคุณลักษณะต่างๆของกูเกิ้ลแอปเอ็นจิ้น

เนื่องจากกูเกิ้ลแอปเอ็นจิ้นมีคุณลักษณะต่างๆมากมายให้เลือกใช้งาน ผู้จัดทำจึงทำการเลือกเฉพาะคุณลักษณะหลักที่สามารถนำมาใช้งานได้สะดวก และไม่มีค่าใช้จ่ายเพิ่มเติม ซึ่งได้ทำการแบ่งคุณลักษณะที่จะศึกษาเป็น 3 กลุ่มด้วยกัน ดังนี้

1. คุณลักษณะที่ใช้ในการพัฒนาเว็บแอปพลิเคชั่นในภาษาจาวา เป็นส่วนที่ใช้สำหรับการสร้างและพัฒนาเว็ปแอปพลิเคชั่น ซึ่งประกอบด้วย

<u>เครื่องมือสำหรับการพัฒนาเว็ปแอปพลิเคชั่น</u> ซึ่งประกอบไปด้วย

- เครื่องมือสำหรับพัฒนาโปรแกรมคือ Eclipse โดยที่ Eclipse นั้นเป็นโปรแกรมประเภท IDE ย่อมาจาก Integrated Development Environments คือเป็นเครื่องมือที่ช่วยในการพัฒนาโปรแกรมโดยมีสิ่งอำนวยความสะดวกต่างๆ รองรับการใช้งานภาษาจาวา
- เครื่องมือสำหรับการทดสอบแอปพลิเคชั่น คือ Development Server โดยที่ Development web server นั้นมีอยู่ใน App Engine Java SDK สำหรับการทดสอบแอปพลิเคชั่นบนโลคอลคอมพิวเตอร์ โดยจะทำการจำลอง Java runtime environment ของ App Engine และ services ทั้งหมดที่มีรวมถึง datastore มาใช้สำหรับการทดสอบแอปพลิเคชั่นด้วย
- เครื่องมือในการอัปโหลดและจัดการแอปพลิเคชั่นคือ App Engine Java SDK ในส่วนนี้จะเป็นการพูดถึงการอัปโหลดและจัดการแอปพลิเคชั่นโดยการใช้ คำสั่งต่างๆซึ่งประกอบไปด้วยการอัปโหลดโค้ดของแอปพลิเคชั่น อัปโหลดการตั้งค่าต่างๆ และ static files รวมไปถึงการใช้คำสั่งในการจัดการกับดาต้าสโตร์อินเดกซ์ และการดาวน์โหลดล็อกดาต้าอีกด้วย

- เครื่องมือในการสร้างและ Deploy เว็บแอปพลิเคชั่น คือกูเกิ้ลแอปเอ็นจิ้นปลั๊กอินซึ่งเป็นปลั๊กอินของกูเกิ้ลแอปเอ็นจิ้นที่จะช่วยนักพัฒนาในการทดสอบ การสร้างหรือการ Deploy เว็บแอปพลิเคชั่นให้สามารถทำได้ง่ายขึ้น ซึ่งหากต้องการใช้ก็จะต้องทำการติดตั้งลงไปในอีคลิปส์ก่อน

พื้นที่สำหรับเก็บข้อมูล ประกอบด้วย

- การจัดเก็บข้อมูลโดยใช้ Java Datastore API นั้นประกอบไปด้วยการใช้งาน Low level API ซึ่งสามารถใช้การดำเนินการอย่างง่าย เช่น get put delete และ query ได้ รวมทั้งสามารถใช้ API นี้ได้โดยตรงภายในแอปพลิเคชั่นและสามารถใช้ในการพัฒนา interface adapters อื่นได้ นอกจากนี้ Java SDK ยังมีอินเตอร์เฟซมาตรฐานในการเข้าถึง Datastore อีก 2 รูปแบบ คือ JDO (Java Data Object) และ JPA (Java Persistence API) ซึ่งทั้งสองรูปแบบนี้ต่างก็ใช้แพลตฟอร์มของ DataNucleus Platform ในการเก็บข้อมูล โดยที่ JDO นั้นเป็นอินเตอร์เฟซมาตรฐานสำหรับการจัดเก็บข้อมูลแบบ Object ไปยังฐานข้อมูล สามารถเรียก Object ได้โดยใช้การ Query และโต้ตอบกับฐานข้อมูลโดยใช้ Transaction เป็นลักษณะการเก็บข้อมูลแบบ Schemaless ซึ่งแตกต่างจากการเก็บข้อมูลที่นิยมใช้ในแบบ Traditional Relation Database ทำให้สามารถรองรับการเก็บข้อมูลเกือบทุกรูปแบบ ส่วน JPA นั้นเป็นอินเตอร์เฟซมาตรฐานสำหรับการเก็บข้อมูลแบบ Object ไปยังฐานข้อมูลเชิงสัมพันธ์ (relational database) สามารถทำการเรียก Object ได้ด้วยการ Query และโต้ตอบกับฐานข้อมูลโดยใช้ Transaction เช่นเดียวกันกับ JDO แต่เนื่องจากการที่ดาต้าสโตร์ของกูเกิ้ลแอปเอ็นจิ้นไม่ได้เป็นฐานข้อมูลเชิงสัมพันธ์ จึงมีบางคุณลักษณะของ JPA ที่กูเกิ้ลแอปเอ็นจิ้นไม่สามารถรองรับได้ เช่น การ Join query และความสัมพันธ์แบบ many-to-many เป็นต้น

- Google Cloud SQL เป็นเว็บเซอร์วิสที่ช่วยในการสร้าง การกำหนดค่า และการใช้งานฐานข้อมูลเชิงสัมพันธ์ที่มีอยู่ในคลาวด์ของกูเกิ้ล เป็นเซอร์วิสที่ช่วยในการเก็บและจัดการข้อมูลข้อมูลที่สามารถให้บริการได้อย่างเต็มประสิทธิภาพ โดยรูปแบบของฐานข้อมูลนั้นคล้างคลึงกับฐานข้อมูล MySQL

- Google Cloud Storage (API) เป็นบริการในการเก็บรักษาข้อมูลบนคลาวด์ของกูเกิ้ล โดยที่ API นี้จะรองรับการอ่านและเขียนข้อมูลไปยังดาต้าสโตร์ แต่หากต้องการให้บริการข้อมูลนั้นแก่แอปพลิเคชั่น จะต้องทำผ่านทาง Blobstore API

Services ต่างๆ ประกอบด้วย

- User API ที่ใช้ในการตรวจสอบผู้ใช้ที่มีบัญชีของกูเกิ้ล หรือการตรวจสอบบัญชีที่อยู่บนโดเมนของแอปพลิเคชั่น หรือการระบุตัวตนโดยใช้ OpenID รวมทั้งใช้ในการตรวจจับว่าผู้ใช้คนปัจจุบันได้ทำการล็อกอิน (ลงชื่อเข้าใช้) เข้ามาหรือไม่ และถ้าหากผู้ใช้ยังไม่ได้ทำการล็อกอินก็สามารถที่จะทำการเชื่อมต่อเส้นทางไปยังหน้าล็อกอินได้ นอกจากนี้ในขณะที่ผู้ใช้ทำการเข้าสู่ระบบ แอปพลิเคชั่นจะสามารถระบุได้ว่าผู้ใช้ที่ล็อกอินเข้ามานี้เป็นผู้ดูแลระบบของแอปพลิเคชั่นหรือไม่อีกด้วย

- Blobstore ที่ใช้ในการจัดเก็บข้อมูลในรูปแบบออบเจ็ก โดยการอัปโหลดผ่าน HTTP request ซึ่งเป็นประโยชน์มากในการให้บริการไฟล์ที่มีขนาดใหญ่แก่ผู้ใช้ เช่น ไฟล์รูปภาพ วิดิโอ เป็นต้น

- Mail API ที่ทำให้แอปพลิเคชั่นสามารถส่งข้อความอีเมลล์ในนามของผู้ดูแลระบบและในนามของผู้ใช้ด้วยการใช้บัญชีผู้ใช้ของกูเกิ้ล โดยแอปพลิเคชั่นจะสามารถส่งอีเมลล์ได้โดยใช้ Java Mail API และสามารถรับข้อความที่เข้ามาได้ในรูปแบบของ http request

- URL Fetch API ที่ทำให้แอปพลิเคชั่นจะสามารถติดต่อสื่อสารกับแอปพลิเคชั่นอื่นๆ หรือเข้าถึง resources อื่นๆบนเว็บได้โดยการใช้ URL Fetch API โดยที่แอปพลิเคชั่นจะใช้ URL Fetch service เพื่อส่งออกและรับการตอบสนองของ HTTP และ HTTPS requests และสำหรับ URL Fetch API นี้จะใช้โครงสร้างเน็ทเวิร์กของกูเกิ้ลเพื่อให้การเข้าถึงมีประสิทธิภาพมากขึ้น

- Memcache API เป็นบริการที่ช่วยให้การเข้าถึงข้อมูลของแอปพลิเคชั่นมีประสิทธิภาพมากขึ้น โดยใช้การเก็บกระจายข้อมูลไว้ในแคช ซึ่งเป็นหน่วยความจำที่จะดึงมาใช้ก่อนที่จะเข้าถึงไปยังหน่วยความจำหลัก

- Channel API เป็นการสร้างการเชื่อมต่อแบบถาวรระหว่างแอปพลิเคชั่น และกูเกิ้ลเซิฟเวอร์เพื่อช่วยให้แอปพลิเคชั่นสามารถส่งข้อความไปยังจาวาสคริปไคลเอนท์ได้แบบเรียลไทม์โดยไม่มีการใช้ polling ซึ่งเป็นประโยชน์กับแอปพลิเคชั่นที่ออกแบบมาเพื่อรองรับการอัปเดทข้อมูลใหม่ของผู้ใช้ได้ทันที หรือเพื่อรองรับการเผยแพร่ข้อมูลไปยังผู้ใช้หลายๆคน เป็นต้น API นี้สามารถนำไปใช้งานสำหรับการทำบอร์ดเซทในเกมที่มีผู้เล่นหลายๆคน หรือในแอปพลิเคชั่นที่มีการใช้งานร่วมกัน เป็นต้น

- Log API เป็นการให้บริการในการเข้าถึงไปยังแอปพลิเคชั่น และเรียกดูบันทึกการใช้งานของแอปพลิเคชั่นได้ โดยที่บันทึกต่างๆนั้นถูกเรียกว่า log

2. คุณลักษณะที่ใช้ในการกำหนดค่าเว็บแอปพลิเคชั่น เป็นส่วนที่ใช้สำหรับการกำหนดค่าต่างๆ สำหรับเว็ปแอปพลิเคชั่น ประกอบด้วย

Deployment Descriptor ซึ่งก็คือไฟล์ web.xml เป็นไฟล์ XML ที่ใช้สำหรับการกำหนดการจับคู่แอปพลิเคชั่นระหว่าง servlet กับที่อยู่ URL เพื่อใช้ในการจัดการกับ request ที่เข้ามา

Application Configuration ซึ่งก็คือไฟล์ XML ที่มีชื่อว่า appengine-web.xml ซึ่งมีไว้สำหรับระบุการตั้งค่าต่างๆของแอปพลิเคชั่นที่เป็นกูเกิ้ลแอปเอ็นจิ้นแอปพลิเคชั่น เช่น ID ของแอปพลิเคชั่นและเวอร์ชั่นของแอปพลิเคชั่น เป็นต้น

Backend Configuration ซึ่งก็คือไฟล์ backends.xml เป็นไฟล์ที่จะช่วยเพิ่มแบ็กเอนด์ไปยังแอปพลิเคชั่นได้ ด้วยการประกาศชื่อและคุณสมบัติที่ต้องการของแต่ละ backend server ในส่วนนี้จึงเป็นการอธิบายถึงประเภทของแบ็กเอ็นด์อินสแตนซ์ต่างๆ และคลาสของแบ็กเอ็นด์ เป็นต้น

Index Configuration ซึ่งก็คือไฟล์ datastore-indexes.xml จะระบุการกำหนดค่าอินเด็กซ์ของดาต้าสโตร์ไว้ เนื่องจากในทุกๆการคิวรีของดาต้าสโตร์นั้นจำเป็นต้องใช้อินเด็กซ์ โดยที่อินเด็กซ์ก็คือตารางผลลัพธ์ของการคิวรีในลำดับที่ต้องการ ในหัวข้อนี้จึงเป็นการอธิบายถึงส่วนประกอบต่างๆในไฟล์อินเด็กซ์และการใช้งาน

Scheduled Task Configuration ซึ่งก็คือ App Engine Cron Service จะช่วยในการกำหนด scheduled tasks ที่จะทำงานในเวลาที่กำหนดไว้ หรือทำงานในช่วงเวลางานปกติ โดยที่ tasks ดังกล่าวนั้นเป็นที่รู้จักกันดีในชื่อ cron jobs และ cron jobs เหล่านี้จะถูกเรียกใช้อย่างอัตโนมัติโดย App Engine Cron Service โดยสามารถทำการกำหนดค่า cron jobs ได้จากไฟล์ cron.xml และในส่วนนี้จะอธิบายถึงส่วนต่างๆในไฟล์ cron.xml และการใช้งาน

Task Queue Config ที่แอปพลิเคชั่นใช้ในการกำหนด task queues โดยใช้ไฟล์ที่ชื่อ queue.xml ในการกำหนดค่าทั้ง push queues และ pull queues ในส่วนนี้จึงเป็นส่วนที่จะอธิบายการใช้งานไฟล์ queue.xml ถึงส่วนประกอบและการใช้งานต่างๆ

3. คุณลักษณะที่ใช้ในการดูแลและจัดการเว็บแอปพลิเคชั่น เป็นส่วนที่ใช้สำหรับการดูแลและจัดการเว็ปแอปพลิเคชั่น ประกอบด้วย

Backends เป็นอินสแตนซ์พิเศษของแอปพลิเคชั่น โดยอินสแตนซ์ก็คือหน่วยการประมวลผลเป็นเวอร์ช่วลแมชชีนที่มีชุดของหน่วยความจำและซีพียูตามที่จำกัดไว้ให้ ซึ่งแบ็กเอ็นด์สามารถเข้าถึงหน่วยความจำและซีพียูได้มากกว่าอินสแตนซ์ทั่วไป และได้รับการยกเว้นจากกฎ request deadline โดยมีกูเกิ้ลแอปเอ็นจิ้นเป็นผู้ดำเนินการทำงานของแบ็กเอ็นด์อย่างอัตโนมัติ มีหน้าที่ในการจัดการกับ request ที่เข้ามายังแอปพลิเคชั่นทั้งจากผู้ใช้และจากส่วนอื่นๆของแอปพลิเคชั่น ซึ่งในหัวข้อนี้จะเป็นการอธิบายถึงการตั้งค่าและการใช้งานแบ็กเอ็นด์อย่างง่าย เป็นต้น

Admin Console แอดมินคอนโซลเป็นส่วนที่จะให้การเข้าถึงไปยังแอปพลิเคชั่นที่ถูกเผยแพร่อยู่บนกูเกิ้ลแอปเอ็นจิ้นแล้ว โดยการเข้าไปที่ https://appengine.google.com/ โดยผู้ที่เป็นแอดมินสามารถใช้แอดมินคอนโซลในการกำหนดค่าพื้นฐานต่างๆของแอปพลิเคชั่น เช่น title หรือ cookie ได้ สามารถกำหนดค่าประสิทธิภาพของแอปพลิเคชั่นได้ สามารถดูการกำหนดใช้งานบริการต่างๆ เปิดปิดการเขียนของดาต้าสโตร์ รวมไปถึงทำการปิดการใช้งานแอปพลิเคชั่นได้ เป็นต้น

โควตาการใช้งาน เป็นส่วนที่ระบุถึงขีดจำกัดในการใช้งานส่วนต่างๆของกูเกิ้ลแอปเอ็นจิ้น เช่น โควตาการใช้งาน Blobstore โควตาการใช้งานพื้นที่เก็บ code และ static file โควตาการใช้งานดาต้าสโตร์ เป็นต้น

3.2 ออกแบบและพัฒนาเว็บแอปพลิเคชั่น

เพื่อให้ผู้อ่านสามารถเข้าใจถึงเนื้อหาอย่างชัดเจน ผู้เขียนจึงได้จัดทำตัวอย่างประกอบการศึกษา เป็นการนำคุณลักษณะต่างๆที่ได้ศึกษาข้างต้น มาใช้งานในการพัฒนาเว็บแอปพลิชั่นด้วยภาษาจาวา โดยมีขั้นตอนการพัฒนา ดังนี้

HelloWorld Application

จุดประสงค์ : (1) เพื่อแสดงการใช้งานโปรแกรมอีคลิปส์สำหรับการสร้างกูเกิ้ลแอปเอ็นจิ้นแอปพลิเคชั่น

(2) เพื่อแสดงการใช้งานไฟล์จาวาเซิร์ฟเล็ท welcome-file และไฟล์ Web.xml เบื้องต้น

(3) เพื่อแสดงตัวอย่างการเริ่มการทำงานของแอปพลิเคชั่นและการหยุดการทำงาน

ก่อนพัฒนา : (1) ติดตั้งโปรแกรม Eclipse IDE

(2) ติดตั้ง GAE Plugin สำหรับ Eclipse IDE

(3) ติดตั้ง GAE Java SDK

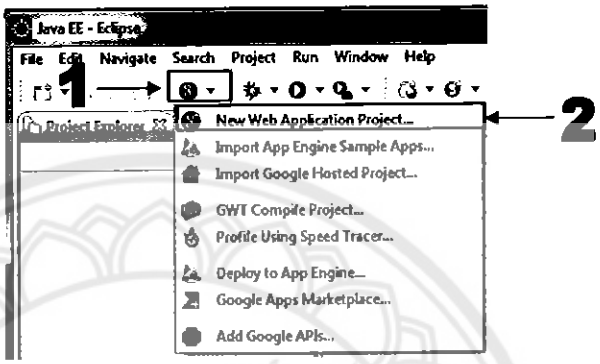
ขั้นตอนการพัฒนา :

1. ภาพรวมของการพัฒนาแอปพลิเคชั่น

รูปที่ 3.1 แสดงภาพรวมของการพัฒนาแอปพลิเคชั่น

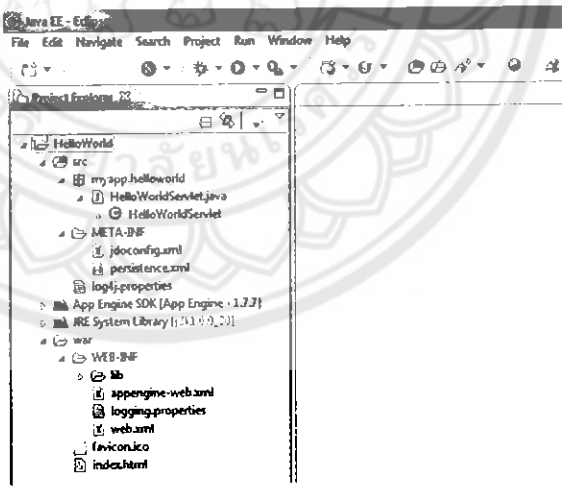
2. การสร้างเว็บแอปพลิเคชั่นโปรเจ็กโดยใช้โปรแกรม Eclipse IDE

ทำการสร้างกูเกิ้ลแอปเอ็นจิ้นเว็บแอปพลิเคชั่นโปรเจ็กด้วย Plugin ของกูเกิ้ลแอปเอ็นจิ้น โดยการคลิกที่ปุ่ม (8) บนแถบเครื่องมือหลักแล้ว New Web Application Project **ดังรูป**



รูปที่ 3.2 แสดงการใช้งาน Ecipse ในการสร้างเว็บแอปพลิเคชั่นใหม่

เมื่อทำการกำหนดค่าต่างๆของแอปพลิเคชั่นเรียบร้อยแล้ว (ดูเพิ่มเติมที่ภาคผนวก **"การใช้งาน** Eclipse IDE") จะได้โครงสร้างของไดเรกทอรี ดังรูป

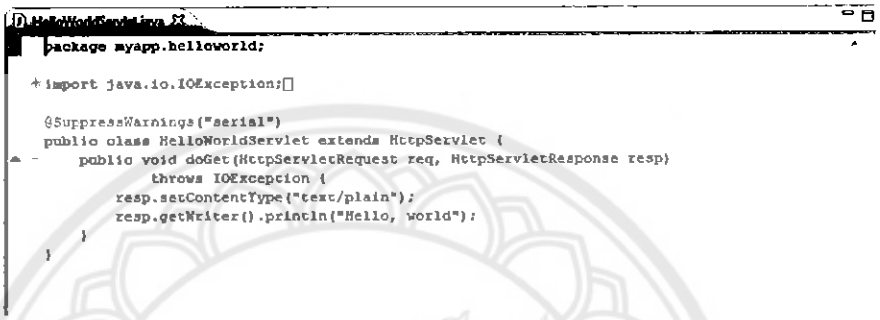


รูปที่ 3.3 แสดงโครงสร้างของไดเรกทอรี

3. สร้างการทำงานของแอปพลิเคชั่นด้วยไฟล์ servlet และ index.html

เมื่อสร้างโปรเจ็กใหม่ โปรแกรมอีคลิปส์จะทำการสร้าง servlet class และ welcome-file ชื่อ index.html สำหรับแอปพลิเคชั่นขึ้นมาโดยอัตโนมัติ ซึ่งไฟล์ทั้งสองนั้นมีเนื้อหาดังนี้

ไฟล์ HelloServlet.java



```
package myapp.helloworld;

import java.io.IOException;

@SuppressWarnings("serial")
public class HelloWorldServlet extends HttpServlet {
    public void doGet(HttpServletRequest req, HttpServletResponse resp)
            throws IOException {
        resp.setContentType("text/plain");
        resp.getWriter().println("Hello, world");
    }
}
```

รูปที่ 3.4 รูปแสดงไฟล์ HelloServlet.java

ไฟล์ Index.html

```
<!-- "Quirks Mode". Replacing this declaration      -->
<!-- with a "Standards Mode" doctype is supported, -->
<!-- but may lead to some differences in layout.   -->

<html>
  <head>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8">
    <title>Hello App Engine</title>
  </head>

  <body>
    <h1>Hello App Engine!</h1>

    <table>
      <tr>
        <td colspan="2" style="font-weight:bold;">Available Servlets:</td>
      </tr>
      <tr>
        <td><a href="helloworld">HelloWorld</a></td>
      </tr>
    </table>
  </body>
</html>
```

รูปที่ 3.5 แสดงไฟล์ Index.html

4. ตั้งค่าแอปพลิเคชั่นด้วยไฟล์ web.xml และ appengine-web.html

เมื่อทำการสร้างโปรเจ็กใหม่ โปรแกรมอีคลิปส์จะทำการสร้างไฟล์ web.xml และ appengine-web.html พร้อมทั้งกำหนดเนื้อหาของไฟล์ดังกล่าวให้โดยอัตโนมัติ ดังนี้

หมายเหตุ : หากต้องการอัปโหลดแอปพลิเคชั่นไปยังกูเกิ้ลแอปเอ็นจิ้นจะต้องทำการสมัครเพื่อรับ Application ID ก่อน อ่านเพิ่มเติมในภาคผนวก "การขอ Application ID"

ไฟล์ web.xml

```
<?xml version="1.0" encoding="utf-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" version="2.5">
    <servlet>
        <servlet-name>HelloWorld</servlet-name>
        <servlet-class>myapp.helloworld.HelloWorldServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>HelloWorld</servlet-name>
        <url-pattern>/helloworld</url-pattern>
    </servlet-mapping>
    <welcome-file-list>
        <welcome-file>index.html</welcome-file>
    </welcome-file-list>
</web-app>
```

รูปที่ 3.6 แสดงไฟล์ web.xml

ไฟล์ appengine-web.xml

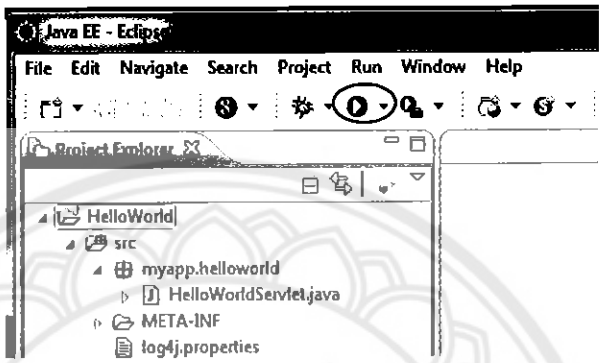
```
<?xml version="1.0" encoding="utf-8"?>
<appengine-web-app xmlns="http://appengine.google.com/ns/1.0">
  <application></application>
  <version>1</version>

  <!--
    Allows App Engine to send multiple requests to one instance in parallel:
  -->
  <threadsafe>true</threadsafe>

  <!-- Configure java.util.logging -->
  <system-properties>
    <property name="java.util.logging.config.file" value="WEB-INF/logging.properties"/
  </system-properties>

  <!--
    HTTP Sessions are disabled by default. To enable HTTP sessions specify:

      <sessions-enabled>true</sessions-enabled>

    It's possible to reduce request latency by configuring your application to
    asynchronously write HTTP session data to the datastore:
```

รูปที่ 3.7 แสดงไฟล์ appengine-web.xml

5. การเริ่มและหยุดการทำงานของแอปพลิเคชั่น

<u>เริ่มการทำงานของแอปพลิเคชั่น</u>

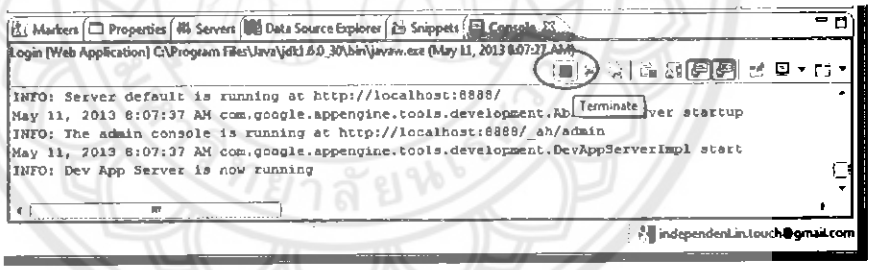
ทำได้โดยการไปที่ปุ่ม บนแถบเครื่องมือหลักของ Eclipse



รูปที่ 3.8 แสดงปุ่มเริ่มการทำงานของ Eclipse

<u>การหยุดการทำงานของแอปพลิเคชั่น</u>

ทำได้โดยการคลิกที่ปุ่ม เพื่อจบการทำงานของเซิร์ฟเวอร์ ดังรูป



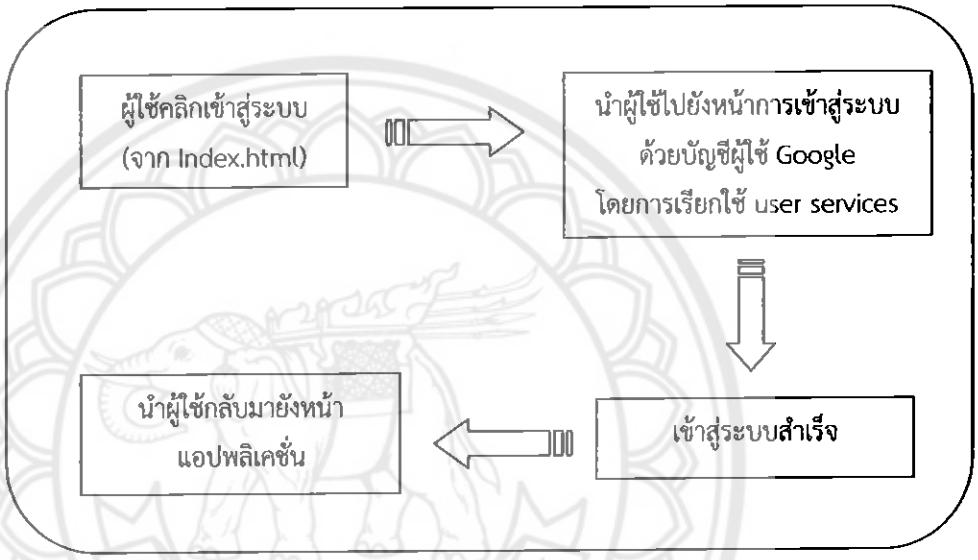
รูปที่ 3.9 แสดงปุ่มสำหรับหยุดการทำงานของ server

Login Application

จุดประสงค์ : เพื่อแสดงตัวอย่างการใช้งาน User API **สำหรับการลงชื่อเข้าใช้งานแอปพลิเคชั่นด้วยบัญชีผู้ใช้ของ** Google

ขั้นตอนการพัฒนา :

1. **ภาพรวมของแอปพลิเคชั่น**



รูปที่ 3.10 แสดงภาพรวมของแอปพลิเคชั่น

2. วิธีการอิมพลีเม้นท์โค้ด servlet .java

เริ่มจากเมื่อผู้ใช้เข้ามายังแอปพลิเคชั่น จะต้องทำการตรวจสอบก่อนว่าผู้ใช้ได้ทำการเข้าสู่ระบบด้วยบัญชีผู้ใช้ของกูเกิ้ลหรือไม่ โดยการเรียกใช้เมธอด getCurrentUser(); ของ userService ดังรูป

```
UserService userService = UserServiceFactory.getUserService();
User user = userService.getCurrentUser();
```

รูปที่ 3.11 แสดงการใช้งานเมธอด getCurrentUser()

โดยหากผู้ใช้ได้ทำการเข้าสู่ระบบด้วยบัญชีของกูเกิ้ลแล้ว แอปพลิเคชั่นจะแสดงข้อความต้อนรับและสามารถ get.Nickname (); เพื่อแสดงชื่อของผู้ใช้ได้ รวมทั้งสามารถทำการลงชื่อออกจากการใช้งานแอปพลิเคชั่นได้ด้วยเมธอด createLogoutURL(); ดังรูป

```
if (user != null) {
msg = "<p>Welcome !, " + user.getNickname() + " <a href=\"" + userService.createLogoutURL(url) +
"\"><br><br> SIGN OUT </a> | <a href= mailva.html> E-mail Service </a></p>";
} else {
```

รูปที่ 3.12 แสดงการใช้งานเมธอด get.Nickname (); และ createLogoutURL();

ถ้าหากผู้ใช้ยังไม่ได้ทำการลงชื่อเข้าสู่ระบบ แอปพลิเคชั่นจะนำผู้ใช้ไปยังหน้าลงชื่อเข้าใช้โดยใช้บัญชีของกูเกิ้ลด้วยเมธอด createLoginURL(); ของ userService ดังรูป

```
} else {
msg = "<p>Welcome ! <a href=\"" + userService.createLoginURL(url) +
"\"><br><br> SIGN IN | REGISTER </a></p>";
}
```

รูปที่ 3.13 แสดงการใช้งานเมธอด createLoginURL();

3. การตั้งค่าให้กับแอปพลิเคชั่น

ไฟล์ Web.xml

ทำการจับคู่ Servlet กับ Class และ url-pattern เพื่อให้รีเควสที่เข้ามาถูกจัดการโดย LoginServlet ดังรูป

```
<servlet>
    <servlet-name>login</servlet-name>
    <servlet-class>login.LoginServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>login</servlet-name>
    <url-pattern>/login</url-pattern>
</servlet-mapping>
```

รูปที่ 3.14 แสดงการใช้งานไฟล์ web.xml

ไฟล์ appengine-web.xml

ทำการกำหนดค่า Application ID และ version ของแอปพลิเคชั่น ดังรูป

```
<application>jahcpe2</application>
<version>Login</version>
```

รูปที่ 3.15 แสดงการใช้งานไฟล์ appengine-web.xml

การกำหนดให้แอปพลิเคชั่นสามารถส่ง multiple request ไปยังอินสแตนซ์เดียวกันพร้อมกันได้

```
<threadsafe>true</threadsafe>
```

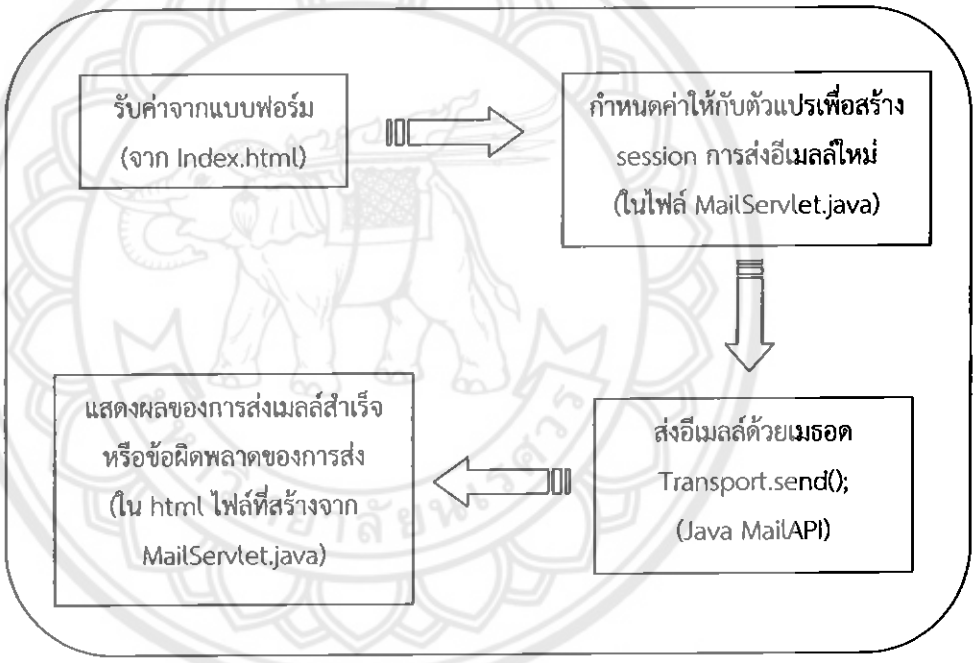
รูปที่ 3.16 แสดงการใช้งาน threadsafe ในไฟล์ appengine-web.xml

Mail Application

จุดประสงค์ : เพื่อแสดงตัวอย่างการใช้งาน Java Mail API สำหรับการส่งอีเมลล์

ขั้นตอนการพัฒนา :

1. ภาพรวมของแอปพลิเคชั่น



รูปที่ 3.17 แสดงภาพรวมของแอปพลิเคชั่น

2. วิธีการอิมพลีเม้นท์โค้ดส่วนต่างๆ

<u>ส่วนของแบบฟอร์มรับค่า</u>

แบบฟอร์มของการส่งอีเมลล์จะต้องประกอบไปด้วย ผู้รับอีเมลล์ หัวเรื่องของอีเมลล์และเนื้อความของอีเมลล์ จึงทำการสร้างแบบฟอร์มอย่างง่ายขึ้นมา โดยทำการกำหนดให้ฟิลด์ "To" รับค่า mailto ฟิลด์ "Subject" รับค่า mailSubject และฟิลด์ "Message" รับค่า mailBody ดังรูป

รูปที่ 3.18 แสดงฟอร์มที่ใช้สำหรับการส่งอีเมลล์

<u>ส่วนโค้ด MailServlet.java</u>

ได้กำหนดให้มีการรับค่าตัวแปร 3 ค่าด้วยกันคือ mailTo สำหรับเก็บค่าที่อยู่อีเมลล์ของผู้ส่ง mailSubject สำหรับเก็บค่าหัวเรื่องของอีเมลล์และ mailBody สำหรับเก็บค่าเนื้อความของอีเมลล์ ดังรูป

```
String mailTo = request.getParameter("mailTo");
String mailSubject = request.getParameter("subject");
String mailBody = request.getParameter("mailBody");
```

รูปที่ 3.19 แสดงการรับค่าตัวแปรจากฟอร์ม

ทำการสร้าง session สำหรับการส่งอีเมลล์ขึ้นมา

```
Properties props = new Properties();

Session session = Session.getDefaultInstance(props, null);
```

รูปที่ 3.20 แสดงการสร้าง session

ทำการสร้าง object ประเภท Message ขึ้นมาเพื่อใช้ในการกำหนดค่าต่างๆสำหรับsession ในการส่งอีเมลล์ ดังรูป

```
Message msg = new MimeMessage(session);
```

รูปที่ 3.21 แสดงการสร้าง object Message

ทำการกำหนดค่าในการส่งอีเมลล์ด้วยการเรียกใช้เมธอด ต่างๆ เพื่อรับค่าจากฟอร์มในไฟล์ index.html ดังรูป

```
msg.addRecipient(Message.RecipientType.TO,new InternetAddress(mailTo));
```

รูปที่ 3.22 แสดงการรับค่าที่อยู่ผู้รับและทำการกำหนดค่าเป็นผู้รับด้วยเมธอด addRecipient();

ทำการกำหนดค่าหัวเรื่องของอีเมลล์ด้วยเมธอด setSubject(); และทำการเข้ารหัสเป็น UTF-8 เพื่อให้สามารถแสดงหัวเรื่องภาษาไทยได้

```
msg.setSubject(mailSubject);
```

```
((MimeMessage)msg).setSubject(mailSubject, "UTF-8");
msg.setText(mailBody);
```

รูปที่ 3.23 แสดงการกำหนดค่า subject

สร้างการบันทึกค่า object msg และใช้เมธอด Transport.send () สำหรับการส่งอีเมลล์

```
msg.saveChanges();
Transport.send(msg);
```

รูปที่ 3.24 แสดงการเรียกใช้เมธอด Transport.send ()

เพื่อที่จะทราบว่าโปรแกรมสามารถส่งอีเมลล์สำเร็จหรือไม่ จึงทำการสร้างส่วน try-catch ขึ้นมา โดยหากการส่งสำเร็จจะทำการแสดงผลบน html ว่า "ส่งอีเมลล์เรียบร้อยแล้ว" และหากไม่สำเร็จจะทำการแสดงข้อผิดพลาดของการส่งออกมา ดังนี้

```
Try {
//สร้าง Object Message ชื่อ msg
Message msg = new MimeMessage(session);
//ตามด้วยการกำหนดค่าให้กับ msg
Msg.setFrom(new InternetAddress (mailFrom));
...
//ทำการส่งอีเมลล์
Transport.send(msg);
Out.println("ส่งอีเมลล์เรียบร้อยแล้ว"); //แสดงข้อความว่าการส่งอีเมลล์ประสบความสำเร็จ
}catch(AddressException ex){ //ตรวจจับข้อผิดพลาดจากการส่งอีเมลล์
Ex.printStackTrace();
Out.println("ไม่สามารถส่งอีเมลล์ได้เนื่องจาก"+ex);
} catch(MessagingException ex){ //ตรวจจับข้อผิดพลาดจากการส่งอีเมลล์
Ex.printStackTrace();
Out.println("ไม่สามารถส่งอีเมลล์ได้เนื่องจาก"+ex);
}
```

รูปที่ 3.25 แสดงตัวอย่างโค้ดส่วน try-catch

3. การตั้งค่าให้กับแอปพลิเคชั่น

Web.xml

ทำการจับคู่ Servlet กับ Class และ url-pattern เพื่อให้รีเควสที่เข้ามาถูกจัดการโดย MailServlet

```
<servlet>
    <servlet-name>mail</servlet-name>
    <servlet-class>mail.MailServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>mail</servlet-name>
    <url-pattern>/mail</url-pattern>
</servlet-mapping>
```

รูปที่ 3.26 แสดงการใช้งานไฟล์ web.xml

Appengine-web.xml

ทำการกำหนดค่า Application ID และ version ของแอปพลิเคชั่น ดังรูป

```
<application>jahcpe2</application>
<version>mail2</version>
```

รูปที่ 3.27 แสดงการค่าในไฟล์ appengine-web.xml

การกำหนดค่าให้แอปพลิเคชั่นส่ง multiple request ไปยังอินสแตนซ์เดียวกันได้พร้อมๆกัน

```
<threadsafe>true</threadsafe>
```

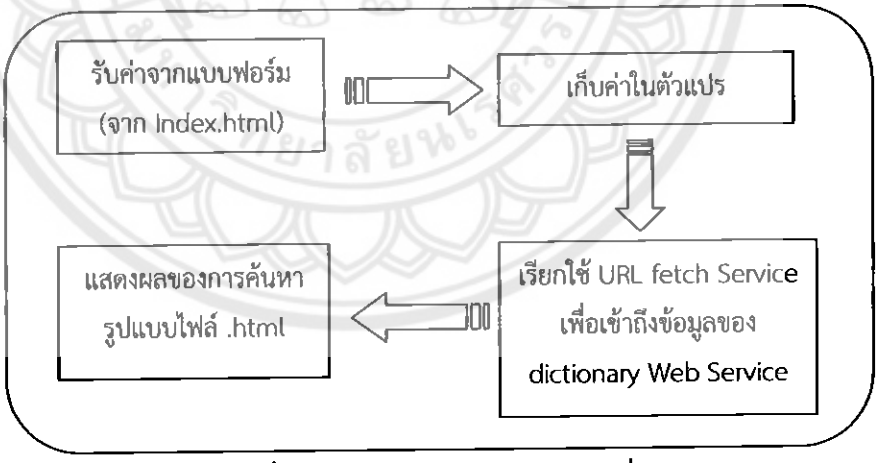
รูปที่ 3.28 แสดงการใช้งาน treadsafe ใน appengine-web.xml

URL Fetch Application

จุดประสงค์ : เพื่อแสดงตัวอย่างการใช้งาน URL Fetch Service เป็นการพัฒนาแอปพลิเคชั่นสำหรับค้นหาความหมายของคำจาก dictionary service ซึ่งเป็น RESTful Web Service โดยทำการเรียกใช้ URL Fetch Service เพื่อดึงข้อมูลจากเว็บไซต์ดังกล่าวมาแสดงผล

ขั้นตอนการพัฒนา :

1. ภาพรวมของแอปพลิเคชั่น

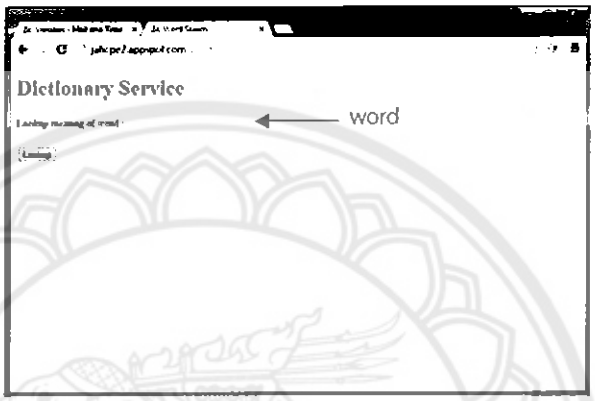


รูปที่ 3.29 แสดงภาพรวมของแอปพลิเคชั่น

2. วิธีการอิมพลีเม้นท์โค้ดส่วนต่างๆ

ส่วนของแบบฟอร์มรับค่า

แบบฟอร์มของการค้นหาคำศัพท์นั้น ต้องการใช้ตัวแปรเดียวเพียงหนึ่งเดียวคือ word ดังนั้นจึงทำการสร้างแบบฟอร์มสำหรับการค้นหาคำศัพท์อย่างง่ายขึ้นมาและทำการกำหนดให้ฟิลด์ "Looking up Meaning of Word" รับค่า word ดังรูป



รูปที่ 3.30 แสดงฟอร์มที่ใช้รับค่า word

ส่วนโค้ด WordServlet.java

ได้กำหนดให้มีการรับค่าตัวแปร 1 ค่าคือ lookupWord สำหรับเก็บตัวแปร "word" จากฟอร์มเพื่อนำไปค้นหาใน dictionary service ดังรูป

```
String lookupWord = request.getParameter("word");
```

รูปที่ 3.31 แสดงการรับค่าจากตัวแปร word

เพื่อที่จะดึงข้อมูลจากเว็บ dictionary service จะต้องทำการสร้าง java.net URL object ขึ้นมาจากนั้นเรียกใช้เมธอด openStream (); เมธอดนี้จะจัดการรายละเอียดในการสร้างการเชื่อมต่อการส่ง HTTP GET request และการเรียกข้อมูลตอบกลับให้ ดังรูป

```
URL url = new URL(urlStr);
BufferedReader reader = new BufferedReader(new InputStreamReader(url.openStream()));
```

รูปที่ 3.32 แสดงการสร้าง url ด้วย java.net

สร้าง response Data แล้วแสดงค่าออกมาในรูปแบบ html

```
StringBuffer responseData = new StringBuffer();
while ((line = reader.readLine()) != null) {
responseData.append(line + "<br>");
}
reader.close();
out.println(responseData);
```

รูปที่ 3.33 แสดงการสร้าง response Data

3. การตั้งค่าให้กับแอปพลิเคชั่น

Web.xml

ทำการจับคู่ Servlet กับ Class และ url-pattern เพื่อให้รีเควสที่เข้ามาถูกจัดการโดย WordServlet

```
<servlet>
    <servlet-name>Word</servlet-name>
    <servlet-class>word.WordServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>Word</servlet-name>
    <url-pattern>/word</url-pattern>
</servlet-mapping>
```

รูปที่ 3.34 แสดงการใช้งานไฟล์ web.xml

Appengine-web.xml

ทำการกำหนดค่า Application ID และ version ของแอปพลิเคชั่น ดังรูป

```
<application>jahcpe2</application>
<version>Word</version>
<threadsafe>true</threadsafe>
```

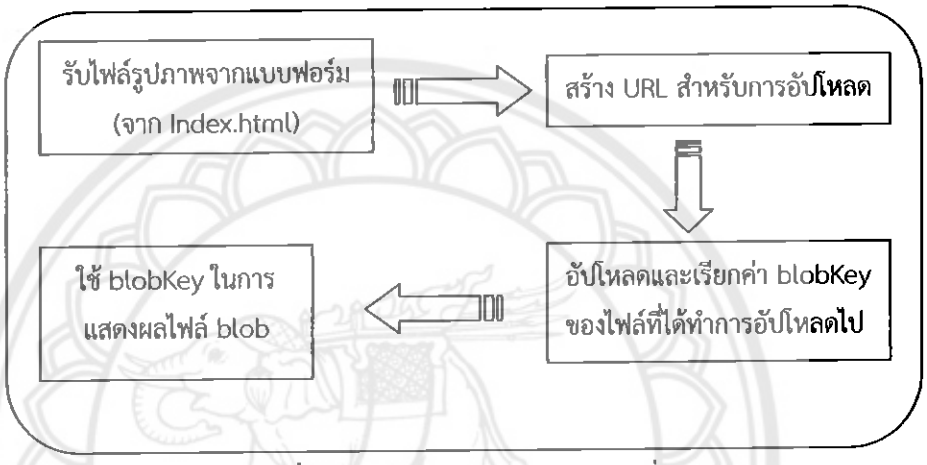
รูปที่ 3.35 แสดงการใช้งานไฟล์ appengine-web.xml

Blobstore Application

จุดประสงค์ : เพื่อแสดงตัวอย่างการใช้งาน Blobstore Service โดยการพัฒนาแอปพลิเคชั่นสำหรับอัปโหลดไฟล์รูปภาพไปยังเว็บแอปพลิเคชั่น และเรียกใช้รูปภาพนั้น

ขั้นตอนการพัฒนา :

1. **ภาพรวมของแอปพลิเคชั่น**



รูปที่ 3.36 แสดงภาพรวมของแอปพลิเคชั่น

2. วิธีการอิมพลีเม้นท์โค้ดส่วนต่างๆ

<u>ส่วนของแบบฟอร์มรับค่า</u>

เพื่อให้ผู้ใช้สามารถอัปโหลดค่า blob ไปยังแอปพลิเคชั่นได้ จำเป็นต้องมีฟอร์ม**สำหรับรับค่าไฟล์**นั้นๆ จึงได้ทำการสร้างฟอร์มสำหรับอัปโหลดไฟล์รูปภาพไปยัง blob โดยกำหนดตัวแปร**myFile เพื่อรับ**ค่าไฟล์ ดังรูป

รูปที่ 3.37 แสดงฟอร์มที่ใช้รับค่าไฟล์เพื่ออัปโหลดไปยัง blobstore

ใช้เมธอด blobstoreService.createUploadUrl ของ blobstore API ในการสร้าง Url สำหรับการอัปโหลดค่า blob ตั้งค่าฟอร์มในการอัปโหลดเป็น post และต้องทำการ enctype ด้วย multipart/form-data เท่านั้น จากนั้นบราวเซอร์จะทำการอัปโหลดไฟล์นั้นโดยตรงผ่านทาง Url ที่สร้างขึ้นมา

```
<form action="<%= blobstoreService.createUploadUrl("/upload") %>" method="post" enctype="multipart/form-data">
```

รูปที่ 3.38 แสดงการอัปโหลดไฟล์ไปยัง blobstore

เมื่อฟอร์มถูกส่ง blob จะถูกสร้างขึ้นด้วย HTTP POST Request ตามเนื้อหาของไฟล์นั้นๆและ Blobstore API จะทำการสร้าง info record สำหรับ blob ขึ้นมาเก็บไว้ในดาต้าสโตร์ และตอบกลับเป็นค่า blob key ซึ่งเอาไว้สำหรับให้บริการ blob แก่ผู้ใช้ โดยเมธอด getUploadedBlobs(); จะตอบกลับเป็นค่า blob ที่ได้ทำการอัปโหลดไปแล้วและ Map object เป็นลิสต์รายชื่อที่เกี่ยวข้องกับฟิลด์ของการอัปโหลดไปยัง blobstore

```
Map<String, BlobKey> blobs = blobstoreService.getUploadedBlobs(req);
BlobKey blobKey = blobs.get("myFile");
```

รูปที่ 3.39 แสดงการเรียกใช้ค่า blob ที่อัปโหลดไปแล้ว

Blob ต้องใช้ค่า blobKey ในการแสดงไฟล์ จึงต้องส่งค่า blobkey ให้กับคลาส serve ต่อ

```
if (blobKey == null) {
    res.sendRedirect("/");
} else {
    res.sendRedirect("/serve?blob-key=" + blobKey.getKeyString());
}
```

รูปที่ 3.40 แสดงโค้ดที่ใช้สำหรับ serve ไฟล์ blob

เรียกใช้งานคลาส serve เพื่อแสดงไฟล์รูปภาพออกมาด้วยเมธอด serve();

```
BlobKey blobKey = new BlobKey(req.getParameter("blob-key"));

blobstoreService.serve(blobKey, res);
```

รูปที่ 3.41 แสดงการเรียกใช้เมธอดเพื่อแสดงไฟล์รูปภาพ

3. การตั้งค่าให้กับแอปพลิเคชั่น

Web.xml

ทำการจับคู่ Servlet กับ Class และ url-pattern ดังรูป

```
<servlet>
    <servlet-name>serve</servlet-name>
    <servlet-class>image.serve</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>serve</servlet-name>
    <url-pattern>/serve</url-pattern>
</servlet-mapping>

<servlet>
    <servlet-name>upload</servlet-name>
    <servlet-class>image.upload</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>upload</servlet-name>
    <url-pattern>/upload</url-pattern>
</servlet-mapping>
```

รูปที่ 3.42 แสดงการใช้งานไฟล์ web.xml

Appengine-web.xml

ทำการกำหนดค่า Application ID และ version ของแอปพลิเคชั่น ดังรูป

```
<application>jahcpe2</application>
<version>Blob</version>
<threadsafe>true</threadsafe>
```

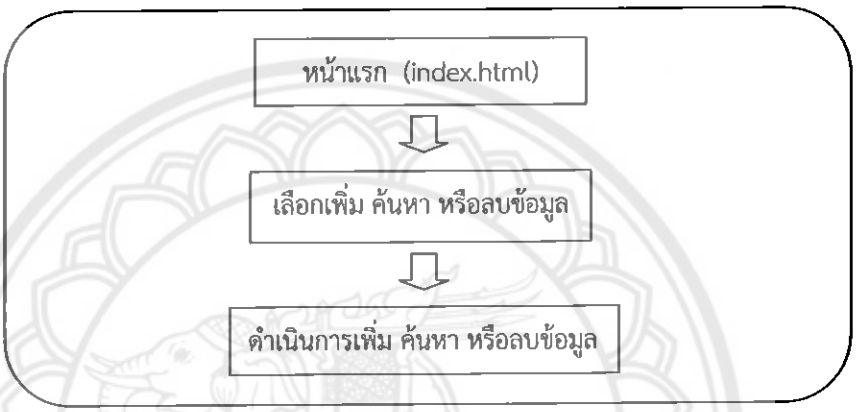
รูปที่ 3.43 แสดงการใช้งานไฟล์ appengine-web.xml

Datastore Application

จุดประสงค์ : เพื่อแสดงตัวอย่างการใช้งาน Java Datastore API โดยการพัฒนาแอปพลิเคชั่นสำหรับเก็บข้อมูลนักศึกษา

ขั้นตอนการพัฒนา :

1. ภาพรวมของแอปพลิเคชั่น



รูปที่ 3.44 แสดงภาพรวมของแอปพลิเคชั่น

2. การเพิ่ม ค้นหา และลบข้อมูล

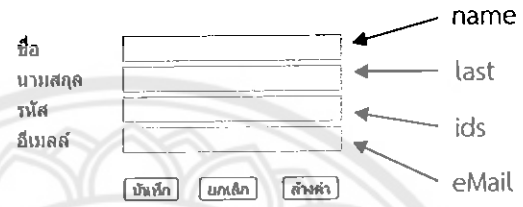
<u>การเพิ่มข้อมูล</u>

การเก็บข้อมูลของแอปพลิเคชั่นในกูเกิ้ลแอปเอ็นจิ้นนั้น ใช้การเก็บข้อมูลเป็น object การเพิ่มข้อมูลนักศึกษานั้นจะเป็นการสร้างเอนติตี้ชนิด student ขึ้นมาใหม่โดยใช้เมธอด new Entity (); แล้วจึงเรียกใช้เมธอด put() ของ DatastoreService เพื่อทำการเพิ่มข้อมูลไปยัง datastore

```
//ส่งลงไปเพิ่มในดาต้าสโตร์โดยใช้ name เป็นคีย์
Entity student = new Entity("Student",name);
student.setProperty("name", name);
student.setProperty("last", last);
student.setProperty("ids", ids);
student.setProperty("eMail", eMail);

datastore.put(student);
```

รูปที่ 3.45 แสดงการเพิ่มข้อมูลไปยังดาต้าสโตร์

เพื่อให้สะดวกต่อการใช้งาน จึงสร้างฟอร์มสำหรับการเพิ่มข้อมูลขึ้นมา และให้ฟอร์มดังกล่าวรับค่าตัวแปรสี่ค่า คือ ฟิลด์ “ชื่อ” รับค่าตัวแปร “name” สำหรับชื่อของนักศึกษา, ฟิลด์ “นามสกุล” รับค่าตัวแปร “last” สำหรับนามสกุลของนักศึกษา, ฟิลด์ “รหัส” รับค่าตัวแปร “ids” สำหรับรหัสนักศึกษา และ ฟิลด์ “อีเมลล์” รับค่าตัวแปร “eMail” ซึ่งเป็นอีเมลล์ของนักศึกษา ดังรูป



รูปที่ 3.46 แสดงฟอร์มที่ใช้รับค่าข้อมูลนักศึกษา

ทำการเรียกค่าจากฟอร์มมาเก็บไว้ในตัวแปร แล้วกำหนดค่าที่ได้รับจากฟอร์มเป็นค่าคุณสมบัติของเอนติตี้ โดยทำการกำหนดค่า “name” ให้เป็นคีย์ของเอนติตี้ด้วย

```
//รับค่าจากไฟล์ input.html
String name = req.getParameter("name");
String last = req.getParameter("last");
String ids = req.getParameter("ids");
String eMail = req.getParameter("eMail");

//ค่าตัวไปใส่ตัวแปรเก็บค่าจากไฟล์ที่จะใช้ name เป็นคีย์
Entity student = new Entity("Student",name);
student.setProperty("name", name);
student.setProperty("last", last);
student.setProperty("ids", ids);
student.setProperty("eMail", eMail);

datastore.put(student);
```

รูปที่ 3.47 แสดงการเรียกค่าจากฟอร์มมาใส่ไว้ในตัวแปร

เมื่อการเพิ่มเอนติตี้ประสบความสำเร็จให้ทำการแสดงรายละเอียดคุณสมบัติต่างๆของเอนติตี้ออกมา พร้อมกับแสดงข้อความ “บันทึกข้อมูลนักศึกษาแล้ว”

```
out.println("<br> คีย์ : " +studentKey);
out.println("<br> ชื่อ : " +student.getProperty("name"));
out.println("<br> นามสกุล : " +student.getProperty("last"));
out.println("<br> รหัส : " +student.getProperty("ids"));
out.println("<br> อีเมลล์ : " +student.getProperty("eMail"));
out.println("<br><br> บันทึกข้อมูลนักศึกษาแล้ว ");
```

รูปที่ 3.48 แสดงโค้ดสำหรับการแสดงรายคุณสมบัติต่างๆของเอนติตี้

การค้นหาข้อมูล

การค้นหาข้อมูลของเอนติตี้นั้น ใช้การรับค่าตัวแปรเพียงแค่ค่าเดียว คือ targetName เนื่องจากแอปพลิเคชั่นนี้ใช้ค่าคีย์เป็น “name” จึงทำการค้นหาโดยใช้ชื่อของเอนติตี้ มีฟอร์มในการรับค่า ดังรูป

รูปที่ 3.49 แสดงฟอร์มสำหรับการค้นหาข้อมูลนักศึกษา

ทำการเรียกค่าจากฟอร์มมาเก็บไว้ในตัวแปรชื่อ targetName

```
//รับค่าจากไฟล์ get.html
String targetName = req.getParameter("targetName");
```

รูปที่ 3.50 แสดงการเก็บค่าในตัวแปร targetName

การดึงข้อมูลในดาต้าสโตร์ออกมาแสดงนั้น จำเป็นต้องใช้ค่าคีย์ในการระบุเอนติตี้ จึงทำการสร้างค่าตัวแปรที่รับมาเป็นค่าคีย์ ดังรูป

```
Key studentKey = KeyFactory.createKey("Student", targetName);
System.out.println("studentKey: "+ studentKey);
out.println("<br> คีย์Key : " +studentKey);
```

รูปที่ 3.51 แสดงการสร้างคีย์

นำค่าคีย์ที่ได้มาใช้ในการดึงข้อมูลจากดาต้าสโตร์ ด้วยเมธอด get (); ดังรูป

```
Entity s = datastore.get(studentKey);
```

รูปที่ 3.52 แสดงการดึงข้อมูลจากดาต้าสโตร์

เมื่อพบข้อมูลให้ทำการแสดงค่าคุณสมบัติของเอนติตี้ออกมายังหน้า html ดังรูป

```
out.println("<br> พบข้อมูล found for"+studentKey);
out.println("<br> Name : " + s.getProperty("name"));
out.println("<br> Last : " + s.getProperty("last"));
out.println("<br> ID : " + s.getProperty("IDS"));
out.println("<br> EMail : " + s.getProperty("eMail"));
out.println("<br> <a href= /delete> ลบรายชื่อ</a>" + "หรือ"+"<a href= input.html > แก้ไข</a>");
out.println("<br> <a href= /> ย้อนกลับไปหน้าที่แล้ว</a>" + "หรือ"+"<a href= index.html > หน้าแรก</a>");
out.println("<br> <a href= /all> ดูรายชื่อทั้งหมด</a>");
```

รูปที่ 3.53 แสดงการเรียกค่าคุณสมบัติของเอนติตี้มาแสดง

หากไม่พบข้อมูลให้ทำการแสดง exception ออกมา ดังรูป

```
catch(EntityNotFoundException ex){
   ex.printStackTrace();
   out.println("ไม่พบข้อมูล not found <br>"+studentKey +"<br>"+ex);
```

รูปที่ 3.54 แสดงการ catch exception

<u>การลบข้อมูล</u>

การลบข้อมูลนั้นจะทำได้จากการส่งค่าคีย์ของเอนติตี้ให้กับเมธอดเช่นเดียวกันกับการดึงข้อมูลจากดาต้าสโตร์ แต่หากไม่ทราบว่าในดาต้าสโตร์มีเอนติตี้ใดอยู่ ก็จะไม่สามารถทำการลบข้อมูลได้ ดังนั้นการลบข้อมูลของเอนติตี้จึงถูกกำหนดให้เป็นการกระทำที่ต่อเนื่องมาจากการค้นหาเอนติตี้ เมื่อเจอเอนติตี้แล้วจึงจะสามารถทำการลบข้อมูลของเอนติตี้นั้นได้ ฉะนั้นค่าคีย์ที่ใช้ในการลบจะต้องเป็นคีย์เดียวกันกับค่าคีย์ในการค้นหา ซึ่งค่าคีย์จากการค้นหานั้นได้มาจากตัวแปร targetName จึงทำการประกาศตัวแปร targetName ให้สามารถใช้ร่วมกันได้ ดังรูป

```
String targetName = req.getParameter("targetName");

Key studentKey = KeyFactory.createKey("Student", targetName);
System.out.println("studentKey: "+ studentKey);
out.println("<br> คีย์Key : " +studentKey);

c = req.getParameter("targetName");
```

รูปที่ 3.55 แสดงการประกาศตัวแปร เพื่อเรียกใช้ targetName

จากนั้นจึงทำการสร้างคีย์จากค่า targetName อีกครั้ง

```
Key studentKey = KeyFactory.createKey("Student", ShowServlet.c);
```

รูปที่ 3.56 แสดงการสร้างค่าคีย์เพื่อลบเอนติตี้

ใช้ค่าคีย์ที่ได้ในการลบเอนติตี้ด้วยเมธอด delete (); ของ DatastoreService

```
datastore.delete(studentKey);
```

รูปที่ 3.57 แสดงการใช้งานเมธอด delete ();

เมื่อต้องการตรวจเช็คว่าการลบเอนติตี้ประสบความสำเร็จหรือล้มเหลวอย่างไร **จึงได้ทำการสร้าง** ส่วน try- catch ขึ้นมา

```
try {
    datastore.delete(studentKey);
    System.out.println("printconsole:"+ studentKey);
    out.println("<br> ทำการลบข้อมูลของนักศึกษา deleted "+studentKey +"เรียบร้อยแล้ว");
    out.println("<br> <a href = get.html> ค้นหา</a>" + "หรือ"+"<a href= input.html > เพิ่มรายชื่อใหม่</a>");

}catch(Exception ex){
    ex.printStackTrace();
    out.println("ไม่สามารถลบข้อมูลได้เนื่องจาก<br>"+ex);
    out.println(msg);
    out.println(msg3);
}
```

รูปที่ 3.58 แสดงการสร้าง try-catch

3. การตั้งค่าแอปพลิเคชั่น

Web.xml

ทำการจับคู่ Servlet กับ Class และ url-pattern ดังรูป

```
<servlet>
    <servlet-name>01</servlet-name>
    <servlet-class>ol.OlServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>01</servlet-name>
    <url-pattern>/ol</url-pattern>
</servlet-mapping>

<servlet>
    <servlet-name>show</servlet-name>
    <servlet-class>ol.ShowServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>show</servlet-name>
    <url-pattern>/show</url-pattern>
</servlet-mapping>

<servlet-mapping>
     <servlet-name>delete</servlet-name>
     <url-pattern>/delete</url-pattern>
</servlet-mapping>
<servlet>
```

รูปที่ 3.59 แสดงการใช้งานไฟล์ web.xml

Appengine-web.xml

ทำการกำหนดค่า Application ID และ version ของแอปพลิเคชั่น ดังรูป

```
<application>jahcpe2</application>
<version>datastore</version>
<threadsafe>true</threadsafe>
```

รูปที่ 3.60 แสดงการใช้งานไฟล์ appengine-web.xml

3.3 จัดทำเนื้อหาในรูปแบบหนังสือคู่มือ

การจัดทำหนังสือคู่มือนั้นเป็นการนำองค์ความรู้ที่ได้จากในขั้นตอนแรกมาใช้งานจริง แล้วจึงสรุปการใช้งานของคุณลักษณะต่างๆออกมา เพื่อให้ผู้อ่านได้ทราบถึงรายละเอียดเกี่ยวกับคุณลักษณะต่างๆของกูเกิ้ลแอปเอ็นจิ้นรวมทั้งการใช้งานคุณลักษณะเหล่านั้นในการพัฒนาจาวาเว็บแอปพลิเคชั่น

ซึ่งก่อนที่จะทำความรู้จักกับคุณลักษณะต่างๆของกูเกิ้ลแอปเอ็นจิ้นนั้น ผู้ใช้จะต้องมีความรู้เบื้องต้นเกี่ยวกับเว็บแอปพลิเคชั่นและสภาพแวดล้อมต่างๆในการพัฒนาเสียก่อน ซึ่งความรู้ในส่วนนี้จะถูกรวบรวมไว้ในหนังสือเช่นกัน และเมื่อนำมารวมกันกับเนื้อหาหลักคือคุณลักษณะต่างๆของกูเกิ้ลแอปเอ็นจิ้น ทำให้ได้โครงเรื่องเนื้อหาหลักของหนังสือ ดังนี้

บทที่ 1 : ความรู้เบื้องต้น

เป็นส่วนของการแนะนำและให้ความรู้ต่างๆที่เกี่ยวข้องกับเว็บแอปพลิเคชั่นและภาษาที่ใช้ในการพัฒนา รวมไปถึงวิธีการติดตั้งและใช้งาน ประกอบไปด้วยเนื้อหา 2 ส่วนหลัก คือ

1. เว็บแอปพลิเคชั่นและภาษาจาวา เป็นการบรรยายถึงความรู้พื้นฐานเกี่ยวกับเว็บแอปพลิเคชั่น ซึ่งอธิบายถึงความหมายของเว็บแอปพลิเคชั่น ส่วนประกอบ หลักการทำงาน และการใช้งานไฟล์ต่างๆร่วมกับการสร้างเว็บแอปพลิเคชั่น เป็นต้น

2. คลาวด์คอมพิวติ้งและกูเกิ้ลแอปเอ็นจิ้น เป็นการบรรยายถึงความรู้ทั่วไปเกี่ยวกับเทคโนโลยีคลาวด์คอมพิวติ้งซึ่งครอบคลุมเนื้อหาในเรื่องความเป็นมาของเทคโนโลยีคลาวด์คอมพิวติ้ง โครงสร้างการใช้งาน และรูปแบบการให้บริการบนคลาวด์ รวมไปถึงประโยชน์และข้อจำกัดการใช้งานของคลาวด์คอมพิวติ้งด้วย อีกทั้งในหัวข้อนี้ยังเป็นส่วนที่อธิบายถึงกูเกิ้ลแอปเอ็นจิ้นในลักษณะโดยภาพรวมและบรรยายถึงเทคโนโลยีในการพัฒนาร่วมกับภาษาจาวาอีกด้วย

บทที่ 2 : คุณลักษณะต่างๆของกูเกิ้ลแอปเอ็นจิ้นและการใช้งาน

เป็นการนำคุณลักษณะต่างๆของกูเกิ้ลแอปเอ็นจิ้นที่ได้ศึกษามาสรุปเป็นองค์ความรู้โดยแบ่งเป็น 3 ส่วนหลักตามคุณลักษณะของกูเกิ้ลแอปเอ็นจิ้น คือ

1. คุณลักษณะต่างๆที่ใช้ในการพัฒนาเว็บแอปพลิเคชั่น ซึ่งประกอบไปด้วย บทความเรื่องเครื่องมือ พื้นที่การเก็บข้อมูล และ Service ที่แอปพลิเคชั่นของกูเกิ้ลแอปเอ็นจิ้นสามารถใช้ได้ เช่น Mail, Image, User Feth, User API เป็นต้น

2. คุณลักษณะต่างๆที่ใช้ในการกำหนดค่าแอปพลิเคชั่น เป็นการบรรยายคุณลักษณะที่ใช้สำหรับการกำหนดค่าต่างๆให้แก่แอปพลิเคชั่น ซึ่งแต่ละไฟล์ต่างก็มีการใช้ในรูปแบบที่แตกต่างกันไป เช่น web.xml appengine-web.xml queue.xml index.xml รวมไปถึงไฟล์ cron.xml ด้วย

3. คุณลักษณะต่างๆที่ใช้ในการดูแลและจัดการแอปพลิเคชั่น เป็นการบรรยายถึงส่วนต่างๆที่ใช้ในการจัดการเว็บแอปพลิเคชั่น เช่น แบ็กเอ็นด์และ Admin Console ซึ่งจะเน้นการอธิบายถึงส่วนต่างๆของแอดมินคอนโซลและการใช้งานต่างๆ เช่น การกำหนดค่าให้กับแอปพลิเคชั่น การแบ็กอัพข้อมูลในดาต้าสโตร์ เป็นต้น

บทที่ 3 : การพัฒนาเว็บแอปพลิเคชั่น

เป็นการกล่าวถึงขั้นตอนในการพัฒนาเว็บแอปพลิเคชั่น อุปกรณ์พื้นฐานและความรู้ที่ต้องใช้ รวมถึงเมื่อทำการพัฒนาเว็บแอปพลิเคชั่นเสร็จสมบูรณ์แล้ว ก็มีอธิบายการทำงานประกอบ ประกอบไปด้วยรายชื่อของแอปพลิเคชั่นต่างๆ การอธิบายถึงขั้นตอนในการพัฒนาและลักษณะของเว็บแอปพลิเคชั่นที่ได้เสร็จสมบูรณ์ รวมไปถึงซอร์สโค้ดและลิ้งค์ของตัวอย่างแอปพลิเคชั่นด้วย

บทที่ 4 : สรุปผลการใช้งาน

เป็นการกล่าวถึงผลของการใช้กูเกิ้ลแอปเอ็นจิ้นในการพัฒนาเว็บแอปพลิเคชั่นว่ามีข้อดีข้อด้อยอย่างไรบ้างและเปรียบเทียบแพลตฟอร์มกูเกิ้ลแอปเอ็นจิ้นกับผลิตภัณฑ์ประเภทเดียวกันของผู้ประกอบการรายอื่น

บทที่ 4
ผลการทดลอง

ในบทนี้จะกล่าวถึงผลจากการศึกษาคุณลักษณะต่างๆของกูเกิ้ลแอปเอ็นจิ้น ผลจากการพัฒนาเว็บแอปพลิเคชั่นขึ้นมาประกอบการอธิบาย และผลของการจัดทำหนังสือคู่มือ ซึ่งมีรายละเอียดเนื้อหาดังต่อไปนี้

4.1 ผลจากการศึกษาคุณลักษณะต่างๆของกูเกิ้ลแอปเอ็นจิ้น

จากการศึกษาถึงคุณลักษณะต่างๆของกูเกิ้ลแอปเอ็นจิ้นพบว่า แต่ละคุณลักษณะนั้นก็มีความสำคัญในประการที่แตกต่างกันไป ดังนี้

ส่วนของการพัฒนาเว็บแอปพลิเคชั่น

1. เครื่องมือ สำหรับการพัฒนาเว็บแอปพลิเคชั่นด้วยจาวา หากใช้เครื่องมือสำหรับช่วยในการพัฒนา เช่น Google App Engine Plugin, App Engine Java SDK และโปรแกรม Eclipse จะทำให้การพัฒนาโปรแกรมสามารถทำได้ง่ายและรวดเร็วขึ้น สำหรับวิธีการติดตั้งและใช้งานจะถูกอธิบายใน (บทที่ 2 : เครื่องมือ) ของหนังสือคู่มือการพัฒนาเว็บแอปพลิเคชั่นด้วยภาษาจาวาบนกูเกิ้ลแอปเอ็นจิ้นแพลตฟอร์ม

2. พื้นที่เก็บข้อมูล สำหรับการเก็บข้อมูลบนนั้นแอปเอ็นจิ้นมีตัวเลือกที่สำคัญอยู่ 3 แบบด้วยกันคือ Java Datastore API (JDO /JPA /Low-level), Google Cloud SQL และ Google Cloud Storage ซึ่งตัวเลือกทั้งสามแบบนี้ก็มีข้อดีข้อเสียแตกต่างกันไป โดย JDO/JPA นั้นเหมาะสำหรับนักพัฒนาที่ต้องการความสะดวกในการใช้งานอินเตอร์เฟสนี้เข้าถึงฐานข้อมูล โดย JDO สามารถเข้ากันกับฐานข้อมูลแบบใดๆก็ได้ แต่ JPA นั้นรองรับเฉพาะฐานข้อมูลแบบ RDBMS สามารถทำการ transaction ได้ โดยทั่วไปอาจจะมองว่า JPA เป็นส่วนหนึ่งของ JDO ก็ได้ เพราะในเรื่องของความยืดหยุ่นแล้ว JDO มีมากกว่า แต่ทั้งสองต่างก็มีข้อจำกัดคือเมื่อมี Entity เยอะขึ้น การแปลงเอนติตี้ไปเป็นออปเจ็กอาจจะต้องใช้เวลามาก จึงอาจจะไม่เหมาะกับฐานข้อมูลที่มีขนาดใหญ่มากๆ สำหรับนักพัฒนาแล้ว JDO และ JPA เป็นตัวเลือกที่ดีในการใช้งาน แต่สำหรับมือใหม่ที่เพิ่งรู้จักกับกูเกิ้ลแอปเอ็นจิ้นอาจจะเลือกใช้งานในส่วนของ Low level API ก็ได้เนื่องจากมีความซับซ้อนน้อยกว่าและจะเป็นพื้นฐานในการนำไปใช้งานต่อในอนาคตได้ ส่วน Google Cloud SQL นั้นคล้ายคลึงกับฐานข้อมูล MySQL อย่างมาก แต่มีข้อเสียคือการใช้งานที่มีค่าใช้จ่ายตั้งแต่เริ่มใช้ ไม่มีโควตาฟรีให้ทดลองใช้งาน ส่วน Google Cloud Storage นั้นเหมาะกับข้อมูลที่มีลักษณะเป็นไฟล์ หรือออปเจ็กซึ่งมีขนาดใหญ่ ซึ่ง Google Cloud Storage นั้นก็มีบริการฟรี

ให้ทดลองใช้งานด้วย แต่เป็นการให้บริการที่มีข้อจำกัดและสามารถสร้างได้แค่เพียงครั้งเดียวเท่านั้น ดังนั้นในการศึกษาคุณลักษณะสำหรับการเก็บข้อมูลจึงเน้นไปที่เรื่องของการใช้งาน Low-level API สำหรับการเก็บข้อมูล ซึ่งการศึกษาและใช้งานจะถูกอธิบายไว้ใน (บทที่ 2: พื้นที่เก็บข้อมูล) ของหนังสือคู่มือการพัฒนาเว็บแอปพลิเคชั่นด้วยภาษาจาวาบนกูเกิ้ลแอปเอ็นจิ้นแพลตฟอร์ม

3. Service ต่างๆของกูเกิ้ลที่มีให้กับแอปพลิเคชั่นที่ทำงานบนกูเกิ้ลแอปเอ็นจิ้น เช่น UserAPI, BlobstoreAPI, MailAPI, URL-FetchAPI, MemcacheAPI, ChannelAPI และ LogAPI ซึ่งจากการศึกษาพบว่าแอปเอ็นจิ้นสามารถใช้ service ทุกอย่างของกูเกิ้ลได้ โดยเริ่มจากโควตาการใช้งานฟรีก่อน และหากมีการใช้งานเกินก็สามารถอัปเกรดบัญชีให้เป็นบัญชีแบบมีค่าใช้จ่ายได้ต่อไป เป็นต้น ซึ่ง Service ต่างๆเหล่านี้เป็น service พื้นฐานที่ Google มีให้แก่นักพัฒนาเพื่อช่วยให้การพัฒนาเว็บแอปพลิเคชั่นสำหรับเชื่อมต่อไปยังการให้บริการในส่วนต่างๆทำได้ง่าย และรวดเร็วขึ้น สำหรับการศึกษาและใช้งาน service ต่างๆจะถูกกล่าวถึงใน (บทที่ 2: Services) ของหนังสือคู่มือการพัฒนาเว็บแอปพลิเคชั่นด้วยภาษาจาวาบนกูเกิ้ลแอปเอ็นจิ้นแพลตฟอร์ม

ส่วนของการกำหนดค่าแอปพลิเคชั่น

กูเกิ้ลแอปเอ็นจิ้นให้การเข้าถึงแก่นักพัฒนาในการเข้าไปจัดการในส่วนต่างๆของแอปพลิเคชั่นได้ตามที่นักพัฒนาต้องการ ซึ่งในบางคุณลักษณะอาจจะมีข้อจำกัดเพิ่มมาจากการใช้งานคุณลักษณะอื่นๆเล็กน้อย แต่โดยรวมแล้วถือว่า คุณลักษณะต่างๆเหล่านี้เป็นประโยชน์ต่อการจัดการแอปพลิเคชั่นอย่างมาก เช่น App Config ที่ช่วยตั้งค่าต่างๆของแอปพลิเคชั่น (เช่น การกำหนดเวอร์ชั่นให้กับแอปพลิเคชั่น และการกำหนดว่าไฟล์ใดเป็น static files) ก่อนที่จะทำการอัปโหลดและเผยแพร่ไปยังกูเกิ้ลแอปเอ็นจิ้นได้ หรือการใช้งาน Task queue ในการจัดการคิวของ Task ที่รอการประมวลผลได้ เป็นต้น สำหรับการศึกษาและใช้งานจะถูกอธิบายไว้ใน (บทที่ 2: คุณลักษณะที่ใช้ในการตั้งกำหนดค่าแอปพลิเคชั่น) ของหนังสือคู่มือการพัฒนาเว็บแอปพลิเคชั่นด้วยภาษาจาวาบนกูเกิ้ลแอปเอ็นจิ้นแพลตฟอร์ม

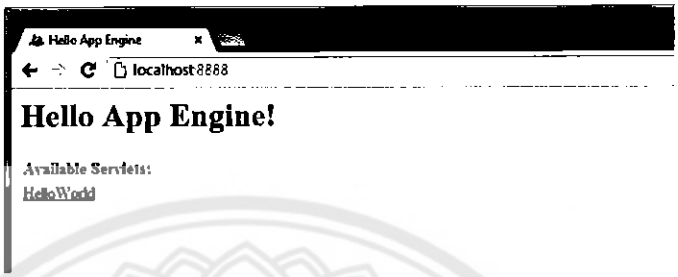
ส่วนของการดูแลจัดการแอปพลิเคชั่น

สำหรับการดูแลจัดการแอปพลิเคชั่น มีวิธีการทำได้ 2 วิธีคือ ทำโดยการใช้แบ็กเอ็นด์ และโดยการใช้ Admin Console ซึ่งสำหรับการใช้งานในแอดมินคอนโซลนั้นจะเป็นการเข้าถึงในรูปแบบของ UI (User Interface) ซึ่งสามารถใช้งานได้ง่ายและสะดวกต่อการดูแลจัดการแอปพลิเคชั่นในเรื่องต่างๆ เช่น ดูปริมาณการเข้าชมแอปพลิเคชั่น ปริมาณพื้นที่เก็บที่ถูกใช้ไปในการเก็บโค้ดและไฟล์ต่างๆ การตรวจดู Log ของแอปพลิเคชั่น การตรวจดูปริมาณค่าโควตาในแต่ละวัน และค่าโควตาการใช้งานแบบรายเดือนได้ด้วย ผลของการศึกษาคุณลักษณะต่างๆต่อไปนี้จะถูกจัดทำออกมาในรูปแบบของหนังสือคู่มือ

4.2 ผลการพัฒนาเว็บแอปพลิเคชั่น

Hello World Application

ผลการทำงานของแอปพลิเคชั่น :



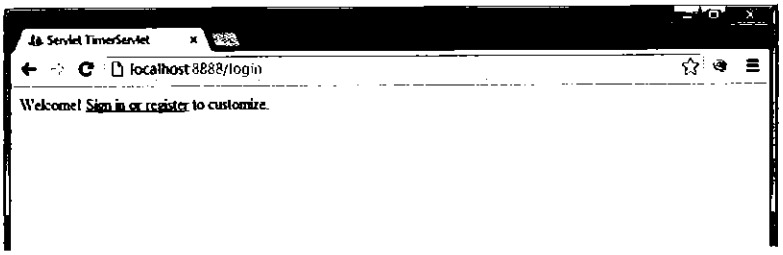
รูปที่ 4.1 แสดงหน้าเว็บของ Hello World เมื่อเข้าไปยัง http://localhost:8888 จากเว็บบราวเซอร์

รูปที่ 4.2 แสดงเนื้อหาหน้าเว็บเมื่อเรียกใช้ Servlet ของ Hello World

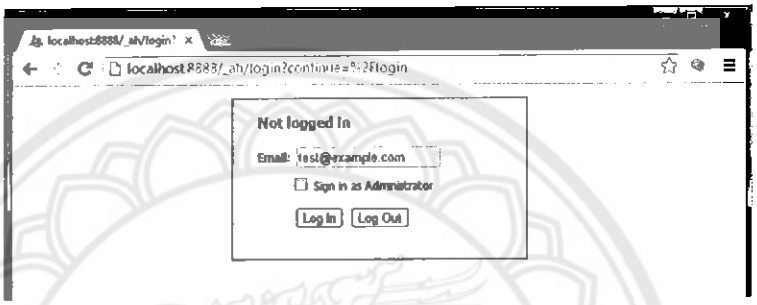
Login Application

ผลการทำงานของแอปพลิเคชั่น :

รูปที่ 4.3 แสดงหน้าเว็บของ Login เมื่อเข้าไปยัง http://localhost:8888 จากเว็บบราวเซอร์

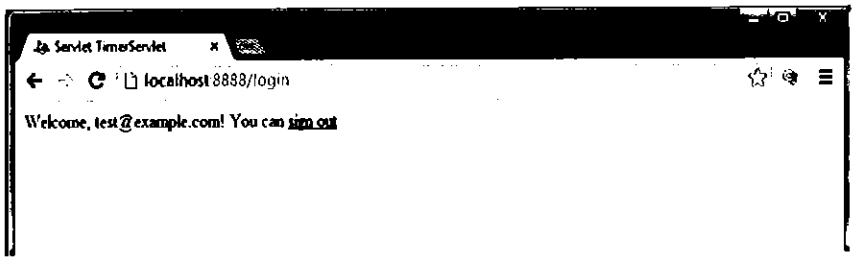


รูปที่ 4.4 แสดงลิ้งค์สำหรับการลงชื่อเข้าสู่ระบบ จากการเรียก Login servlet

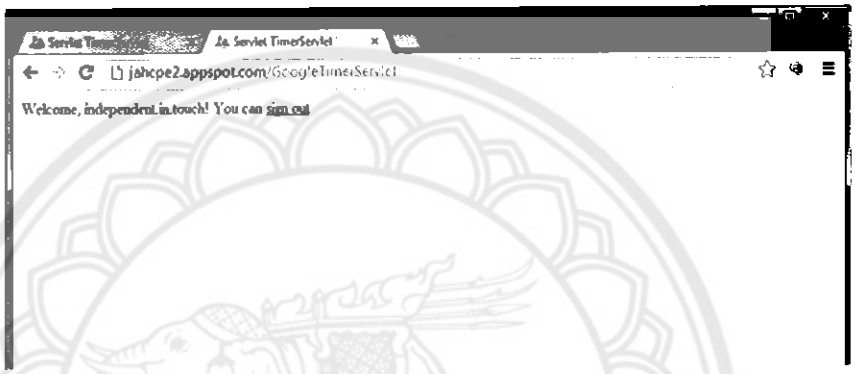


รูปที่ 4.5 แสดงหน้าสำหรับการลงชื่อเข้าใช้เมื่อรันที่ http://localhost:8888 จำลองหน้าลงชื่อเข้าสู่ระบบโดยใช้ App Engine Java SDK

รูปที่ 4.6 แสดงหน้าสำหรับการลงชื่อเข้าใช้จริงบน Google App Engine โดยใช้ Google Account (จะทำได้ก็ต่อเมื่อทำการ Deploy เว็บแอปพลิเคชั่นไปยัง Google App Engine แล้วเท่านั้น)



รูปที่ 4.7 แสดงหน้าเว็บเมื่อทำการลงชื่อเข้าใช้ใน http://localhost:8888



รูปที่ 4.8 แสดงหน้าเว็บเมื่อทำการลงชื่อเข้าใช้บน Google App Engine โดยใช้ Google Account

รูปที่ 4.9 แสดงหน้าเว็บเมื่อคลิกที่ลิ้งค์ Sign out ซึ่งเป็นการออกจากระบบของผู้ใช้แอปพลิเคชั่นจะนำผู้ใช้กลับไปหน้าแรกก่อนการ Sign in

Mail Application

1. หมายเหตุ :

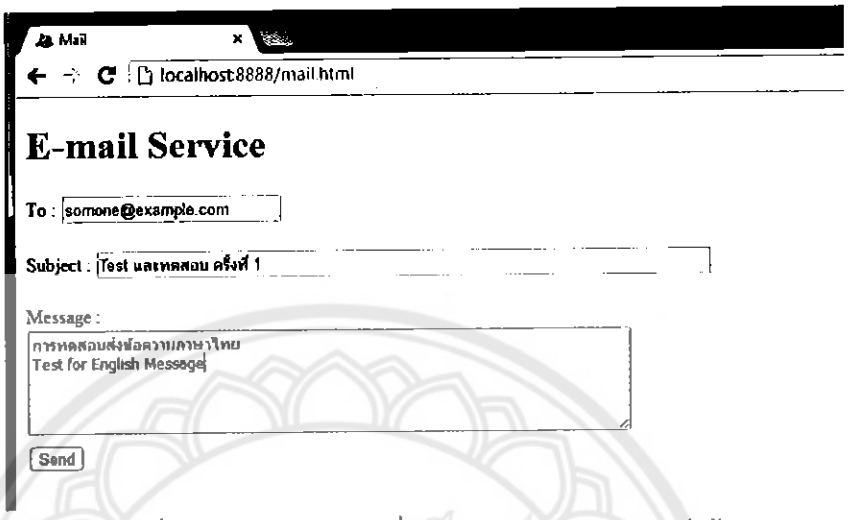
(1) สำหรับ Mail แอปพลิเคชั่นนั้น Development Server จะไม่สามารถทำการส่งอีเมลล์จริงได้ แต่นักพัฒนาสามารถตรวจสอบค่าที่ได้จากการดูที่คอนโซลสำหรับการเริ่มต้นการทำงานของเซิร์ฟเวอร์ของแอปพลิเคชั่นใน Eclipse

(2) แอปพลิเคชั่นเมลล์จะสามารถส่งค่าอีเมลล์ใน Development Server ได้เมื่อมีการลงชื่อเข้าใช้แอปพลิเคชั่นแล้ว (จาก Login Application)

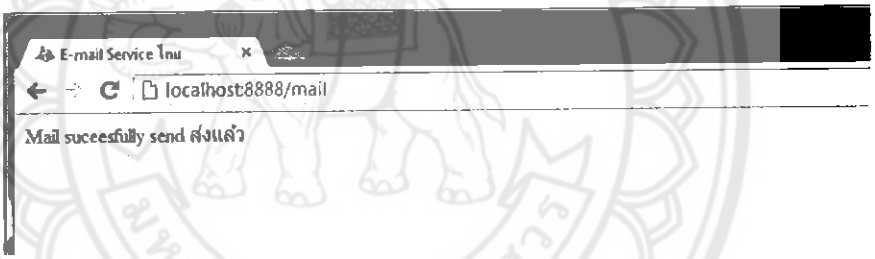
2. ผลการทำงานของแอปพลิเคชั่น :

รูปที่ 4.10 แสดงหน้าเว็บของแอปพลิเคชั่น Mail เมื่อเข้าไปยัง http://localhost:8888

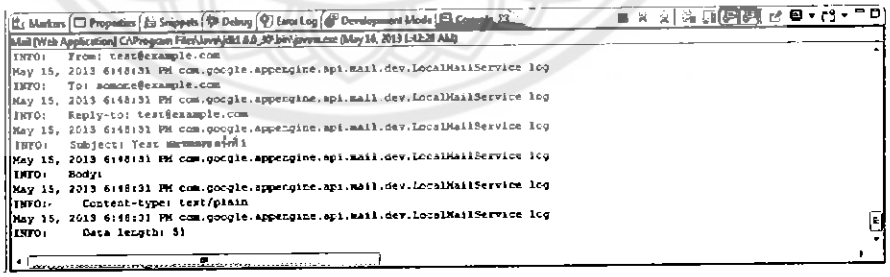
รูปที่ 4.11 แสดงหน้า mail.html ซึ่งเป็นฟอร์มสำหรับการส่ง Email เมื่อเรียกใช้งาน Mail Servlet



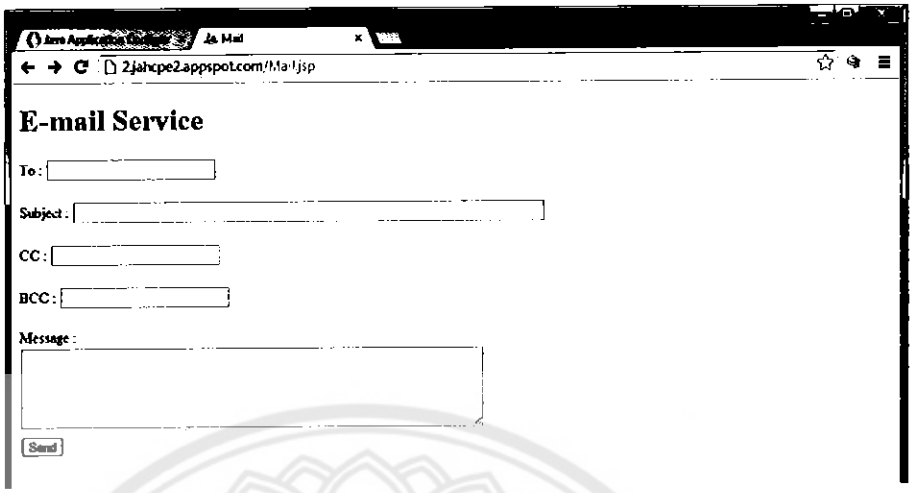
รูปที่ 4.12 แสดงหน้าฟอร์มที่กรอกข้อความจนครบทุกฟิลด์แล้ว



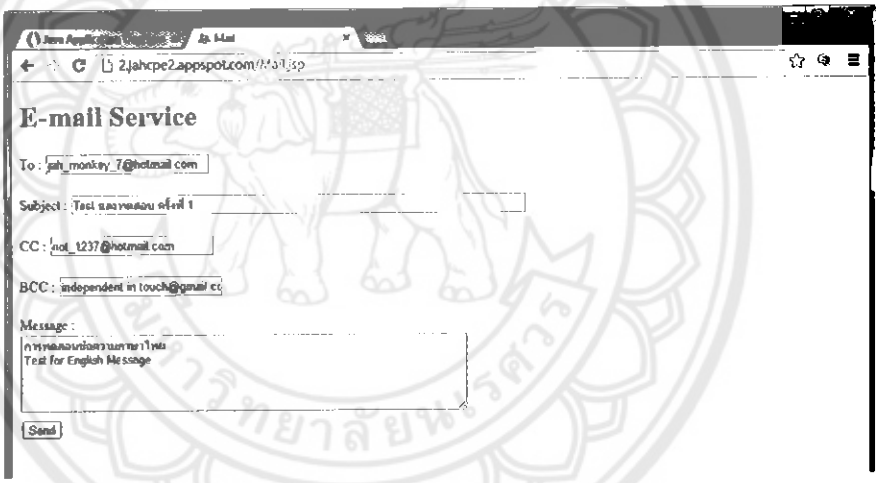
รูปที่ 4.13 แสดงหน้าเว็บหลังจากคลิกปุ่ม send



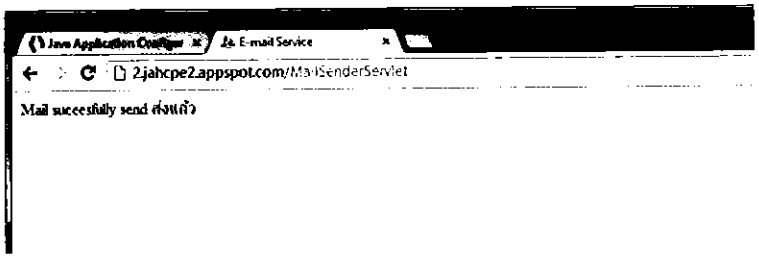
รูปที่ 4.14 แสดงการจับข้อมูลที่หน้า console หลังจากคลิกปุ่ม send



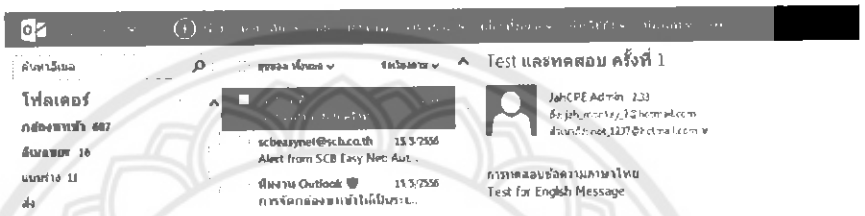
รูปที่ 4.15 แสดงฟอร์มการส่งอีเมลล์ของแอปพลิเคชั่นที่ทำงานอยู่บน Google App Engine



รูปที่ 4.16 แสดงการกรอกข้อมูลจนครบหมดทุกฟิลด์



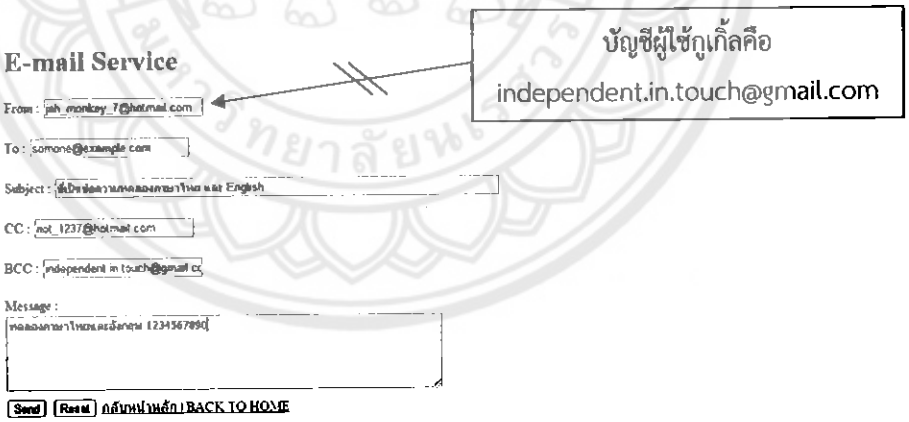
รูปที่ 4.17 แสดงหน้าเว็บหลังจากที่กดปุ่ม Send



รูปที่ 4.18 แสดงข้อความที่ได้รับในกล่องรับข้อความ

3. การทำงานของแอปพลิเคชั่นเมื่อเกิด Exception

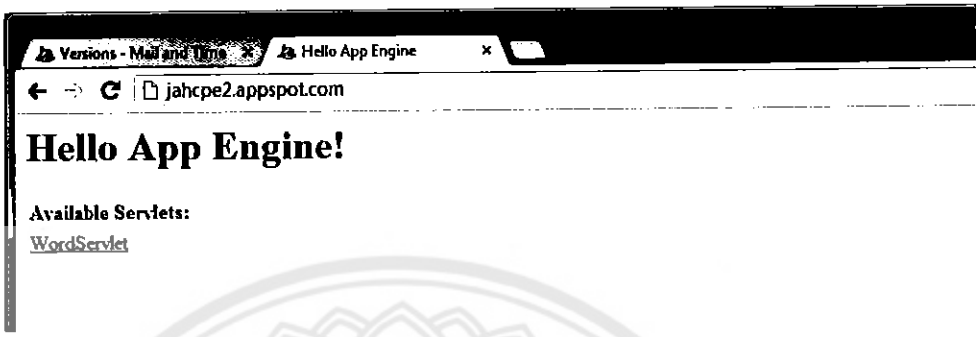
เมื่อที่อยู่อีเมลล์ของผู้ส่งไม่ตรงกันกับบัญชีผู้ใช้ของกูเกิ้ล จะทำให้การส่งอีเมลล์ล้มเหลว เนื่องจากไม่สามารถระบุผู้ส่งอีเมลล์ได้



รูปที่ 4.19 แสดงการใส่ค่าที่อยู่อีเมลล์ที่ไม่ตรงกันกับบัญชีผู้ใช้กูเกิ้ล

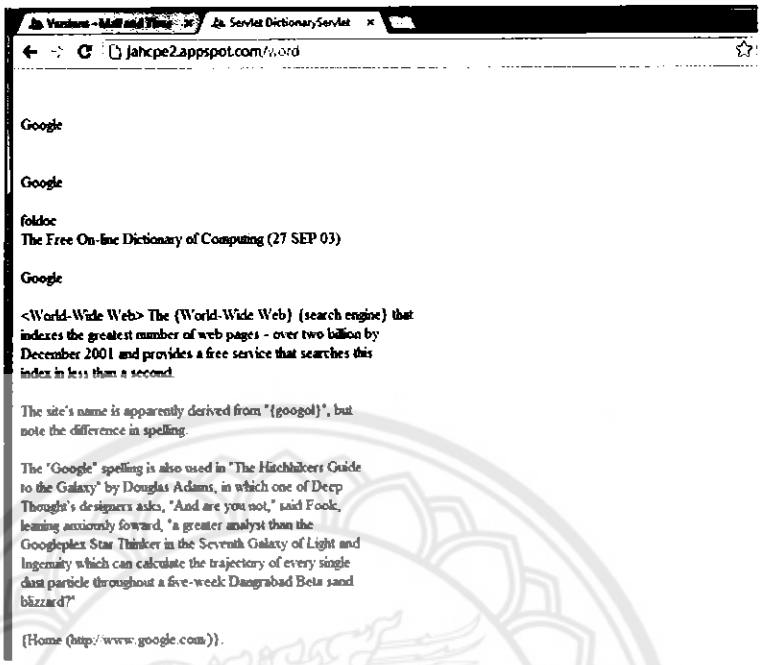
Word Search Application

ผลการทำงานของแอปพลิเคชั่น :



รูปที่ 4.20 แสดงหน้าเว็บของแอปพลิเคชั่นเมื่อถูกอัปโหลดไปยัง Google App Engine

รูปที่ 4.21 แสดงการเรียกใช้งาน WordServlet และพิมพ์ที่ฟิลด์กรอกข้อมูลว่า Google เพื่อค้นหาความหมายของคำ

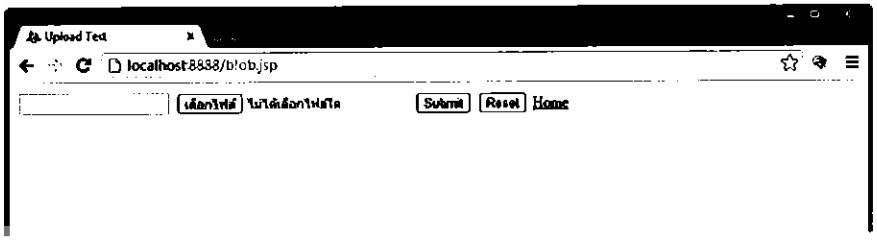


รูปที่ 4.22 แสดงผลลัพธ์ของการค้นหาความหมายคำว่า Google

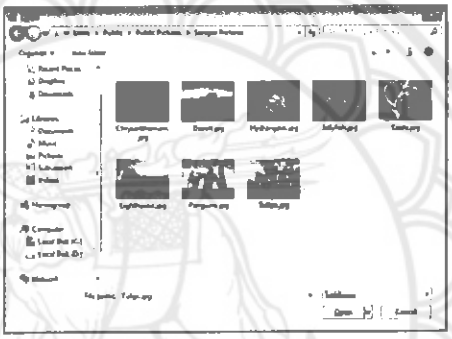
รูปที่ 4.23 แสดงผลของการค้นหาที่ผลลัพธ์ออกมาเป็นค่าว่าง เนื่องจากไม่พบข้อมูล

Blobstore Application

ผลการทำงานของแอปพลิเคชั่น :



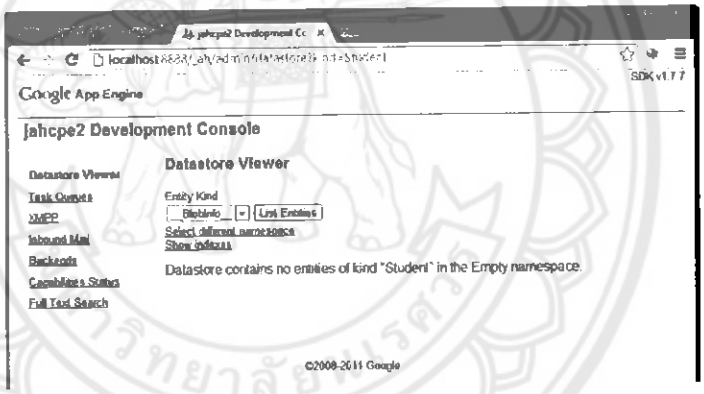
รูปที่ 4.24 แสดงหน้าเว็บของแอปพลิเคชั่น Blobstore เมื่อเข้าไปยัง http://localhost:8888



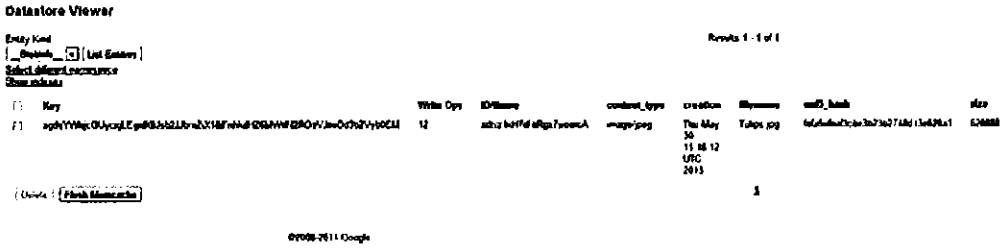
รูปที่ 4.25 แสดงหน้าต่าง Open ให้เลือกเปิดไฟล์เมื่อคลิกที่ปุ่มเลือกไฟล์ ให้ทำการเลือกไฟล์แล้วกด Open

รูปที่ 4.26 แสดงผลหน้าเว็บหลังจากทำการเลือกรูปภาพแล้ว การกดปุ่ม Submit จะเป็นการอัปโหลดไฟล์รูปภาพไปยัง blobstore

รูปที่ 4.27 แสดงหน้าเว็บหลังจากทำการกดปุ่ม Submit



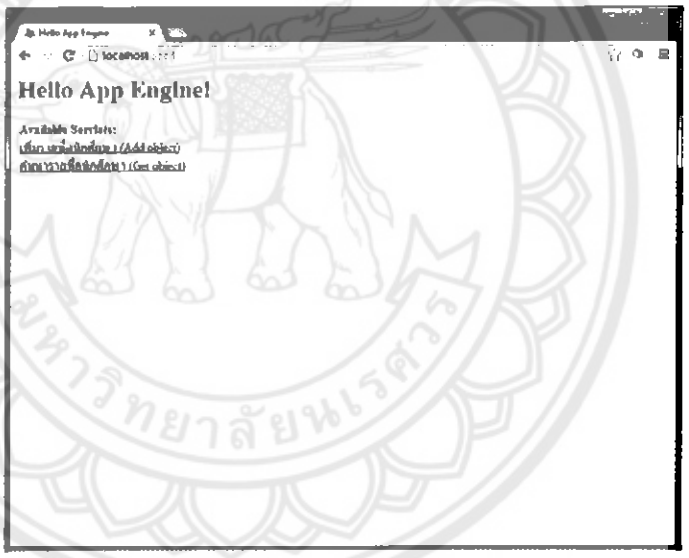
รูปที่ 4.28 แสดงหน้า Admin Console ที่มี Entity _BlobInfo_ ถูกสร้างขึ้น



รูปที่ 4.29 แสดงข้อมูลต่างๆของรูปภาพที่ได้ทำการอัปโหลดไปยัง Blobstore สำเร็จ ใน Admin Console ส่วนของ Datastore Viewer

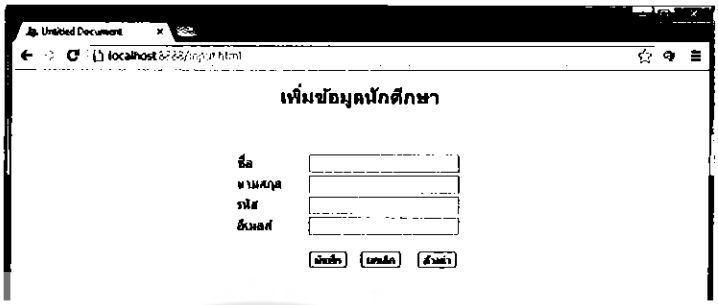
Datastore Application

1. ผลการทำงานของแอปพลิเคชั่น

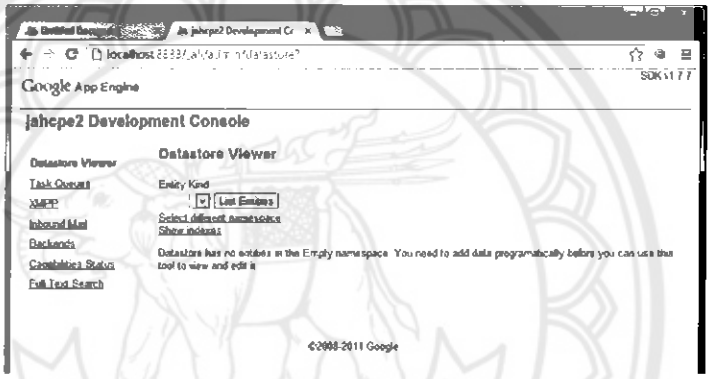


รูปที่ 4.30 แสดงหน้าเว็บของแอปพลิเคชั่น Datastore เมื่อเข้าไปยัง http://localhost:8888

- การเพิ่มข้อมูลนักศึกษา (1)

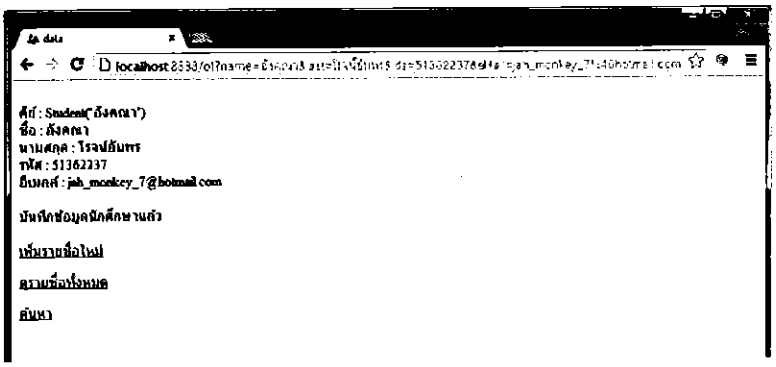


รูปที่ 4.31 แสดงแบบฟอร์มสำหรับการเพิ่มข้อมูลนักศึกษา

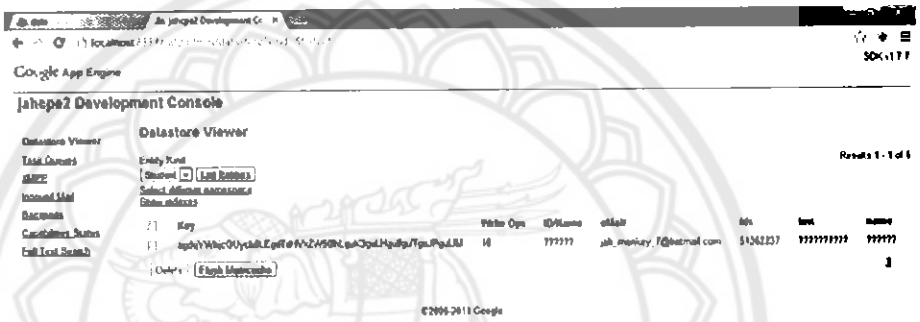


รูปที่ 4.32 แสดงส่วนของ Datastore viewer ในหน้า Adminconsole

รูปที่ 4.33 แสดงการกรอกข้อมูลในฟอร์มการเพิ่มข้อมูลนักศึกษา



รูปที่ 4.34 แสดงผลหน้าเว็บผลลัพธ์เมื่อทำการเพิ่มข้อมูลนักศึกษาไปยังดาต้าสโตร์



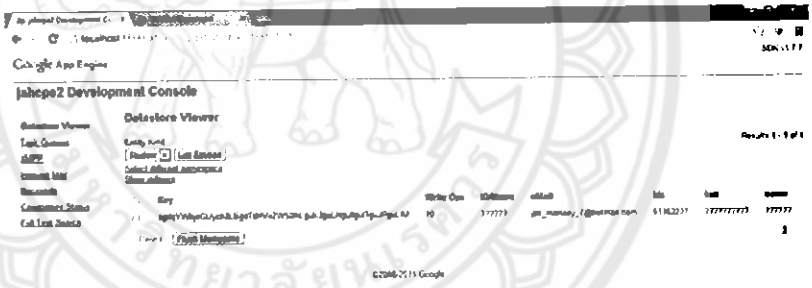
รูปที่ 4.35 แสดงผลลัพธ์ที่เกิดขึ้นในส่วน Datastore Viewer ของ Admin Console

(หมายเหตุ: GAE Java SDK ไม่รองรับการแสดงผลในภาษาไทยในส่วน Addmin Console ของ localhost จึงทำให้ไม่สามารถแสดงผลภาษาไทยได้อย่างถูกต้อง)

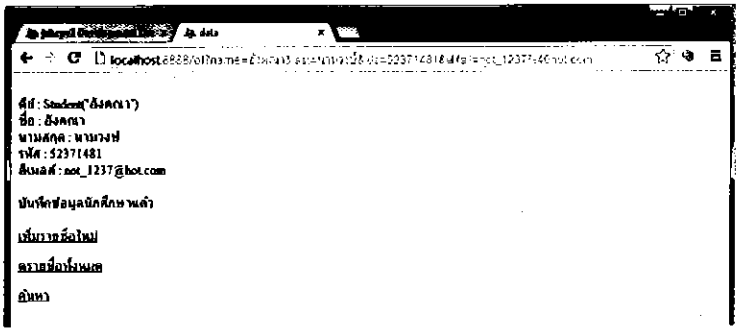
- การเพิ่มข้อมูลของนักศึกษา (2)

เนื่องจากเอนติตี้ที่ใช้สำหรับเก็บข้อมูลของนักศึกษาประเภท student จะใช้ค่าคีย์เป็นประเภทของเอนติตี้ คือ student ประกอบกับค่า “name” ของ student รวมกันเป็นค่า “key” ของเอนติตี้ ซึ่งค่าคีย์ของเอนติตี้ในประเภทเดียวกันนั้นจะต้องไม่ซ้ำกัน และหากได้ทำการเพิ่มเอนติตี้หนึ่งด้วยค่าคีย์ Student “อังคณา” ไปแล้ว เมื่อทำการเพิ่มเอนติตี้ด้วยค่าคีย์ Student “อังคณา” อีกครั้งจะถือเป็นการอัปเดตเอนติตี้ที่มีคีย์คือ Student “อังคณา” แทนดังตัวอย่าง

รูปที่ 4.36 แสดงการเพิ่มเอนติตี้ไปยังดาต้าสโตร์ด้วยคีย์ Student “อังคณา”



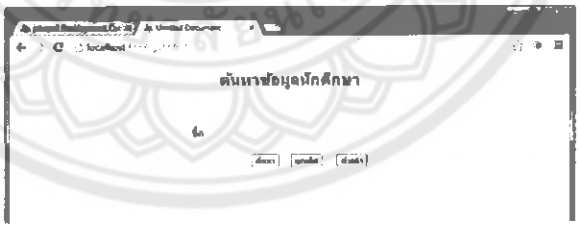
รูปที่ 4.37 แสดงผลลัพธ์ของการเพิ่มเอนติตี้ไปยังดาต้าสโตร์ด้วยคีย์ Student “อังคณา”



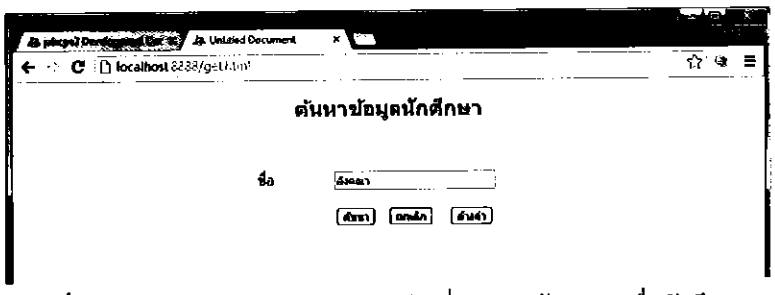
รูปที่ 4.38 แสดงการเพิ่มเอนติตี้ไปยังดาต้าสโตร์ด้วยคีย์ Student “อังคณา” อีกครั้ง

รูปที่ 4.39 แสดงผลลัพธ์ที่เกิดขึ้นใน Datastore เมื่อทำการเพิ่มเอนติตี้ไปยังดาต้าสโตร์ด้วยคีย์ Student “อังคณา” อีกครั้ง

- การค้นหาข้อมูลนักศึกษา



รูปที่ 4.40 แสดงแบบฟอร์มสำหรับค้นหาข้อมูลนักศึกษา

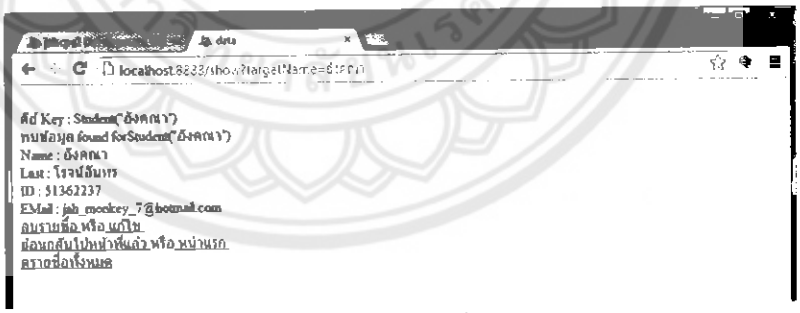


รูปที่ 4.41 แสดงการกรอกข้อมูลในฟอร์มเพื่อทำการค้นหารายชื่อนักศึกษา

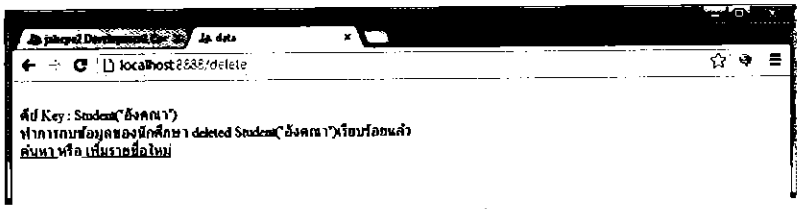
รูปที่ 4.42 แสดงผลลัพธ์ของการค้นหารายชื่อนักศึกษาจากดาต้าสโตร์

- การลบข้อมูลนักศึกษา

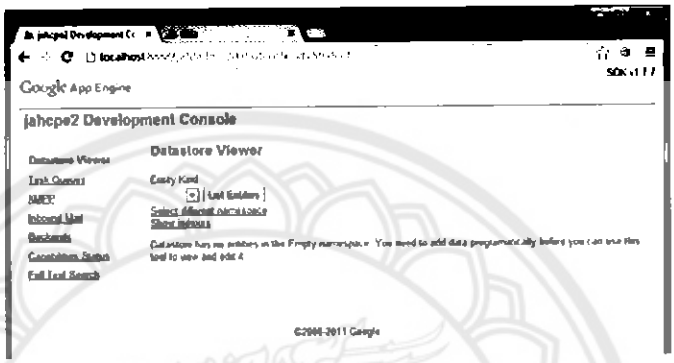
การที่จะลบข้อมูลนักศึกษาได้นั้น จำเป็นจะต้องทราบเสียก่อนว่ามีรายชื่อนักศึกษานี้อยู่ จึงต้องทำการค้นหาเอนติตี้นักศึกษาที่มีคีย์เป็นชื่อนั้นก่อน แล้วจึงจะสามารถเรียกใช้เมธอดนี้ได้



รูปที่ 4.43 แสดงผลของการค้นหานักศึกษาที่มีคีย์เป็น Student("อังคณา")



รูปที่ 4.44 แสดงผลของการลบรายชื่อนักศึกษาที่มีคีย์เป็น Student("อังคณา")



รูปที่ 4.45 แสดงผลลัพธ์ที่เกิดขึ้นในส่วน Datastore Viewer ของ Admin Console หลังจากที่ทำการลบเอนติตี้ที่มีคีย์เป็น Student("อังคณา")

4.3 ผลจากการนำองค์ความรู้มาจัดทำเป็นหนังสือคู่มือ

จากการศึกษาคุณลักษณะต่างๆและการพัฒนาเว็บแอปพลิเคชั่น ทำให้สามารถจัดทำเนื้อหาของหนังสือคู่มือการใช้งานการสร้างเว็บแอปพลิเคชั่นด้วย Google App Engine ได้ โดยครอบคลุมเนื้อหาด้านต่างๆดังนี้

ตารางที่ 4.1 แสดงเนื้อหาหนังสือที่ได้ในบทที่ 1

บทที่ 1 : บทนำ		
No.	เรื่อง	ความสำคัญ
1	รู้จักกับเว็บแอปพลิเคชั่น - เว็บแอปพลิเคชั่นคืออะไร - ส่วนประกอบของเว็บแอปพลิเคชั่น - หลักการทำงานของเว็บแอปพลิเคชั่น	ความรู้เบื้องต้นเกี่ยวกับสิ่งที่จะทำการพัฒนา
2	รู้จักกับภาษาจาวาและชุดพัฒนาภาษาจาวา - จาวาคืออะไร - ชุดพัฒนาภาษาจาวา - การติดตั้งตัวแปรภาษาจาวา	ความรู้เบื้องต้นเกี่ยวกับภาษาที่ใช้ในการพัฒนาแอปพลิเคชั่น
3	รู้จักกับคลาวด์คอมพิวติ้ง - คลาวด์คอมพิวติ้งคืออะไร - โครงสร้างของคลาวด์คอมพิวติ้ง -รูปแบบการให้บริการบนคลาวด์คอมพิวติ้ง -ประโยชน์และข้อจำกัดการใช้งาน	ความรู้เบื้องต้นเกี่ยวกับเทคโนโลยีที่เป็นพื้นฐานของกูเกิ้ลแอปเอ็นจิ้น
4	รู้จักกับกูเกิ้ลแอปเอ็นจิ้น - กูเกิ้ลแอปเอ็นจิ้นคืออะไร - Environment ในการพัฒนาของแอปพลิเคชั่น - Java Environment	ความรู้เบื้องต้นเกี่ยวกับเกี่ยวกับกูเกิ้ลแอปเอ็นจิ้น

ตารางที่ 4.2 แสดงเนื้อหาหนังสือที่ได้ในบทที่ 2

บทที่ 2 : คุณลักษณะต่างๆของกูเกิ้ลแอปเอ็นจิ้น		
No.	เรื่อง	ความสำคัญ
1	คุณลักษณะที่ใช้ในการพัฒนาเว็บแอปพลิเคชั่น	
	เครื่องมือ - การพัฒนาภาษาจาวา : Eclipse IDE - การสร้างและอัปโหลด : Google App Plugin - การทดสอบแอปพลิเคชั่น : Development Server - การอัปโหลดและจัดการแอปพลิเคชั่น : Java SDK	ความรู้เบื้องต้นเกี่ยวกับ เครื่องมือที่ช่วยในการพัฒนาเว็บแอปพลิเคชั่น
	พื้นที่เก็บข้อมูล - JDO/JPA - Google Cloud SQL - Google Storage API	ความรู้เบื้องต้นเกี่ยวกับพื้นที่เก็บข้อมูลของแอปเอ็นจิ้นและการใช้งาน
	Services - User API - Mail API - URL Fetch API - Blobstor API - Java Datastore API	ความรู้เบื้องต้นเกี่ยวกับ services ต่างๆของกูเกิ้ลแอปเอ็นจิ้นและการใช้งาน
	ส่วนเพิ่มเติมสำหรับการสร้างเว็บแอปพลิเคชั่น - Java Servlet และ JSP - Static file และการใช้งาน	ความรู้เบื้องต้นเกี่ยวกับเทคโนโลยีช่วยในการทำงานด้วยภาษาจาวา
2	คุณลักษณะที่ใช้ในการกำหนดค่าแอปพลิเคชั่น - Deployment Descriptor (web.xml) - Application Configuration (appengine-web.xml) - Backends Configuration (backend.xml) - Index Configuration (index.aml) - Scheduled Tasks (cron.xml) -Task Queue Configuration (queue.xml)	ความรู้เบื้องต้นเกี่ยวกับไฟล์ตั้งค่าต่างๆของแอปพลิเคชั่นและการใช้งาน
3	คุณลักษณะที่ใช้ในการดูแลจัดการแอปพลิเคชั่น - Backends และการใช้งาน - Administrator Console และการใช้งาน	ความรู้เบื้องต้นในการดูแลจัดการแอปพลิเคชั่น
4	โควตาและขีดจำกัดการใช้งาน	ปริมาณการใช้งาน

ตารางที่ 4.3 แสดงเนื้อหาหนังสือที่ได้ในบทที่ 3

บทที่ 3 : การพัฒนาเว็บแอปพลิเคชั่น		
No.	เรื่อง	ความสำคัญ
1	อธิบายขั้นตอนการพัฒนาเว็บแอปพลิเคชั่นพื้นฐาน	
2	ตัวอย่างแอปพลิเคชั่น	
	Hello World - ลักษณะของแอปพลิเคชั่นและการทำงาน - ขั้นตอนในการพัฒนา	แนะนำการใช้งาน Servlet , html และการใช้งาน Eclipse ในการสร้าง เริ่มการทำงานและหยุดการทำงานของแอปพลิเคชั่น
	Login - ลักษณะของแอปพลิเคชั่นและการทำงาน - ขั้นตอนในการพัฒนา	การใช้งาน User API ในการลงชื่อเข้าและออกจากระบบด้วยบัญชีกูเกิ้ล
	E-mail - ลักษณะของแอปพลิเคชั่นและการทำงาน - ขั้นตอนในการพัฒนา - การพัฒนาปรับปรุง	การส่ง-รับอีเมลล์โดยใช้ Java Mail API การปรับปรุงการรับค่าต่างๆ การใช้ภาษาไทย การตั้งค่าผู้รับหลายคน
	Word Search - ลักษณะของแอปพลิเคชั่น และการทำงาน - ขั้นตอนในการพัฒนา - การประยุกต์ใช้กับ web service อื่น	การเข้าถึง Webservice Http get request Http post request
	Blobstore - ลักษณะของแอปพลิเคชั่น และการทำงาน - ขั้นตอนในการพัฒนา - การพัฒนาปรับปรุง	การใช้งาน Blobstore API ในการอัปโหลดไฟล์รูปภาพ และผลลัพธ์ที่เกิดขึ้นใน datastore
	Datastore - ลักษณะของแอปพลิเคชั่น และการทำงาน - ขั้นตอนในการพัฒนา	การใช้งาน Java Datastore API ในการเก็บข้อมูลและผลลัพธ์ที่เกิดขึ้นใน datastore
3	ขั้นตอนการ Deploy ไปยัง Google App Engine - การลงทะเบียนสมัคร Application ID - การอัปโหลดไปยัง Google App Engine	ขั้นตอนการลงทะเบียนและอัปโหลดแอปพลิเคชั่นไปยังกูเกิ้ลแอปเอ็นจิ้นเซิร์ฟเวอร์

ตารางที่ 4.4 แสดงเนื้อหาหนังสือที่ได้ในบทที่ 4

บทที่ 4 : สรุปผลการใช้งาน		
No.	เรื่อง	ความสำคัญ
1	Google App Engine กับ Platform อื่น	ตารางเปรียบเทียบระหว่าง Google App Engine Windows Azure และ Lunacloud
2	Google App Engine ในมุมมองนักพัฒนา	จุดเด่น – จุดด้อยในมุมมองของนักพัฒนา

บทที่ 5
บทสรุปและข้อเสนอแนะ

ในบทนี้กล่าวถึงการสรุปผลการทดสอบทั้งหมดที่ได้มาจากการดำเนินโครงการและทดสอบระบบในส่วนต่างๆ นอกจากนี้ยังกล่าวรวมถึงปัญหาที่พบพร้อมทั้งแนวทางแก้ไข อีกทั้งแนวทางในการพัฒนาต่อซึ่งได้มาจากผลของการทดลอง

5.1 สรุปผลการดำเนินงาน

สำหรับการศึกษาคุณลักษณะต่างๆของกูเกิ้ลแอปเอ็นจิ้นได้ดำเนินการศึกษาคุณลักษณะดังต่อไปนี้ได้แก่ Environment, Tools, Datastore, Services, Application Management, Configuration fileและ Quota ซึ่งเป็นคุณลักษณะที่สำคัญซึ่งได้นำความรู้มาประยุกต์ใช้กับการพัฒนาเว็บแอปพลิเคชั่นบนกูเกิ้ลแอปเอ็นจิ้นและจัดทำหนังสือคู่มือ

ในการพัฒนาเว็บแอปพลิเคชั่นบนกูเกิ้ลแอปเอ็นจิ้นได้นำความรู้จากการศึกษาคุณลักษณะต่างๆมาพัฒนาเว็บแอปพลิเคชั่นประกอบไปด้วย

1. Hello World Application เป็นการเริ่มต้นพัฒนาแอปพลิเคชั่นบนกูเกิ้ลแอปเอ็นจิ้น
2. Login Application เป็นการพัฒนาแอปพลิเคชั่นบนกูเกิ้ลแอปเอ็นจิ้นโดยใช้ User API
3. Mail Application เป็นการพัฒนาแอปพลิเคชั่นบนกูเกิ้ลแอปเอ็นจิ้นโดยใช้ Mail API
4. URL Fetch Application เป็นการพัฒนาแอปพลิเคชั่นบนกูเกิ้ลแอปเอ็นจิ้นโดยใช้ URL Fetch API
5. Blob Application เป็นการพัฒนาแอปพลิเคชั่นบนกูเกิ้ลแอปเอ็นจิ้นโดยใช้ Blobstore API
6. Datastore Application เป็นการพัฒนาแอปพลิเคชั่นบนกูเกิ้ลแอปเอ็นจิ้นโดยใช้ Datastore API

แอปพลิเคชั่นที่ได้พัฒนาขึ้นเป็นการนำความรู้ความเข้าใจจากการศึกษาข้อมูลต่างๆมาประยุกต์ใช้ในการพัฒนา ผู้ดำเนินโครงการจึงได้รับความรู้ความเข้าใจเพิ่มมากยิ่งขึ้น อีกทั้งสามารถพัฒนาเว็บแอปพลิเคชั่นบนกูเกิ้ลแอปเอ็นจิ้นได้ ทั้งนี้ความรู้ความเข้าใจและการพัฒนาเว็บแอปพลิเคชั่นทั้งหมดก็จะนำไปเขียนในหนังสือคู่มือต่อไป ในส่วนของการดำเนินการจัดทำหนังสือคู่มือเป็นการรวบรวมความรู้ความเข้าใจต่างๆที่ได้ศึกษาเกี่ยวกับกูเกิ้ลแอปเอ็นจิ้น รวมถึงการพัฒนาเว็บแอปพลิเคชั่นที่ผู้ดำเนินโครงการได้เลือกพัฒนาขึ้นมา ยิ่งไปกว่านั้นในหนังสือคู่มือที่ได้จัดทำขึ้นยังมี API อื่นๆ คุณลักษณะอื่นๆ อีกมากมาย ที่ไม่ได้เลือกมาทำอธิบายเอาไว้ให้ในหนังสือคู่มืออีกด้วย

5.2 ปัญหาที่พบ

เครื่องคอมพิวเตอร์ที่ใช้ในการดำเนินโครงการมีประสิทธิภาพที่ต่ำ ในการดำเนินโครงการจึงพบกับอุปสรรคคือโปรแกรมที่ใช้ในการดำเนินโครงการมักค้างอยู่บ่อยครั้ง ทำให้การดำเนินโครงการล่าช้าและกูเกิ้ลแอปเอ็นจิ้นเป็นเทคโนโลยีคลาวด์คอมพิวติ้ง ซึ่งเป็นเทคโนโลยีใหม่และยังมีข้อมูลให้ศึกษาไม่มากนัก

5.3 แนวทางการแก้ไข

ใช้คอมพิวเตอร์ที่มีประสิทธิภาพสูงมากยิ่งขึ้น ทำให้โปรแกรมที่ใช้ในการดำเนินโครงการไม่ค้างบ่อย การดำเนินโครงการจะทำได้สะดวก รวดเร็วมากยิ่งขึ้น

พยายามสืบค้นข้อมูลจากแหล่งข้อมูลต่างประเทศและศึกษาทำความเข้าใจด้วยตนเอง

5.4 แนวทางการพัฒนาต่อในอนาคต

นำคุณลักษณะต่างๆของกูเกิ้ลแอปเอ็นจิ้น รวบรวมเป็นสื่อสารสอนออนไลน์บนเว็บแอปพลิเคชั่นซึ่งพัฒนาด้วยกูเกิ้ลแอปเอ็นจิ้น เพื่อให้ผู้ที่สนใจสามารถศึกษาความรู้เกี่ยวกับการพัฒนาเว็บแอปพลิเคชั่นด้วยกูเกิ้ลแอปเอ็นจิ้นผ่านระบบออนไลน์ได้

เอกสารอ้างอิง

[1] "การประมวลผลแบบกลุ่มเมฆ". [ออนไลน์]. เข้าถึงได้จาก http://th.wikipedia.org/wiki/การประมวลผลแบบกลุ่มเมฆ

[2] ธีระนันท์ บิลตะเย็บ. (30 กรกฎา 2555). "การประมวลผลกลุ่มเมฆหรือ cloud computing" เข้าถึงได้จาก http://bteeranan.wordpress.com/2012/07/30/การประมวลผลกลุ่มเมฆ-หรี/

[3] "Cloud Computing เขามีรูปแบบการให้บริการอะไรบ้าง มาดูกัน" [ออนไลน์]. เข้าถึงได้จาก http://rattanasak.jigsawoffice.com/content/content.php?mid=87&did=369&tid=1&0

[4] "Cloud computing คืออะไร" .[ออนไลน์] .เข้าถึงได้จาก http://www.manacomputers.com/what-is-cloud-computing/

[5] "Web Application คืออะไร". [ออนไลน์]. เข้าถึงได้จากhttp://www.igetugot.com/topics/Web+Application+คืออะไร-838-1-1.html

[6] ทินกร วัฒนเกษมกุล. (2548). คัมภีร์ JSP: ส่วนประกอบของเว็บแอปพลิเคชั่น กรุงเทพฯ : สำนักพิมพ์เคทีพี.

[7] "เอพีไอ". [ออนไลน์]. เข้าถึงได้จาก http://th.wikipedia.org/wiki/เอพีไอ

[8] "Sdk คืออะไร". [ออนไลน์]. เข้าถึงได้จาก www.mindphp.com/คู่มือ/73-คืออะไร/2261-sdk-คืออะไร.html

[9] "JSP&servlet คืออะไร". [ออนไลน์]. เข้าถึงได้จาก http://www.martroutine.com/2011/04/jsp-servlet-คืออะไร/

[10] ทินกร วัฒนเกษมกุล. (2548). คัมภีร์ JSP: JSP กรุงเทพฯ: สำนักพิมพ์ เคทีพี.

[11] "Figure3. How JavaServer Pages Work".[ออนไลน์] .เข้าถึงได้จาก http://www.itmelody.com/tu/introjsp.htm

[12] "Google App Engine คืออะไร". [ออนไลน์] .เข้าถึงได้จาก http://www.chakkrit.com/what-is-google-app-engine/

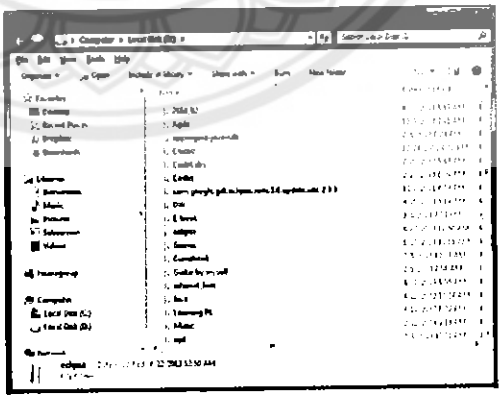
ภาคผนวก

การติดตั้ง Eclipse

ขั้นตอนที่ 1: ดาวน์โหลด Eclipse

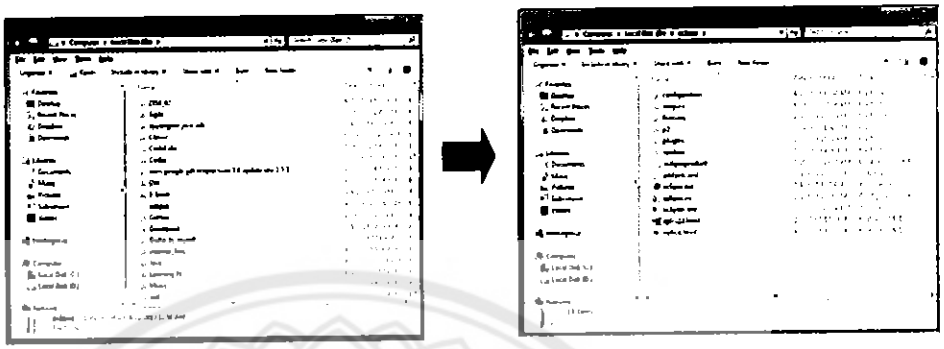
1. ดาวน์โหลด Eclipse จาก http://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/heliossr2 ซึ่งเป็น Eclipse IDE for Java EE Developer เวอร์ชัน 3.4 Helios Packages จากนั้นทำการเลือกสำหรับระบบปฏิบัติการที่ต้องการ โดยตัวอย่างได้เลือก Windows-32 bit (ที่ต้องเลือก Java EE เนื่องจากต้องการใช้งานในเชิงของ Java Web Developer)

2. จะได้ไฟล์ซิป eclipse-jee-helios-SR2-win32.zip ให้ทำการแตกไฟล์ซิปและนำไปวางไว้ในที่ที่สะดวกสำหรับการใช้งาน และสามารถเปลี่ยนชื่อโฟลเดอร์ได้ตามต้องการ โดยไม่ส่งผลใดๆกับการใช้งาน (ตัวอย่างเลือกวางไว้ที่ไดร์ฟ D และเปลี่ยนชื่อเป็น eclipse เพื่อให้จำง่าย)

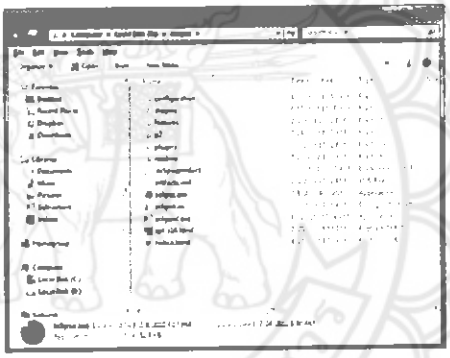


ขั้นตอนที่ 2: ตรวจสอบความพร้อมในการใช้งานของโปรแกรม

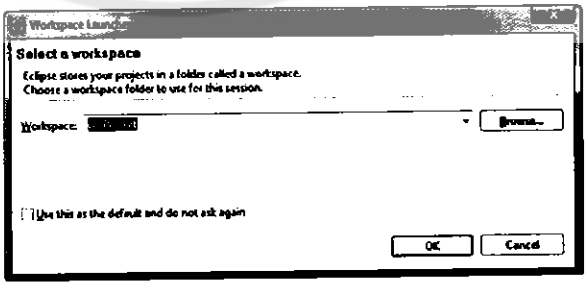
1. ไปยังที่อยู่ของไฟล์ที่ได้ทำการติดตั้งไปในขั้นตอนที่แล้ว



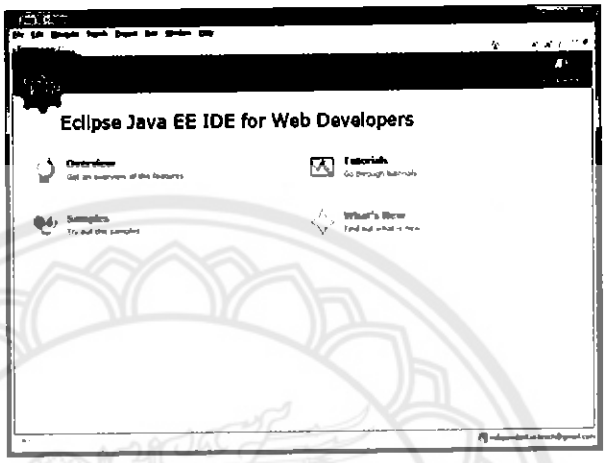
2. ดับเบิ้ลคลิกที่ eclipse.exe เพื่อเริ่มต้นการทำงานของ Eclipse



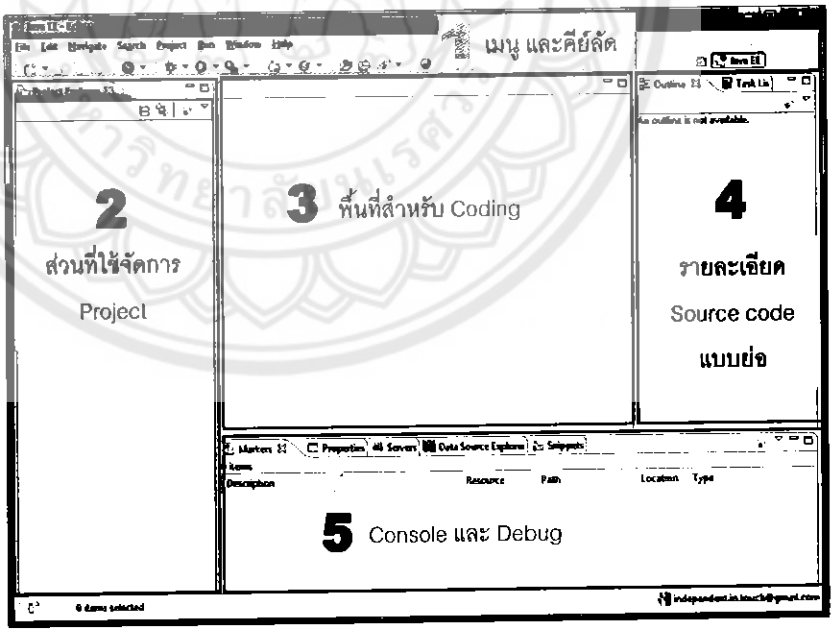
3. Eclipse จะถามถึงพื้นที่ในการเก็บ Project ที่จะสร้างขึ้นใช้งาน (เรียกว่า workspace) ซึ่งสามารถเลือกใช้โฟลเดอร์ที่มีอยู่แล้ว หรืออาจจะสร้างขึ้นมาใหม่เพื่อเป็น workspace ของ eclipse โดยเฉพาะก็ได้ และเมื่อเลือกเสร็จแล้วให้คลิก OK
(ตัวอย่าง เก็บ Workspace ไว้ที่ D:\Project)



4. โปรแกรม eclipse จะทำการโหลด workbench และนำมาสู่หน้าจอต้อนรับดังภาพ ซึ่งจะมีทางลัดไปยังรายละเอียดเกี่ยวกับตัวของโปรแกรมเอง รวมทั้งตัวอย่างและ Tutorials สำหรับให้ศึกษาและทดลองใช้งานด้วย



5. เมื่อปิดหน้าจอต้อนรับไป จะพบกับหน้าจอ editor ซึ่งจะเป็นส่วนที่ต้องใช้สำหรับการพัฒนาโปรแกรมในอนาคต ซึ่งประกอบไปด้วย Panel ต่างๆดังภาพ



Google App Engine Plug-in สำหรับ Eclipse และการติดตั้ง

สำหรับการพัฒนา Web Application ด้วย Eclipse นั้น การติดตั้ง Plug-in เพิ่มเข้าไปจะช่วยให้การพัฒนาทำได้ง่ายขึ้น เนื่องจาก Plug-in ได้รวมเอาทุกสิ่งทุกอย่างที่จำเป็นต้องใช้ไว้ให้แล้ว ไม่ว่าจะเป็นในการสร้าง การทดสอบ หรือในการ deploy Web Application ไปยัง Google App Engine ผ่านทาง Eclipse ก็ตาม

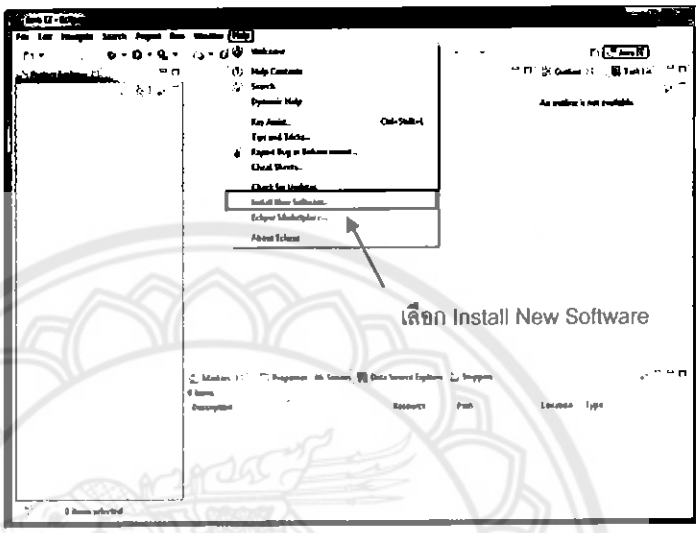
Plug-in นี้สามารถใช้งานได้กับ Eclipse ตั้งแต่เวอร์ชัน 3.3 จนถึงเวอร์ชันปัจจุบัน และการติดตั้งนั้นสามารถทำได้โดยการใช้ Software Update feature ของ Eclipse ซึ่งแต่ละเวอร์ชั่นจะมีที่อยู่ในการติดตั้ง plug-in ที่ต่างกัน ดังนี้

- The Google Plugin for Eclipse, for Eclipse 3.3 (Europa):
  https://dl.google.com/eclipse/plugin/3.3
- The Google Plugin for Eclipse, for Eclipse 3.4 (Ganymede)
  https://dl.google.com/eclipse/plugin/3.4
- The Google Plugin for Eclipse, for Eclipse 3.5 (Galileo):
  https://dl.google.com/eclipse/plugin/3.5
- The Google Plugin for Eclipse, for Eclipse 3.6 (Helios):
  https://dl.google.com/eclipse/plugin/3.6
- The Google Plugin for Eclipse, for Eclipse 3.7 (Indigo)
  https://dl.google.com/eclipse/plugin/3.7
- The Google Plugin for Eclipse, for Eclipse 3.8/4.2 (Juno)
  https://dl.google.com/eclipse/plugin/4.2

โดยที่อยู่เหล่านี้นอกจากจะสามารถใช้ในการติดตั้ง Plug-in ของ Google App Engine แล้วยังสามารถใช้ในการติดตั้ง SDKs ของ Google App Engine ได้อีกด้วย ทั้งนี้สามารถติดตั้งทั้ง Plug-in และ SDKs ได้ในเวลาเดียวกัน โดยใช้ขั้นตอนในการติดตั้งดังต่อไปนี้

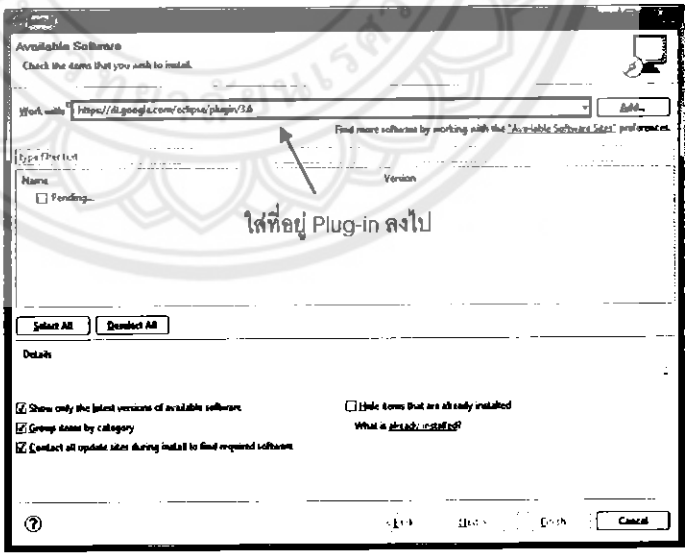
การติดตั้ง Google App Engine Plug-in สำหรับ Eclipse

1. เปิดโปรแกรม Eclipse ขึ้นมา แล้วไปที่ Help > Install New Software

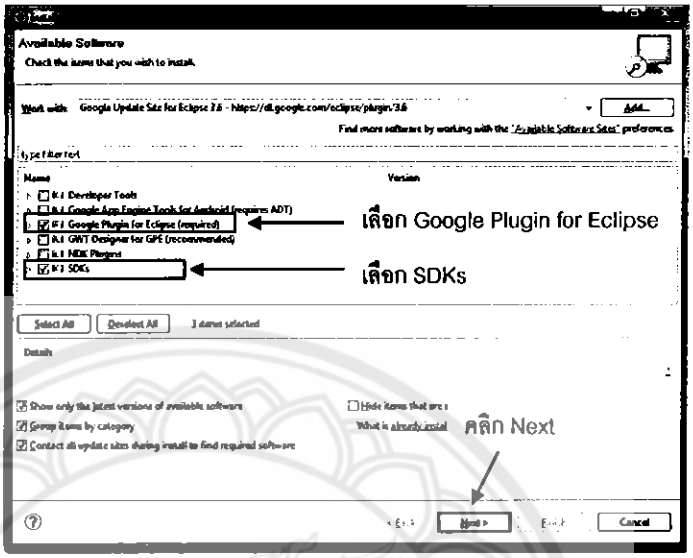


2. ใส่ที่อยู่ของ Plug-in ลงไป

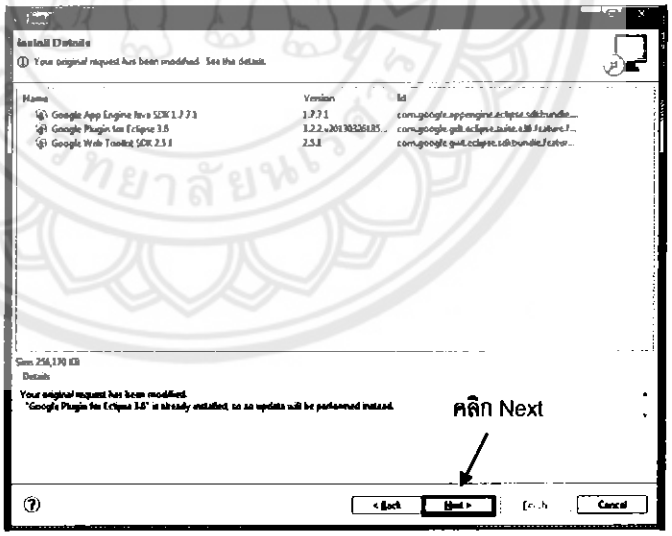
(ตัวอย่างใช้ Eclipse 3.6 ที่อยู่ในการติดตั้งคือ https://dl.google.com/eclipse/plugin/3.6)



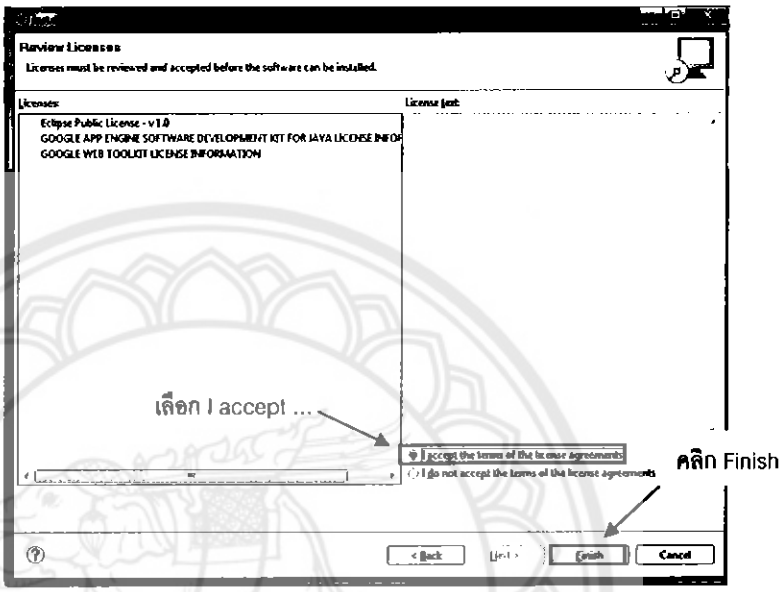
3. จะพบกับรายชื่อ Plug-in ดังภาพ ให้เลือก Plug-in ที่ต้องการใช้แล้วคลิก Next



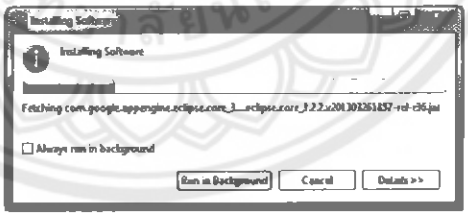
4. Eclipse จะทำการ download และตรวจสอบ Plug-in ที่เลือกไป และแสดงรายละเอียดในการติดตั้งให้ดูดังภาพ ให้คลิก next



5. ก่อนที่ Eclipse จะทำการติดตั้ง Plug-in จะต้องทำการยอมรับข้อตกลงในการใช้งานก่อน(license agreement) โดยคลิกที่ I accept the terms of license agreements จากนั้นคลิก finish เพื่อเริ่มการติดตั้ง

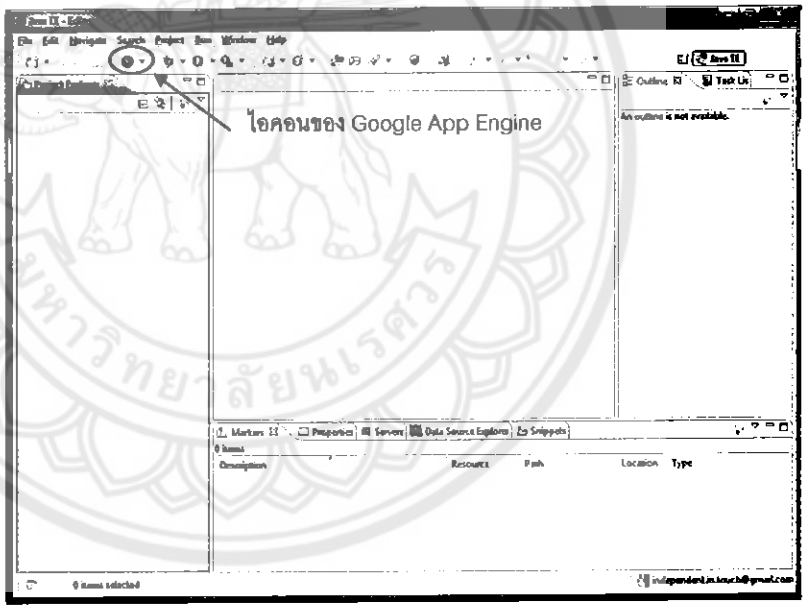


6. Eclipse จะเริ่มทำการติดตั้ง Plug-in โดยจะใช้เวลาสักระยะหนึ่ง ซึ่งเวลาที่ใช้ในการติดตั้งอาจจะแตกต่างกันไป ขึ้นอยู่กับความเร็วของอินแทอร์เน็ตที่ใช้ด้วย



7. เมื่อการติดตั้งเสร็จสิ้น Eclipse จะเตือนให้ทำการ Restart โปรแกรมก่อนที่จะใช้งาน ให้เลือก Restart Now เพื่อให้การติดตั้งเมื่อสักครู่สามารถใช้งานได้และเพื่อป้องกันการเกิด Error ที่มาจากการติดตั้งอีกด้วย

8. เมื่อการ Restart เสร็จสิ้นลงจะพบว่าบริเวณแถบเครื่องมือของ Eclipse มีไอคอนของ Google App Engine ปรากฏขึ้นมานั่นแสดงว่า Eclipse ได้ทำการติดตั้ง Plugin เสร็จเรียบร้อยและพร้อมสำหรับการใช้งานแล้ว



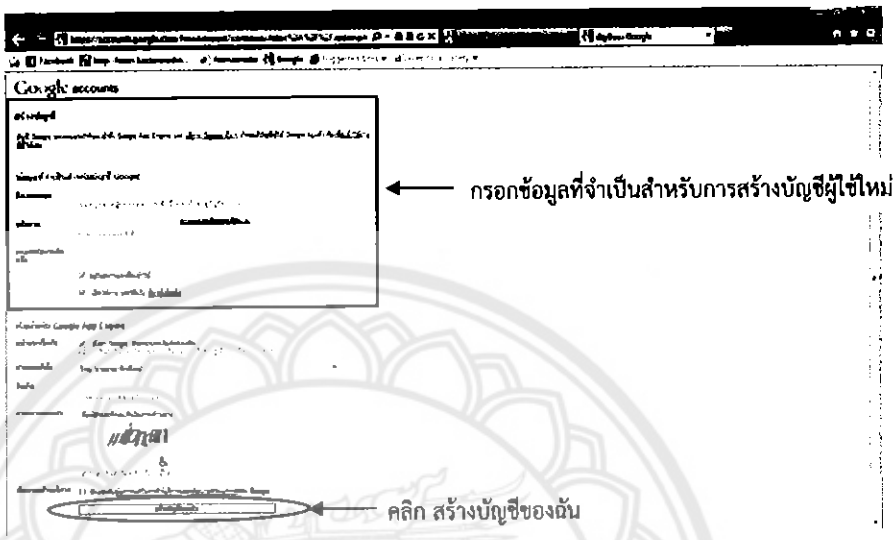
การลงทะเบียนใช้งาน Google App Engine

สำหรับการจะเข้าใช้งาน Google App Engine ได้นั้นจำเป็นต้องมีบัญชี (account) ของ Google เสียก่อน สำหรับผู้ที่ยังไม่มีบัญชีสามารถสมัครตามขั้นตอนดังต่อไปนี้ (สำหรับผู้มีบัญชีแล้วให้ข้ามไปขั้นตอนที่ 4)

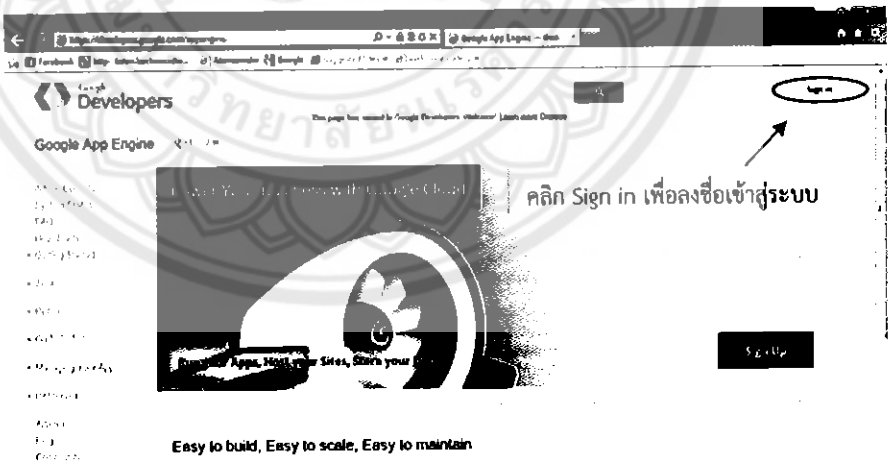
1. เข้าไปที่ www.google.co.th แล้วเลือก **ลงชื่อเข้าสู่ระบบ**

2. จากนั้นจะปรากฏหน้าเว็บดังภาพ ให้เลือกที่ **สมัครใช้งาน**

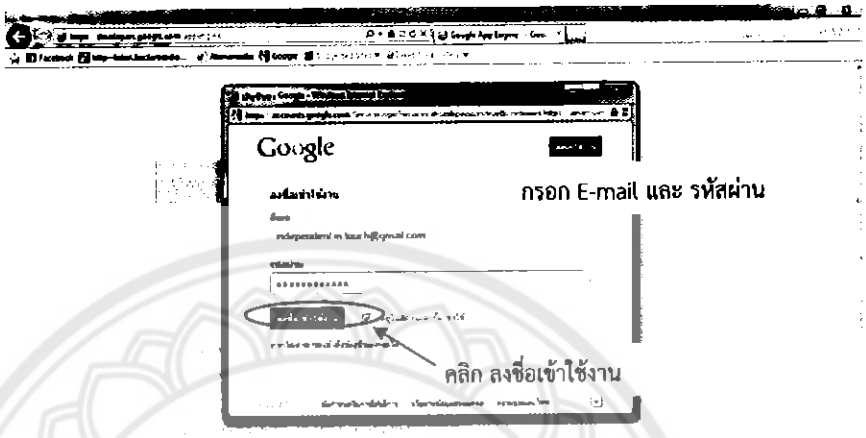
3. จากนั้นจะพบหน้าเว็บดังภาพ ให้กรอกข้อมูลสำคัญให้ครบถ้วน แล้วเลือกที่ **สร้างบัญชีของฉัน** เพื่อสร้างบัญชีผู้ใช้ Google ขึ้นมาใหม่



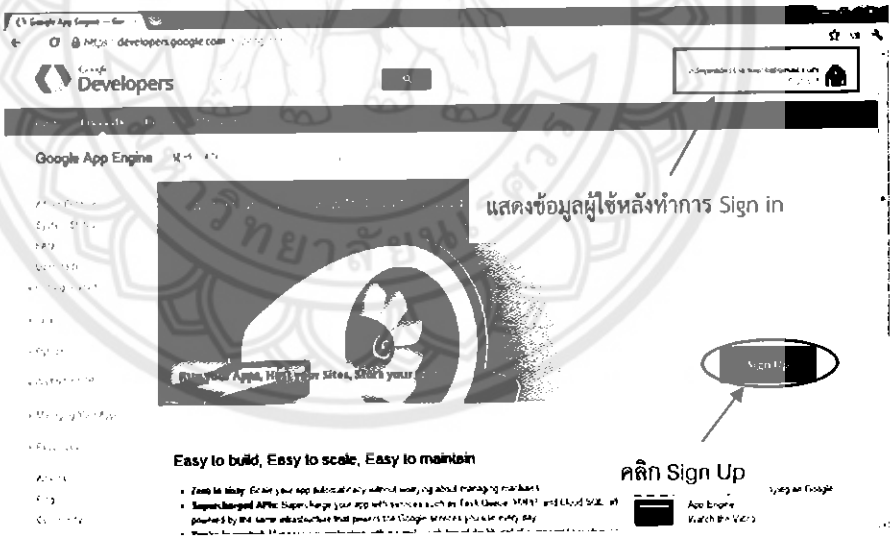
4. เมื่อได้บัญชีผู้ใช้ของ Google มาแล้ว ให้เข้าไปยัง http://developers.google.com/appengine หน้าเว็บไซต์ของ Google App Engine เพื่อสมัครใช้งาน โดยเริ่มจากการ Sign in **เข้าสู่ระบบก่อน**



5. หลังจากคลิก Sign in จะปรากฏหน้าต่างสำหรับกรอกอีเมลล์และรหัสผ่าน เพื่อลงชื่อเข้าใช้งานบัญชี เมื่อกรอกเสร็จให้เลือก ลงชื่อเข้าใช้งาน



6. หลังลงชื่อเข้าใช้งานแล้วเลือก sign up เพื่อสมัครเข้าใช้งาน Google App Engine

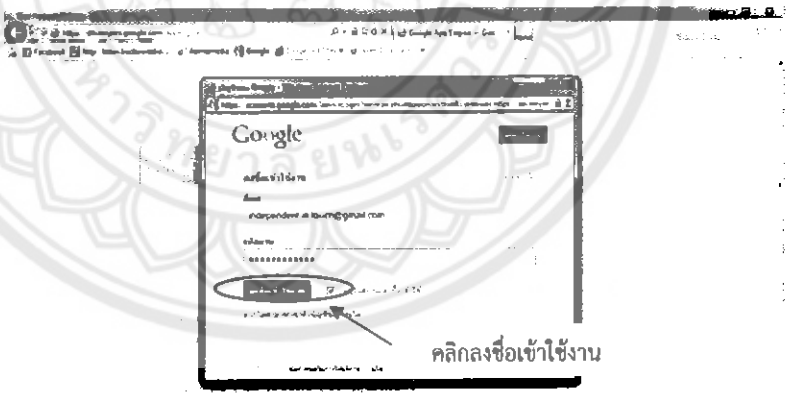


7. หลังจากที่ Sign up เข้าไปแล้วจะพบกับหน้าเว็บ Welcome to Google App Engine ดังภาพ ซึ่งมีหัวข้อ Getting Started Guide, FAQ และ Developer's Guide ให้เรียนรู้เพิ่มเติมและยังปรากฏปุ่ม Create Application สำหรับสร้างแอปพลิเคชั่น ID อีกด้วย

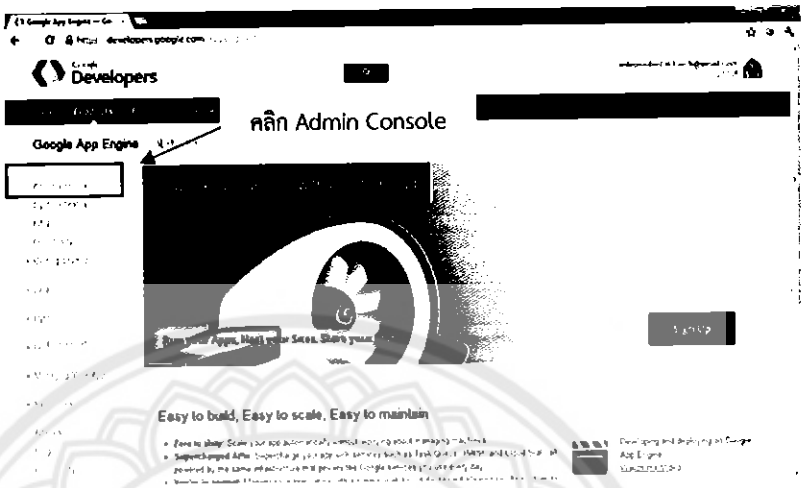
การลงทะเบียนเพื่อรับ Application Identifier

1. เข้าไปที่ https://developers.google.com/appengine/ หน้าเว็บหลักของ Google App Engine จากนั้นทำการ **ลงชื่อเข้าสู่ระบบ หรือ Sign in ก่อน**

2. หลังจากคลิก Sign in จะปรากฏหน้าต่างสำหรับกรอกอีเมลล์และรหัสผ่าน เพื่อลง**ชื่อเข้าใช้งานบัญชี** เมื่อกรอกเสร็จให้เลือก ลงชื่อเข้าใช้งาน

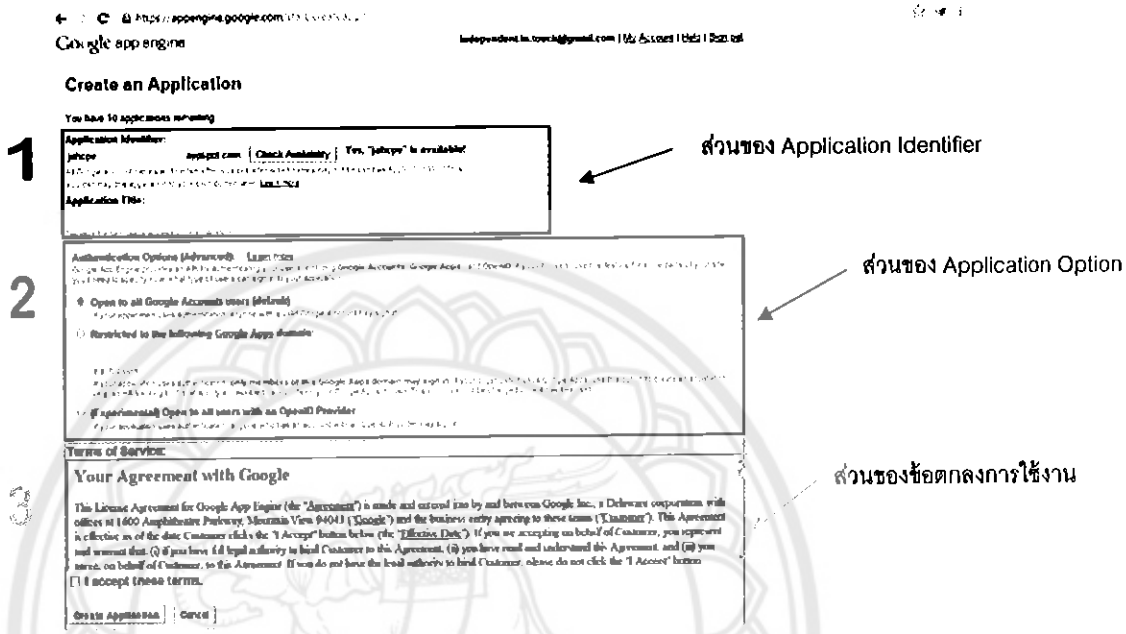


3. หลังลงชื่อเข้าใช้งานแล้วไปที่ Admin Console ทางด้านซ้ายมือ



4. หลังจากคลิกเข้ามาแล้วจะพบกับหน้าเว็บ Welcome to Google App Engine **ดังภาพให้เลือก** Create Application

5. ในหน้า Create Application นั้นจะต้องกรอกรายละเอียดต่างๆเกี่ยวกับ Application ที่ต้องการจะสร้างให้ครบถ้วน ซึ่งในส่วนที่ต้องกรอกนั้นจะประกอบไปด้วยส่วนต่างๆดังนี้



5.1 ส่วนของ Application Identifier ประกอบไปด้วยข้อมูล 2 ส่วนที่ต้องกรอก ดังนี้

a. ชื่อ Application ซึ่งจะถูกต่อท้ายด้วยโดเมน .appspot.com และจะเป็นที่อยู่ของ Application ต่อไป

b. Application Title ซึ่งจะแสดงขึ้นเมื่อมีผู้ใช้เข้ามายัง Application ซึ่งสามารถละไว้แล้วมาแก้ไขภายหลังได้

5.2 ส่วนของ Application Option ซึ่งจะเป็นการเลือกระบุชนิดของ User ที่สามารถ sign in เข้ามายังโดเมน โดยมีให้เลือก 3 แบบ ดังนี้

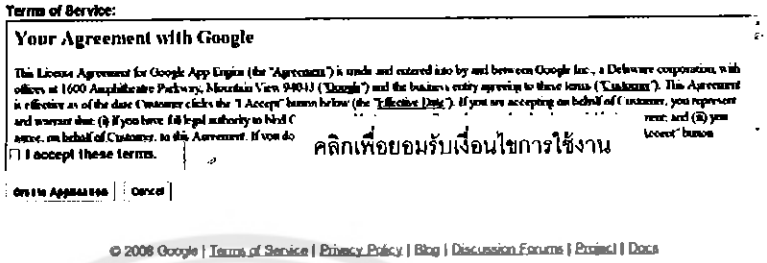
a. Open to all Google Accounts users (default) คือเปิดให้ผู้ใช้ทุกคนที่มีบัญชีของ Google

b. Restricted to the following Google Apps domain คือเปิดให้เฉพาะสมาชิกของ Google App โดเมนนั้นๆเท่านั้น แต่ควรระวังในการเลือกตัวเลือกนี้ไว้อย่างหนึ่งคือ เมื่อใดก็ตามที่เลือกตัวเลือกนี้ จะไม่สามารถเปลี่ยนมันไปเป็นแบบอื่นได้อีก

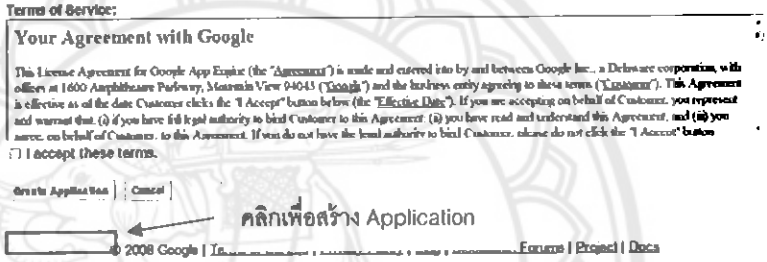
c. (Experimental) Open to all users with an OpenID Provider คือเปิดให้ผู้ใช้ที่มี OpenID สามารถเข้าใช้งานได้

5.3 ส่วนของ Term of Services

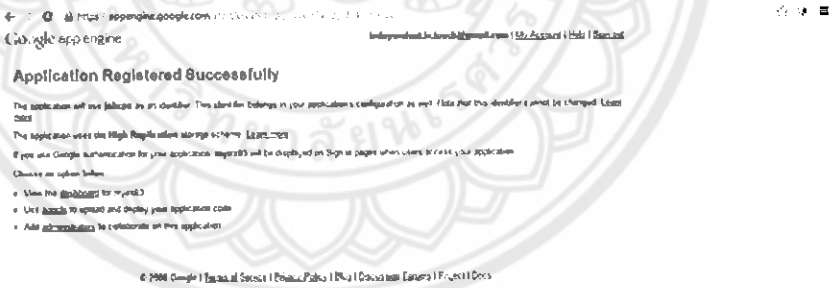
ซึ่งเป็นข้อตกลงการใช้งานกับ Google เมื่อทำการอ่านรายละเอียดเรียบร้อยแล้วให้คลิกถูกที่ I accept these terms.



เมื่อกรอกรายละเอียดครบหมดแล้วให้คลิกที่ Creat Application



6. จากนั้นจะพบกับหน้าที่แจ้งว่าลงทะเบียน Application เรียบร้อยแล้ว



7. เมื่อการลงทะเบียนเสร็จสิ้นลง Google App Engine จะแจ้ง Application ID และรายละเอียดอื่นๆ แบบสรุปให้ทราบ และสำหรับการเข้าถึง Application นั้นสามารถเข้าถึงได้จาก Application ID ตามด้วย .appspot.com เช่น ตัวอย่างมี Application ID เป็น jahcpe เมื่อลงทะเบียนสำเร็จแล้วจะสามารถเข้าถึง Application ได้จาก http://jahcpe.appspot.com เป็นต้น