ระบบจ่ายเงินอิเล็กทรอนิกส์โดยใช้โทรศัพท์มือถือแทนบัตรเครดิต

A SYSTEM FOR ELECTRONICS PAYMENT BY SMART PHONE INSTEAD OF USING CREDIT CARD

นายปัญญา แต่งงาม รหัส 46360053

นายสุเมธ แสงแผน รหัส 46360194

นายเอนกพงษ์ อินไชยเทพ รหัส 46360251

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2549

ใบรับรองโครงงานวิศวกรรม

หัวข้อโครงงาน	ระบบจ่ายเงินอิเล็กทรอนิกส์โดยใช้โทรศัพท์มือถือแทนบัตรเครดิต		
ผู้ดำเนินโครงงาน	นายปัญญา	แต่งงาม	รหัส 46360053
	นายสุเมธ	แสงแผน	รหัส 46360194
	นายเอนกพงษ์	อินไชยเทพ	รหัส 46360251
อาจารย์ที่ปรึกษา	อาจารย์ภาณุพงศ์ สอนคม		
สาขาวิชา	วิศวกรรมคอมพิวเตอร์		
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์		
ปีการศึกษา	2549		

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้โครงงานฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์

คณะกรรมการสอบโครงงานวิศวกรรม

..... ประธานกรรมการ

(อาจารย์ภาณุพงศ์ สอนคม)

..... กรรมการ

(ดร.สุรเดช จิตประไพกุลศาล)

..... กรรมการ

(อาจารย์จิราพร พุกสุข)

หัวข้อโครงงาน	ระบบจ่ายเงินอิเล็กทรอนิกส์โดยใช้โทรศัพท์มือถือแทนบัตรเครดิต		
ผู้ดำเนินโครงงาน	นายปัญญา	แต่งงาม	รหัส 46360053
	นายสุเมธ	แสงแผน	รหัส 46360194
	นายเอนกพงษ์	อินไชยเทพ	รหัส 46360251
อาจารย์ที่ปรึกษา	นายภาณุพงศ์ สอนคม		
สาขาวิชา	วิศวกรรมคอมพิวเตอร์		
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์		
ปีการศึกษา	2549		

บทคัดย่อ

โครงงานนี้ได้พัฒนาระบบการจ่ายเงินอิเล็กทรอนิกส์ผ่านมือถือแทนการใช้บัตรเครดิต ทำให้ผู้ใช้สามารถนำโทรศัพท์มือถือซึ่งเป็นอุปกรณ์ที่พกติดตัวเป็นประจำมาใช้ชำระค่าสินค้าและบริการอีกทั้งยังมีความปลอดภัยสูงกว่าการใช้บัตรเครดิต ในการพัฒนาระบบเน้นทางด้านความปลอดภัยของการรับส่งข้อมูลซึ่งเป็นการส่งข้อมูลระหว่างโทรศัพท์มือถือ (Client) กับเซิร์ฟเวอร์ (Server) โดยใช้โปรโตคอล SSL ในการรับส่งข้อมูลและใช้มาตรฐาน ECDSA ในการลงลายมือชื่อดิจิตอล ระบบที่ได้เป็นระบบที่มีความปลอดภัย เชื่อถือได้ ง่ายต่อการใช้งาน และสามารถนำไปประยุกต์ใช้ได้จริง

Project Title	A System For Electronics Payment By Smart Phone Instead Of Using Credit Card		
Name	Mr.Panya	Tang-ngarm	ID 46360053
	Mr.Sumet	Sangphan	ID 46360194
	Mr.Anekpong	Inchaitep	ID 46360251
Project Advisor	Mr.Panupong	Sornkom	
Major	Computer Engineering		
Department	Electrical and Computer Engineering		
Academic Year	2006		

Abstract

This project develops the electronics payment system instead of using credit card. User can take smart phone for pay goods and service charge so it's secure than credit card. This project is emphasizes in security of data transfer between smart phone (Client) and Server by uses the SSL protocol in data transfer and ECDSA algorithm in digital signature. So this developed system is very secure, reliable and usable.

กิตติกรรมประกาศ

ขอขอบพระคุณ อาจารย์ภาณุพงศ์ สอนคม อาจารย์ที่ปรึกษาโครงงานนี้ ที่ให้ความกรุณาแนะนำวิธีในการทำงานให้เข้าใจถึงการศึกษาอย่างเป็นระบบขั้นตอน อีกทั้งสละเวลาเพื่อตรวจสอบการทำงานและชี้แนวทางแก้ไขในทุกขั้นตอนตลอดการทำโครงงาน และสุดท้ายนี้ขอขอบพระคุณอาจารย์ทุกท่านและเพื่อนๆ ทุกคนที่ไม่ได้เอ่ยนามที่คอยให้ความช่วยเหลือและคำแนะนำต่างๆ จนโครงงานนี้สำเร็จได้ด้วยดี

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

ตารางที่ หน้า

สารบัญรูป

สารบัญรูป(ต่อ)

รูปที่ หน้า

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของโครงงาน

ระบบการชำระเงินด้วยเงินสดเป็นวิธีการชำระเงินที่มีมาตั้งแต่อดีต มีใช้แพร่หลายทั่วโลก ด้วยรูปแบบที่เรียบง่ายไม่มีวิธีการซับซ้อน อีกยังมีความปลอดภัยในระดับหนึ่ง จึงทำให้การชำระเงินสดเป็นกลไกหลักของระบบการชำระเงิน แต่ว่าระบบการชำระเงินด้วยเงินสดก็ยังมีปัญหาอยู่ เช่น การพกพาเงินสดเป็นจำนวนมากจะมีความเสี่ยงสูงในการเกิดอาชญากรรมต่อทั้งทางด้านร่างกายและทรัพย์สินรวมถึงความไม่สะดวกในการพกพา อีกทั้งวิวัฒนาการอันรวดเร็วของเทคโนโลยีในปัจจุบัน จึงทำให้มีการนำเทคโนโลยีมาประยุกต์ใช้กับระบบการชำระเงิน เป็นระบบการชำระเงินแบบใหม่ที่เรียกว่า ระบบการชำระเงินอิเล็กทรอนิกส์

ระบบการชำระเงินอิเล็กทรอนิกส์ได้เข้ามามีบทบาทสำคัญในตลาดเงิน ด้วยรูปแบบที่หลากหลายในการให้บริการ ซึ่งก็แล้วแต่ความสะดวกของแต่ละบุคคล เช่น ระบบชำระเงินผ่านบัตรเครดิต ระบบการชำระเงินออนไลน์ เป็นต้น โดยระบบการชำระเงินอิเล็กทรอนิกส์ดังกล่าวนี้สามารถตอบสนองความต้องของคนส่วนใหญ่ในสังคมได้เป็นอย่างดี ทั้งเรื่องการอำนวยความสะดวก ความปลอดภัย และในเรื่องของเวลา แต่ว่าระบบการชำระเงินอิเล็กทรอนิกส์เหล่านี้ก็ยังมีข้อเสียอยู่เช่นกัน คือ ในเรื่องของสถานที่ๆไม่สามารถใช้ได้ในทุกๆที่ได้ตามต้องการ ระบบการชำระเงินออนไลน์จำเป็นต้องอยู่ในสถานที่ๆ มีอินเตอร์เน็ตติดตั้งอยู่ ระบบชำระเงินผ่านบัตรเครดิตก็จำเป็นต้องมีเครื่องรูดบัตรเครดิตติดตั้งอยู่ในบริเวณที่จะใช้งานเช่นกัน ด้วยข้อเสียเหล่านี้ จึงมีการนำเทคโนโลยีแบบไร้สายเข้ามาใช้ เพื่อกำจัดข้อเสียของระบบการชำระเงินอิเล็กทรอนิกส์ในเรื่องของสถานที่ให้หมดไป

โทรศัพท์มือถือเป็นอุปกรณ์สื่อสารไร้สาย ซึ่งมีความนิยมมากที่สุดในปัจจุบัน จนถือได้ว่าเป็นปัจจัยที่ 5 ที่มีความจำเป็นในชีวิตประจำวันของคนไทยก็ว่าได้ เพราะว่าในครอบครัวหนึ่งๆจะมีมือถือใช้กันเกือบจะทุกคน ด้วยรูปแบบที่ง่ายในการสื่อสาร ราคาโทรศัพท์มือที่ไม่แพง มี Application ต่างๆให้เลือกใช้มากมาย รวมถึงกระแสแห่งเทคโนโลยีและแฟชั่น เป็นปัจจัยหนุนให้โทรศัพท์มือถือเป็นที่นิยม ดังนั้นโทรศัพท์มือถือจึงมีความเหมาะสมเป็นอย่างยิ่ง ที่จะถูกนำมาผนวกเข้ากับระบบการระบบการชำระเงินอิเล็กทรอนิกส์

ด้วยเหตุนี้ ทางคณะผู้จัดทำจึงตกลงกันว่า จะนำโทรศัพท์มือถือมาใช้ร่วมกับระบบการชำระเงินอิเล็กทรอนิกส์ เป็นระบบชำระเงินที่ใช้ผ่านงานผ่านทางโทรศัพท์มือถือ โดยระบบชำระ

เงินที่ใช้งานผ่านทางโทรศัพท์มือถือนี้จะเน้นเรื่องความปลอดภัย ความเสถียรของระบบ ความสะดวกสบาย ง่ายต่อการใช้งานเป็นหลักสำคัญ

1.2 วัตถุประสงค์ของโครงงาน

1. สามารถออกแบบระบบการชำระเงินอิเล็กทรอนิกส์ที่ใช้ผ่านทางผ่านทางโทรศัพท์มือถือ
2. ศึกษาการเข้ารหัสข้อมูลแบบไร้สาย แล้วนำมาประยุกต์ใช้กับระบบรักษาความปลอดภัยของระบบการชำระเงินอิเล็กทรอนิกส์ที่ใช้ผ่านโทรศัพท์มือถือ
3. พัฒนา Application บนโทรศัพท์มือถือเพื่อติดต่อกับระบบชำระเงินอิเล็กทรอนิกส์ที่ใช้ผ่านทางโทรศัพท์มือถือ
4. พัฒนา Application ของระบบชำระเงินอิเล็กทรอนิกส์บน Server ที่ติดต่อผ่านทางโทรศัพท์มือถือ

1.3 ขอบข่ายการทำงาน

1. ศึกษาข้อมูลเกี่ยวกับระบบการชำระเงินอิเล็กทรอนิกส์ในปัจจุบัน
2. ศึกษาข้อมูลในเรื่องการเข้ารหัสข้อมูลที่เน้นเกี่ยวกับความปลอดภัยในการรับส่งข้อมูลแบบไร้สาย
3. สร้างและพัฒนา Application บนโทรศัพท์มือถือ และ Application บน Server เกี่ยวกับระบบการชำระเงินอิเล็กทรอนิกส์ที่ใช้ผ่านทางโทรศัพท์มือถือ โดยใช้ภาษา J2ME และ JSP รวมถึงสร้างและพัฒนาการเข้ารหัสข้อมูลแบบไร้สายโดยใช้ภาษา JAVA

1.4 ขั้นตอนการดำเนินงาน

1. ศึกษาข้อมูลเกี่ยวกับหลักการและทฤษฎีต่างๆ ดังนี้
   1.1 ระบบของการชำระเงินอิเล็กทรอนิกส์ต่างๆ ที่มีในปัจจุบัน
   1.2 การทำงานของภาษา J2ME บนโทรศัพท์มือถือ
   1.3 การทำงานของ Server และภาษา JSP
   1.4 หลักการการเข้ารหัสข้อมูลที่ใช้ภาษา JAVA
2. ออกแบบและพัฒนาระบบ
3. ทดสอบและแก้ไขระบบ
4. วิเคราะห์การทดสอบพร้อมทั้งสรุปผล
5. จัดทำรายงาน

1.5 แผนการดำเนินงาน

กิจกรรม	ปี 2548		ปี 2549									
	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย	พ.ค.	มิ.ย.	ก.ค.	ส.ค.	ก.ย	ต.ค
1. ศึกษาข้อมูลเกี่ยวกับหลักการและทฤษฎีต่างๆ ดังนี้	←			→								
2. ออกแบบและพัฒนาระบบ					←			→				
3. ทดสอบและแก้ไขระบบ									←	→		
4. วิเคราะห์การทดสอบและสรุปผล											←	→
5. จัดทำรายงาน												↔

1.6 ผลที่คาดว่าจะได้รับ

1. เข้าใจหลักการทำงานของระบบการชำระเงินแบบต่างๆ
2. ออกแบบระบบการชำระเงินอิเล็กทรอนิกส์ที่ใช้ผ่านทางผ่านทางโทรศัพท์มือถือ
3. การทำให้ข้อมูลมีความปลอดภัย เพื่อป้องกันบุคคลอื่นสามารถเข้าถึงข้อมูลได้
4. ได้ Application บนโทรศัพท์มือถือเพื่อติดต่อกับระบบชำระเงินอิเล็กทรอนิกส์ที่ใช้ผ่านทางโทรศัพท์มือถือ
5. ได้ Application ของระบบชำระเงินอิเล็กทรอนิกส์บน Server ที่ติดต่อผ่านทางโทรศัพท์มือถือ
6. ได้ Open Source Software

1.7 งบประมาณ

1.	ค่าวัสดุสำนักงาน	เป็นเงิน	300	บาท
2.	ค่าวัสดุคอมพิวเตอร์	เป็นเงิน	1,000	บาท
3.	ค่าถ่ายเอกสาร	เป็นเงิน	1,000	บาท
4.	ค่าวัสดุอุปกรณ์	เป็นเงิน	700	บาท
รวมเป็นเงินทั้งสิ้น			3,000	บาท

หมายเหตุ ถัวเฉลี่ยทุกรายการ

บทที่ 2
หลักการและทฤษฎี

2.1 ระบบการชำระเงินอิเล็กทรอนิกส์

2.1.1 ระบบการชำระเงินสำหรับการพาณิชย์อิเล็กทรอนิกส์ในปัจจุบัน

ระบบการชำระเงิน (Payment System) เป็นองค์ประกอบสำคัญในการทำธุรกรรมที่เกี่ยวข้องกับพาณิชย์อิเล็กทรอนิกส์ ระบบชำระเงินที่มีประสิทธิภาพ มีต้นทุนต่ำและมีความปลอดภัยเท่านั้นที่จะช่วยให้การพาณิชย์อิเล็กทรอนิกส์สามารถแพร่หลายไปได้ในวงกว้างในปัจจุบัน การชำระเงินผ่านบัตรเครดิตเป็นวิธีหนึ่งในการชำระเงินที่สะดวกที่สุดของการพาณิชย์อิเล็กทรอนิกส์ระหว่างธุรกิจและผู้บริโภค (B-to-C E-Commerce) โดยทั่วไปในการชำระเงินด้วยวิธีการดังกล่าวมักจะมีการรักษาความปลอดภัยของข้อมูลที่เกี่ยวข้องด้วยเทคโนโลยีการเข้ารหัส (Encryption Technology) และใช้โปรโตคอล SSL ในสื่อสารระหว่างเครื่องคอมพิวเตอร์ที่เกี่ยวข้องเพื่อเข้ารหัสข้อมูลดังกล่าว อย่างไรก็ตามการชำระเงินผ่านบัตรเครดิตในปัจจุบันยังมีข้อจำกัดในการใช้ในการพาณิชย์ อิเล็กทรอนิกส์คือ การชำระเงินผ่านบัตรเครดิตด้วยโปรโตคอลแบบ SSL ยังไม่ได้แก้ไขปัญหาความปลอดภัยอย่างสมบูรณ์ เนื่องจากในกระบวนการดังกล่าวร้านค้าไม่สามารถแน่ใจได้ว่าผู้สั่งซื้อสินค้าเป็นบุคคลตามที่กล่าวอ้างและเป็นเจ้าของบัตรเครดิตนั้นจริงหรือไม่

2.1.2 ระบบการชำระเงินของการพาณิชย์อิเล็กทรอนิกส์ในอนาคต

จากข้อจำกัดดังกล่าวข้างต้น จะเห็นว่าในปัจจุบันยังไม่มีวิธีการในการชำระเงินที่สะดวกปลอดภัยและมีประสิทธิภาพสำหรับการซื้อขายสินค้าในยุคของการพาณิชย์อิเล็กทรอนิกส์ อย่างไรก็ตามได้มีการพัฒนาและทดลองใช้ระบบชำระเงินใหม่ๆ หลายระบบ เราอาจแบ่งระบบการชำระเงินที่ได้รับการพัฒนาขึ้นมาใหม่ออกเป็น 2 กลุ่มใหญ่ๆ คือ กลุ่มระบบการชำระเงินอิเล็กทรอนิกส์ (Electronic Payment System) กับกลุ่มเงินสดเล็กทรอนิกส์ (Electronic Cash) กลุ่มระบบการชำระเงิน

- **เช็คอิเล็กทรอนิกส์ (Electronic Check)** ซึ่งพัฒนาขึ้นมาจากระบบเช็คในปัจจุบันให้มีความเร็วมากขึ้นจากการปรับปรุงให้เป็นระบบอิเล็กทรอนิกส์ทั้งตัวเช็คการลงลายมือชื่อและการจัดส่งโดยใช้เทคโนโลยีเข้ารหัสข้อมูล (Encryption) ในการรักษาความปลอดภัย เช็คอิเล็กทรอนิกส์มีลักษณะคล้ายเช็คในปัจจุบันคือผู้รับชำระจะเป็นผู้รับความเสี่ยงในกรณีที่ผู้สั่งจ่ายไม่โอนเงินเข้าบัญชี

ระบบการชำระเงินโดยเช็คระหว่างธนาคารมีวัตถุประสงค์หลักให้การเคลียร์เช็คระหว่างธนาคารรวดเร็วขึ้น โดยใช้การส่งข้อมูลผ่านสื่ออิเล็กทรอนิกส์ ซึ่งสมาชิกของสำนักหักบัญชีสามารถใช้ได้ทั้งระบบออนไลน์และออฟไลน์ อย่างไรก็ตามปัจจุบันนี้สมาชิกของสำนักหักบัญชี

ทั้งหมดล้วนใช้ระบบออนไลน์กลไกการตัดบัญชีระหว่างสมาชิกเริ่มต้นจากผู้ได้รับเช็คสั่งจ่ายสามารถนำเช็คนั้นไปเข้าบัญชีที่ธนาคารของตนเพื่อให้เรียกเก็บจากธนาคารของผู้ออกเช็คให้เจ้าหน้าที่ธนาคารจะรับเช็คเข้าเครื่องอ่านอักษรแม่เหล็ก (MICR) ซึ่งเคลือบบนเช็คเพื่อบอกรหัสของธนาคารเจ้าของเช็ค และบันทึกรายการให้กับผู้เข้าเช็ค แต่ผู้เข้าเช็คยังไม่สามารถเบิกเงินในวันนั้นได้ทันที ข้อมูลที่บันทึกผ่านเครื่อง MICR และยอดเงินจะถูกส่งให้กับสำนักงานใหญ่ของธนาคารที่รับเช็ค ซึ่งจะรับข้อมูลออนไลน์จากสาขาต่างๆ แล้วรวมยอดส่งให้กับสำนักหักบัญชีเพื่อประมวลผลแบบ Batch โดยสำนักหักบัญชีอนุญาติให้ธนาคารส่งข้อมูลนี้ได้จนถึง 15.30 น. ของทุกวันทำการส่วนกระบวนการส่งตัวเช็คมาที่สำนักงานใหญ่นั้นเป็นไปตามที่ธนาคารจะบริหารภายในแต่ส่วนใหญ่จะให้ส่งภายในเย็นวันนั้น สำนักหักบัญชีจะทำการประมวลข้อมูลจากธนาคารสมาชิกต่างๆ เพื่อหายอดเงินสุทธิระหว่างธนาคารและทำการโอนเงินตามยอดเงินสุทธิแต่ละวันในบัญชีกระแสรายวันที่ทุกธนาคารเปิดไว้ที่ธนาคารแห่งประเทศไทย อย่างไรก็ตามแต่ละธนาคารยังไม่สามารถถอนเงินตามยอดนี้ได้ จนกระทั่งตัวเช็คถูกสำนักงานใหญ่ของแต่ละธนาคารรวบรวมส่งมาที่ธนาคารแห่งประเทศไทย โดยจะมีการจัดแยกเช็คตามธนาคารของผู้ออกเช็คและมอบเช็คให้กับธนาคารนั้นไป ในเช้าวันรุ่งขึ้นทางสำนักงานใหญ่ของแต่ละธนาคารจะทำการแยกเช็คตามสาขาและกระจายส่งกลับไปให้สาขาของตนเพื่อ ตรวจสอบความถูกต้องของเช็ค เช่นลายมือชื่อตรงกับที่เจ้าของบัญชีให้กับธนาคารไว้ หากสาขาต้องการปฏิเสธการจ่ายเงินเนื่องจากสาเหตุใดก็ตาม จะต้องแจ้งให้ธนาคารแห่งประเทศไทยทราบภายในเวลา 10.30 น. ในกรณีที่ไม่มีการคืนเช็คผู้นำเช็คเข้าฝากจะสามารถถอนเงินจากบัญชีได้ในเวลา 13.00 น. ในกรณีที่มีการเข้าบัญชีเช็คข้ามจังหวัดจะใช้ระยะเวลาเพิ่มขึ้นในการหักบัญชี เพราะต้องเสียเวลาในการส่งเช็คไปให้สาขาของธนาคารที่ออกเช็ค การใช้เช็คในการชำระเงินถือเป็นสื่อที่ได้รับความนิยมอย่างสูงในประเทศไทย อย่างไรก็ตามจะเห็นว่ายังมีข้อเสียคือใช้เวลานาน เพราะต้องมีการตรวจสอบลายมือชื่อของผู้ออกเช็ค โดยต้องแลกเปลี่ยนเช็คจากธนาคารผู้รับไปสู่ธนาคารผู้สั่งจ่าย และหากมีการคืนเช็ค กระบวนการนี้ก็จะย้อนกลับอีกครั้ง แทนที่จะมีเพียงการแลกเปลี่ยนข้อมูลอิเล็กทรอนิกส์เพียงอย่างเดียว

- **บัตรเครดิต (Credit Card)** ซึ่งมีการพัฒนาขึ้นมาจากบัตรเครดิตที่ใช้อยู่ในปัจจุบันและยังคงยึดหลักการเดิมคือบริษัทบัตรเครดิตเป็นผู้รับความเสี่ยงในกรณีที่ผู้ซื้อสินค้าไม่ชำระเงิน แต่ปรับปรุงให้มีความปลอดภัยมากขึ้น ตัวอย่างของการชำระเงินด้วยบัตรเครดิตที่ได้รับความสนใจในวงกว้างได้แก่ระบบ SET ซึ่งได้เพิ่มกลไกในการรักษาข้อมูลส่วนตัวของผู้ใช้บัตร และเพิ่มความปลอดภัยโดยการใช้ใบรับรองอิเล็กทรอนิกส์ (Electronic Certificate) เพื่อให้ร้านค้าและผู้บริโภคสามารถยืนยันกันได้ว่าทั้งสองฝ่ายเป็นบุคคลตามที่กล่าวอ้างจริง

การใช้บัตรเครดิตเป็นระบบชำระเงินที่ธนาคารให้วงเงินแก่ผู้ถือบัตรเพื่อซื้อหาสินค้าจากร้านค้าที่เป็นสมาชิกร้านค้าของธนาคาร การเรียกเก็บจะกระทำเป็นงวด เดือนละ 1 ครั้ง ปัจจุบันมี

บัตรเครดิต 2 ประเภทใหญ่ๆ ในประเทศไทย คือบัตรเครดิตที่ออกโดยธนาคารในประเทศ ซึ่งจะมีขอบเขตการใช้งานและร้านค้าที่รับบัตรน้อย

ในกระบวนชำระเงินด้วยบัตรเครดิตจะมีตัวกลาง 3 ฝ่ายคือ บริษัท VISA ซึ่งเชื่อมระหว่างธนาคารผู้ออกบัตรแก่ผู้ซื้อสินค้า (Issuer Bank) และธนาคารของร้านค้าซึ่งรับบัตร (Acquirer Bank) อาจกล่าวได้ว่าธนาคารผู้ออกบัตรนั้นเป็นตัวแทนฝั่งผู้ซื้อ และธนาคารผู้รับบัตรเป็นตัวแทนฝั่งผู้ขาย ในกระบวนการนี้ลูกค้าใช้บัตรเครดิตชำระเงินผ่านเครื่องรับบัตร (Electronic Data Capture) ซึ่งเชื่อมกับเครือข่ายของ VISA ในการตรวจสอบสถานะว่าบัตรนั้นถูกแจ้งอายัดหรือไม่ วงเงินที่ขออนุมัตินั้นมากกว่าวงเงินที่เจ้าของบัตรเหลืออยู่หรือไม่ การตรวจสอบส่วนใหญ่ใช้วิธีติดต่อผ่านโมเด็มไปยังธนาคารผู้รับบัตร เมื่อวงเงินได้รับการอนุมัติ เครื่องจะพิมพ์สลิปรายการขึ้นเพื่อให้ผู้ซื้อลงชื่อกำกับ ผู้ขายสามารถนำรายการเหล่านี้แต่ละวันไปเบิกเงินกับธนาคารผู้รับบัตรที่ผู้ขายมีบัญชีสมาชิกร้านค้า (Merchant Account) อยู่ได้ ซึ่งธนาคารผู้รับบัตรจะทำการโอนเงินตามยอดที่ร้านค้าส่ง หลังจากหักค่าธรรมเนียมของธนาคารซึ่งอยู่ในราว 2-3% ให้แก่ร้านค้าภายในระยะเวลา 1 วันจากวันที่ผู้ขายส่งยอด จะสังเกตเห็นว่าภายในระบบนี้ ธนาคารผู้รับบัตรนั้นได้ออกเงินล่วงหน้าให้แก่ผู้ขายไปก่อนที่จะรับชำระจากผู้ซื้อ เนื่องจากต้องรอจนถึงรอบเก็บเงินของผู้ซื้อ โดยอาจกินเวลาถึง 30 วันนับจากวันสั่งซื้อสินค้า เครือข่าย VISA สามารถเชื่อมโยงการสื่อสารระหว่างธนาคารผู้ซื้อและผู้ขายแม้ว่าจะอยู่คนละประเทศก็ตาม ภายใต้ระบบนี้ ผู้ซื้อจะได้รับใบเรียกเก็บเงินจากธนาคารผู้ออกบัตรประจำรอบของตน และทำการชำระเงิน หากมีข้อพิพาทเช่นผู้ซื้อปฏิเสธการชำระราคาสินค้าตามรายการนั้น ผู้ซื้อจะต้องแจ้งให้ธนาคารผู้ออกบัตรทราบเป็นลายลักษณ์อักษร ซึ่งธนาคารผู้ออกบัตรจะติดต่อธนาคารผู้รับบัตรผ่านเครือข่าย VISA เมื่อได้รับแจ้งแล้วผู้ขายจะได้รับการติดต่อจากธนาคารผู้รับบัตรทางโทรศัพท์เพื่อให้ตรวจสอบธุรกรรม การแก้ข้อพิพาทส่วนใหญ่จะนำสลิปซึ่งมีลายเซ็นของผู้ซื้อส่งเป็นสำเนาให้ผู้ซื้อรับทราบ ปัญหาส่วนใหญ่มักเกิดจากการที่ชื่อร้านค้าในใบเรียกเก็บเงินนั้นต่างไปจากชื่อทางการค้าที่ผู้ซื้อคุ้นเคย ในกรณีที่ไม่สามารถแสดงหลักฐานเพื่อแก้ข้อพิพาทได้ ธนาคารผู้รับบัตรจะขอให้ร้านค้าคืนเงินที่ธนาคารเคยให้ไปพร้อมทั้งค่าธรรมเนียม ภายใต้กฎของ VISA ผู้ขายมีความรับผิดชอบต่อการโต้แย้งของผู้ซื้อถึง 6 เดือนดังนั้นในทางปฏิบัติธนาคารจะให้ผู้ขายฝากเงินค้ำประกันก้อนหนึ่งก่อนเปิดบัญชีร้านค้าให้ใช้บริการได้

- **บัตรเดบิต (Debit Card)** ซึ่งคล้ายกับบัตรเดบิตที่ใช้อยู่ในปัจจุบัน และยังคงยึดหลักการเดิมคือไม่มีผู้ใดแบกรับความเสี่ยง เนื่องจากจะมีการโอนเงินเข้าบัญชีของร้านค้าในทันทีและผู้สั่งจ่ายไม่สามารถสั่งจ่ายเกินกว่ายอดเงินในบัญชีของตนได้บัตรเดบิตสำหรับการพาณิชย์อิเล็กทรอนิกส์จะมีข้อตกต่างจากบัตรเดบิตในปัจจุบันตรงที่สามารถใช้กับเครือข่ายอินเทอร์เน็ตได้โดยใช้เทคโนโลยีการเข้ารหัสในการรักษาความปลอดภัย

บัตรเดบิตเป็นส่วนผสมระหว่างบัตร ATM และบัตรเครดิต โดยผู้ใช้บัตรจะต้องมีบัญชีเงินฝากจากธนาคารก่อนเมื่อต้องการใช้เงินผู้ถือบัตรเดบิตสามารถถอนเงินจากเครื่อง ATM โดยใช้บัตร

ของตนเช่นเดียวกับบัตร ATM แต่ส่วนที่คล้ายบัตรเครดิตคือผู้ถือบัตรสามารถใช้บัตรเดบิตในการชำระราคาสินค้าหรือบริการตามร้านค้าเช่นเดียวกับบัตรเครดิต โดยธนาคารจะหักบัญชีเงินฝากของเจ้าของบัตร เดบิตไปสู่บัญชีร้านค้าในทันที ในกรณีที่ผู้ถือบัตรต้องการสินเชื่อจากธนาคาร ผู้ถือบัตรเดบิตจะต้องสมัครเพิ่มและธนาคารจะพิจารณาตามหลักเกณฑ์เช่นเดียวกับการออกบัตรเครดิต โดยปกติธนาคารจะให้วงเงินคิดเป็น 2 เท่าของเงินเดือน ธนาคารพาณิชย์บางแห่งในประเทศไทยได้พัฒนาระบบรับชำระเงินของร้านค้าพาณิชย์อิเล็กทรอนิกส์ให้สามารถรับชำระเงินจากลูกค้า

ในระบบพาณิชย์อิเล็กทรอนิกส์ได้โดยใช้บัตรเดบิตหรือบัตร ATM อย่างไรก็ตาม วิธีนี้มีข้อจำกัดคือร้านค้าพาณิชย์อิเล็กทรอนิกส์นั้นจะต้องสมัครเป็นสมาชิกร้านค้ากับธนาคารเดียวกับเจ้าของบัตรด้วย ทำให้มีต้นทุนการดำเนินการสูง เพราะหากร้านค้าต้องการรับบัตรเดบิตจากหลายธนาคาร ก็จะต้องสมัครกับทุกธนาคาร

2.1.3 เงินสดอิเล็กทรอนิกส์

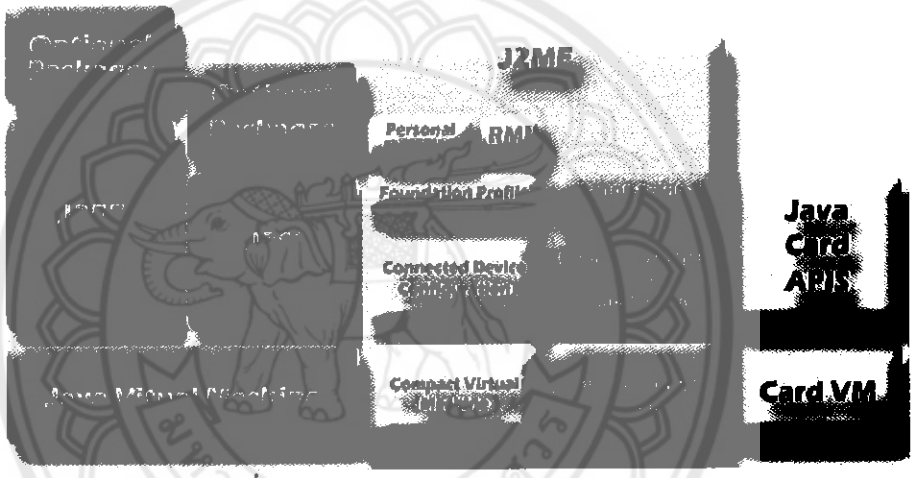
"เงินสดอิเล็กทรอนิกส์" (Electronic Cash) หรือ เรียกสั้นๆ ว่า "เงินอิเล็กทรอนิกส์" เป็นเทคโนโลยีที่ได้รับการพัฒนาขึ้นมาให้มีความปลอดภัย และเหมาะสมกับการชำระเงินมูลค่าต่ำๆ ที่เรียกว่า "การชำระเงินขนาดย่อม" (Micro Payment) ซึ่งจะสามารถรองรับการพาณิชย์อิเล็กทรอนิกส์ที่เกี่ยวข้องกับสินค้าหรือบริการที่มีมูลค่าต่ำเช่นบทความสั้นๆ หรือการเช่าซอฟต์แวร์ทางเครือข่าย เป็นต้นเงินอิเล็กทรอนิกส์ ซึ่งหมายถึงการใช้เทคโนโลยีในการเก็บข้อมูลในรูปแบบดิจิตอล และข้อมูลนั้นเป็นตัวแทนของมูลค่า (Stored Value) ซึ่งผู้ถือข้อมูลนั้นได้ชำระไว้แล้วล่วงหน้า (Prepaid) โดยที่ข้อมูลนั้นสามารถนำไปใช้ชำระเงินด้วยวิธีต่างๆ ไม่ว่าจะเป็นการชำระค่าสินค้าหรือบริการ ณ จุดขาย (Point of Sale) หรือเป็นการเปลี่ยนมือจากผู้หนึ่งไปยังอีกผู้หนึ่งโดยผ่านอุปกรณ์บางอย่าง (Direct Transfer) หรือผ่านเครือข่ายเครือข่ายอินเทอร์เน็ต เป็นต้น นอกจากนี้สิ่งที่เราจะถือว่าเป็นเงินอิเล็กทรอนิกส์จะต้องมีคุณสมบัติของความเป็นเงิน (Moneyness) ซึ่งได้แก่ ความเชื่อถือและความสามารถใช้ได้อย่างกว้างขวาง อันเป็นคุณสมบัติที่จะแยกระหว่างเงินอิเล็กทรอนิกส์ออกจากบัตรซื้อสินค้าที่ต้องจ่ายเงินล่วงหน้า (Prepaid Card) อื่นๆ เช่น คูปองของศูนย์อาหาร หรือบัตรโทรศัพท์ เป็นต้นเงินอิเล็กทรอนิกส์เหล่านี้อาจเป็นเงินที่ใช้ได้ในแบบออฟไลน์ (Off-Line) คือโอนเงินระหว่างสมาร์ทคาร์ดกับอุปกรณ์พิเศษ หรือในแบบออนไลน์ (On-Line) ผ่านเครือข่ายอินเทอร์เน็ต แบบสามารถเปลี่ยนมือได้ (Open loop) ซึ่งสามารถถ่ายโอนต่อไปได้โดยอิสระ หรือแบบที่ไม่สามารถเปลี่ยนมือได้โดยไม่ผ่านตัวกลาง (Closed loop) แบบทราบตัวผู้ใช้ (Onymous) หรือแบบที่ไม่ทราบตัวผู้ใช้ (Anonymous) เป็นต้น

เงินอิเล็กทรอนิกส์จึงไม่รวมวิธีการชำระเงินอิเล็กทรอนิกส์ต่างๆ เช่น บัตรเครดิต ซึ่งเป็นการชำระเงินจริงๆ หลังจากการซื้อสินค้าหรือบริการ (Pay-Later) และบัตรเดบิตซึ่งเป็นการชำระเงินพร้อมกับการซื้อสินค้าหรือบริการ (Pay-Now) เพราะระบบการชำระเงินทั้งสองไม่ใช่ระบบที่ชำระเงินล่วงหน้า (Pay-Before หรือ Prepaid) นอกจากนี้เงินอิเล็กทรอนิกส์ตามคำจำกัดความดังกล่าวยัง

ไม่ครอบคลุมถึงระบบการชำระเงินอื่นๆ ซึ่งไม่มีการชำระเงินล่วงหน้า (Prepaid) เช่น การโอนเงินผ่านเครือข่ายของธนาคารพาณิชย์

2.2 J2ME

การพัฒนาโปรแกรมเพื่อรองรับการทำงานของอุปกรณ์ไร้สาย เป็นอีกความพยายามหนึ่งของบริษัท Sun Microsystems ที่สามารถเปลี่ยนโฉมหน้าเทคโนโลยีและบทบาทของอินเทอร์เน็ตไปอย่างสิ้นเชิง จุดมุ่งหมายที่สำคัญของการออกแบบภาษาจาวา คือ โปรแกรมต้องทำงานบนเครื่องต่างระบบกันได้ โดยเรียกคุณสมบัตินี้ว่า "ความไม่ขึ้นกับระบบ" ซึ่งเป็นภาษาเชิงออบเจ็กต์ที่สามารถทำงานได้บนทุกระบบปฏิบัติการ "Write once, Run Anywhere"



รูปที่ 2.1 โครงสร้างของ JAVA Technology

รูปจาก http://www.thai-programmer.com/image/jme2.jpg

JAVA เวอร์ชันล่าสุดหรือ JAVA2 นั้นได้ถูกพัฒนาออกมา 3 รุ่น เพื่อความเหมาะสมในการเขียนโปรแกรมบนอุปกรณ์ที่มีทรัพยากรแตกต่างกัน ดังนี้

- **J2SE (JAVA 2 Platform, Standard Edition)** ใช้สำหรับการเขียนโปรแกรมบนคอมพิวเตอร์เดสก์ท็อปทั่วๆ ไป
- **J2EE (JAVA 2 Platform, Enterprise Edition)** ใช้สำหรับการเขียนโปรแกรมบนระบบงานใหญ่ๆ โดยเพิ่มศักยภาพของ J2SE ให้สามารถรองรับการทำงานแบบ Server Side ซึ่งมีการใช้งานจากผู้ใช้ (Client) เป็นจำนวนมาก
- **J2ME (JAVA 2 Platform, Micro Edition)** ใช้สำหรับเขียนโปรแกรมบนอุปกรณ์ขนาดเล็กซึ่งมีทรัพยากร เช่น การแสดงผล ขนาดของหน่วยความจำ และความสามารถในประมวลผลจำกัด โดยตัวอย่างอุปกรณ์พวกนี้ ได้แก่ โทรศัพท์มือถือ และ PDA เป็นต้น

นอกจากนี้ J2ME ยังสามารถใช้พัฒนาแอพพลิเคชันให้ทำงานบนอุปกรณ์ที่ไม่ได้เป็นอุปกรณ์ไร้สายได้อีกด้วยถ้าไม่มีข้อจำกัดเพื่อดาวน์โหลดและติดตั้งซอฟท์แวร์ เช่น กล่องรับสัญญาณดาวเทียมสำหรับทีวี อินเทอร์เน็ตทีวี เป็นต้น

บริษัท Sun Microsystems เป็นผู้ริเริ่มในการพัฒนาเทคโนโลยี J2ME แต่ปัจจุบันรับการสนับสนุนจากผู้ผลิตอุปกรณ์อิเล็กทรอนิกส์ชั้นนำของโลกอยู่หลายบริษัท โดยอยู่ภายใต้การดูแลของ JCP (JAVA Community Process) เพื่อให้การพัฒนา J2ME เป็นไปในทิศทางเดียวกัน และอยู่ภายใต้มาตรฐาน JAVA ของ Sun

โปรแกรม JAVA ทุกตัวจะต้องทำงานภายใต้ JAVA Virtual Machine (JVM) เสมอ เมื่อเราคอมไพล์โปรแกรมเป็นไบต์โค้ด (ไฟล์ .class) แล้ว JVM จะทำหน้าที่แปลงไบต์โค้ดเหล่านี้ไปเป็นภาษาเครื่องและทำงานตามคำสั่งนั้นๆ ต่อไป ด้วยวิธีนี้โปรแกรม JAVA จึงสามารถทำงานได้บนทุกระบบปฏิบัติการ ขอเพียงแต่มี JVM บนระบบปฏิบัติการนั้นๆ ก็พอ ซึ่ง JVM นี้ก็จะเปลี่ยนไปตามระบบปฏิบัติการ ของอุปกรณ์แต่ละชนิด ซึ่ง J2ME ได้ใช้ Configuration เป็นตัวกำหนด JVM ดังนี้

ตารางที่ 2.1 Configuration และ JVM ของ J2ME

Configuration	**JVM**
CDL	CVM (Compact Virtual Machine)
CLDC	KVM (Kilobyte Virtual Machine)

2.2.1 การแบ่งอุปกรณ์ตาม Configuration

Configuration เป็นตัวระบุ Virtual Machine และคลาสไลบรารีพื้นฐาน ซึ่งจะมีเหมือนกันในอุปกรณ์ทุกตัวที่ถูกจัดอยู่ในกลุ่มเดียวกัน โดย Configuration ใน J2ME ได้แบ่งกลุ่มของอุปกรณ์ออกเป็น 2 กลุ่ม โดยใช้คุณสมบัติของหน่วยความจำ การแสดงผล และความสามารถในการประมวลผลเป็นตัวกำหนด ดังนี้

1) **CDC (Connected Device Configuration)** คุณสมบัติของอุปกรณ์ในกลุ่ม ได้แก่
- มีหน่วยความจำตั้งแต่ 2-16 MB
- มีหน่วยประมวลผลขนาด 32 บิต เป็นอย่างน้อย
- ความเร็วในการเชื่อมต่อเครือข่ายค่อนข้างสูง
- ตัวอย่างอุปกรณ์ เช่น Pocket PC และ Set-TOP BOX ของเคเบิลทีวี

2) **CLDC (Connected Limited Device Configuration)** คุณสมบัติของอุปกรณ์ในกลุ่ม ได้แก่
- มีหน่วยความจำ 160-512 KB
- มีหน่วยประมวลผลขนาด 16-32 บิต ซึ่งมีความเร็วอย่างน้อย 25 MHz

- มีข้อจำกัดในการแสดงผล
- ความเร็วในการเชื่อมต่อเครือข่ายค่อนข้างต่ำ
- ตัวอย่างอุปกรณ์ เช่น โทรศัพท์มือถือ เพจเจอร์

ฟังก์ชันต่างๆ ของ CDC และ CLDC ส่วนใหญ่จะสืบทอดมาจาก J2SE และมีส่วนที่เพิ่มเข้ามาเพื่อให้เหมาะสมกับการทำงานบนอุปกรณ์ขนาดเล็กที่มีทรัพยากรจำกัด ซึ่งความสัมพันธ์ระหว่างคลาสไลบรารีของ J2SE และ J2ME สามารถเขียนอธิบายได้ ดังรูปด้านล่าง

รูปที่ 2.2 ความสัมพันธ์ระหว่าง J2SE และ J2ME Configuration
รูปจาก http://www.thai-programmer.com/image/jme1.jpg

กลุ่มคลาสพื้นฐานที่มีให้เรียกใช้ซึ่งกำหนดเอาไว้ใน Configuration ของอุปกรณ์มือถือหรือ CLDC มีดังนี้

ตารางที่ 2.2 คลาสพื้นฐานที่มีให้เรียกใช้ใน Configuration ของอุปกรณ์มือถือหรือ CLDC

Package ใน CLDC	รายละเอียด
ที่สืบทอดมาจาก J2SE	
JAVA.io	กลุ่มคลาสสำหรับรับส่งข้อมูล
JAVA.lang	กลุ่มคลาสสำหรับภาษา JAVA
JAVA.util	กลุ่มคลาสต่างๆ
ที่เพิ่มเติมใน CLDC	
JAVAx.microedition.io	กลุ่มคลาสสำหรับส่งข้อมูลผ่านระบบเครือข่าย

2.2.2 Profile

Profile เป็นตัวกำหนดกลุ่มของไลบรารีที่เพิ่มเติมจาก Configuration เพื่อรองรับข้อแตกต่างของอุปกรณ์แต่ละชนิด ดังนั้น Profile จึงเกี่ยวข้องกับคุณลักษณะทางด้านฮาร์ดแวร์ของอุปกรณ์แต่ละตัว เช่น อุปกรณ์มีช่องทาง (Interface) ติดต่อกับผู้ใช้อย่างไร หรืออุปกรณ์ติดต่อกับเครือข่ายได้อย่างไร เป็นต้น

Profile เป็นส่วนของ API และ Class ที่ใช้งานได้บนตัวของอุปกรณ์ แต่ละประเภท ซึ่งเป็นการขยายความสามารถของ CDC และ CLDC ให้มากขึ้น และมีส่วนของการทำงานที่เป็นลักษณะเฉพาะของอุปกรณ์นั้นๆ ตัวอย่างของ Profile เช่น

- **MIDP (Mobile Information Device Profile)** หมายถึง ประเภท Device ที่มีคุณสมบัติ Small Display (min. 96 x 54 pixels) มี Touch Screen หรือ Keyboard สามารถ Connect Mobile Network ด้วย Band With ที่จำกัด MIDP ประกอบด้วย APIs ที่ทำหน้าที่ต่อไปนี้
  - User Interface จัดการเกี่ยวกับการแสดงผล
  - Persistent Storage จัดการเกี่ยวกับการเก็บข้อมูลและฐานข้อมูล
  - Network จัดการเกี่ยวกับการเชื่อมต่อเน็ตเวิร์ค
  - Application Life-Cycle จัดการเกี่ยวกับลำดับขั้นตอนการทำงาน

  MIDP Packages จะมีอยู่ 3 Packages คือ
  - JAVAx.microedition.midlet เป็น API ในการสร้างโปรแกรมหลัก
  - JAVAx.microedition.lcdui เป็น API ในการจัดการ User Interface
  - JAVAx.microedition.rms (RMS : Record Management System) เป็นส่วนของการเก็บข้อมูลเช่นเดียวกับฐานข้อมูล
- **PDA Profile (Personal Digital Assistant Profile)** สำหรับอุปกรณ์ประเภท Organizer เช่น เครื่อง Palm
- **Foundation Profile** สำหรับอุปกรณ์ในกลุ่มของ High-End Device, เป็นส่วนขยายเพิ่มเติมเฉพาะด้านให้กับ CDC ซึ่งจะประกอบด้วย API และ Function พื้นฐาน
- **Personal Profile** สำหรับอุปกรณ์ในกลุ่มของ High-End Device, เป็นส่วนขยายเพิ่มเติมเฉพาะด้านให้กับ Foundation Profile ซึ่งจะประกอบด้วย การจัดการด้าน GUI
- **RMI Profile** สำหรับอุปกรณ์ในกลุ่มของ High-End Device, เป็นส่วนขยายเพิ่มเติมเฉพาะด้านให้กับ Foundation Profile ซึ่งจะประกอบด้วย การจัดการด้าน RMI (Remote Method Invocation)

2.3 ระบบความปลอดภัย

2.3.1 มาตรการการรักษาความปลอดภัยของข้อมูล

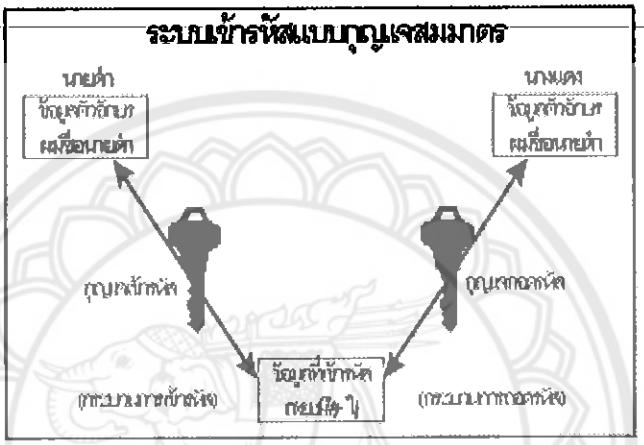
จุดประสงค์ของระบบความปลอดภัยจะคำนึงถึงหลักสำคัญดังต่อไปนี้

- **การพิสูจน์ตัวจริง (Authentication)** คือ การระบุตัวบุคคลที่ติดต่อว่าเป็น บุคคลตามที่ได้กล่าวอ้างไว้จริง และมี อำนาจหน้าที่ตามที่ได้กล่าวอ้างไว้จริง (เปรียบได้กับการแสดงตัวด้วยบัตรประจำตัวซึ่งมีรูปติดอยู่ด้วย หรือการใช้ระบบล็อคซึ่งผู้ที่จะเปิดได้จะต้องมีกุญแจอยู่เท่านั้น เป็นต้น)
- **การรักษาความสมบูรณ์ (Integrity)** คือ การรับรองว่าข้อมูลจะไม่ถูกเปลี่ยนแปลงหรือทำลายไม่ว่าจะเป็นโดย อุบัติเหตุหรือโดยเจตนา (เปรียบเทียบได้กับการเขียนด้วยหมึกซึ่งถ้าถูกลบแล้วจะก่อให้เกิดรอยลบขึ้น)
- **การห้ามปฏิเสธความรับผิดชอบ (Non-Repudiation)** คือ วิธีการสื่อสารซึ่งผู้ส่งข้อมูลได้รับหลักฐานว่าได้มีการส่งข้อมูลแล้วและผู้รับก็ได้รับการยืนยันว่าผู้ส่งเป็นใคร ดังนั้นทั้งผู้ส่งและผู้รับจะไม่สามารถปฏิเสธได้ว่าไม่มีความเกี่ยวข้องกับข้อมูลดังกล่าวในภายหลัง (เปรียบเทียบได้กับการส่งจดหมายลงทะเบียน เป็นต้น)
- **การรักษาความลับของข้อมูล (Confidentiality)** คือการรับรองว่าจะมีการเก็บข้อมูลไว้เป็นความลับ และผู้มีสิทธิเท่านั้นจึงจะเข้าถึงข้อมูลนั้นได้ (เปรียบเทียบได้กับ การปิดผนึกซองจดหมาย การใช้ซองจดหมายที่ทึบแสง การเขียนหมึกที่มองไม่เห็น เป็นต้น)
- **ความพร้อมใช้ (Availability)** คือการรับรองว่าข้อมูลและบริการการสื่อสารต่าง ๆ พร้อมที่จะใช้ได้ในเวลาที่ต้องการใช้งาน

2.3.2 วิธีการรักษาความปลอดภัย

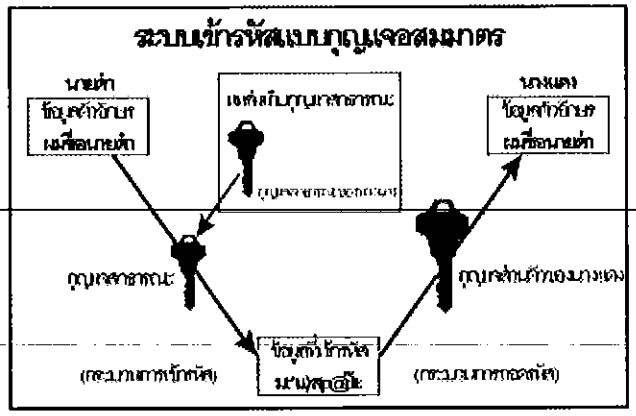
- **การเข้ารหัสลับ (Encryption)** คือ การทำให้ข้อมูลที่จะส่งผ่านไปทางเครือข่ายอยู่ในรูปแบบที่ไม่สามารถอ่านออกได้ ด้วยการเข้ารหัสลับ (Encryption) ทำให้ข้อมูลนั้นเป็นความลับ ซึ่งผู้ที่มีสิทธิ์จริงเท่านั้นจะสามารถอ่านข้อมูลนั้นได้ด้วยการถอดรหัสลับ (Decryption) นั่นคือสามารถรักษาข้อมูลให้เป็นความลับ (Confidentiality) การพิสูจน์ตัวจริงและการให้อำนาจ (Authentication & Authorization) สำหรับการเข้ารหัสลับและถอดรหัสลับนั้นจะอาศัยสมการทางคณิตศาสตร์ที่ซับซ้อน ซึ่งต้องอาศัยกุญแจซึ่งอยู่ในรูปของพารามิเตอร์ที่กำหนดไว้ (สำหรับตัวกุญแจนั้นจะมีความยาวเป็น บิต(Bit) ยิ่งกุญแจมีความยาวมาก ยิ่งปลอดภัยมาก เนื่องจากจะต้องใช้เวลามากขึ้นในการคาดเดากุญแจของผู้ลุกคาม) ในการเข้าและถอดรหัสลับ สามารถแบ่งออกเป็น 2 ประเภท คือ การเข้ารหัสลับแบบกุญแจสมมาตร (Symmetric Key Cryptography หรือ Secret Key Cryptography) และ การเข้าแบบอสมมาตร (Asymmetric Key Cryptography หรือ Public Key Cryptography)

1) การเข้าแบบกุญแจสมมาตร หมายถึง การเข้าและถอดรหัสลับโดยใช้กุญแจลับที่เหมือนกัน ซึ่งมีขั้นตอนแสดงดังตัวอย่าง ในรูปที่ 1 คือ นายดำเป็นผู้ส่ง จะทำการส่งข้อความ "ผมชื่อนายดำ" ผ่านไปยัง ผู้รับคือนางแดง โดยที่นายดำทำการเข้ารหัสข้อความ "ผมชื่อนายดำ" ด้วยกุญแจลับ โดยข้อความนั้นจะเปลี่ยนเป็นข้อความที่เข้ารหัสลับแล้ว (Cipher Text) "ก\ยd-ฟ)+ใ" แล้วถูกส่งไปยังนางแดง จากนั้นนางแดงก็ใช้กุญแจลับเดียวกันกับที่นายแดงใช้เข้ารหัสมาทำการถอดรหัสลับออกมาเป็นข้อความเดิมคือ "ผมชื่อนายดำ" ในกรณีนี้กุญแจลับจะเป็นกุญแจเดียวกัน ซึ่งจะต้องเป็นที่รู้กันเพียงผู้ส่งและผู้รับเท่านั้น



รูปที่ 2.3 ระบบการเข้า และ ถอดรหัส แบบกุญแจสมมาตร
รูปจาก http://www.ecommerce.or.th/faqs/faq3-1.html#1

2) การเข้ารหัสลับแบบกุญแจอสมมาตร หมายถึง การเข้าและการถอดรหัสลับด้วยกุญแจต่างกัน ซึ่งมีขั้นตอนดังตัวอย่างที่แสดงไว้ในรูปที่ 2 คือ นายดำเป็นผู้ส่งทำการเข้ารหัสข้อความ "ผมชื่อนายดำ" ไปเป็น "มt*แ)สp@ด๊ะ" ด้วยกุญแจสาธารณะของผู้รับได้แก่ นางแดง ซึ่งนายดำขอกุญแจนั้นมาจากองค์กรกลางที่เก็บกุญแจสาธารณะของบุคคลต่างๆไว้ จากนั้นข้อความที่เข้ารหัสลับแล้วถูกส่งไปยัง นางแดง นางแดงจะทำการถอดรหัสลับข้อความด้วยกุญแจส่วนตัวของนางแดง และนางแดงเท่านั้นจะเป็นผู้มีสิทธิ์ เนื่องจากนางแดงจะเป็นผู้เดียวที่มีกุญแจส่วนตัวของนางแดงเอง นั่นคือ ในการส่งข้อความด้วยการเข้ารหัสลับแบบกุญแจอสมมาตร จะเน้นที่ผู้รับเป็นหลัก คือ จะใช้กุญแจสาธารณะของผู้รับซึ่งเป็นที่เปิดเผยในการเข้ารหัสลับ และจะใช้กุญแจส่วนตัวของผู้รับในการถอดรหัสลับ



รูปที่ 2.4 ระบบการเข้า และ ถอดรหัสลับ แบบกุญแจอสมมาตร

รูปจาก http://www.ecommerce.or.th/faqs/faq3-1.html#1

- **ลายมือชื่อดิจิทัล (Digital Signature)** ในการส่งข้อมูลผ่านเครือข่ายนั้น นอกจากจะทำให้ข้อมูลที่ส่งนั้นเป็นความลับสำหรับผู้ไม่มีสิทธิ์โดยการใช้เทคโนโลยีการรหัสแล้ว สำหรับการทำนิติกรรมสัญญาโดยทั่วไป ลายมือชื่อจะเป็นสิ่งที่ใช้ในการพิสูจน์ตัวจริง (Authentication) และยังแสดงถึงเจตนาในการยอมรับเนื้อหาในสัญญานั้นๆซึ่งเชื่อมโยงถึงการป้องกันการปฏิเสธความรับผิดชอบ (Non-Repudiation) สำหรับในการทำธุรกรรมทางอิเล็กทรอนิกส์นั้นจะใช้ลายมือชื่ออิเล็กทรอนิกส์ (Electronic Signature) ซึ่งมีรูปแบบต่างๆ เช่น สิ่งที่ระบุตัวบุคคลทางชีวภาพ (ลายพิมพ์นิ้วมือ เสียง ม่านตา เป็นต้น) หรือจะเป็นสิ่งที่มอบให้แก่บุคคลนั้นๆในรูปแบบของรหัสประจำตัว ตัวอย่างที่สำคัญของลายมือชื่ออิเล็กทรอนิกส์ที่ได้รับการยอมรับกันมากที่สุดอันหนึ่งคือ ลายมือชื่อดิจิทัล (Digital Signature) ซึ่งจะเป็นองค์ประกอบหนึ่งในโครงสร้างพื้นฐานกุญแจสาธารณะ (Public Key Infrastructure, PKI)

รูปที่ 2.5 ลายมือชื่อดิจิทัลเป็นตัวอย่างหนึ่งของลายมือชื่ออิเล็กทรอนิกส์

รูปจาก http://www.ecommerce.or.th/faqs/faq3-1.html#1

ลายมือชื่อดิจิทัล (Digital Signature) คือ ข้อมูลอิเล็กทรอนิกส์ที่ได้จากการเข้ารหัสข้อมูลด้วยกุญแจส่วนตัวของผู้ส่ง ซึ่งเปรียบเสมือนเป็นลายมือชื่อของผู้ส่ง คุณสมบัติของลายมือชื่อดิจิทัลนอกจากจะสามารถระบุตัวบุคคลและเป็นกลไกการป้องกันการปฏิเสธความรับผิดชอบแล้ว ยัง

สามารถป้องกันข้อมูลที่ส่งไปไม่ให้ถูกแก้ไข หรือหากถูกแก้ไขไปจากเดิมก็สามารถล่วงรู้ได้ กระบวนการสร้างและลงลายมือชื่อดิจิทัลมีขั้นตอนแสดงดังในรูปที่ 4 คือ

เริ่มจากการนำเอาข้อมูลอิเล็กทรอนิกส์ต้นฉบับที่จะส่งไปนั้นมาผ่านกระบวนการทางคณิตศาสตร์ที่เรียกว่า ฟังก์ชันย่อยข้อมูล (Hash Function) เพื่อให้ได้ข้อมูลที่สั้นๆ ที่เรียกว่า ข้อมูลที่ย่อยแล้ว (Digest) ก่อนที่จะทำการเข้ารหัสลับ เนื่องจากข้อมูลต้นฉบับมักจะมีความยาวมากซึ่งจะทำให้กระบวนการเข้ารหัสลับใช้เวลานานมาก

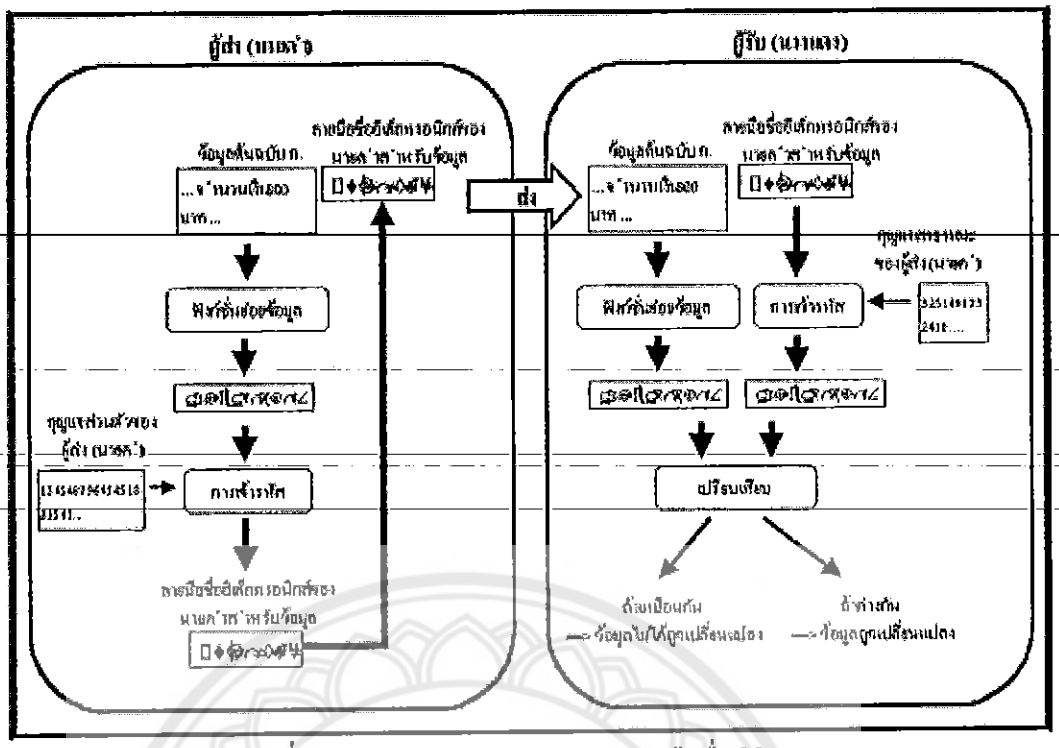
จากนั้นจึงทำการเข้ารหัสลับด้วยกุญแจส่วนตัวของผู้ส่งเอง ซึ่งจุดนี้เปรียบเสมือนการลงลายมือชื่อของผู้ส่งเพราะผู้ส่งเท่านั้นที่มีกุญแจส่วนตัวของผู้ส่งเอง แล้วจะได้ข้อมูลที่เข้ารหัสลับแล้ว เรียกว่า ลายมือชื่อดิจิทัล

จากนั้นก็ทำการส่งลายมือชื่อไปพร้อมกับข้อมูลต้นฉบับไปยังผู้รับ ผู้รับก็จะทำการตรวจสอบว่าข้อมูลที่ได้รับถูกแก้ไขระหว่างทางหรือไม่ โดยการนำข้อมูลต้นฉบับที่ได้รับ มาผ่านกระบวนการย่อยด้วยฟังก์ชันย่อยข้อมูล จะได้ข้อมูลที่ย่อยแล้วอันหนึ่ง แล้วนำลายมือชื่อดิจิทัล มาทำการถอดรหัสด้วย กุญแจสาธารณะของผู้ส่ง ก็จะได้ข้อมูลที่ย่อยแล้วอีกอันหนึ่ง แล้วทำการเปรียบเทียบ ข้อมูลที่ย่อยแล้วทั้งสองอัน ถ้าหากว่าเหมือนกัน ก็แสดงว่าข้อมูลที่ได้รับนั้นไม่ได้ถูกแก้ไข แต่ถ้าข้อมูลที่ย่อยแล้วแตกต่างกัน ก็แสดงว่า ข้อมูลที่ได้รับถูกเปลี่ยนแปลงระหว่างทาง จากกระบวนการลงลายมือชื่อดิจิทัลข้างต้นมีข้อพึงสังเกตดังต่อไปนี้

1. ลายมือชื่อดิจิทัลจะแตกต่างกันไปตามข้อมูลต้นฉบับและบุคคลที่จะลงลายมือชื่อ ไม่เหมือนกับลายมือชื่อทั่วไปที่จะต้องเหมือนกันสำหรับบุคคลนั้นๆ ไม่ขึ้นอยู่กับเอกสาร

2. กระบวนการที่ใช้จะมีลักษณะคล้ายคลึงกับการเข้ารหัสแบบอสมมาตร แต่การเข้ารหัสจะใช้ กุญแจส่วนตัวของผู้ส่ง และการถอดรหัสจะใช้กุญแจสาธารณะของผู้ส่ง ซึ่งสลับกันกับการเข้าและถอดรหัสแบบกุญแจอสมมาตรในการรักษาข้อมูลให้เป็นความลับ

ในรูปที่ 2.4 แสดงถึงกระบวนการลงลายมือชื่อดิจิทัล แต่ในการใช้งานจริงข้อมูลต้นฉบับที่ส่งไปก็ควรจะถูกเข้ารหัสด้วยเพื่อทำให้ข้อมูลเป็นความลับสำหรับผู้ที่ไม่มีสิทธิ์



รูปที่ 2.6 แผนภาพกระบวนการลงลายมือชื่อดิจิตอล

ที่มา http://www.ecommerce.or.th/faqs/faq3-1.html#1

2.4 Secure Socket Layer (SSL)

Secure Sockets Layer (SSL) เริ่มพัฒนาโดย Netscape Communications เพื่อใช้ในโพรโตคอลระดับแอพพลิเคชันคือ Hypertext Transfer Protocol (HTTP) ซึ่งเป็นการสื่อสารผ่านเว็บให้ปลอดภัย พัฒนาในช่วงต้นของยุคการค้าอิเล็กทรอนิคส์กำลังได้รับความนิยมในโลกอินเตอร์เน็ต

SSL ทำให้เกิดการสื่อสารอย่างปลอดภัยระหว่างไคลเอ็นต์และเซิร์ฟเวอร์ โดยการอนุญาตให้มีกระบวนการพิสูจน์ตัวตนรวมกับการใช้งานลายเซ็นดิจิตอลสำหรับการรักษาความถูกต้องของข้อมูลและการเข้ารหัสข้อมูลเพื่อป้องกันความเป็นส่วนตัวระหว่างการสื่อสารข้อมูล

โพรโตคอล SSL อนุญาตให้สามารถเลือกวิธีการในการเข้ารหัส วิธีสร้างไดเจสต์ [*1] และลายเซ็นดิจิตอล ได้อย่างอิสระก่อนการสื่อสารจะเริ่มต้นขึ้น ตามความต้องการของทั้งเว็บเซิร์ฟเวอร์และบราวเซอร์ ทั้งนี้เพื่อเพิ่มความยืดหยุ่นในการใช้งาน เปิดโอกาสให้ทดลองใช้วิธีการในการเข้ารหัสวิธีใหม่ รวมถึงลดปัญหาการส่งออกวิธีการเข้ารหัสไปประเทศที่ไม่อนุญาต

Netscape เริ่มพัฒนา SSL เวอร์ชันแรกคือเวอร์ชัน 2.0 และเวอร์ชันถัดมาเป็น 3.0 ซึ่งสนับสนุนความสามารถด้านความปลอดภัยมากขึ้น และเป็นเวอร์ชันสุดท้ายก่อนที่จะเป็นมาตรฐานกลางของโพรโตคอลบนอินเตอร์เน็ต โดยเปลี่ยนชื่อเป็น Transport Layer Security หรือ TLS ซึ่งดูแลมาตรฐานโดย Internet Engineering Task Force (IETF) อธิบายเวอร์ชั่นของ SSL และผู้พัฒนาได้ตามตาราง

ตารางที่ 2.3 ผู้พัฒนาและเวอร์ชั่นของ SSL

เวอร์ชัน	ผู้พัฒนา	จุดเด่น	บราวเซอร์ที่สนับสนุน
SSL v2.0	Netscape Corp. [SSL2]	โพรโตคอล SSL รุ่นแรกที่พัฒนาบนบราวเซอร์	NS Navigator 1.x/2.x MS IE 3.x Lynx/2.8 + OpenSSL
SSL v3.0	Netscape Corp. เป็น Internet Drafted รุ่นก่อนเป็นมาตรฐานกลาง [SSL3]	ปรับปรุงใหม่เพิ่มความปลอดภัยมากขึ้น สนับสนุนการใช้ Non-RSA Ciphers ในการเข้ารหัส และห่วงโซ่ Certificate[*2]	NS Navigator 2.x/3.x/4.x MS IE 3.x/4.x Lynx/2.8 + OpenSSL
TLS v1.0	IETF กำลังเสนอให้เป็นมาตรฐานโพรโตคอลบนอินเตอร์เน็ต (Proposed Internet Standard)	ปรับปรุงจาก SSL v3.0 สนับสนุนการทำงานในชั้น MAC และ HMAC เพิ่ม Padding ชนิด Block และวิธีการจัดลำดับข้อมูล และเพิ่มระดับการแจ้งเตือน	Lynx/2.8 + OpenSSL

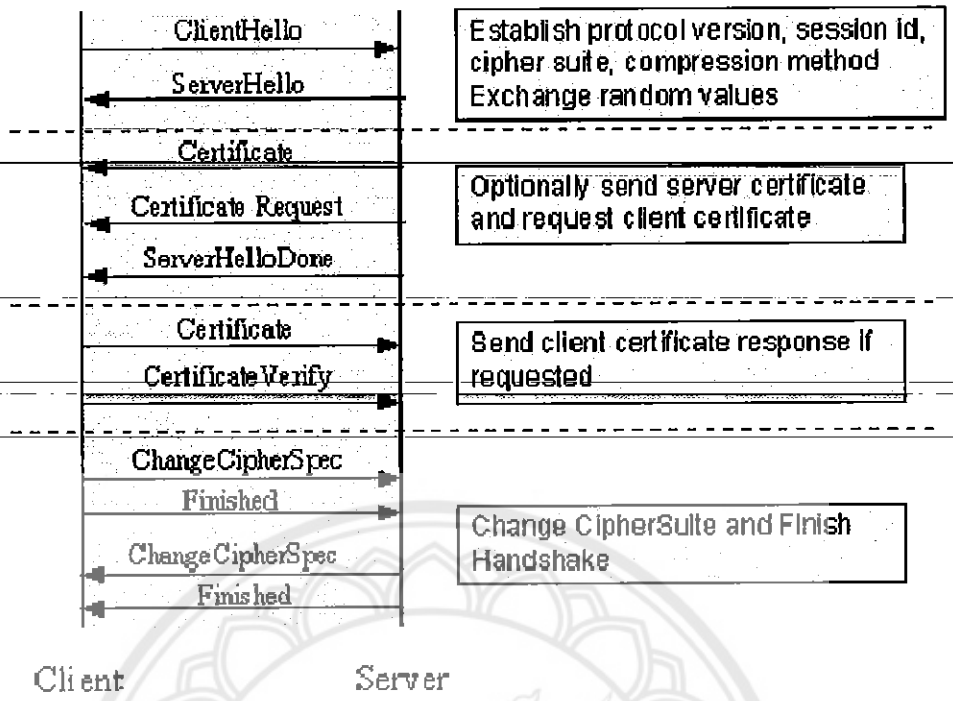
หมายเหตุ

[*1] ไดเจสต์ (Digest) คือข้อความที่เกิดจากการเข้ารหัสข้อมูลด้วยฟังก์ชันแฮชเช่น MD5 หรือ SHA-1

[*2] ห่วงโซ่ Certificate (Certificate Chain) คือการเพิ่มข้อมูล Certificate ที่เกี่ยวเนื่องกันเมื่อใช้ในขั้นตอนแลกเปลี่ยนข้อมูล ซึ่งจะช่วยลดเวลาในการค้นหา Certificate จากผู้ให้บริการ Certificate Authority (CA) ที่เกี่ยวเนื่องกันมากกว่า 1 ขึ้นไป

2.4.1 กระบวนการในการเริ่มต้นการสื่อสารผ่านชั้น SSL แบ่งเป็น 4 ขั้นตอน คือ

1) ประกาศชุดวิธีการเข้ารหัส ไดเจสต์ และลายเซ็นดิจิตอลที่สนับสนุนของทั้งไคลเอ็นต์และเซิร์ฟเวอร์
2) การพิสูจน์ตัวตนของเซิร์ฟเวอร์ต่อไคลเอ็นต์
3) การพิสูจน์ตัวตนของไคลเอ็นต์ต่อเซิร์ฟเวอร์ ถ้าจำเป็น
4) ไคลเอ็นต์และเซิร์ฟเวอร์ตกลงชุดวิธีการเข้ารหัส การสร้างไดเจสต์ และการใช้ลายเซ็นดิจิตอลตามรูป 2.7



รูปที่ 2.7 กระบวนการเริ่มต้นการติดต่อสื่อสารของโพรโตคอล SSL

ที่มา www.thaicert.nectec.or.th/paper/authen/authentication_guide.php

ขั้นตอน 1 ประกาศชุดวิธีการเข้ารหัส ไดเจสต์ และลายเซ็นดิจิตอลที่สนับสนุนของทั้งไคลเอ็นต์และเซิร์ฟเวอร์

ไคลเอ็นต์และเซิร์ฟเวอร์ส่งข้อความเริ่มต้นการสื่อสาร (Hello Message) ซึ่งประกอบไปด้วยเวอร์ชันของโพรโตคอลที่ใช้ วิธีการเข้ารหัสที่เว็บเซิร์ฟเวอร์และไคลเอ็นต์สนับสนุน หมายเลขระบุการสื่อสาร (Session Identifier) รวมถึงวิธีการบีบอัดข้อมูลในการสื่อสารที่สนับสนุน หมายเลขระบุการสื่อสารที่เกิดขึ้น ใช้สำหรับตรวจสอบการเชื่อมต่อระหว่างไคลเอ็นต์และเซิร์ฟเวอร์ ถ้ามีการเชื่อมต่อก่อนหน้านี้เกิดขึ้น แสดงว่าได้มีการตกลงวิธีการสื่อสารแล้ว สามารถเริ่มต้นส่งข้อมูลได้ทันที เป็นการลดเวลาติดต่อสื่อสารลง

ขั้นตอน 2 การพิสูจน์ตัวตนของเซิร์ฟเวอร์ต่อไคลเอ็นต์

ถัดมาเว็บเซิร์ฟเวอร์ทำการส่ง Certificate หรือใบยืนยันความมีตัวตนของเซิร์ฟเวอร์ ไคลเอ็นต์จะทำการตรวจสอบ Certificate กับผู้ให้บริการ Certificate Authority ที่ได้ตั้งค่าไว้ เพื่อยืนยันความถูกต้องของ Certificate ของเซิร์ฟเวอร์

ขั้นตอน 3 การพิสูจน์ตัวตนของไคลเอ็นต์ต่อเซิร์ฟเวอร์

ถ้าจำเป็นเซิร์ฟเวอร์สามารถร้องขอ Certificate จากไคลเอ็นต์เพื่อตรวจสอบความถูกต้องของ Client ด้วยก็ได้ ใช้ในกรณีที่มีการจำกัดการใช้งานเฉพาะไคลเอ็นต์ที่ต้องการเท่านั้น ซึ่ง SSL

สนับสนุนการตรวจสอบได้จากทั้งเซิร์ฟเวอร์และไคลเอ็นต์ ขึ้นอยู่กับการเลือกใช้งานในขณะติดต่อสื่อสารที่เกิดขึ้นนั้น

ขั้นตอน 4 ไคลเอ็นต์และเซิร์ฟเวอร์ตกลงชุดวิธีการเข้ารหัส การสร้างไดเจสต์ และการใช้ลายเซ็นดิจิตอล

ขั้นตอนการตรวจสอบ Certificate ที่เซิร์ฟเวอร์ร้องขอจากไคลเอ็นต์จะมีหรือไม่มีก็ได้ ขึ้นอยู่กับการตั้งค่าบนเซิร์ฟเวอร์ หลังจากขั้นตอนการตรวจสอบเสร็จสิ้น เซิร์ฟเวอร์และไคลเอ็นต์จะตกลงการใช้งานวิธีการเข้ารหัสระหว่างกันโดยใช้ค่าที่ได้จากการประกาศในขั้นตอนแรก

วิธีการแลกเปลี่ยนกุญแจในการเข้ารหัส (Key Exchange Method) คือการกำหนดกลไกการแลกเปลี่ยนกุญแจที่ใช้ในการเข้ารหัสระหว่างการสื่อสาร โดยทั้งไคลเอ็นต์และเซิร์ฟเวอร์จะใช้กุญแจนี้ในการเข้ารหัสและถอดรหัสข้อมูล ใน SSL เวอร์ชัน 2.0 จะสนับสนุนวิธีการแลกเปลี่ยนกุญแจแบบ RSA ส่วน SSL เวอร์ชัน 3.0 ขึ้นไปจะสนับสนุนวิธีการอื่นๆ เพิ่มเติมเช่นการใช้ RSA ร่วมกับการใช้ Certificate หรือ Diffie-Hellman เป็นต้น

วิธีการเข้ารหัสในปัจจุบันแบ่งเป็นสองวิธีคือ การใช้กุญแจเดียวกันในการเข้ารหัสและถอดรหัส อาจเรียกกุญแจนี้ว่า Session Key หรือ Secret Key ส่วนอีกวิธีการคือ การใช้กุญแจคนละตัวในการเข้ารหัสและถอดรหัส ประกอบไปด้วยกุญแจสาธารณะและกุญแจส่วนตัวซึ่งเป็นคู่กันเสมอ เข้ารหัสด้วยกุญแจใด จะต้องถอดรหัสด้วยกุญแจที่คู่กันและตรงกันข้ามเท่านั้น มักใช้วิธีการเข้ารหัสด้วยกุญแจคนละตัวมาใช้ในการเข้ารหัส Session Key และส่งไปให้ฝั่งตรงข้ามก่อนการสื่อสารจะเกิดขึ้นรวมเรียกว่าวิธีการแลกเปลี่ยนกุญแจในการเข้ารหัส

SSL ใช้วิธีการเข้ารหัสด้วยกุญแจสมมาตร หรือกุญแจเดียวในการเข้ารหัสและถอดรหัสตามที่กล่าวข้างต้น วิธีการเข้ารหัสคือ การเข้ารหัสด้วย DES และ 3DES (Data Encryption Standard), วิธีการเข้ารหัสด้วย IDEA ส่วน RC2 และ RC4 เป็นวิธีการเข้ารหัสของ RSA รวมถึงวิธีการเข้ารหัสแบบ Fortezza สำหรับความยาวของการเข้ารหัสที่ใช้คือ 40 บิต, 96 บิต และ 128 บิต

การสร้าง Message Authentication Code (MAC) เพื่อใช้สำหรับการยืนยันความถูกต้องของข้อมูลระหว่างการสื่อสารและป้องกันการปลอมข้อมูล ส่วนฟังก์ชันสร้างไดเจสต์ที่ SSL สนับสนุนและเลือกใช้ได้ในปัจจุบันคือ MD5 ขนาด 128 บิต และ SHA-1 (Secure Hash Algorithm) ขนาด 160 บิต

ซึ่งจะได้วิธีการที่ทั้งสองฝ่ายสนับสนุนและเหมาะสมซึ่งเป็นขั้นตอนสุดท้ายก่อนการสื่อสารที่มีการเข้ารหัสจะเริ่มต้นขึ้น

2.5 The Elliptic Curve Digital Signature Algorithm (ECDSA)

The Elliptic Curve Digital Signature Algorithm (ECDSA) ได้รับการเสนอครั้งแรกเมื่อปี 1992 โดย Scott Vanstone และเป็นอัลกอริทึมที่ได้รับการยอมรับในหลายองค์กร เช่น

- ในปี ค.ศ. 1998 ได้รับยอมรับจาก ISO (International Standard Organization) เรียกว่ามาตรฐาน ISO148888-3
- ในปี ค.ศ. 1999 ได้รับการยอมรับจาก ANSI (American National Standard Institute) เรียกว่ามาตรฐาน ANSI X9.62
- ในปี ค.ศ. 2000 ได้รับการยอมรับจาก IEEE (Institute of Electrical and Electronics Engineers) เรียกว่ามาตรฐาน IEEE 1363-2000
- ในปี ค.ศ. 2000 ได้รับการยอมรับจาก FIPS standard เรียกว่ามาตรฐาน FIPS 186-2

2.5.1 The Digital Signature Algorithm

DSA ได้รับการนำเสนอในเดือนสิงหาคม ปี ค.ศ. 1991 โดย The U.S. National Institute of Standards and Technology (NIST) และได้รับการยอมรับใน U.S. Government Federal Information Processing Standards (FIPS 186) เรียกว่า The Digital Signature Standard (DSS) DSA สามารถมองสิ่งที่เปลี่ยนแปลงของ Elgamal Signature Scheme ความปลอดภัยของมันขึ้นอยู่กับความยากของการคำนวณของ Discrete Algorithm ใน Prime-order subgroup ของ Z_p^*

- การสร้างโดเมนพารามิเตอร์ (Domain parameters) ของ DSA ซึ่งกำหนดสำหรับแต่ละ Entity ใน Particular Security Domain
  1. เลือก 160-bit Prime q 1 ตัว และ 1024-bit Prime p 1 ตัว ด้วยคุณสมบัติที่ q | p -1
  2. (เลือก Generator g 1 ตัว ของ Unique Cyclic Group ของ Order q ใน Z_p^*) เลือก h หนึ่งตัวของ $h \in Z_p^*$ แล้วคำนวณหา $g = h^{(p-1)/q} \bmod p$ (ทำซ้ำจนกระทั่ง $g \neq 1$)
  3. โดเมนพารามิเตอร์ (Domain parameters) คือ p, q และ g
- การสร้าง Key Pair ของ DSA แต่ละ Entity A ในโดเมนซึ่งโดเมนพารามิเตอร์ (Domain parameters) คือ (p, q, g) เป็นดังนี้
  1. สุ่มหรือสุ่มเทียม (Pseudorandom) จำนวนเต็ม k ซึ่ง $1 \leq k \leq q-1$
  2. คำนวณหา $y = g^x \bmod p$
  3. Public ของ A คือ y : Private Key ของ A คือ x
- การสร้าง DSA Signature เพื่อที่จะ Sign ข้อความ m, A จะเป็นดังนี้
  1. สุ่มหรือสุ่มเทียม (Pseudorandom) จำนวนเต็ม k ซึ่ง $1 \leq k \leq q-1$
  2. คำนวณหา $X = g^k \bmod p$ และ $r = X \bmod q$ ถ้า $r = 0$ ให้กลับไปทำข้อ 1.
  3. คำนวณหา $k^{-1} \bmod q$
  4. คำนวณหา $e = SHA\text{-}1(m)$

5. คำนวณหา s = k-1{e + xr} mod q ถ้า s = 0 ให้กลับไปทำข้อ 1.

- **การตรวจสอบยืนยัน DSA Signature** เพื่อที่จะยืนยันความถูกต้องของลายเซ็นของ A (r, s) สำหรับ m, B ได้รับสำเนาของโดเมนพารามิเตอร์ (p, q, g) ของ A และ Public Key y จะทำตามขั้นตอนดังนี้

1. พิสูจน์ยืนยันความถูกต้อง ว่า r และ s คือ เลขจำนวนจริงในช่วง [1, q - 1]
2. คำนวณหา e = SHA-1(m)
3. คำนวณหา w = s-1 mod q
4. คำนวณหา u1 = ew mod q และ u2 = rw mod q
5. คำนวณหา $X = g^{u_1} y^{u_2} \bmod p$ และ v = X mod q
6. ยอมรับลายเซ็นก็ต่อเมื่อ v = r

2.5.2 ECDSA Key Pairs

ECDSA Key Pair เกี่ยวข้องโดยเป็นส่วนหนึ่งของ EC Domain Parameter โดย Public Key เป็นการสุ่มหลายค่าของค่าพื้นฐาน ส่วน Private Key เป็นการรวม

- **การสร้าง Key Pair**

Key pair ของ A เกี่ยวข้องเป็นส่วนหนึ่งกับของ EC Domain Parameter นั่นคือ D = (q, FR, a, b, g, n, h) การสร้าง Key Pair มีขั้นตอนดังนี้

1) สุ่มค่าจำนวนเต็ม d ในช่วง [1, n-1]
2) คำนวณ Q = dG
3) Public Key ของ A คือ Q ส่วน Private Key ของ A คือ d

- **การยืนยัน Public Key**

การรับประกันว่า Public Key Q นั้นถูกต้อง มีวิธีการดังนี้

1) A ปฏิบัติตาม Algorithm 6 (แสดงดังหัวข้อข้างล่าง)
2) A สร้าง Q ขึ้นมาโดยใช้ระบบที่น่าเชื่อถือ
3) A รับการรับรองจากองค์กร T ที่น่าเชื่อถือ เช่น CA โดยให้ T รับรองว่า A ปฏิบัติตาม Algorithm 6
4) A รับการรับรองจากองค์กร T ว่า Q ถูกสร้างขึ้นโดยระบบที่น่าเชื่อถือ

Algorithm 1

Input: A มี Public Key $Q = (x_Q, y_Q)$ เกี่ยวเนื่องกับ Domain Parameter (q, FR, a, b, g, n, h)

Output: การยอมรับหรือปฏิเสธความถูกต้องของ Q

1) เช็คว่า Q ไม่เท่ากับศูนย์

2) เช็คว่า x_Q และ y_Q แสดงองค์ประกอบอย่างเหมาะสมของ F_q ยกตัวอย่างจำนวนเต็มในช่วง [0, p-1] ในกรณีที่ q = p และบิตสตริงของความยาว m บิตในกรณี $q = 2^m$

3) เช็คว่า Q เป็นเท็จบน elliptic curve นิยามโดย a กับ b

4) เช็คว่า nQ = 0

5) ถ้าทุกๆ การเช็คไม่ถูกต้อง แสดงว่า Q ไม่ถูกต้อง มิเช่นนั้นถือว่าถูกต้อง

- **พิสูจน์ความเป็นเจ้าของ Private Key**

ถ้า C สามารถรับรอง Public Key Q ของ A ว่าเป็นของตน C สามารถอ้างได้ว่าข้อความที่ถูกเซ็นต์โดย A นั้นมาจาก C เพื่อป้องกันการเกิดเช่นนี้ CA ควรจะมีสิ่งที่เกี่ยวข้องกับ A ทั้งหมดเพื่อพิสูจน์ความเป็นเจ้าของ Private Key ว่าสอดคล้องกับ Public Key ก่อนที่ CA จะรับรอง Public Key ให้กับ A

2.5.3 การสร้างและการพิสูจน์ลายเซ็นโดยใช้ ECDSA

เป็นการอธิบายการสร้างและการตรวจสอบลายเซ็นต์โดยใช้ ECDSA

- **การสร้างลายเซ็นต์โดยใช้ ECDSA:** ในการเซ็นต์ข้อความ m โดย A กับ Domain Parameter (q, FR, a, b, g, n, h) และเชื่อมโยงกับ Key Pair (d, Q)

1) สุ่มจำนวนเต็ม k จากช่วง $1 \leq k \leq n-1$

2) คำนวณ $kG = (x_1, y_1)$ และแปลงค่า x_1 เป็นจำนวนเต็ม $\overline{x_1}$

3) คำนวณ $r = x_1 \bmod n$ ถ้า r = 0 กลับไปข้อ 1

4) คำนวณ $k^{-1} \bmod n$

5) คำนวณ SHA-1(m) แล้วแปลงค่าจากบิตสตริงไปเป็นจำนวนเต็ม e

6) คำนวณ $s = k^{-1}(e + dr) \bmod n$ ถ้า s = 0 กลับไปข้อ 1

7) ลายเซ็นต์ของ A สำหรับข้อความ m คือ (r,s)

- **การตรวจสอบลายเซ็นต์โดยใช้ ECDSA:** เพื่อตรวจสอบลายเซ็นต์ A (r,s) สำหรับข้อความ m โดย A กับ Domain Parameter (q, FR, a, b, g, n, h)

1) ตรวจสอบว่า r กับ s เป็นจำนวนเต็มในช่วง [1, n-1]

2) คำนวณ SHA-1(m) แล้วแปลงค่าจากบิตสตริงไปเป็นจำนวนเต็ม e

3) คำนวณ $w = s^{-1} \bmod n$

4) คำนวณ $u_1 = ew \bmod n$ และ $u_2 = rw \bmod n$

5) คำนวณ $X = u_1G + u_2Q$

6) ถ้า X = 0 ให้ปฏิเสธลายเซ็นต์ ถ้าไม่ให้เปลี่ยน x-coordinate x_1 ของ X เป็นจำนวนเต็ม $\overline{x_1}$ และคำนวณ $v = \overline{x_1} \bmod n$

7) ยอมรับลายเซ็นต์ถ้า v = r เท่านั้น

2.6 Apache Tomcat

Apache Tomcat คือ Web Server อย่างหนึ่ง ที่รองรับการทำงานกับภาษา JSP และสามารถใช้งานเป็น HTTP Server ธรรมดาได้ ซึ่งดูจากคุณลักษณะ Tomcat ก็อาจเปรียบได้เป็น IIS บน PC

2.6.1 SSL in Tomcat

การเซตค่าให้ Tomcat นั้นเป็นสิ่งสำคัญที่ทำให้ Tomcat ได้รับความปลอดภัยจาก SSL เมื่อรัน Tomcat เป็น Web Server ซึ่งจำเป็นที่ Web Server ต้องเชื่อมต่อแบบ SSL กับผู้ใช้ โดยจะทำให้ Tomcat สามารถติดต่อฟังก์ชันทั้งหมดของ SSL ที่เกี่ยวข้อง ต่อมา Tomcat คืน cleartext ที่ถูกเข้ารหัสก่อนที่จะคืนสู่เบราเซอร์ของผู้ใช้ ในขั้นตอนนี้ Tomcat จะรู้ว่าการเชื่อมต่อระหว่าง Web Server และ Client เป็นการเชื่อมต่อที่มีความปลอดภัยเพียงแต่ Tomcat ไม่มีส่วนในการเข้าหรือถอดรหัสด้วยตนเอง

2.6.2 Certificate in Tomcat

การนำ SSL มาใช้นั้น จำเป็นที่ Web Server ต้องมี Certificate โดยเฉพาะก่อนได้รับข้อมูลสำคัญ เปรียบดัง Certificate เป็น "ใบอนุญาตแบบดิจิตอล" สำหรับอินเตอร์เน็ต ซึ่ง Certificate ก็คือการเข้ารหัสและลงชื่อโดยผู้เป็นเจ้าของ และเป็นไปได้ยากที่จะปลอมแปลงให้เหมือนต้นฉบับ เหมาะสำหรับธุรกิจพาณิชย์ที่เล็งเห็นว่าการแลกเปลี่ยนติดต่อนั้นต้องการความปลอดภัยสูง

Certificate ได้รับการรับรองจาก Certificate Authority (CA) อย่างเช่น VeriSign หรือ Thawte ซึ่งสามารถถูกตรวจสอบได้ทางอิเล็คทรอนิกส์ ดังนั้นจึงสามารถเชื่อได้ว่า Certificate นั้นถูกต้อง ถ้าได้รับการรับรองจาก CA แล้ว

อย่างไรก็ตาม ความกังวลที่แท้จริงไม่ได้มาจากความการรับประกัน ผู้ส่งอาจจะแค่ต้องการทราบว่าข้อมูลที่ถูกส่งออกไป หรือที่ได้รับจาก Server นั้นเป็นความรับและไม่ได้ถูกลักลอบอ่านโดยผู้ที่ลอบขโมยข้อมูลขณะที่มีการเชื่อมต่ออยู่ โชคดีที่ JAVA มี Command-line ที่เรียกว่า Keytool ซึ่งสามารถสร้าง Certificate กับลายเซ็นต์ด้วยตัวเอง โดยผู้ใช้สามารถสร้าง Certificate ที่ไม่ต้องผ่านทาง CA โดยตรง

2.7 ระบบฐานข้อมูล

ข้อมูลเป็นสิ่งที่มีความสำคัญเป็นอย่างมาก จึงต้องเก็บข้อมูลไว้เป็นอย่างดีเพื่อให้สามารถนำมาใช้ได้เมื่อต้องการ โดยในอดีตข้อมูลจะถูกเก็บอยู่ในไฟล์ แต่ไฟล์ก็มีข้อจำกัดเนื่องจากไม่สะดวกต่อการเปลี่ยนแปลงโครงสร้างข้อมูลภายหลัง และมีโอกาสทำให้เก็บข้อมูลซ้ำซ้อนกันทำให้เกิดปัญหาในการอัพเดตข้อมูลตามมา และการเก็บข้อมูลในไฟล์ยังเสี่ยงต่อความปลอดภัยที่ใครๆ อาจจะแอบมาคัดลอกข้อมูลไปใช้ได้ง่ายๆ

ฐานข้อมูลจึงถูกพัฒนาขึ้นเพื่อแก้ปัญหาต่างๆ ที่เป็นข้อจำกัดของไฟล์ โดยสามารถแก้ปัญหาการขึ้นกับข้อมูลที่ต้องแก้ไขโปรแกรมทุกครั้งที่โครงสร้างข้อมูลเปลี่ยนไปได้ ป้องกันโอกาสที่จะเก็บข้อมูลซ้ำซ้อนกันเนื่องจากต้องออกแบบฐานข้อมูลก่อน ไม่มีข้อมูลที่มีค่าขัดแย้งกัน เพราะข้อมูลหนึ่งๆ จะถูกเก็บอยู่ที่เดียว สามารถใช้ข้อมูลร่วมกันได้โดยอาจจะใช้งานพร้อมๆ กันได้ด้วย มีการควบคุมสิทธิ์ในการใช้ข้อมูล ผู้ไม่เกี่ยวข้องจึงไม่สามารถเห็นข้อมูลที่เราต้องการเก็บเป็นความลับได้ ง่ายต่อการนำข้อมูลมาใช้ และมีกลไกกู้ข้อมูลในกรณีที่ฐานข้อมูลมีปัญหาได้

2.7.1 ความรู้พื้นฐานเกี่ยวกับระบบฐานข้อมูล

ระบบฐานข้อมูล (Database System) หมายถึงโครงสร้างสารสนเทศที่ประกอบด้วยรายละเอียดของข้อมูลที่เกี่ยวข้องกันที่จะนำมาใช้ในระบบต่างๆ ร่วมกัน ประกอบด้วย 4 ส่วนหลัก คือ

- **ฐานข้อมูล (Database)** ซึ่งเป็นส่วนหนึ่งของระบบฐานข้อมูล มีหน้าที่สำหรับเก็บข้อมูลรวมทั้งความสำพันธ์ของข้อมูลนั้นๆ โดยในระบบฐานข้อมูลหนึ่งๆ อาจจะมีฐานข้อมูลได้หลายตัวก็ได้เพื่อประโยชน์การใช้งานที่แตกต่างกันออกไป
- **ซอฟต์แวร์จัดการระบบฐานข้อมูล (DBMS)** ฐานข้อมูลเป็นที่สำหรับจัดเก็บข้อมูลต่างๆ เท่านั้น การนำข้อมูลเข้าและออกจากฐานข้อมูลจึงเป็นหน้าที่ของซอฟต์แวร์จัดการระบบฐานข้อมูลหรือ Database Management System

โดย DBMS จะทำหน้าที่เป็นตัวกลางระหว่างฐานข้อมูลกับโปรแกรมที่มาใช้งานฐานข้อมูลและผู้ใช้งานในการติดต่อไปยังฐานข้อมูลเพื่อทำงานที่ผู้ใช้งานสั่งมาให้สำเร็จ ไม่ว่าจะเป็นการเพิ่มข้อมูลลงไปในฐานข้อมูล การค้นหาข้อมูลที่ต้องการ หรือการลบข้อมูลที่ไม่ต้องการแล้วออกจากฐานข้อมูล เช่น Access, FoxPro, Clipper, dBase, FoxBASE, Oracle, SQL เป็นต้น โดยแต่ละโปรแกรมจะมีความสามารถต่างกัน บางโปรแกรมใช้ง่ายแต่จะจำกัดขอบเขตการใช้งาน บางโปรแกรมใช้งานยากกว่า แต่จะมีความสามารถในการทำงานมากกว่า

- **โปรแกรมใช้งานฐานข้อมูล (Application Programs)** เป็นโปรแกรมหรือแอพพลิเคชันที่พัฒนาขึ้นมาเพื่อใช้ประโยชน์จากข้อมูลที่เก็บไว้ในฐานข้อมูล โดยอาจจะเป็นโปรแกรมที่ทำงานบนเครื่องคอมพิวเตอร์หรือทำงานบนเว็บผ่านอินเตอร์เน็ตก็ได้
- **ผู้ใช้งาน (Users)** คือทุกๆ คนที่เกี่ยวข้องกับฐานข้อมูลไม่ว่าจะเป็นผู้พัฒนาโปรแกรมขึ้นมาใช้งานฐานข้อมูล (Application Programmer), ผู้ออกแบบฐานข้อมูล (Database Designer), ผู้ดูแลระบบฐานข้อมูล (DBA) หรือผู้ใช้งานทั่วๆ ไป (End User)

2.7.2 การนำ ER Diagram มาใช้

ER Diagram เป็นไดอะแกรมที่ใช้แสดงความสัมพันธ์ของตารางข้อมูล ซึ่งมีประโยชน์ในการออกแบบฐานข้อมูลจะทำให้เราเห็นความสัมพันธ์ระหว่างข้อมูลที่ต้องจัดเก็บและมองเห็น

รายละเอียดข้อมูลอย่างชัดเจนจึงทำให้วิเคราะห์ความสัมพันธ์ของข้อมูลได้อย่างถูกต้องและไม่ลืมที่จะเก็บข้อมูลสำคัญบางตัวไป โดย ER Diagram จะประกอบด้วย

- **เอนทิตี้ (Entity)** หมายถึง ตัวข้อมูลที่เราสนใจจะเก็บลงฐานข้อมูล เป็นข้อมูลที่มีส่วนสำคัญให้ระบบงานดำเนินต่อไปได้ ในระบบงานหนึ่งๆ จะมี Entity อยู่หลายชนิดโดยแต่ละชนิดก็จะมี Entity ที่แตกต่างกันในรายละเอียดอยู่หลายตัว ตัวอย่างเช่น ในระบบงานทะเบียนนักศึกษา "นักศึกษา" ถือเป็น Entity ชนิดหนึ่งซึ่งมีอยู่ด้วยกันหลายตัวตามจำนวนนักศึกษาทั้งหมด โดยเราสามารถแยกนักศึกษาแต่ละคนออกจากกันได้ด้วยรหัสประจำตัวนักศึกษาซึ่งเป็นสิ่งที่ทำให้นักศึกษาแต่ละคนแตกต่างกันนั่นเอง

เอนทิตี้ชนิดอ่อนแอ (Weak Entity) เป็น Entity ที่ไม่มีความหมาย หากขาดเอนทิตี้อื่นในฐานข้อมูล

ความสัมพันธ์ระหว่าง Entity แบ่งออกเป็น 3 ประเภท คือ

- ความสัมพันธ์แบบหนึ่งต่อหนึ่ง (One-to-one Relationships) เป็นการแสดงความสัมพันธ์ของข้อมูลในเอนทิตี้หนึ่งที่มีความสัมพันธ์กับข้อมูลในอีกเอนทิตี้หนึ่ง ในลักษณะหนึ่งต่อหนึ่ง (1: 1)

- ความสัมพันธ์แบบหนึ่งต่อกลุ่ม (One-to-many Relationships) เป็นการแสดงความสัมพันธ์ของข้อมูลในเอนทิตี้หนึ่ง ที่มีความสัมพันธ์กับข้อมูลหลาย ๆ ข้อมูลในอีกเอนทิตี้หนึ่ง ในลักษณะ (1: M) ตัวอย่างเช่น

- ความสัมพันธ์แบบกลุ่มต่อกลุ่ม (Many-to-many Relationships) เป็นการแสดงความสัมพันธ์ของข้อมูลสองเอนทิตี้ในลักษณะกลุ่มต่อกลุ่ม (M:M) เช่น เอนทิตี้ใบสั่งซื้อแต่ละใบจะสามารถสั่งสินค้าได้มากกว่าหนึ่งชนิด ความสัมพันธ์ของข้อมูลจากเอนทิตี้ใบสั่งซื้อไปยังเอนทิตี้สินค้า จึงเป็นแบบหนึ่งต่อกลุ่ม (1:M) ในขณะที่สินค้าแต่ละชนิด จะถูกสั่งอยู่ในใบสั่งซื้อหลายใบ ความสัมพันธ์ของข้อมูลจากเอนทิตี้สินค้าไปยังอินทิตี้ใบสั่งซื้อ จึงเป็นแบบหนึ่งต่อกลุ่ม (1:M) เช่นกัน ดังนั้นความสัมพันธ์ของเอนทิตี้ทั้งสอง จึงเป็นแบบกลุ่มต่อกลุ่ม (M:M)

- **แอททริบิวต์ (Attribute)** หมายถึง คุณสมบัติเฉพาะของ Entity แต่ละตัวโดยถึงแม้จะเป็น Entity ชนิดเดียวกันก็อาจจะมีค่าของ Attribute เหมือนหรือต่างกันก็ได้ และ Entity สามารถมี Attribute ได้มากมายขึ้นอยู่กับความจำเป็นที่เราต้องจัดเก็บลงฐานข้อมูลเพื่อนำมาใช้ในระบบงาน ตัวอย่างเช่น ใน Entity นักศึกษา อาจจะมี Attribute ได้แก่

  - Attribute รหัสนักศึกษา
  - Attribute ชื่อนักศึกษา-นามสกุล
  - Attribute วันเดือนปีเกิด
  - Attribute คณะ
  - Attribute ที่อยู่นักศึกษา เป็นต้น

- **ความสัมพันธ์ (Relationships)** ในระบบงานหนึ่งๆ จะต้องมี Entity อย่างน้อย 2 ชนิด โดย Entity ทั้งหมดก็จะเกี่ยวข้องกันไม่ทางใดก็ทางหนึ่ง ความเกี่ยวข้องกันระหว่าง Entity นี้เรียกว่า Relationship

Relationship ในระบบงานจะมีอะไรบ้างนั้นขึ้นอยู่กับความเกี่ยวข้องหรือความสัมพันธ์ระหว่าง Entity ซึ่ง Entity แต่ละคู่ก็อาจจะมีความสัมพันธ์มากกว่า 1 ก็ได้ และถึงแม้ว่าจะเป็นระบบงานเดียวกันแต่ถ้าอยู่คนละที่ก็อาจจะมี Relationship ไม่เหมือนกันขึ้นอยู่กับลักษณะงานและความเป็นจริงที่เกิดขึ้นในการทำงานนั้นๆ ที่อาจมีข้อจำกัดหรือความต้องการไม่เหมือนกันนั่นเอง

2.7.3 แบบจำลองข้อมูล (Data Model) มีไว้เพื่อนำเสนอข้อมูลและความสัมพันธ์ระหว่างข้อมูลในรูปแบบที่เข้าใจได้ง่าย มีดังนี้คือ

- **ฐานข้อมูลแบบลำดับชั้น (Hierarchical Model)** ฐานข้อมูลแบบลำดับชั้น เป็นโครงสร้างที่จัดเก็บข้อมูลในลักษณะความสัมพันธ์แบบพ่อ-ลูก (Parent-Child Relationship Type : PCR Type) หรือเป็นโครงสร้างรูปแบบต้นไม้ (Tree) กล่าวคือ Record ที่อยู่ด้านบนของโครงสร้างหรือพ่อนั้นสามารถมีลูกได้มากกว่าหนึ่งคน แต่ลูกจะไม่สามารถมีพ่อมากกว่าหนึ่งคนได้ ด้วยความสัมพันธ์ของข้อมูลแบบนี้ได้ช่วยลดการเก็บข้อมูลซ้ำซ้อนกันลงไปได้มากแต่การจะใช้ข้อมูลได้นั้นผู้ใช้ยังคงต้องรู้โครงสร้างการเก็บข้อมูลในไฟล์อยู่ดี ซึ่งต้องรู้ว่า Tree นั้นๆ มีข้อมูลอะไรอยู่บ้าง และต้องรู้ด้วยว่าแต่ละลำดับชั้นนั้นเก็บข้อมูลอะไรอยู่ ซึ่งยุ่งยากมากในการค้นหาข้อมูล

- **ฐานข้อมูลแบบเครือข่าย (Network Model)** จะเป็นการนำทฤษฎีเซต (Set) ทางคณิตศาสตร์มาใช้ นั่นคือสมาชิกของเซตหนึ่งๆ สามารถเป็นสมาชิกของเซตอื่นๆ ได้ด้วย ซึ่งทำขึ้นเพื่อรับรองความสัมพันธ์แบบ Many-to-many

แต่ถึงแม้ว่าการเก็บข้อมูลด้วย Network Model จะช่วยแก้ปัญหาซับซ้อนของข้อมูลให้หมดไปได้ก็ตาม แต่ความสัมพันธ์ของข้อมูลที่โยงกันไปมาก็ทำให้ยากต่อการใช้งาน และผู้ใช้ยังคงต้องเข้าใจโครงสร้างข้อมูลเพื่อให้สามารถนำข้อมูลมาใช้ได้เหมือนเดิม ปัญหาจึงยังไม่หมดไป

Network Model จึงเหมาะกับ Programmer ที่คุ้นเคยเป็นอย่างดีกับโครงสร้างข้อมูลแบบต่างๆ ทั้งแบบง่ายๆ และแบบที่ซับซ้อนอย่างที่ใช้ใน Network Model จึงไม่เหมาะกับผู้ใช้งานทั่วๆ ไปซึ่งต้องการแบบจำลองข้อมูลที่สามารถทำความเข้าใจและใช้งานได้ง่าย

- **ฐานข้อมูลเชิงสัมพันธ์ (Relational Model)** เป็นแบบจำลองข้อมูลที่เราใช้กันอยู่ในปัจจุบัน โครงสร้างข้อมูลที่ใช้แสดงความสัมพันธ์ของข้อมูลนั้นเป็นตารางซึ่งเก็บข้อมูลซึ่งมีลักษณะเหมือนกันไว้ เช่น ข้อมูลของนักเรียน ข้อมูลของอาจารย์ โดยแต่ละตารางจะมีความสัมพันธ์กันผ่านข้อมูลในคอลัมน์ที่มีค่าเหมือนกันไว้ เช่น ข้อมูลของนักศึกษา ข้อมูลของอาจารย์ โดยแต่ละตารางจะมีความสัมพันธ์กันผ่านข้อมูลในคอลัมน์ที่มีค่าเหมือนกัน ดังตัวอย่าง

ตารางที่ 2.4 ตัวอย่างตารางฐานข้อมูลเชิงสัมพันธ์

รหัสนักศึกษา	ชื่อนักศึกษา	วันเดือนปีเกิด	คณะ	ที่อยู่นักศึกษา
46360000	นายสมชาย	5/05/2527	แพทย์	กรุงเทพ
46360001	นายสมหญิง	23/01/2528	พยาบาล	เชียงใหม่
46360002	นายเอกวิทย์	9/09/2528	วิศวกรรมศาสตร์	สกลนคร

ตารางข้อมูลที่เห็นและเข้าใจว่าใช้เก็บข้อมูลอยู่นั้นไม่จำเป็นต้องรู้ว่าตารางเก็บข้อมูลอย่างไรและเก็บไว้ที่ไหน ก็สามารถนำข้อมูลออกมาใช้ได้ โดยตารางจะมีชื่อเรียกเพื่อให้อ้างถึงเวลาต้องการข้อมูลในตารางนั้น และเมื่อเราต้องการข้อมูลในตารางนั้นเราก็จะใช้วิธีเปรียบเทียบค่าของข้อมูลแทน โดยแค่บอกกับ DBMS ว่าต้องการข้อมูลจากตารางนักศึกษา ที่มีค่าในคอลัมน์คณะ เป็น "แพทย์" เท่านั้นก็จะได้ข้อมูลที่ต้องการทันที

2.7.4 ความสำคัญของการประมวลผลแบบระบบฐานข้อมูล

จากการจัดเก็บข้อมูลรวมเป็นฐานข้อมูลจะก่อให้เกิดประโยชน์ดังนี้

- สามารถลดความซ้ำซ้อนของข้อมูลได้ การเก็บข้อมูลชนิดเดียวกันไว้หลายๆ ที่ ทำให้เกิดความซ้ำซ้อน (Redundancy) ดังนั้นการนำข้อมูลมารวมเก็บไว้ในฐานข้อมูล จะช่วยลดปัญหาการเกิดความซ้ำซ้อนของข้อมูลได้ โดยระบบจัดการฐานข้อมูล (Database Management System: DBMS) จะช่วยควบคุมความซ้ำซ้อนได้ เนื่องจากระบบจัดการฐานข้อมูลจะทราบได้ตลอดเวลาว่ามีข้อมูลซ้ำซ้อนกันอยู่ที่ใดบ้าง
- หลีกเลี่ยงความขัดแย้งของข้อมูลได้ หากมีการเก็บข้อมูลชนิดเดียวกันไว้หลาย ๆ ที่และมีการปรับปรุงข้อมูลเดียวกันนี้ แต่ปรับปรุงไม่ครบทุกที่ที่มีข้อมูลเก็บอยู่ก็จะทำให้เกิดปัญหาข้อมูลชนิดเดียวกัน อาจมีค่าไม่เหมือนกันในแต่ละที่ที่เก็บข้อมูลอยู่ จึงก่อให้เกิดความขัดแย้งของข้อมูลขึ้น (Inconsistency)
- สามารถใช้ข้อมูลร่วมกันได้ ฐานข้อมูลจะเป็นการจัดเก็บข้อมูลรวมไว้ด้วยกัน ดังนั้นหากผู้ใช้ต้องการใช้ข้อมูลในฐานข้อมูลที่มาจากแฟ้มข้อมูลต่างๆ ก็จะทำได้โดยง่าย
- **สามารถรักษาความถูกต้องเชื่อถือได้ของข้อมูล** การจัดเก็บข้อมูลในฐานข้อมูล อาจมีข้อผิดพลาดที่เกิดขึ้น เช่น จากการที่ผู้ป้อนข้อมูลป้อนข้อมูลผิดพลาด คือป้อนจากตัวเลขหนึ่งไปเป็นอีกตัวเลขหนึ่ง โดยเฉพาะกรณีมีผู้ใช้หลายคนต้องใช้ข้อมูลจากฐานข้อมูลร่วมกัน หากผู้ใช้คนใดคนหนึ่งแก้ไขข้อมูลผิดพลาดก็ทำให้ผู้อื่นได้รับผลกระทบตามไปด้วย ในระบบจัดการฐานข้อมูล (DBMS) จะสามารถใส่กฎเกณฑ์เพื่อควบคุมความผิดพลาดที่เกิดขึ้น
- **สามารถกำหนดความเป็นมาตรฐานเดียวกันของข้อมูลได้** การเก็บข้อมูลร่วมกันไว้ในฐานข้อมูลจะทำให้สามารถกำหนดมาตรฐานของข้อมูลได้รวมทั้งมาตรฐานต่าง ๆ ในการจัดเก็บ

ข้อมูลให้เป็นไปในลักษณะเดียวกันได้ เช่นการกำหนดรูปแบบการเขียนวันที่ ในลักษณะ วัน/เดือน/ปี หรือ ปี/เดือน/วัน ทั้งนี้จะมีผู้ที่คอยบริหารฐานข้อมูลที่เรียกว่า ผู้บริหารฐานข้อมูล (Database Administrator: DBA) เป็นผู้กำหนดมาตรฐานต่างๆ

- **สามารถกำหนดระบบความปลอดภัยของข้อมูลได้** ระบบความปลอดภัยในที่นี้เป็นการป้องกันไม่ให้ผู้ใช้ที่ไม่มีสิทธิมาใช้ หรือมาเห็นข้อมูลบางอย่างในระบบ ผู้บริหารฐานข้อมูลจะสามารถกำหนดระดับการเรียกใช้ข้อมูลของผู้ใช้แต่ละคนได้ตามความเหมาะสม
- **เกิดความเป็นอิสระของข้อมูล** ในระบบฐานข้อมูลจะมีตัวจัดการฐานข้อมูลที่ทำหน้าที่เป็นตัวเชื่อมโยงกับฐานข้อมูล โปรแกรมต่าง ๆ อาจไม่จำเป็นต้องมีโครงสร้างข้อมูลทุกครั้ง ดังนั้นการแก้ไขข้อมูลบางครั้ง จึงอาจกระทำเฉพาะกับโปรแกรมที่เรียกใช้ข้อมูลที่เปลี่ยนแปลงเท่านั้น ส่วนโปรแกรมที่ไม่ได้เรียกใช้ข้อมูลดังกล่าว ก็จะเป็นอิสระจากการเปลี่ยนแปลง

2.7.5 การเชื่อมโยงกับฐานข้อมูล

ฐานข้อมูลมีความสำคัญในการสร้างเว็บแอปพลิเคชั่นในปัจจุบันเป็นอย่างมาก เพราะข้อมูลต่างๆ ที่อยู่ในเว็บแอปพลิเคชั่นส่วนใหญ่จะถูกเก็บไว้ในฐานข้อมูล การใช้งานฐานข้อมูลกับภาษาสคริปต์ JSP สามารถใช้งานผ่านทาง JDBC (JAVA Database Connectivity) เพื่อช่วยในการติดต่อกับฐานข้อมูล

- ความหมายของ JDBC

JDBC (JAVA Database Connectivity) เปรียบเสมือนตัวกลางที่ใช้ในการเชื่อมโยงระหว่างภาษา JAVA กับฐานข้อมูล การใช้งาน JDBC API (JDBC Application Programming Interface) ช่วยให้เข้าถึงฐานข้อมูลไม่ว่าจะเป็น การเพิ่มข้อมูล การแสดงข้อมูล หรือการปรับปรุงแก้ไขข้อมูล การใช้งาน JDBC จะถูกนำไปใช้ร่วมกับ Structured Query Language (SQL) ซึ่งเป็นภาษาที่ใช้ในการจัดการกับฐานข้อมูล จากการที่ JDBC ใช้มาตรฐานของภาษา SQL นี้เอง ทำให้ JDBC สามารถสนับสนุนการใช้งาน Database ได้เป็นจำนวนมาก เช่น Access, SQL Server 2000 หรือ MySQL เป็นต้น

- **ขั้นตอนการใช้งาน JDBC**

1) **ติดตั้ง Driver JDBC (MySQL)** www.MySQL.com/products/connector/j/ (Download ที่เว็บดังกล่าว) เสร็จแล้ว Unzip แล้วนำ jar ไฟล์ที่มีชื่อว่า MySQL-connector-java-ชื่อเวอร์ชั่นไปเก็บในโฟลเดอร์ WEB-INF/lib/ ใน Tomcat

2) **โหลด Driver JDBC** การโหลดสามารถทำได้โดยใช้เมธอด forName () จากคลาส java.lang.Class เพื่อโหลดชื่อของคลาส Driver ที่เหมาะสมมีรูปแบบดังนี้

```
Class.forName ("org.gjt.mm.MySQL.Driver");
```

3) สร้างการเชื่อมต่อกับฐานข้อมูล ทำได้โดยใช้เมธอด getConnection () ที่อยู่ในคลาส java.sql.DriverManager เพื่อสร้างการติดต่อ แล้วนำเก็บไว้ในอ็อบเจ็กต์ Connection มีรูปแบบดังนี้

```
Connection ชื่ออ็อบเจ็กต์=DriverManager.getConnection (String URL);
```

พารามิเตอร์ URL จะเป็นการรวม URL, User name และ Password เข้าด้วยกันดังตัวอย่างต่อไปนี้

```
String URL="jdbc: MySQL: //localhost/test?User=root&Password=1234";
Connection con=DriverManager.getConnection (URL);
```

4) สร้างอ็อบเจ็กต์ Statement โดยการใช้เมธอด createStatement () จากอ็อบเจ็กต์ Connection แล้วนำไปเก็บไว้ในอ็อบเจ็กต์ Statement เพื่อส่งคำสั่ง SQL ไปยังฐานข้อมูลมีรูปแบบดังนี้

```
Statement ชื่ออ็อบเจ็กต์= Connection อ็อบเจ็กต์.createStatement
```

แสดงตัวอย่างการใช้งานดังนี้

```
Statement stmt= con.createStatement ();
```

5) จัดการกับคำสั่ง Query หลังจากสร้าง Statement แล้วต่อไปเป็นการจัดการกับคำสั่ง Query ซึ่งมีเมธอดที่ใช้จัดการคือ

- ExecuteQuery (String SQL); เมธอดนี้จะคืนค่ากลับมาเป็นอ็อบเจ็กต์ ResultSet ซึ่งเป็นผลลัพธ์ที่ได้จากการหาข้อมูล ดังนั้นจะต้องสร้างอ็อบเจ็กต์ ResultSet มารับค่าที่หาได้ เมธอดนี้จะใช้กับคำสั่ง SELECT ยกตัวอย่างเช่น

```
ResultSet rs= stmt.executeQuery ("SELECT * FROM test");
```

- ExecuteUpdate (String SQL); เมธอดนี้จะคืนค่าเป็นจำนวนแถว ดังนั้นถ้าต้องการค่าที่ได้ไปใช้งานจะต้องนำตัวแปร int ไปรับเมธอดนี้ ใช้กับคำสั่ง SQL ที่ทำให้ข้อมูลในฐานข้อมูลมีการเปลี่ยนแปลง เช่น INSERT, UPDATE, DELETE หรือ CREATE TABLE เป็นต้น ยกตัวอย่างเช่น

```
int row= stmt.executeQuery("INSERT INTO test VALUES('SUMET','NOOB')");
```

6) ประมวลผลที่ได้จากการติดต่อ (ResultSet) ข้อมูลที่อยู่ในอ็อบเจ็กต์ ResultSet จะมีลักษณะโครงสร้างเหมือนตาราง ใช้เก็บผลลัพธ์ที่ได้จากการติดต่อกับฐานข้อมูล อ็อบเจ็กต์นี้มีเมธอดที่น่าสนใจคือ next () ใช้แสดงข้อมูลของแถวถัดไป getXXX (หมายเลขคอลัมน์) หรือ getXXX (ชื่อเลขคอลัมน์) ใช้คืนค่าของคอลัมน์ที่เลือก แสดงตัวอย่างการใช้งานดังนี้

```
ResultSet rs= stmt.executeQuery("SELECT * FROM test");
while(rs.next()){
        rs.getString("name");
```

7) ปิดการเชื่อมต่อ ขั้นตอนสุดท้ายคือ ปิดการเชื่อมต่อกับฐานข้อมูล ใช้ในกรณีที่ในเพจนั้นๆ ไม่ต้องการเชื่อมต่อกับฐานข้อมูลแล้ว เพื่อประหยัดทรัพยากรของระบบ (Resource) แสดงตัวอย่างดังนี้

```
stmt.close();
con.close();
```

- **คำสั่ง Query เบื้องต้น**

1) คำสั่งสร้างฐานข้อมูล

```
CREATE DATABASE [ชื่อฐานข้อมูล];
```

2) คำสั่งการเลือกข้อมูลที่ต้องการ

```
SELECT [ชื่อ field ที่ต้องการดู] FROM [ชื่อ Table]
WHERE [เงื่อนไขที่เราต้องการให้ Query กรองข้อมูลออกมา];
```

3) คำสั่งเพิ่มข้อมูลเข้าไปในฐานข้อมูล

```
INSERT INTO [ชื่อ Table] VALUES ('[ข้อมูล field1]', '[ข้อมูล field2]'... '[ข้อมูล fieldสุดท้าย]');
```

4) คำสั่งปรับปรุงแถวข้อมูลในฐานข้อมูล

```
UPDATE [ชื่อ Table] SET [field1] = [ข้อมูล], ..., [fieldสุดท้าย]=[ข้อมูล]
WHERE [เงื่อนไขที่เราต้องการให้ กรองข้อมูลออกมา];
```

5) คำสั่งในการลบแถวข้อมูล

```
DELETE FROM [ชื่อ Table]
WHERE [เงื่อนไขที่เราต้องการให้ Query กรองข้อมูลออกมา];
```

2.8 Servlet กับ JSP

Servlet เป็นแอปพลิเคชั่นที่ทำงานทางฝั่งเซิร์ฟเวอร์ (Server Side Application) มีรูปแบบการทำงานคล้ายๆ กับภาษา CGI มีความสามารถในการจัดการกับเว็บแอปพลิเคชั่นแบบ Dynamic Content และถูกสร้างขึ้นจากภาษา JAVA ส่งผลให้ Servlet ยังคงคุณสมบัติของ Object Oriented โดย Servlet ที่ถูกสร้างขึ้นจะมาทำงานอยู่ใน Servlet Engine โดยใน Servlet Engine หนึ่งๆ อาจประกอบไปด้วยหลายๆ Servlet เช่น Servlet ที่ทำหน้าที่ในการเก็บข้อมูลสมาชิก หรือ Servlet ที่ทำหน้าที่ในการตรวจสอบการ Login เป็นต้น

2.8.1 ขั้นตอนการทำงานของ Servlet

- Clients เช่น Web Brower เข้า Web Server และส่ง HTTP Request
- Web Server ได้รับ Request และส่งต่อไปที่ Servlet Container
- Servlet Container จะพิจารณาว่า ควรจะใช้งาน Servlet ตัวใด โดยพิจารณาจาก Configuration ของ Servlet นั้นๆ และจะเรียกใช้งาน Servlet โดยพิจารณาจาก Request ที่ได้จาก Client และ Response ที่ได้รับจาก Server
- Servlet ทราบถึงข้อมูลต่างๆ ผ่านทาง Request Object ที่ได้รับจาก Sever หลังงานนั้น Servlet จะทำการประมวลผลและส่งผลลัพธ์กลับไปยัง Client ผ่านทาง Request Object
- หลังจากที่ Servlet ประมวลผลเสร็จเรียบร้อย Servlet Container จะทำการตรวจสอบว่า Response มีความเรียบร้อยสมบูรณ์ แล้วจึงส่งหน้าที่กลับไปให้ Web Server ทำการส่ง Response กลับไปยัง Client ต่อไป

2.8.2 โครงสร้างของ Servlet

Servlet Interface เป็นองค์ประกอบสำคัญของ Servlet API โดยทุก Servlet จะสนับสนุนการทำงานของ Interface นี้และคลาสที่สืบทอดคุณสมบัติของ Interface นี้มีอยู่ 2 คลาสด้วยกันคือ คลาส GenericServlet ซึ่งอยู่ในแพ็กเกจ javax.servlet และคลาส HTTPServlet ซึ่งอยู่ในแพ็กเกจ javax.servlet.HTTP แต่ส่วนใหญ่จะใช้คลาส HTTP Servlet ในการพัฒนา Servlet คลาส HTTP Servlet จะมีเมธอดนอกเหนือจากที่มีอยู่ใน Servlet Interface เพื่อช่วยจัดการกับการประมวลผลของ HTTP Request เมธอดเหล่านั้นได้แก่

- เมธอด doGet() ใช้สำหรับจัดการการร้องขอ HTTP GET
- เมธอด doPost() ใช้สำหรับจัดการการร้องขอ HTTP POST
- เมธอด doPut() ใช้สำหรับจัดการการร้องขอ HTTP PUT
- เมธอด doDelete() ใช้สำหรับจัดการการร้องขอ HTTP DELETE
- เมธอด doOptions() ใช้สำหรับจัดการการร้องขอ HTTP OPTION
- เมธอด doTrace() ใช้สำหรับจัดการการร้องขอ HTTP TRACE

2.8.3 หน้าที่หลักของ Servlet

- อ่านข้อมูลที่ได้รับจาก Client: ข้อมูลส่วนใหญ่จะได้รับมาจากฟอร์มบน Web Page นอกจากนี้ยังสามารถนำมาจาก JAVA Applet หรือโปรแกรม HTTP Client อื่นๆ
- ตรวจสอบข้อมูลต่างๆ ที่เกี่ยวกับ Request ที่อยู่ใน HTTP Request: ตัวอย่างของข้อมูลเหล่านี้ได้แก่ ความสามารถของ Browser, Cookies, Host Name ของ Client และข้อมูลอื่นๆ
- ประมวลผล: ขั้นตอนนี้ Servlet อาจจะต้องติดต่อกับฐานข้อมูลหรือโปรแกรมอื่นๆ หรืออาจจะคำนวณผลลัพธ์โดยตรง
- จัดการกับรูปแบบของผลลัพธ์: โดยทั่วไปแล้วขั้นตอนนี้เป็นการจัดการกับข้อมูลใน HTML Page
- กำหนด HTTP Response Parameters ที่เหมาะสม: ขั้นตอนนี้ Servlet จะบอก Browser ถึงชนิดของเอกสารที่ส่งกลับ การ Set Cookies และ Parameters รวมไปถึงงานอื่นๆ ที่เกี่ยวข้อง
- ส่งเอกสารกลับไปยัง Client: เอกสารที่ได้รับการส่งกลับไปยัง Client นั้นอาจจะเป็น HTML Format หรือรูปภาพก็ได้

2.8.4 ข้อดีของ Servlet

เมื่อพิจารณาถึงหน้าที่ของ Servlet แล้วพบว่า Servlet มีการทำงานในลักษณะเดียวกับ Common Gateway Interface (CGI) Programs และ Server Extensions อื่นๆ เช่น Netscape Server API (INSAPI) หรือ Apache Modules อย่างไรก็ตาม Servlet มีข้อดีเหนือเทคโนโลยีดังกล่าวดังต่อไปนี้

- Servlet มีการทำงานที่รวดเร็วกว่า CGI Scripts เนื่องจากขั้นตอนการทำงานที่แตกต่างกัน : การทำงานของ CGI Programs นั้น จะมีการเริ่ม Process ใหม่ทุกครั้งที่มี HTTP Request และถ้าหากมี Request จำนวน N ครั้งไปที่ CGI Programs นั้น Code ของ CGI Programs นั้นจะถูก Load เข้าสู่ Memory เป็นจำนวน N ครั้งเช่นเดียวกัน ซึ่งเป็นการสิ้นเปลือง Resource มาก สำหรับ Servlet นั้น ทุกๆ Request จะเป็น Lightweight JAVA Thread ซึ่งได้รับการควบคุมโดย JAVA Virtual Machine และถ้ามี Request จำนวน N ครั้งไปที่ Servlet นั้น Servlet Class จะถูก Load เพียงครั้งเดียว ถึงแม้ว่า จะมี N Threads

- Servlet ใช้ Standard API ที่ได้รับการสนับสนุนจากหลายๆ Web Servers : ปัจจุบันมีบริษัท Third Party หลายบริษัทที่นำเสนอ Web Server ที่สนับสนุนการทำงานของ Servlet และ JSP ตัวอย่างเช่น Apache Web Server, iPlanet Web Server และ Microsoft IIS เป็นต้น นอกจากนี้ Servlet Container ยังสามารถนำไปผนวกเข้ากับ Web-Enabled Application Server เช่น BEA Web Logic Application Server, IBM WebSphere และ iPlanet Application Server เป็นต้น Servlet สามารถพูดคุยกับ Web Server ได้โดยตรง ในขณะที่ CGI Programs ไม่สามารถทำได้ถ้าไม่ใช้ Server-Specific API การสื่อสารกับ Web Server โดยตรงมีข้อดีหลายประการ เช่น ทำให้การแปลง Relative URLS ไปเป็น Path Names ที่ถูกต้องง่ายขึ้น Servlet หลายๆ โปรแกรมยังสามรถใช้ข้อมูลร่วมกัน ทำให้การพัฒนา Database Connection Pooling และ Resource Sharing สะดวกขึ้น นอกจากนี้ Servlet ยังสามารถรักษาข้อมูลจาก Request หนึ่งไปยังอีก Request หนึ่งได้โดยการใช้เทคนิคของ Session Tracking และ Computation Caching

- Servlet สามารถใช้ประโยชน์จาก JAVA Programming Languages ในเรื่องของความสะดวกในการพัฒนา และความเป็นอิสระจาก Platform ใดๆ:

- Servlet ถูกเขียนขึ้นจากภาษา JAVA และตรงกับ Standard API ทำให้ Servlet สามารถเข้าถึง APIs ที่มีอยู่มากมายของ JAVA Platform ด้วยเหตุผลนี้ทำให้ Servlet สามารถ Run อยู่บน Web Server ต่างชนิดกันโดยไม่ต้องมีการแก้ไข Code ตัวอย่างเช่น Servlet ที่ Run บน JAVA Web Server สามารถ Run บน Apache Tomcat ได้ โดยไม่ต้องมีการแก้ไขเปลี่ยนแปลง Code ในปัจจุบันมี Web Server จำนวนมากมายที่สนับสนุน JAVA 2 Platform, Enterprise Edition (J2EE) ซึ่งจะเห็นแนวโน้มของการใช้ Servlet ที่เพิ่มขึ้นด้วย

2.9 JAVA Server Page (JSP)

JAVA Server Page หรือเรียกสั้นๆ ว่า JSP เป็นเทคโนโลยีที่ทำงานบนฝั่งเซิร์ฟเวอร์ (Server Side Script) มีความสามารถในการจัดการกับเว็บแอปพลิเคชั่นแบบ Dynamic Content โดย JSP ถูกพัฒนามาจาก Servlet เพื่อแก้ไขปัญหาหนึ่งที่เกิดขึ้นกับ Servlet คือ Servlet จะเป็นการผสม

ข้อมูลในส่วนของ Business Logic (ข้อมูลทางตรรกะ เช่น JAVABean, Database) กับ Presentation Layer (ข้อมูลในส่วนของการแสดงผล) รวมเข้าด้วยกัน นอกจากนี้ Servlet ยังเปรียบเสมือน JAVA File ที่มีการฝังแท็ก HTML ลงไป จากปัญหาดังกล่าวทำให้ผู้ที่จะพัฒนาจำเป็นต้องมีความรู้ทางด้านภาษา JAVA มากพอสมควร และการแก้ไขในส่วนของหน้าตาที่ใช้แสดงผลจะทำได้ยาก ส่วน JSP จะมีการแยกข้อมูล Business Logic และ Presentation Layer ออกจากกัน นอกจากนี้ JSP ยังเปรียบเสมือน HTML Page ที่มีการฝัง JAVA Code ลงไปทำให้การเขียนโปรแกรมมีประสิทธิภาพมากขึ้น โดยการแยกหน้าที่ของผู้พัฒนาตามความถนัด เช่น หากถนัดเขียนโค้ด ก็ให้ทำงานในส่วนของ Business Logic แต่หากถนัดที่จะออกแบบหน้าตาของเว็บเพจ ก็ให้ทำงานในส่วนของ Presentation Layer

2.9.1 ขั้นตอนการทำงานของ JSP

1) เริ่มแรกจะรับการร้องขอจาก Web Browser โดย Client หรือ Web Server

2) JSP Container จะตรวจสอบว่าไฟล์ JSP ที่ถูกร้องขอนั้นเคยถูกแปลงเป็นไฟล์ Servlet และคอมไพล์เป็นไฟล์คลาสแล้วหรือยัง ถ้ายังไม่เคย JSP Container จะทำการแปลงไฟล์ JSP เป็นไฟล์ Servlet และคอมไพล์เป็นไฟล์คลาสตามลำดับ แต่ถ้าเคยแปลงแล้ว จะตรวจสอบเพิ่มเติมว่าไฟล์ JSP ที่ถูกร้องขอมีการเปลี่ยนแปลงหรือไม่ ถ้ามีการเปลี่ยนแปลง จะทำการแปลงเป็นไฟล์ Servlet และคอมไพล์เป็นไฟล์คลาสใหม่ แต่ถ้าไม่มีการเปลี่ยนแปลง จะส่งผลลัพธ์ที่ได้ไปยังเว็บเซิร์ฟเวอร์

3) เว็บเซิร์ฟเวอร์จะตอบสนองการร้องขอกลับไปยัง Client เป็นลำดับสุดท้าย

2.9.2 การใช้งานแท็กต่างๆ (Tags)

เนื่องจาก JSP เป็นเอกสารที่มีการผสมผสานกันระหว่างข้อมูลที่ไม่มีการเปลี่ยนแปลง (Static Content) กับข้อมูลที่มีการเปลี่ยนแปลง (Dynamic Content) ในส่วนของ Static Content จะเขียนด้วยภาษา HTML ส่วน Dynamic Content จะเขียนด้วยภาษา JAVA โดยมีการแทรกแท็กลงในเอกสาร JSP เพื่อให้ Server ทราบว่าโค้ดในส่วนนี้เป็น Dynamic Content ดังนั้นจึงแบ่งแยกแท็กเหล่านี้ตามลักษณะการใช้งานได้ดังนี้คือ

- **แท็กพื้นฐาน (Scripting Elements)**

แท็กพื้นฐาน เป็นแท็กที่ใช้แยกโค้ดที่เป็น JAVA กับโค้ดที่เป็น HTML ภายในเอกสาร JSP มีอยู่ 3 ชนิด คือ

- Scriptlets ใช้สำหรับแทรกโค้ด JAVA ลงในเอกสาร JSP มีรูปแบบในการใช้งาน 2 แบบดังนี้

แบบ JSP:	<% Code %>
แบบ HML:	<jsp: scriptlet> Code </jsp: scriptlet>

- Declarations ใช้สำหรับสร้างเมธอดหรือตัวแปร เมื่อสร้างเสร็จแล้ว เมธอดหรือตัวแปรเหล่านั้นจะสามารถใช้ได้ทุกที่ในเอกสาร JSP ที่สร้างขึ้นมีรูปแบบในการใช้งาน 2 แบบดังนี้

```
แบบ JSP:    <%! Code %>
แบบ HML:    <jsp:declaration> Code </jsp:declaration >
```

- Expressions ใช้สำหรับนำค่าที่อยู่ในตัวแปรหรือค่าในอ็อบเจ็กต์ต่างๆ ออกมาแสดงผลทางจอภาพในรูปของสตริง การใช้งานแท็ก Expressions นี้ไม่ต้องใช้คำสั่ง out.println() เนื่องจาก JSP Engine จะทำให้อัตโนมัติเมื่อพบแท็กนี้ยกตัวอย่างเช่น การแสดงวันเวลาปัจจุบัน ถ้าเป็นแท็ก Scriptlets จะต้องใช้คำสั่ง <% out.println(new JAVA.util.Date()); %> แต่ถ้าเป็นแบบ Expression จะใช้คำสั่ง <% =new JAVA.util.Date() %> แทน มีรูปแบบการใช้งานดังนี้

```
แบบ JSP:    <%= Code %>
แบบ HML:    <jsp: expression> Code </jsp: expression >
```

- **แท็กหมายเหตุ (Comment)**

ใช้แสดงหมายเหตุต่างๆ เพื่อให้ผู้พัฒนาเองหรือผู้ที่พัฒนาต่อสามารถเข้าใจการทำงานของเพจได้ง่ายขึ้น โดยแท็กเหล่านี้จะไม่ถูกนำไปประมวลผล มี 3 แบบคือ

- แท็กหมายเหตุแบบ JSP จะเขียนไว้ลอยๆ ไม่สามารถเขียนไว้ในแท็ก Scriptlet, Expression หรือ Declaration ได้ มีรูปแบบดังนี้

```
<%-- หมายเหตุ -- %>
```

- แท็กหมายเหตุแบบ JAVA จะเขียนไว้ในแท็ก Scriptlet หรือแท็ก Declaration มีรูปแบบดังนี้

```
<%
 //หมายเหตุ กรณีมี 1 บรรทัด
/*
หมายเหตุ  กรณีมีหลายบรรทัด
หมายเหตุ  กรณีมีหลายบรรทัด
*/
```

- แท็กหมายเหตุแบบ HTML จะเขียนไว้ในแท็ก HTML หรือ XML มีรูปแบบดังนี้

```
<! -- หมายเหตุแบบ HTML -->
```

- **แท็ก Directives**

ใช้สำหรับกำหนดคุณสมบัติต่างๆ หรือเรียกใช้ไฟล์จากภายนอก มีรูปแบบการใช้งานดังนี้

```
<@ ชื่อ directive [ชื่อแอททริบิวต์ = "ค่าของแอททริบิวต์"] %>
```

- Page ใช้กำหนดคุณสมบัติต่างๆ ของเพจ JSP ยกตัวอย่างเช่น การกำหนดชื่อภาษาที่จะใช้งานในเพจ หรือการกำหนดค่าของตัวแปร Session เป็นต้นมีรูปแบบการใช้งานดังนี้

```
<@page ชื่อแอททริบิวต์ = "ค่าของแอททริบิวต์" %>
```

- Include Directive ใช้สำหรับนำไฟล์อื่นที่อยู่ภายนอกเข้ามาในไฟล์ปัจจุบัน จะเป็นไฟล์ JSP หรือไฟล์ HTML ก็ได้ ข้อดีคือทำให้ไม่เสียเวลาในการสร้างเพจที่ถูกเรียกใช้อยู่บ่อยๆ แต่มีข้อเสียคือ ถ้าหากมีการแก้ไขไฟล์ที่ถูกเรียกใช้และไฟล์ปัจจุบันไม่มีการเปลี่ยนแปลง ถ้าเรียกใช้ไฟล์ปัจจุบันผลที่ได้จะเหมือนกับตอนที่ยังไม่ได้แก้ไขไฟล์ที่ถูกเรียกใช้ เหตุที่เป็นเช่นนี้เพราะการนำไฟล์จากภายนอกเข้ามาในไฟล์ปัจจุบันเป็นช่วงเวลา Translation หมายความว่า ถ้าเรียกใช้ไฟล์ปัจจุบันเป็นครั้งที่ 2 JSP Container จะตรวจสอบว่าไฟล์ที่ใช้มีการเปลี่ยนแปลงหรือไม่ ถ้าไม่มีการเปลี่ยนแปลงก็จะนำ Servlet ตัวเดิมออกมาแสดง แต่ข้อเสียดังกล่าวได้ถูกแก้ไขแล้วตั้งแต่ Tomcat เวอร์ชั่น 5 เป็นต้นไป มีรูปแบบการใช้งานดังนี้

```
<%@ include file = "ชื่อที่อยู่ของไฟล์" %>
```

- Taglib Directive ทำหน้าที่บอกให้ JSP Container ทราบว่าจะมีการใช้แท็กที่สร้างขึ้นมาเอง มีรูปแบบการใช้งานดังนี้

```
<%@ taglib uri= "ชื่อที่อยู่ของไฟล์ TLD" prefic="ชื่อตัวแปร" %>
```

2.9.3 ข้อดีของ JSP

JAVA Server Page มีข้อดีหลายอย่างอันเนื่องมาจากการใช้เทคโนโลยีของ JAVA หรือ JAVA Servlet ดังที่กล่าวไว้แล้วในข้างต้น ไม่ว่าจะเป็นความเป็นอิสระจาก Platform หรือความสะดวกในการพัฒนาเป็นต้น

ข้อดีของ JSP ถ้าเปรียบเทียบกับเทคโนโลยีอื่นๆ เช่น

- **Active Server Page (ASP)**

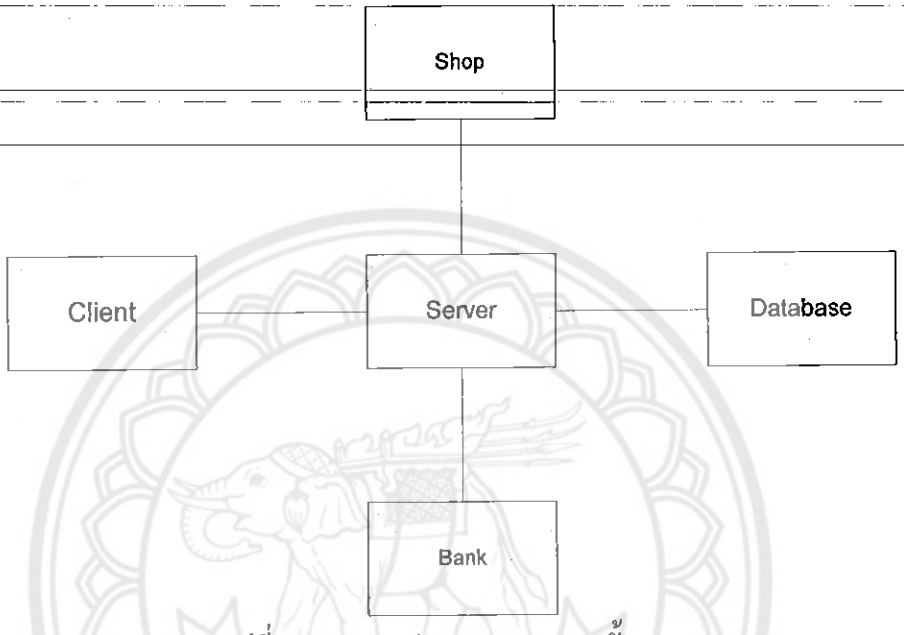
เป็นเทคโนโลยีที่รันทางฝั่งเซิร์ฟเวอร์เหมือนกับ JSP แต่ข้อดีของ JSP ที่เหนือกว่า ASP คือ JSP จะไม่ยึดติดกับระบบปฏิบัติการ (Operating Systems) หรือ Web Servers ใดๆ

- **Servlet**

JSP มีข้อดีที่เหนือกว่า Servlet คือ สามารถพัฒนาเว็บแอปพลิเคชั่นได้สะดวกรวดเร็วกว่า Servlet เนื่องจาก JSP จะมีการแยกส่วนแสดงผล (Presentation Layer) กับส่วนการทำงาน (Business Logic Layer) ออกจากกัน นอกจากนี้ JSP ยังเปรียบเสมือน HTML Page ที่มีการฝัง JAVA Code ลงไป ตรงกันข้ามกับ Servlet ที่เปรียบเสมือน JAVA File ที่มีการฝัง HTML Tag ลงไป ส่งผลให้ JSP สามารถทำความเข้าใจโค้ด และแก้ไขได้ง่ายกว่า Servlet

บทที่ 3
วิธีการดำเนินงาน

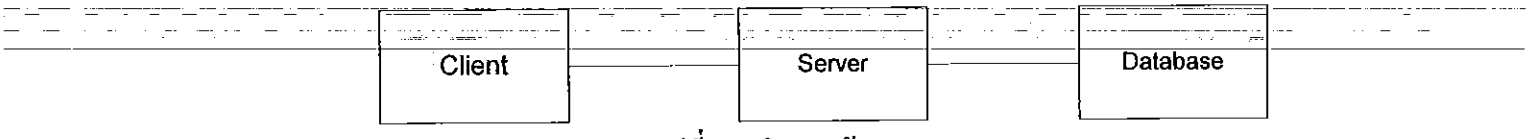
3.1 โครงสร้างของระบบ



รูปที่ 3.1 Model จำลองของระบบทั้งหมด

จากรูปที่ 3.1 เป็น Model โครงสร้างของระบบการจ่ายเงินผ่านมือถือทั้งหมด โดยจะให้ Server เป็นจุดศูนย์กลางในการติดต่อระหว่าง Client (ผู้ซื้อ) กับ Shop (ร้านค้า) และ Bank (ธนาคาร) ส่วนข้อมูลระหว่างการติดต่อการติดต่อจะเก็บไว้ที่ Database (ฐานข้อมูล)

ระบบการจ่ายเงินผ่านมือถือที่ผู้จัดทำจัดทำขึ้นนั้นเป็นเพียง Model จำลองเท่านั้น จากการวิเคราะห์โดยภาพรวมแล้วสามารถแบ่งองค์ประกอบในการพัฒนาเป็นได้เป็น 3 ส่วนหลักๆ คือ



รูปที่ 3.2 โครงสร้างของระบบ

3.1.1 Client ซึ่งเป็น Application ที่ทำงานบนโทรศัพท์มือนั้น จะเลือกใช้ Java Wireless Toolkit 2.5 beta เป็นตัว Simulate พัฒนาโดยใช้ภาษา J2ME

3.1.2 Server ซึ่งเป็น Application บน Server นั้น จะใช้ Apache Tomcat Server 5.5 เป็นตัว Simulate พัฒนาโดยใช้ภาษา JSP

3.1.3 Database ของ Server ซึ่งในระบบจำลองนั้นเป็นฐานข้อมูลขนาดเล็ก จึงจะเลือกใช้ MySQL

3.2 การออกแบบรูปแบบการติดต่อและการใช้งานระหว่าง Client กับ Server

3.2.1 การออกแบบการติดต่อระหว่างมือถือ (Client) กับเซิร์ฟเวอร์ (Server)

การออกแบบการติดต่อระหว่างมือถือ (Client) กับเซิร์ฟเวอร์นั้นยึดหลักในการพัฒนา 3 ประการ คือ

1) ความปลอดภัย ระบบการจ่ายเงินผ่านมือถือที่ออกแบบต้องมีความปลอดภัยในระดับที่สากลยอมรับและสามารถเชื่อถือได้ โดยการที่จะให้ระบบที่ออกแบบนี้มีความปลอดภัยดังกล่าวจึงได้นำโปรโตคอล SSL และอัลกอริทึมการลงลายมือชื่อดิจิตอล ECDSA รวมทั้งหลักการอื่นๆ มาใช้ในระบบ ดังนี้

- ในการใช้จ่ายผ่านระบบจ่ายเงินผ่านมือถือที่ออกแบบ ผู้จ่ายเงินจะต้องเป็นผู้ใช้ (เจ้าของเครื่องโทรศัพท์มือถือ) เท่านั้น จึงจะสามารถทำรายการได้ เนื่องจากผู้ใช้แต่ละคนจะมี Private Key เพียง 1 ตัวที่เก็บอยู่ในมือถือเท่านั้น
- ผู้ใช้จะไม่สามารถปฏิเสธความรับผิดชอบในการใช้จ่ายเงินในแต่ละครั้งได้ เนื่องจากในการใช้จ่ายเงินในแต่ละครั้งนั้น ผู้ใช้จะต้องใช้ Private Key ในการสร้าง Digital Signature โดยใช้อัลกอริทึมการลงลายมือชื่อดิจิตอล ECDSA
- ในทุกขั้นตอนของการจ่ายเงินผ่านระบบจ่ายเงินผ่านมือถือนี้ การติดต่อ/ส่งข้อมูลระหว่าง Server และ Client ผ่านโปรโตคอล SSL ทั้งหมด ซึ่ง SSL เป็นโปรโตคอลที่ได้รับการยอมรับจากสากลอย่างแพร่หลาย จึงมั่นใจได้ว่าระบบที่ออกแบบมีความปลอดภัย

2) **ง่ายต่อการใช้งาน** โดยระบบที่จะพัฒนาขึ้นมาได้นี้ผู้ใช้งานต้องสามารถใช้งานได้โดยง่าย มีขั้นตอนในการใช้งานไม่มาก ซึ่งทั้งหมดนี้ต้องขึ้นอยู่กับมาตรฐานของความปลอดภัยเป็นหลัก

3) **ใช้เวลาไม่นานต่อการติดต่อในแต่ละครั้ง** โดยจะนำมาตรฐานของการลงลายมือชื่อดิจิตอลที่เรียกว่า ECDSA มาใช้ซึ่งเป็น Algorithm การเข้ารหัสที่ใช้จำนวนกุญแจในการเข้ารหัสน้อยกว่า Algorithm อื่นๆ ที่ในระดับความปลอดภัยเท่ากัน ซึ่งจะทำให้ระบบจ่ายเงินผ่านมือถือที่จะพัฒนาขึ้นมานั้นใช้ระยะเวลารวมในการจ่ายเงินในแต่ละครั้งไม่นาน

3.3 Secure Socket Layer (SSL)

ระบบจ่ายเงินผ่านมือถือนั้น User กับ Server จะติดต่อกันผ่าน Internet ซึ่งในปัจจุบันมีความปลอดภัยน้อยมาก เนื่องจากการติดต่อผ่านโปรโตคอลทั่วไปคือ HTTP นั้นไม่มีการเข้ารหัสข้อมูล ผู้ที่ดักจับข้อมูลได้ก็สามารถอ่านข้อมูลนั้นๆ ออกมาได้เลย

การทำให้การติดต่อกันผ่าน Internet มีความปลอดภัยสามารถทำได้สามารถทำได้ 2 วิธี คือ

1) การนำข้อมูลที่ส่งระหว่าง User กับ Server ไปเข้ารหัสโดยใช้ Algorithm ต่างๆที่มีให้ใช้งาน

2) การใช้ Protocol https ซึ่ง Apache Tomcat และ Wireless Toolkit มีให้ใช้งาน

ซึ่งจากการวิเคราะห์ดูแล้วจึงเลือกใช้วิธีที่ 2 ด้วยเหตุผลดังนี้

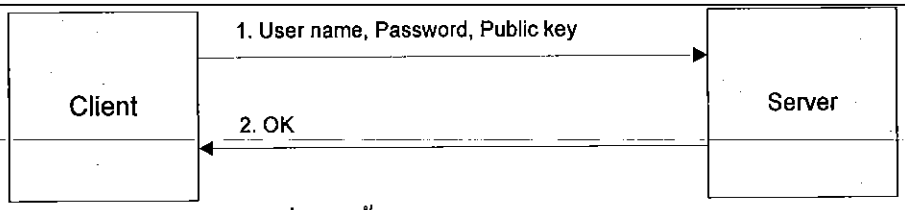
- ง่ายต่อการนำไป Implement
- มีความปลอดภัยสูงในระดับที่น่าเชื่อถือได้

3.4 Digital Signature

สำหรับการทำธุรกรรมทางด้านการเงิน ลายเซ็นถือว่าเป็นสิ่งสำคัญมาก เพราะจะเป็นหลักฐานสำคัญ แต่ว่าการทำธุรกรรมออนไลน์นั้น Digital Signature นั้นถือว่าสิ่งสำคัญเทียบเท่ากับลายเซ็น แต่จากการศึกษาพบว่า การทำ Digital Signature นั้น ต้องอาศัยการคำนวณมาก โดยเฉพาะอย่างยิ่งในมือถือแล้วการประมวลผลมาก จะไม่เหมาะเป็นอย่างยิ่งเพราะจะต้องใช้เวลานานมาก ดังนั้นการทำ Signature ที่ใช้การคำนวณที่ไม่มาก แต่มีความปลอดภัยสูงจึงเป็นสิ่งที่สำคัญมาก และจากการศึกษาพบว่า การใช้ Signature RSA และ DSA ที่มี Key ขนาด 1024 bit ปรากฏว่าการประมวลผลช้ามาก สำหรับความปลอดภัยในระดับที่สามารถยอมรับได้ ดังนั้นผู้จัดทำจึงลองศึกษา Algorithm แบบต่างๆ สำหรับ Signature จึงได้ทดลอง ECDSA โดยมีคุณสมบัติคือใช้ Key ที่สั้นกว่าสำหรับในระดับความปลอดภัยเดียวกันกับ DSA จึงทำให้ประมวลเร็วกว่าในขณะที่ความปลอดภัยเท่ากัน ซึ่งเป็นสิ่งจำเป็นมากสำหรับการทำงานบนมือถือ และใช้ Library สำเร็จรูปที่เป็น Free ที่รองรับ ECDSA ของ Bouncy Castle (คือ Library สำหรับการเข้ารหัสการลงลายมือชื่อ)

3.5 รูปแบบของ Protocol

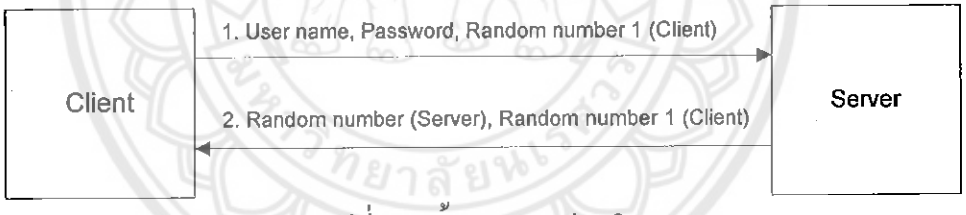
3.5.1 ขั้นตอนการสมัครใช้บริการ



รูปที่ 3.3 ขั้นตอนการสมัครใช้บริการ

1. Client ส่ง User name, Password, Public key ไปที่ Server เพื่อให้ Server เก็บ Public key ของ Client ไว้ใน Database ของ Server
2. Server ส่ง OK กลับไปให้ Client เพื่อบอกว่า Server เก็บ Public key ของ Client ลงใน Database ของ Server เรียบร้อยแล้ว

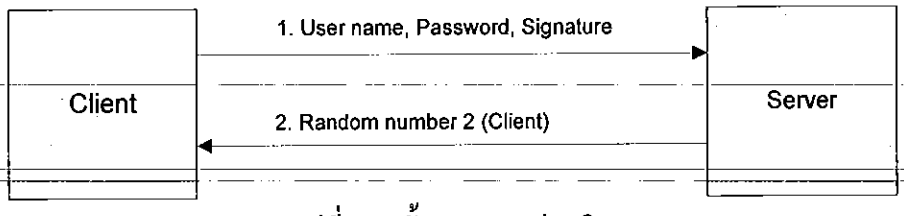
3.5.2 ขั้นตอนการจ่ายเงิน



รูปที่ 3.4 ขั้นตอนการจ่ายเงิน

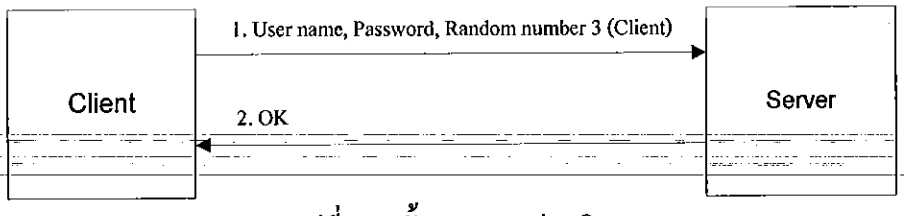
1. จากรูปที่ 3.4 Client ส่ง Request การจ่ายเงินไปที่ Server โดยมีข้อมูล User name ของ Client, Password ของ Client และ Random number 1 ของ Client
2. Server ตอบ Request การจ่ายเงินของ Client กลับไปให้ Client โดยมีข้อมูล Random number 1 (Client) และ Random number ที่ Server สร้างขึ้นมา
3. การที่ Client ส่ง Random number 1 (Client) ไปให้ Server และ Server ก็ส่ง Random number 1 (Client) กลับไปให้ Client นั้น เพราะว่า Client จะใช้ Random number 1 (Client) เป็นตัวตรวจสอบว่า การส่ง Request การจ่ายเงินนั้นไปถึง Server จริง เพราะจะมีแต่ Server เท่านั้นที่สามารถเข้ารหัสและถอดรหัส Random user1 ได้

4. การที่ Client ส่ง Request การขอจ่ายเงินไปที่ Server นั้น ก็เพราะว่าต้องการนำ Random number (Server) มาเข้ารหัสเป็น Signature เพื่อป้องกันการ Replay Signature (คือ การนำ Signature ตัวเดิมมาใช้จ่ายเงินซ้ำๆ)



รูปที่ 3.5 ขั้นตอนการจ่ายเงิน

1. จากรูปที่ 3.5 Client ส่ง User name, Password, Signature ไปให้ Server โดยที่ Signature จะประกอบไปด้วยข้อมูล User name, Password, Shop Id, Price, Random number (Server), Random number 2 (Client), Random number 3 (Client)
2. Server ส่ง Random number 2 (Client) ไปให้ Client เพื่อเป็นการบอก Client ว่าการตรวจสอบ Signature ถูกต้อง
3. การที่นำ Random number 2 (Client), Random number 3 (Client) มาใช้นั้นก็เพื่อให้ Server และ Client ทำการตรวจสอบกันเองว่าต่างฝ่ายต่างเป็นตัวจริง เพราะว่า Random number 2 (Client), Random number 3 (Client) มีเพียง Client เท่านั้นที่สามารถเข้ารหัสเป็น Signature ได้แต่เพียงผู้เดียว และมีเพียง Server เท่านั้นที่สามารถถอดรหัส Signature ได้



รูปที่ 3.6 ขั้นตอนการจ่ายเงิน

1. Client ส่ง User name, Password, Random number 3 (Client) เพื่อเป็นการ Confirm ว่าต้องการจ่ายเงินจริงตาม Signature ที่ส่งมาก่อนหน้านี้
2. Server ส่ง OK ไปให้ Client เพื่อบอกว่าการจ่ายเงินของ Client เสร็จเรียบร้อยแล้ว

3.6 การตั้งค่า SSL

3.6.1 การตั้งค่า Protocol HTTPS มีวิธีการดังนี้

- **Tomcat**

1) แก้ไข Configuration File ของ Tomcat เพื่อให้ Tomcat สามารถรองรับ Protocol HTTPS
2) สร้าง Private Key/Public Key ของ Tomcat โดยใช้ Algorithm MD5 กับ RSA
3) สร้าง Certificate ของ Tomcat โดยใช้ Public Key ที่สร้างขึ้นในข้อ 1.2

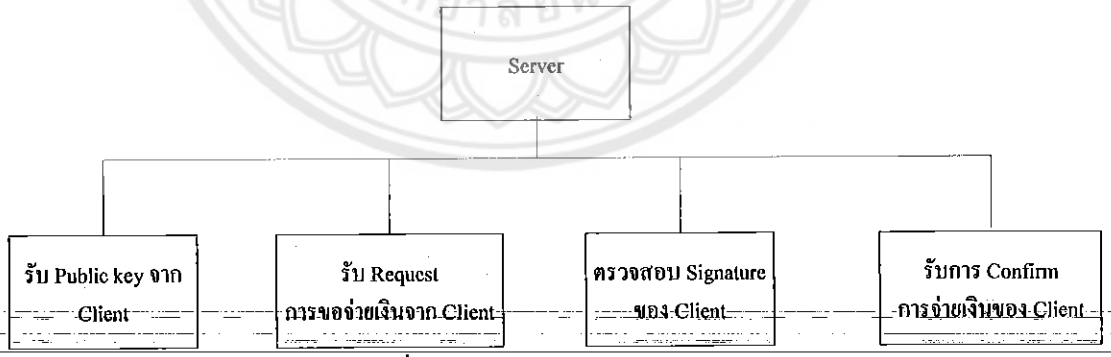
- **Wireless Toolkit**

ส่วนของการแก้ไขในโปรแกรม Wireless Toolkit แก้ไขโดยทำการ Import Certificate ของ Tomcat เข้าไป เพื่อให้ Wireless Toolkit สามารถรองรับ Protocol HTTPS ได้

ส่วนในเรื่องความปลอดภัยของข้อมูลที่ User ส่งให้ Server และการป้องกันการโกงของ User ผู้จัดทำจะใช้ลายเซ็นดิจิตอลเข้ามาช่วย โดยจากการศึกษาเปรียบเทียบ Algorithm แล้วพบว่าในความปลอดภัยระดับเดียวกันแล้ว Algorithm ECDSA ใช้จำนวน Key ที่สั้นกว่า และยังใช้เวลาในการประมวลผลน้อยกว่าอีกด้วย ดังนั้นจึงตัดสินใจใช้ Algorithm ECDSA

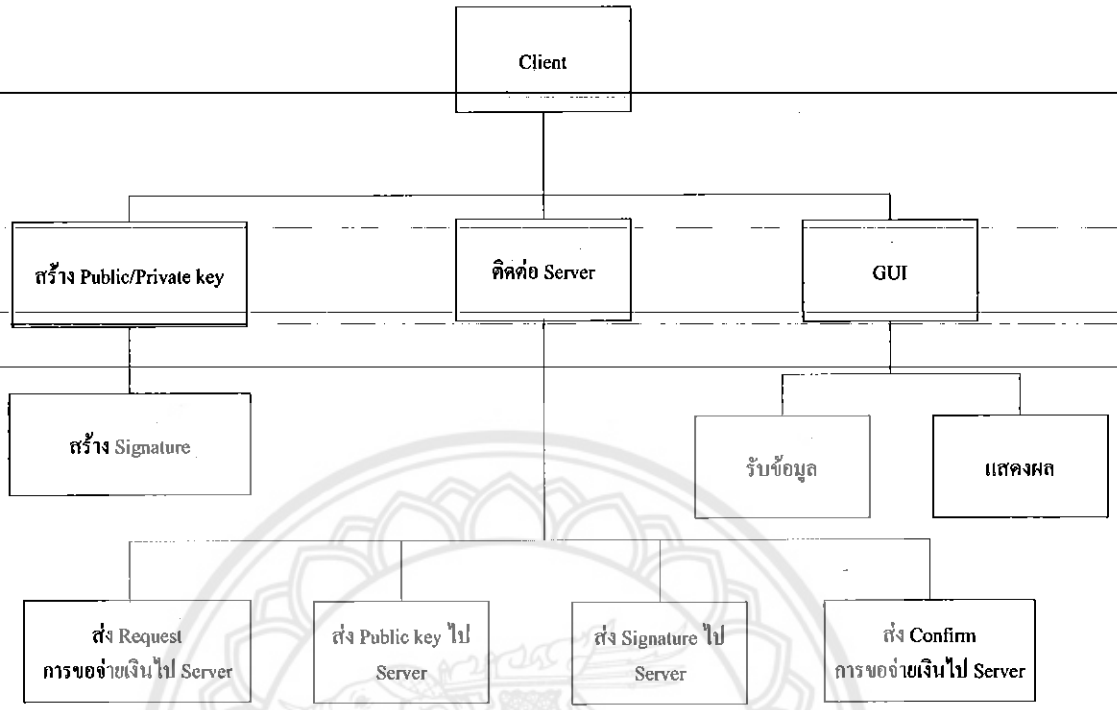
3.7 คอมโพเนนท์ของ Server และ Client

ส่วนประกอบของ Server มีดังนี้



รูปที่ 3.7 ส่วนประกอบของ Server

ส่วนประกอบของ Client มีดังนี้



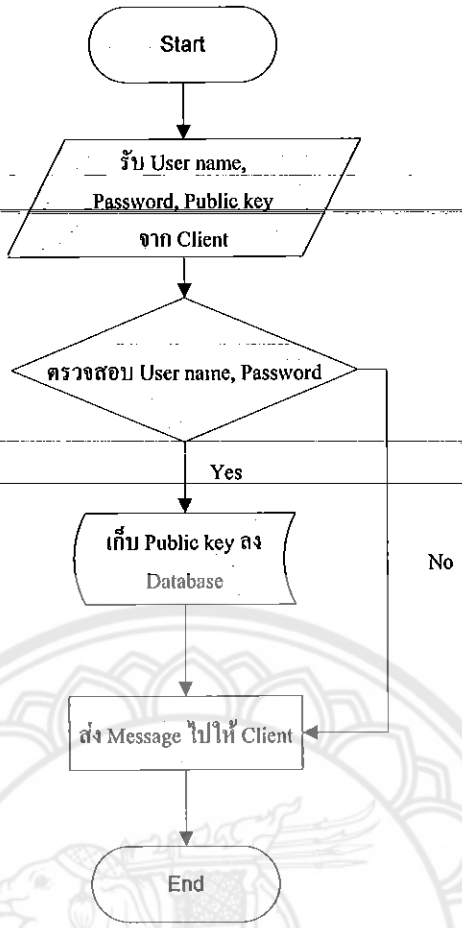
รูปที่ 3.8 ส่วนประกอบของ Client

3.8 การทำงานของโปรแกรม Apache Tomcat 5.5

การทำงานของ Tomcat ได้แบ่งออกเป็นดังนี้

3.8.1 การสมัครใช้บริการระบบจ่ายเงินผ่านมือถือ

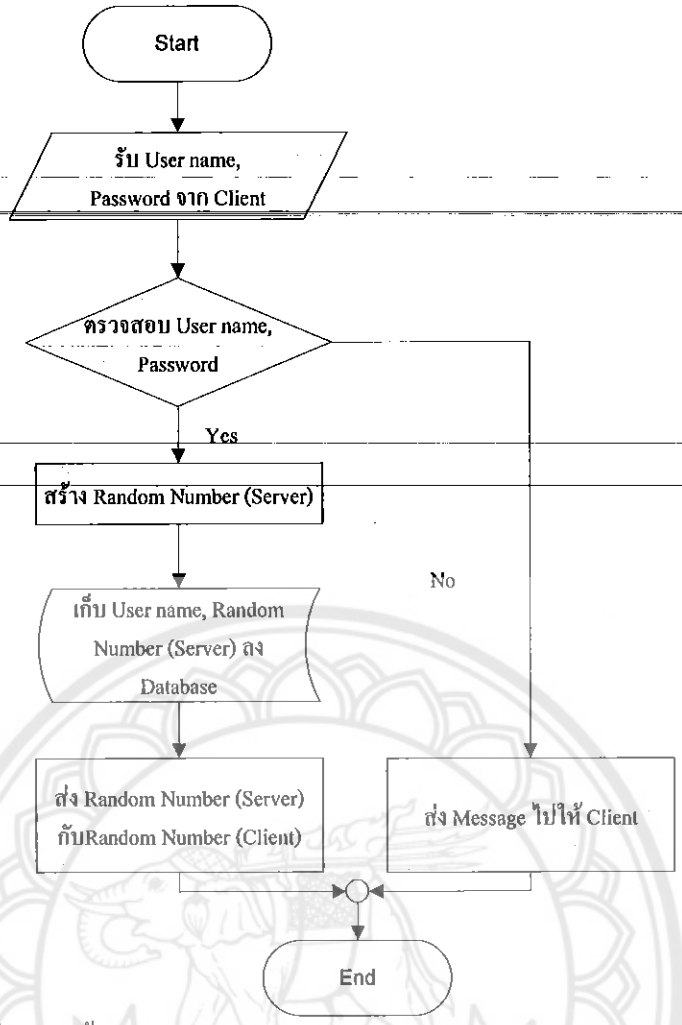
1) รับ User name, Password, Public Key จาก Client แล้วทำการตรวจสอบ User name, Password ของ Client กับข้อมูลของ User ใน Database ว่าถูกต้องหรือไม่ ถ้าถูกต้อง ทำการเก็บ Public Key ของ User นั้นๆลงใน Database และส่ง Message Ok ไปให้ Client ถ้าไม่ถูกต้องก็ Message อื่นๆไปให้ Client อื่นๆ ที่บอกถึงความผิดพลาดแล้วจบการทำงาน



รูปที่ 3.9 การสมัครใช้บริการระบบจ่ายเงินผ่านมือถือ (Apache Tomcat)

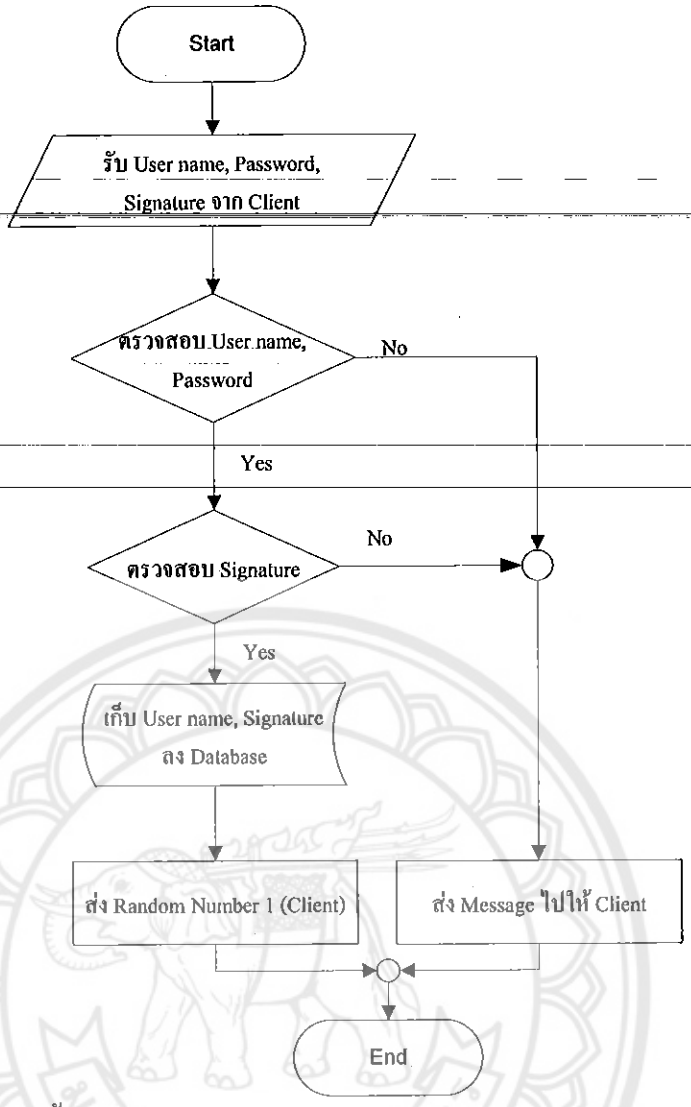
3.8.2 ขั้นตอนการจ่ายเงินของระบบจ่ายเงินผ่านมือถือ

1) รับ Request การขอจ่ายเงินของ Client โดยประกอบด้วยข้อมูล User name, Password, Random Number 1 (Client) แล้วเช็ค User name, Password ของ Client กับ ข้อมูลของ User ใน Database ว่าถูกต้องหรือไม่ ถ้าถูกต้องทำการสร้าง Random Number (Server) และเก็บ ข้อมูลการ Request การขอจ่ายเงินของ User ซึ่งประกอบด้วย User name, Random Number 1 (Client), Random Number (Server) นั้นลงใน Database และส่ง Random Number (Client), Random Number (Server) ไปให้ Client ถ้าไม่ถูกต้องก็ส่ง Message อื่นๆ ที่บอกว่าผิดพลาดอย่างไร แล้วก็จบการทำงาน



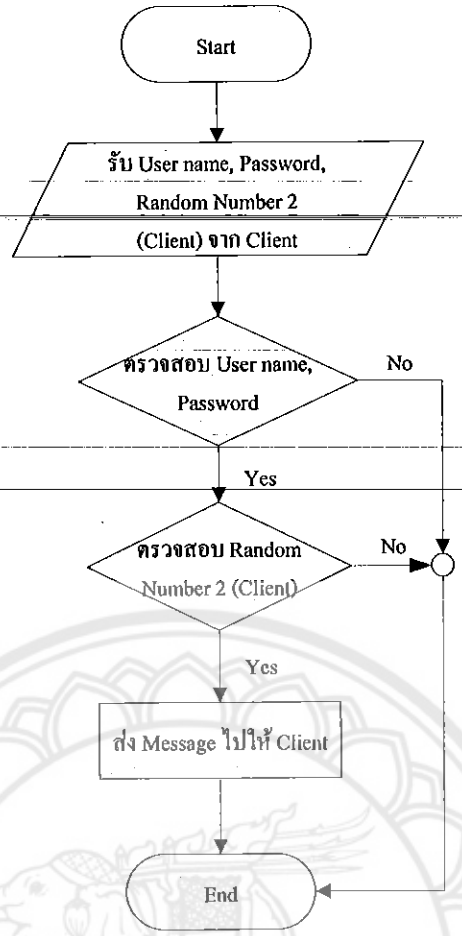
รูปที่ 3.10 ขั้นตอนการจ่ายเงินของระบบจ่ายเงินผ่านมือถือ (Apache Tomcat)

2) รับ User name, Password, Signature แล้วเช็ค User name, Password ของ Client กับข้อมูลของ User ใน Database ว่าถูกต้องหรือไม่ ถ้าถูกต้องแล้วทำการตรวจสอบ Signature ว่าถูกต้องหรือไม่ ถ้าถูกต้องทำการเก็บข้อมูล Signature การขอจ่ายเงินของ User นั้นลงใน Database และส่งการ Random Number 1 (Client) ไปให้ User ถ้าการตรวจสอบทั้ง 2 ขั้นตอนไม่ถูกต้องก็ส่ง Message อื่นๆที่บอกถึงความผิดพลาดแล้วจบการทำงาน



รูปที่ 3.11 ขั้นตอนการจ่ายเงินของระบบจ่ายเงินผ่านมือถือ (Apache Tomcat)

3) รับการ Confirm การขอจ่ายเงินของ User โดยประกอบด้วยข้อมูล User name, Password, Random Number 2 (Client) แล้วเช็ค User name, Password ของ Client กับข้อมูลของ User ใน Database ว่าถูกต้องหรือไม่ แล้วทำการ Check การเช็ค Random Number 2 (Client) นั้นๆ ว่าถูกต้องหรือไม่ ถ้าถูกต้องทำการเก็บข้อมูลการขอจ่ายเงินของ User นั้นลงใน Database แล้วส่งข้อมูลการจ่ายเงินไปยังธนาคาร ถ้าไม่ถูกต้อง Message อื่นๆที่บอกถึงความผิดพลาดแล้วจบการทำงาน



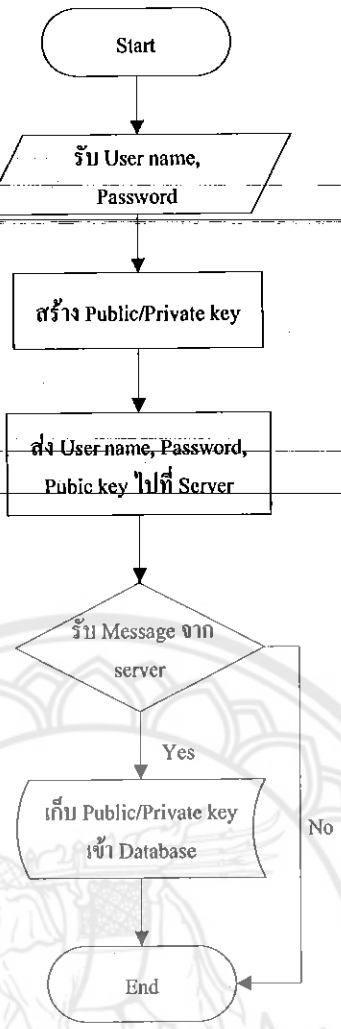
รูปที่ 3.12 ขั้นตอนการจ่ายเงินของระบบจ่ายเงินผ่านมือถือ (Apache Tomcat)

3.9 การทำงานของ Wireless toolkit

การทำงานของ Wireless Toolkit ได้แบ่งออกเป็นดังนี้

3.9.1 การสมัครใช้บริการระบบจ่ายเงินผ่านมือถือ

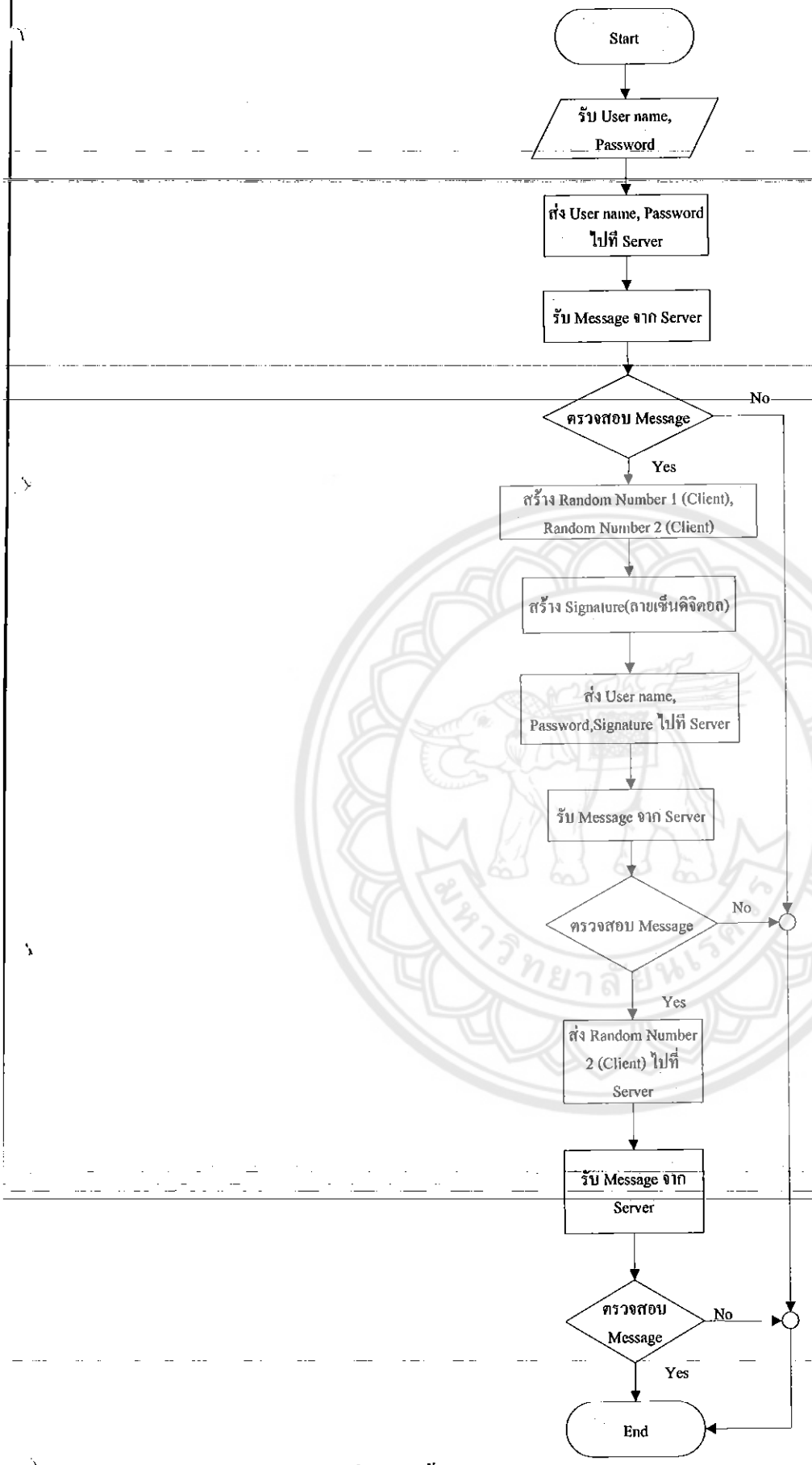
1) รับ User name, Password
2) ทำการสร้าง Public/Private Key โดยใช้ Algorithm ECDSA แล้วส่ง User name, Password, Public Key ไปให้ Server
3) รับ Message จาก Server แล้วทำการเช็ค Message ถ้าถูกต้องก็เก็บ Public/Private key เข้า Database ของมือถือ



รูปที่ 3.13 การสมัครใช้บริการระบบจ่ายเงินผ่านมือถือ (Wireless Toolkit)

3.9.2 ขั้นตอนการจ่ายเงินของระบบจ่ายเงินผ่านมือถือ

1) รับ User name, Password, ชื่อร้านค้า, ราคา
2) ทำการส่ง request การขอจ่ายเงินไปยัง Server และรอการตอบกลับจาก Server ถ้าได้รับการ Confirm การจ่ายเงินจาก Server ก็ทำงานต่อ แต่ถ้าไม่ได้รับการ Confirm การจ่ายเงินจาก Server ก็จบการทำงาน
3) ทำการสร้าง Signature แล้วส่งไปที่ Server และรอการตอบกลับจาก Server ถ้าได้รับการ Confirm การจ่ายเงินจาก Server ก็ทำงานต่อ แต่ถ้าไม่ได้รับการ Confirm การจ่ายเงินจาก Server ก็จบการทำงาน
4) ถ้าต้องจ่ายเงินจริงตามก็ส่งการ Confirm ไปยัง Server ถ้าไม่ต้องการจ่ายเงินก็จบการทำงานไป



รูปที่ 3.14 ขั้นตอนการจ่ายเงินของระบบจ่ายเงินผ่านมือถือ (Wireless Toolkit)

3.10 ระบบฐานข้อมูล (Database)

ระบบฐานข้อมูลที่เลือกนำมาใช้ในระบบจ่ายเงินผ่านมือถือนี้คือ mySQL ซึ่งในฐานข้อมูลมีทั้งหมด 4 ตาราง โดยมีโครงสร้างดังนี้

ตารางที่ 3.1 ฐานข้อมูล: ตาราง Person

ชื่อ	ชนิดของข้อมูล	ขนาดของข้อมูล	ข้อมูล
User_name	Varchar	30	User name
Password	Varchar	30	Password
Q	Varchar	255	Key Q
A	Varchar	255	Key A
B	Varchar	255	Key B
N	Varchar	255	Key N
GG	Varchar	255	Key GG
QQ	Varchar	255	Key QQ

หมายเหตุ Key Q A B N GG และ QQ คือส่วนประกอบของ Public Key

ตารางที่ 3.2 ฐานข้อมูล: ตาราง logg

ชื่อ	ชนิดของข้อมูล	ขนาดของข้อมูล	ข้อมูล
User_name	Varchar	255	User name
Digest	Varchar	255	ข้อมูลที่ไม่ได้เข้ารหัส
Digest_encrypt	Varchar	255	ข้อมูลที่เข้ารหัสด้วย BASE 64
Sig1	Varchar	255	ข้อมูลของ Signature ส่วนที่ 1
Sig2	Varchar	255	ข้อมูลของ Signature ส่วนที่ 2
Random_User	Varchar	255	Random User
Random_Server	Varchar	255	Random Server

ตารางที่ 3.3 ฐานข้อมูล: ตาราง request

ชื่อ	ชนิดของข้อมูล	ขนาดของข้อมูล	ข้อมูล
User_name	Varchar	255	User name
Random_user	Varchar	255	Random User
Random_server	Varchar	255	Random Server

ตารางที่ 3.4 ฐานข้อมูล: ตาราง result

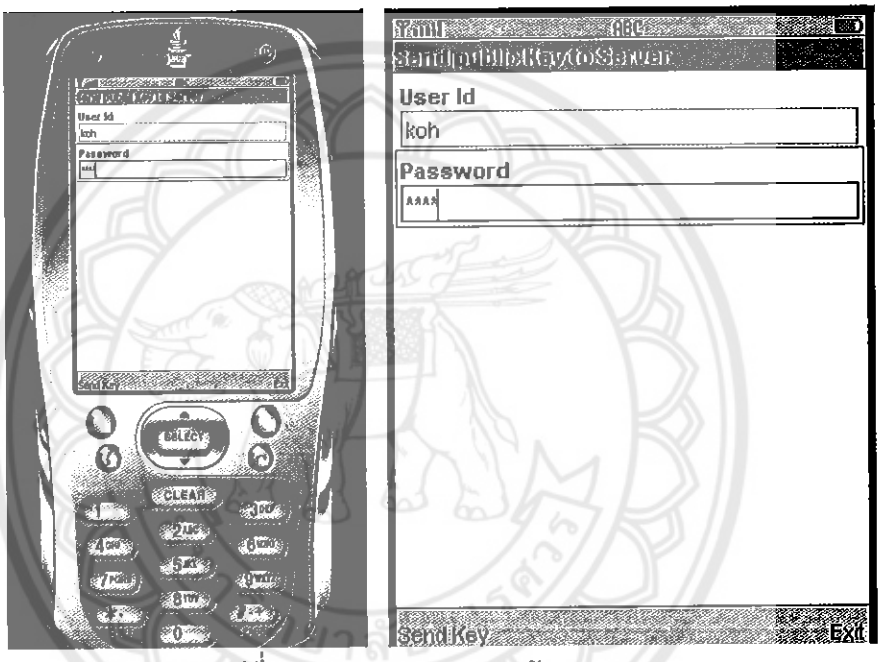
ชื่อ	ชนิดของข้อมูล	ขนาดของข้อมูล	ข้อมูล
User_name	Varchar	255	User name
Digest	Varchar	255	ข้อมูลที่ไม่ได้เข้ารหัส
Digest_encrypt	Varchar	255	ข้อมูลที่เข้ารหัสด้วย BASE 64
Sig1	Varchar	255	ข้อมูลของ Signature ส่วนที่ 1
Sig2	Varchar	255	ข้อมูลของ Signature ส่วนที่ 2

บทที่ 4

ผลการทดลอง

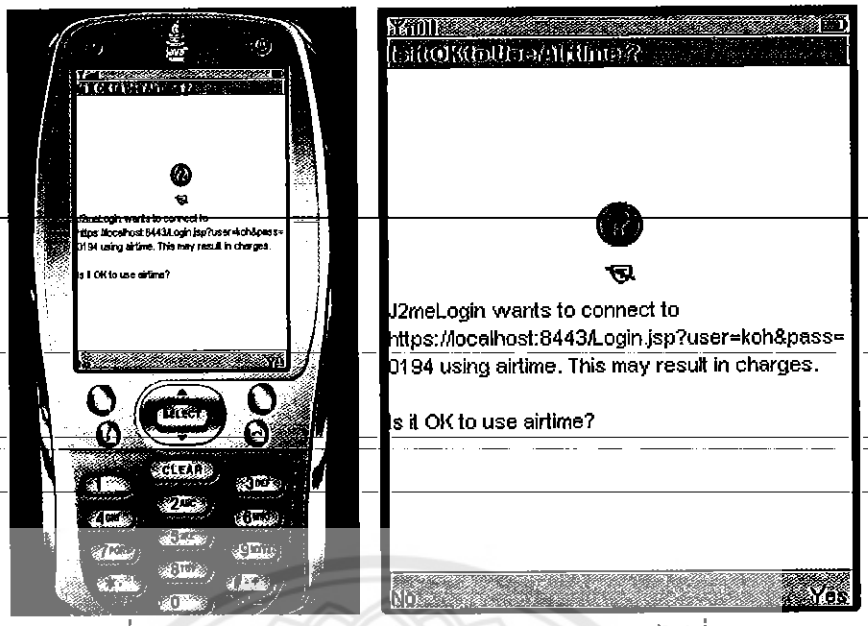
ในการทดสอบนั้น ได้ทำการยกตัวอย่าง โดยสร้าง User name ที่ชื่อ koh และ Password เป็น 0194 ลงในฐานข้อมูลของระบบ แล้วได้ทำการทดสอบ โดยเริ่มจาก

การทำงานของโปรแกรมในขั้นแรกนั้น โปรแกรมจะทำการตรวจสอบว่า ในโปรแกรมมี Private Key เก็บไว้ในฐานข้อมูลของมือถือแล้วหรือยัง ถ้ายังก็จะทำการสร้าง Private Key ก่อน โดยเริ่มจาก User ทำการกรอก User name และ Password ดังรูปที่ 4.1



รูปที่ 4.1 การกรอก User Id กับ Password

ต่อจากนั้น โปรแกรมก็จะทำการสร้าง Private / Public Key โดยใช้ Algorithm ECDSA เมื่อสร้าง Key เสร็จแล้ว ก็จะทำการส่ง User name, Password, Public Key ไปที่ Server เพื่อที่จะให้ทาง Server เก็บ Public ของ User นั้นๆไว้ในฐานข้อมูลของ Server โดยที่ 1 User จะมี Public Key ได้เพียง 1 Key เท่านั้น ดังรูปที่ 4.2

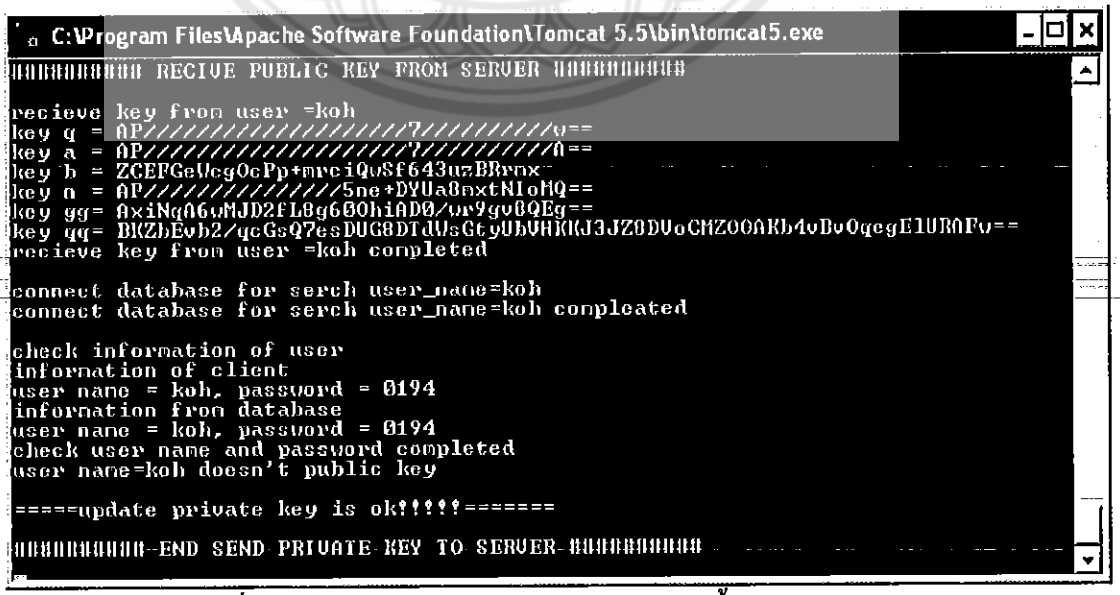


รูปที่ 4.2 การส่ง User name, Password, Public Key ไปที่ Server

หลังจากนั้น User จะรอการตอบกลับจากทาง Server

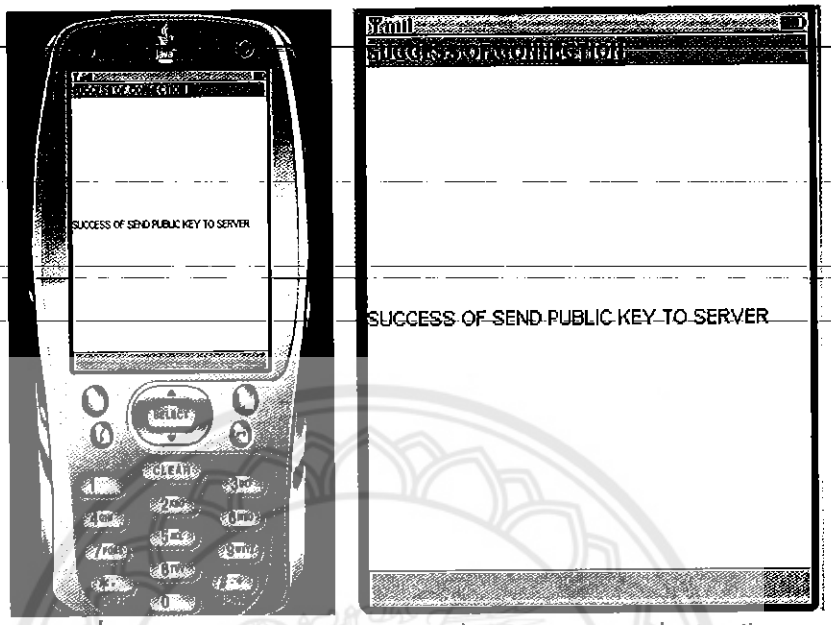
โดยทางฝั่ง Server นั้นก็จะทำการเช็ค User name, Password ว่าถูกต้องหรือเปล่า ถ้าไม่ถูกต้องก็ตอบกลับไปทาง User ว่าไม่พบ User name, Password นี้ในฐานข้อมูลแต่ถูกต้องก็ทำการเช็คต่อไปว่า User name นี้ มี Public Key อยู่ในฐานข้อมูลแล้วหรือยัง ถ้ายังก็ทำการเก็บ Public Key ของ User name นั้นลงในฐานข้อมูล และตอบกลับไปยัง User ว่าได้เก็บ Public Key ของ User ลงในฐานข้อมูลเสร็จแล้ว

ดังรูปที่ 4.3



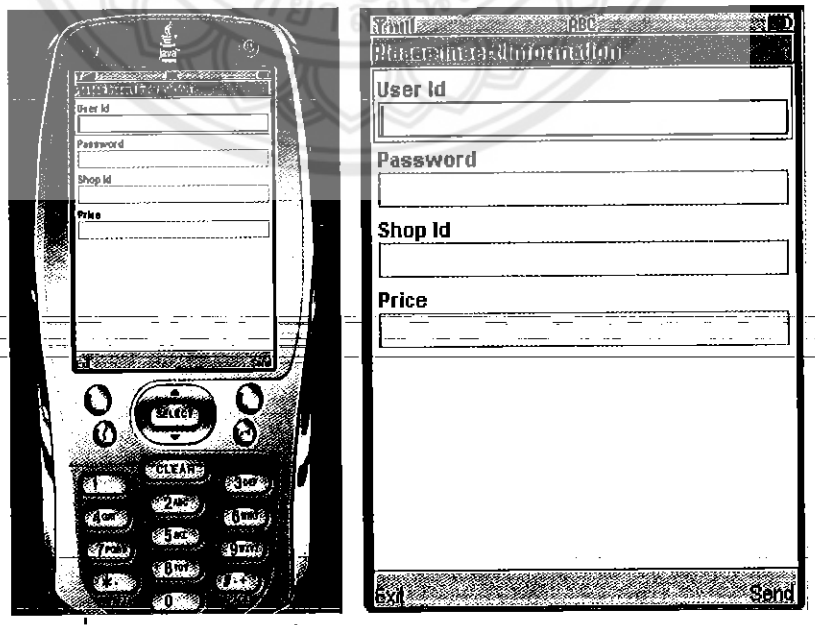
รูปที่ 4.3 การเก็บ Public Key ของ User name นั้นลงในฐานข้อมูล

เมื่อทางฝั่ง Server ตอบกลับมาว่าการเก็บ Public Key เสร็จเรียบร้อยแล้วก็จะปรากฏหน้าจอดังรูปที่ 4.4



รูปที่ 4.4 Server ตอบกลับมาว่าการเก็บ Public Key เสร็จเรียบร้อย

หลังจากนั้นก็จะปรากฏหน้าจอดังรูปด้านล่าง ซึ่งจะเป็นการแสดงว่า ในฐานข้อมูลของมือถือมี Private Key ของ User เรียบร้อย และจะไม่การปรากฏหน้าจอที่สร้าง Public / Private Key ขึ้นอีก ดังรูปที่ 4.5



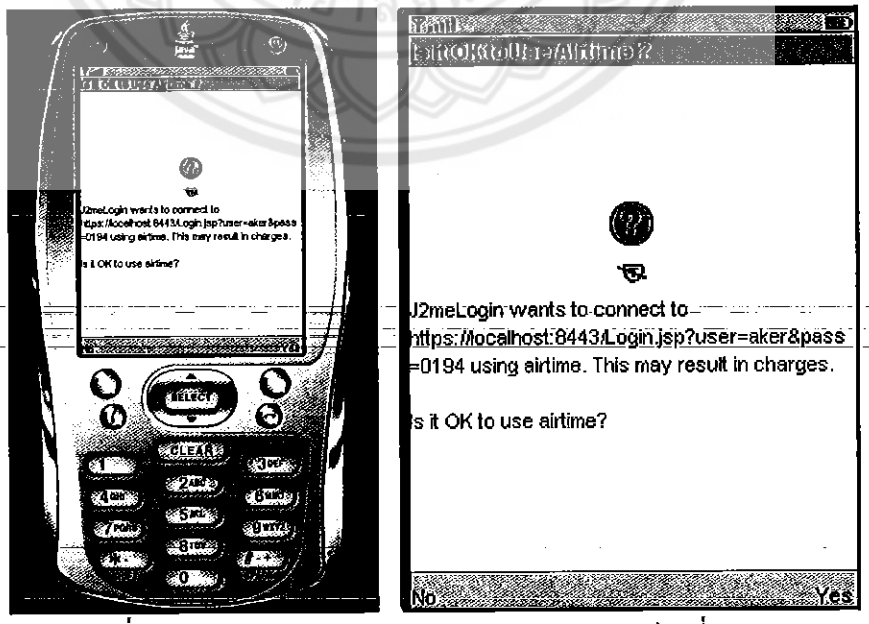
รูปที่ 4.5 หน้าจอการใช้งานปกติหลังจากการตรวจสอบ Public Key

4.1 กรณี User name หรือ Password ผิด

ต่อมาจะเป็นการสร้าง Public / Private Key ของ User ที่ไม่มี User name และ Password อยู่ในฐานข้อมูลของทางฝั่ง Server โดยในที่นี้จะใช้ User name ที่ชื่อว่า aker และมี Password เป็น 0194 ซึ่งแสดงได้ดังรูปที่ 4.6

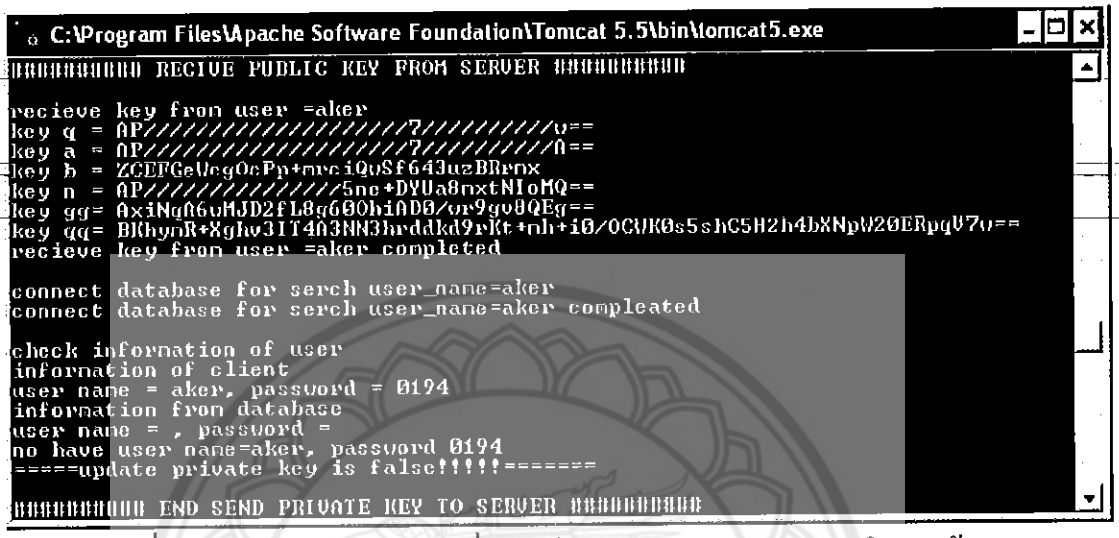
รูปที่ 4.6 ใช้ User name ที่ชื่อว่า aker และมี Password เป็น 0194

โปรแกรมทำการสร้าง Public / Private Key แล้วส่งไปยัง Server ดังรูปที่ 4.7



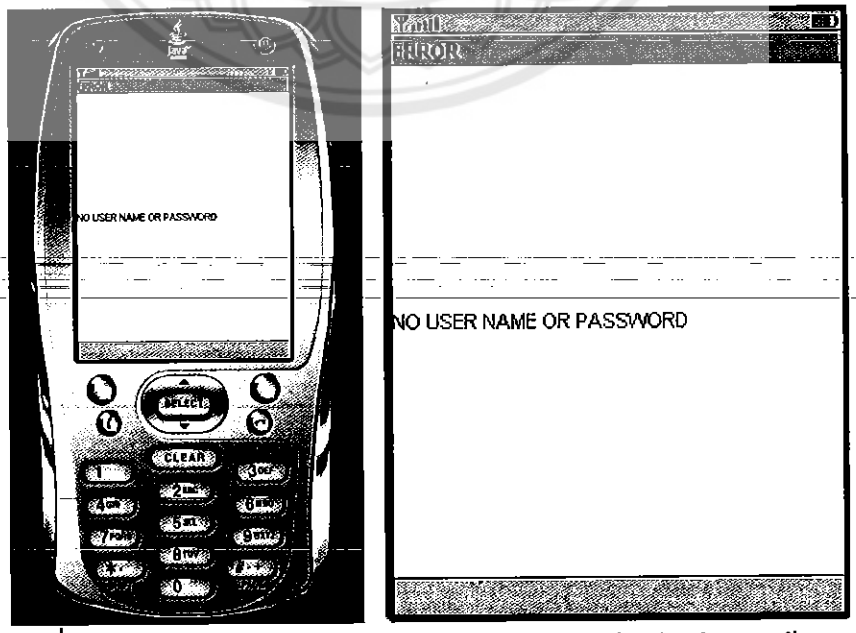
รูปที่ 4.7 การส่ง User name, Password, Public Key ไปที่ Server

ทางฝั่ง Server ก็จะทำการเช็ค User name, Password ของ User ว่าถูกต้องหรือไม่ ซึ่งในที่นี้ User name ที่ชื่อ aker และมี Password เป็น 0194 ไม่มีในฐานข้อมูล ทาง Server ก็จะตอบกลับไปทาง Server ว่าไม่มี User name, Password ในฐานข้อมูล และจะไม่มีการเก็บ Public Key ลงในฐานข้อมูลของ Server ดังรูปที่ 4.8



รูปที่ 4.8 ไม่บันทึก Public Key เนื่องจากไม่มี User name, Password ในฐานข้อมูล

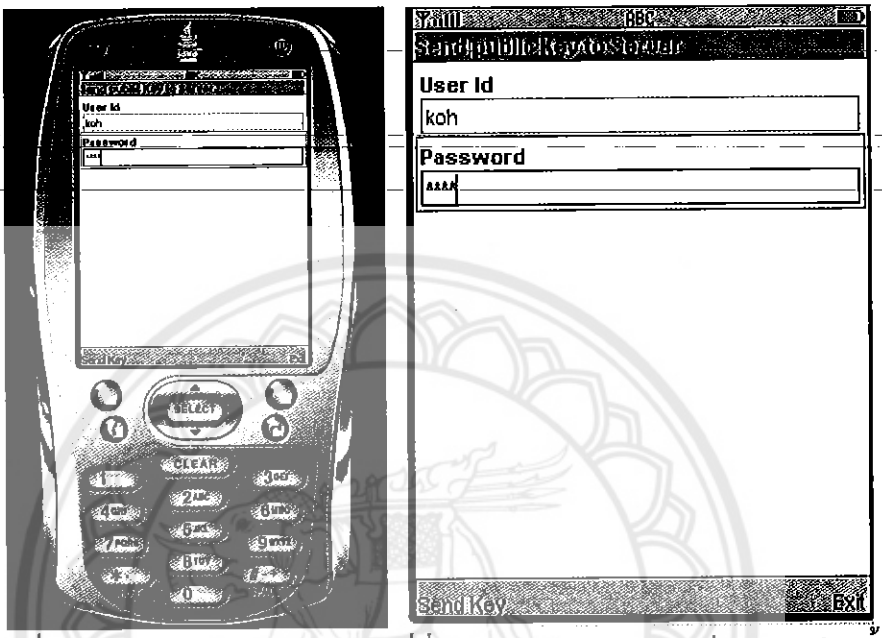
เมื่อทางฝั่ง Server ตอบกลับมาว่า User name, Password ไม่มีอยู่ในฐานข้อมูลก็จะปรากฏหน้าจอดังรูปที่ 4.9 และจะไม่มีการเก็บ Private Key ที่สร้างขึ้นลงในฐานข้อมูลของมือถือ พร้อมกันนั้น User ก็ไม่สามารถทำการจ่ายเงินผ่านทางมือถือได้ จนกว่าจะมีการสร้าง Public / Private Key ของ User name ที่ถูกต้อง



รูปที่ 4.9 Server ตอบกลับมาว่า User name, Password ไม่มีอยู่ในฐานข้อมูล

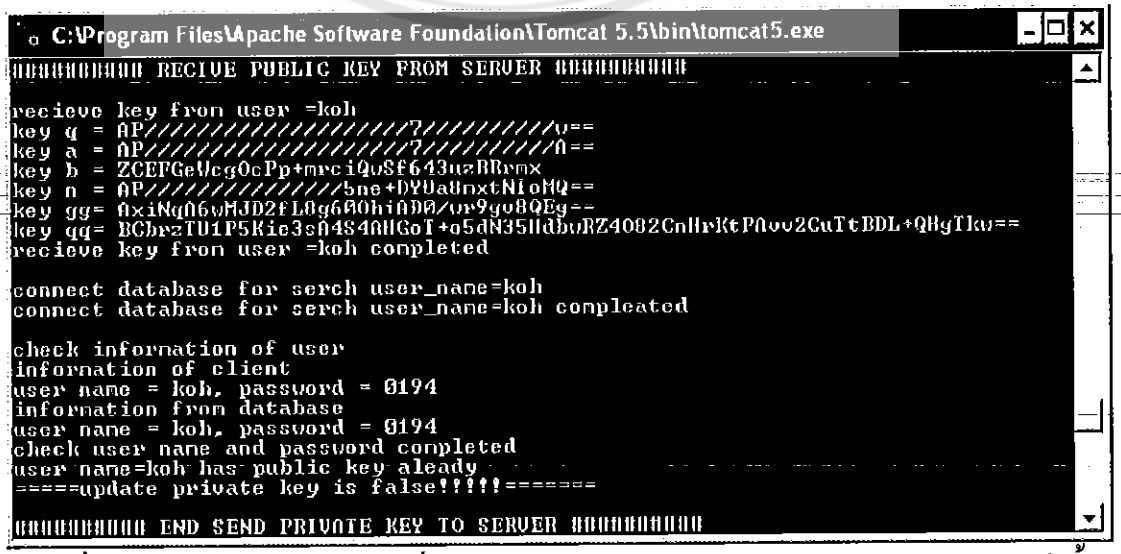
4.2 กรณี User name และ Password นี้มี Public / Private Key แล้ว

จากข้างบน User name ที่ชื่อ koh และมี Password เป็น 0194 มีสร้าง Public / Private Key แล้ว แต่มียังมี User name ที่ชื่อ koh และมี Password เป็น 0194 ส่ง Public Key เข้ามายัง Server อีก ซึ่งแสดงดังรูปที่ 4.10



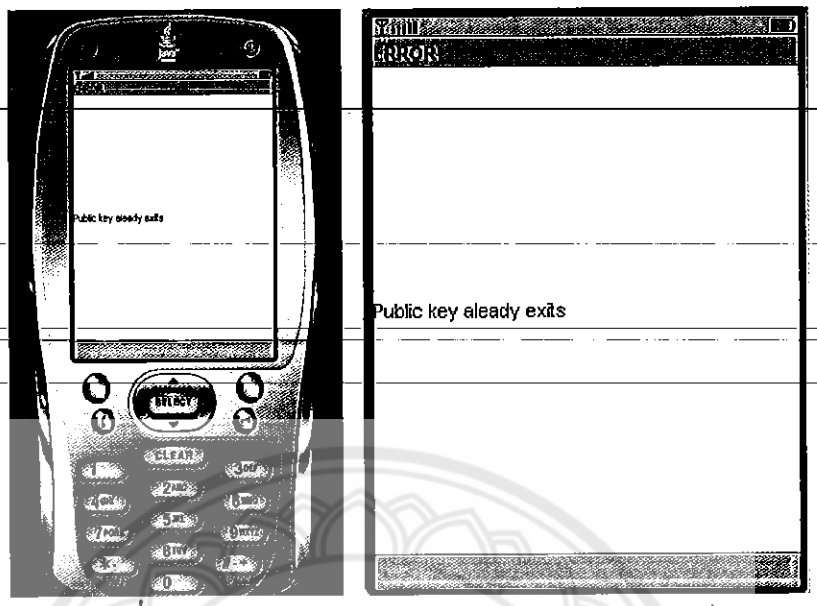
รูปที่ 4.10 เป็นการกรอก User name ที่ชื่อ koh และมี Password เป็น 0194 อีกครั้ง

ทางฝั่ง Server ก็จะทำการตรวจสอบ User name, Password ซึ่งถูกต้อง เสร็จแล้วทำการเช็ค Public Key ของ User ซึ่งปรากฏว่า User name นี้มี Public Key ในฐานข้อมูลของระบบแล้ว ทาง Server ก็จะตอบกลับไปยัง User ว่า User มี Public อยู่แล้ว ซึ่งแสดงได้ดังรูปที่ 4.11



รูปที่ 4.11 ไม่บันทึก Public Key เนื่องจากมี User name, Password ในฐานข้อมูลอยู่ก่อนหน้านี้

เมื่อทาง Server ตอบกลับว่า User มี Public Key อยู่แล้ว ก็จะปรากฏหน้าจอดังรูปที่ 4.12

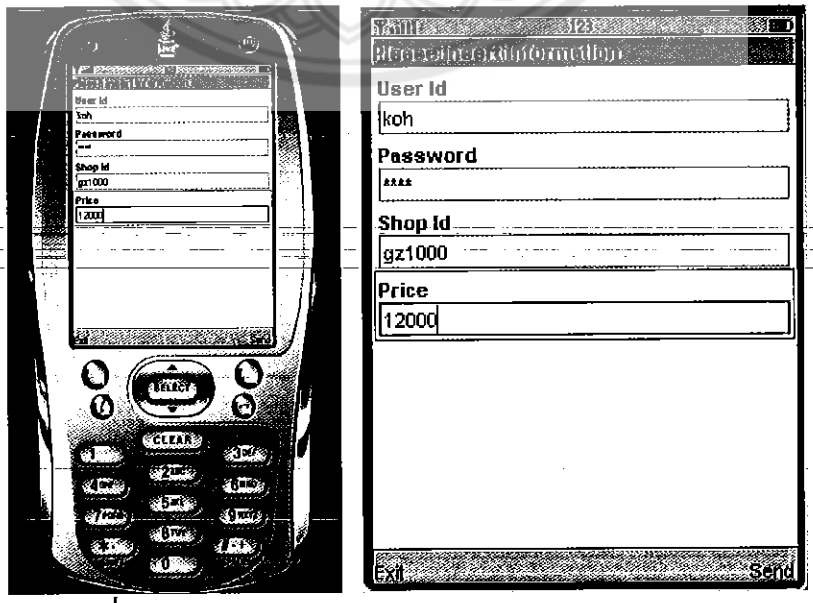


รูปที่ 4.12 Server ตอบกลับว่า User มี Public Key อยู่แล้ว

4.3 ขั้นตอนการจ่ายเงิน

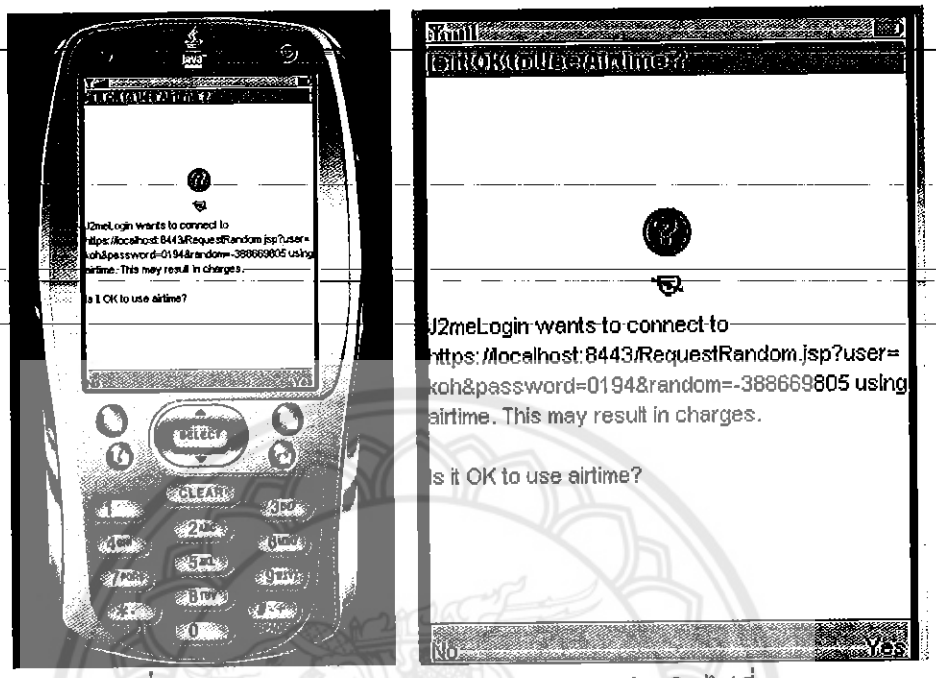
ขั้นตอนการจ่ายเงินนี้ จะเกิดขึ้นก็ต่อเมื่อ User ได้ทำการสร้าง Public Key / Private Key แล้วส่ง Public Key ไปเก็บยัง Server เสร็จแล้วเท่านั้น จึงจะปรากฏหน้าจอการจ่ายเงินขึ้น

ที่หน้าจอการจ่ายเงินจะมีช่องให้กรอกข้อมูลที่ประกอบไปด้วย User name (User Id), Password, Shop Id (รหัสร้านค้า) และ Price (ราคาของสินค้า หน่วยเป็นบาท) ซึ่ง User ต้องใส่ข้อมูลให้ครบ ซึ่งแสดงดังรูปที่ 4.13



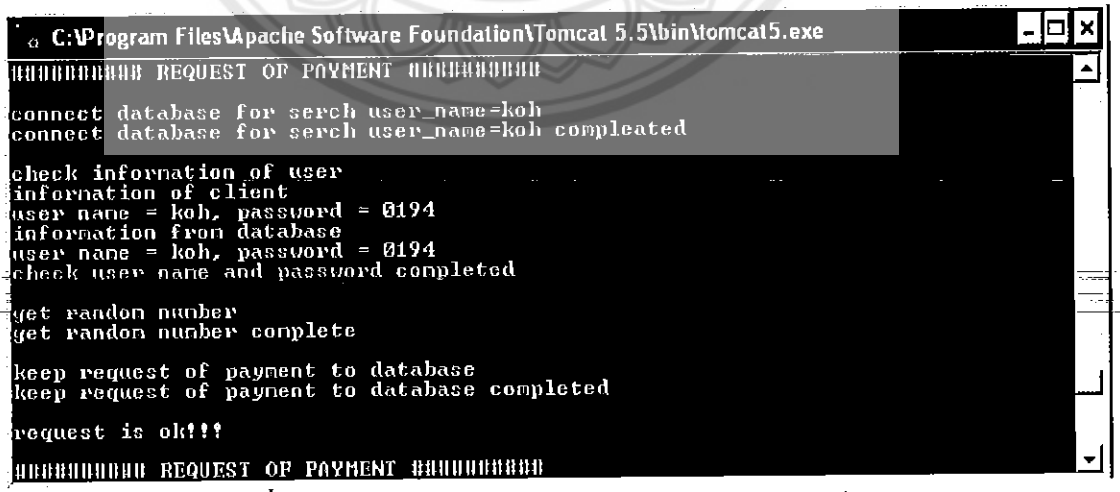
รูปที่ 4.13 กรอกข้อมูล User name, Password, Shop Id , Price

เมื่อใส่ข้อมูลครบแล้วกด Send โปรแกรมก็จะทำการส่ง Request การขอจ่ายเงินไปที่ Server ซึ่งแสดงดังรูปที่ 4.14



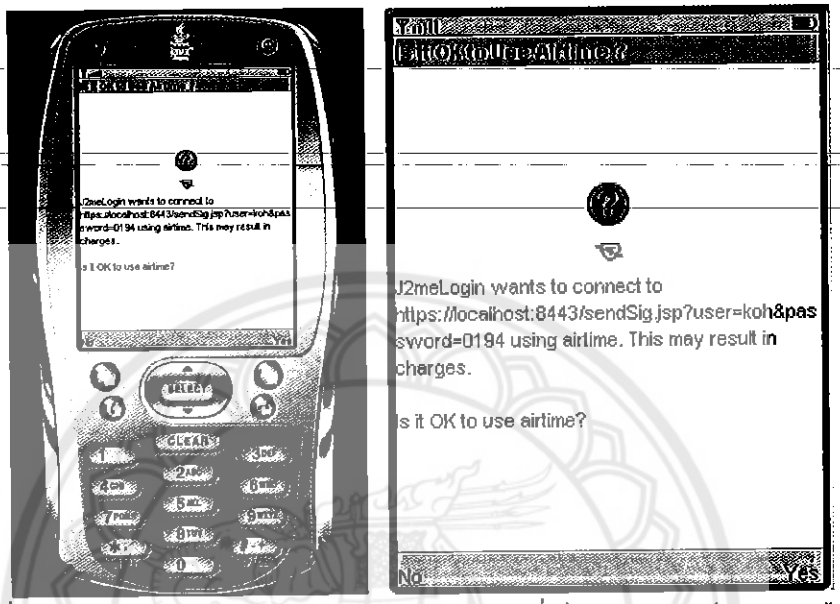
รูปที่ 4.14 โปรแกรมทำการส่ง Request การขอจ่ายเงินไปที่ Server

ทาง Server จะทำการเช็ค User name, Password ถ้าถูกต้อง ก็จะทำการเก็บข้อมูล Request การขอจ่ายเงินลงฐานข้อมูล แล้วตอบกลับไปยัง User ว่าการ Request เสร็จเรียบร้อยแล้ว ซึ่งแสดงดังรูปที่ 4.15



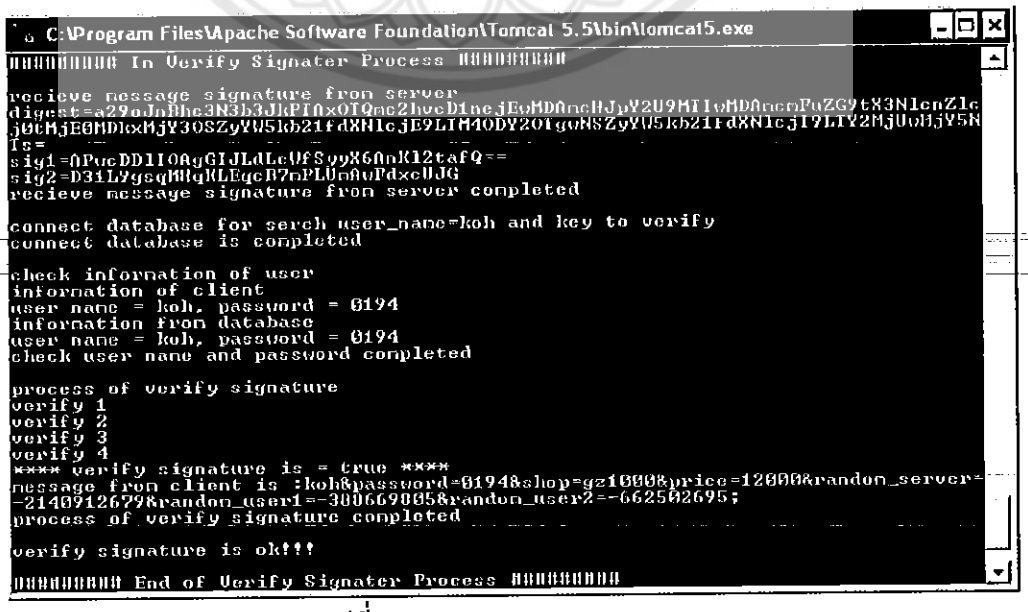
รูปที่4.15 การเก็บข้อมูล Request การขอจ่ายเงินลงฐานข้อมูล

ทาง ฝั่ง User เมื่อได้รับการตอบกลับมาว่า การ Request เสร็จเรียบร้อยแล้ว ก็จะทำการสร้าง Signature โดยใช้ Private Key ที่เก็บไว้ในฐานข้อมูลของมือถือ โดยใน Signature จะมีข้อมูลเป็น User name, Password, Shop id, Price เมื่อสร้างเสร็จแล้วก็ส่ง Signature ไปยัง Server ซึ่งแสดงดังรูปที่ 4.16



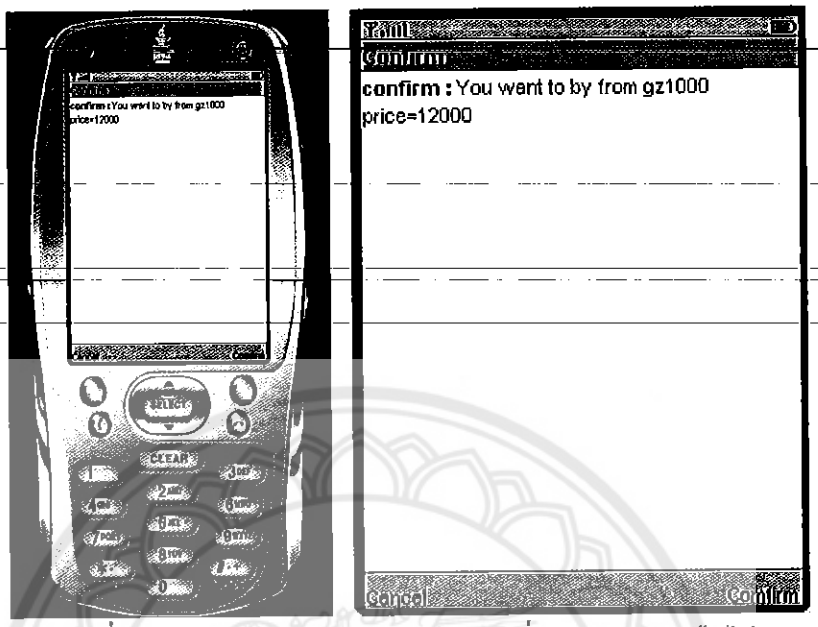
รูปที่ 4.16 การสร้าง Signature โดยใช้ Private Key ที่เก็บไว้ในฐานข้อมูลของมือถือ

ทาง Server ก็จะทำการตรวจสอบ Signature ถ้าขั้นตอนการตรวจสอบ Signature ถูกต้อง Server ก็จะตอบกลับไปทาง User ว่าการตรวจสอบ Signature ถูกต้อง และจะถาม User ว่าต้องการจ่ายเงินตามข้อมูลที่ส่งมาจริงหรือเปล่า ซึ่งแสดงดังรูปที่ 4.17



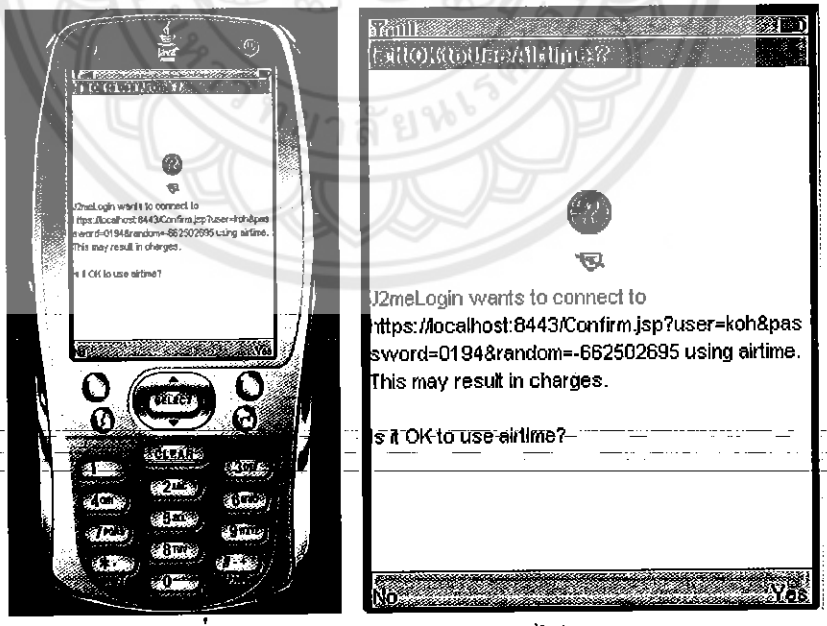
รูปที่ 4.17 การตรวจสอบ Signature

เมื่อ Server ตอบกลับมาว่าต้องการที่จะ Confirm หรือไม่ ก็จะปรากฏหน้าจอพร้อมด้วยข้อมูลดังรูปที่ 4.18



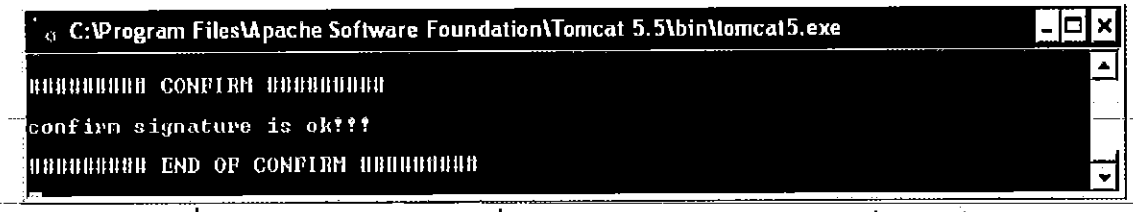
รูปที่ 4.18 Server ตอบกลับมาว่าต้องการที่จะ Confirm หรือไม่

เมื่อ User ต้องการ Confirm ก็จะทำการส่งการ Confirm ไปยัง Server ดังรูปที่ 4.19



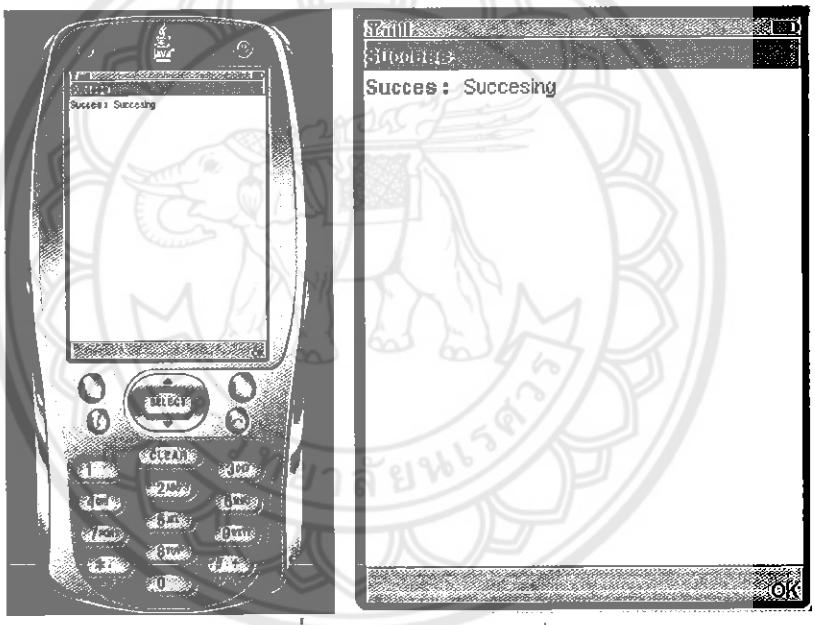
รูปที่ 4.19 ทำการส่งการ Confirm ไปยัง Server

เมื่อ Server ได้รับการ Confirm จาก User ก็จะเก็บข้อมูลการจ่ายเงินที่ User ส่งมาลงฐานข้อมูลของระบบ และ Server ตอบกลับไปที่ User ว่าการจ่ายเงินของ User เสร็จเรียบร้อยแล้ว รูปที่ 4.20



รูปที่ 4.20 Server ตอบกลับไปที่ User ว่าการจ่ายเงินของ User เสร็จเรียบร้อย

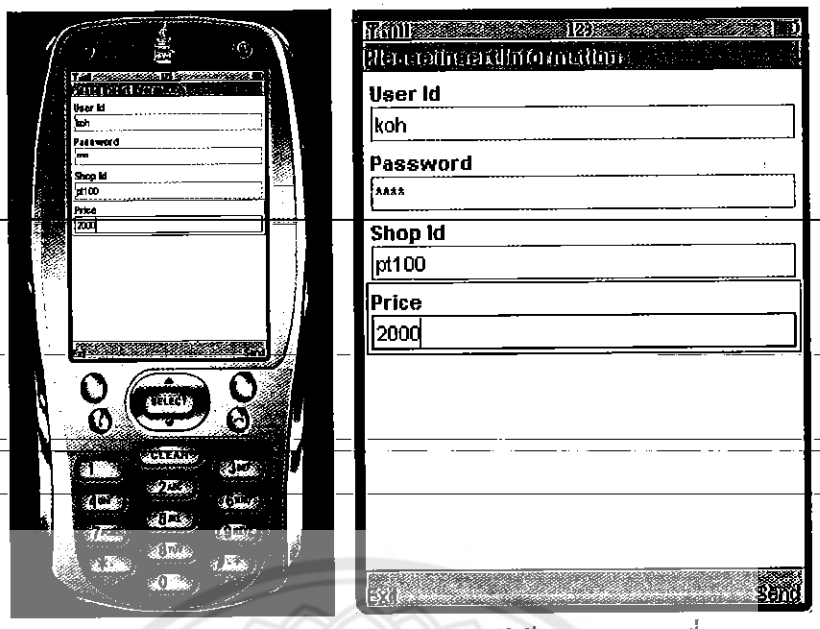
เมื่อ User ได้รับการตอบกลับจาก Server ก็จะปรากฏหน้าจอดังรูปที่ 4.21



รูปที่ 4.21 การจ่ายเงินสำเร็จ

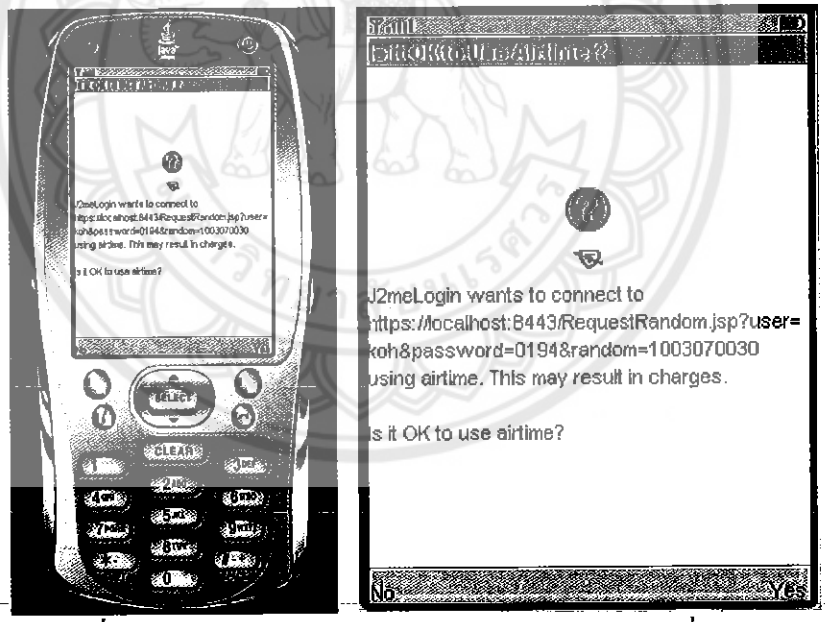
4.4 ขั้นตอนการจ่ายเงินโดยที่ Signature ผิด

ขั้นตอนนี้จะทดสอบโดยจะสร้าง User name ที่ชื่อว่า keng มี Password เป็น 0251 ลงในฐานข้อมูลแล้วทำการสร้าง Public Key / Private Key ส่งไปเก็บที่ Server โดยที่เราจะใช้ User name ที่ชื่อ koh มี Password เป็น 0194 แต่ใช้ Private Key ของ User name ที่ชื่อ keng ในการสร้าง Signature ซึ่งแสดงในรูปที่ 4.22



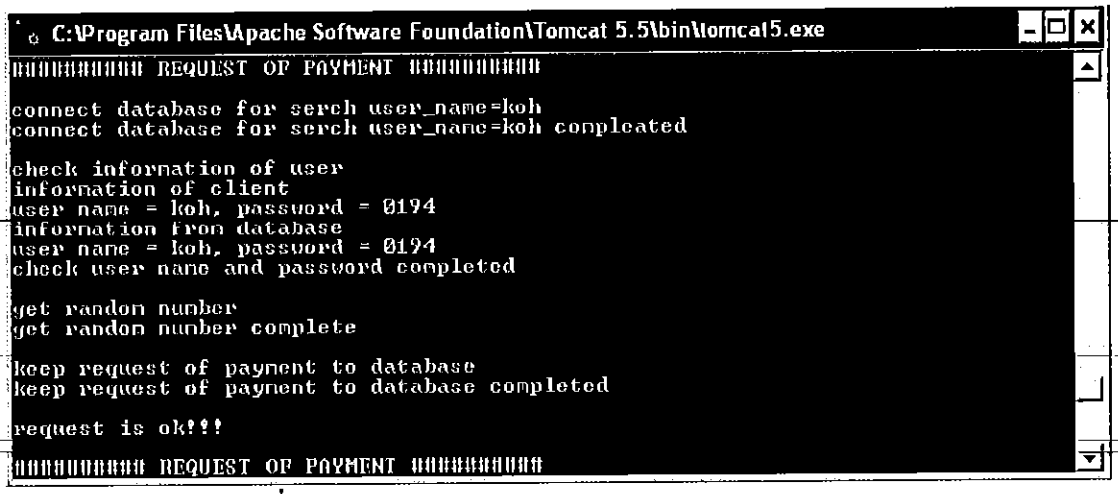
รูปที่ 4.22 ใช้ User name = koh, Password = 0194 แต่ใช้ Private Key ที่ User name = keng

ส่ง Request การขอจ่ายเงินไปยัง Server ดังรูปที่ 4.23



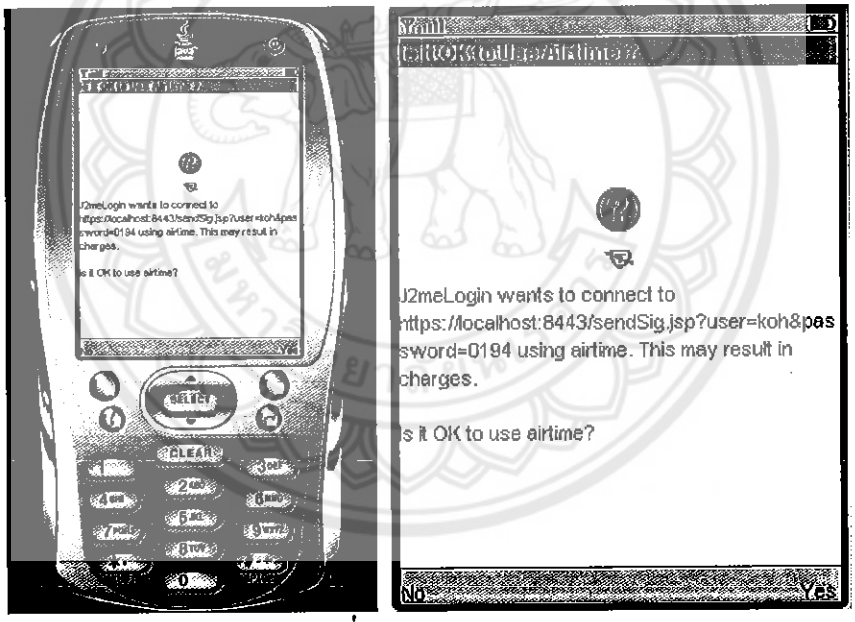
รูปที่ 4.23 การส่ง User name, Password, Public Key ไปที่ Server

ทาง Server ทำการตรวจสอบ Request การขอจ่ายเงิน เสร็จแล้วส่งการตอบกลับไปยัง Server ดังที่แสดงในรูปที่ 4.24



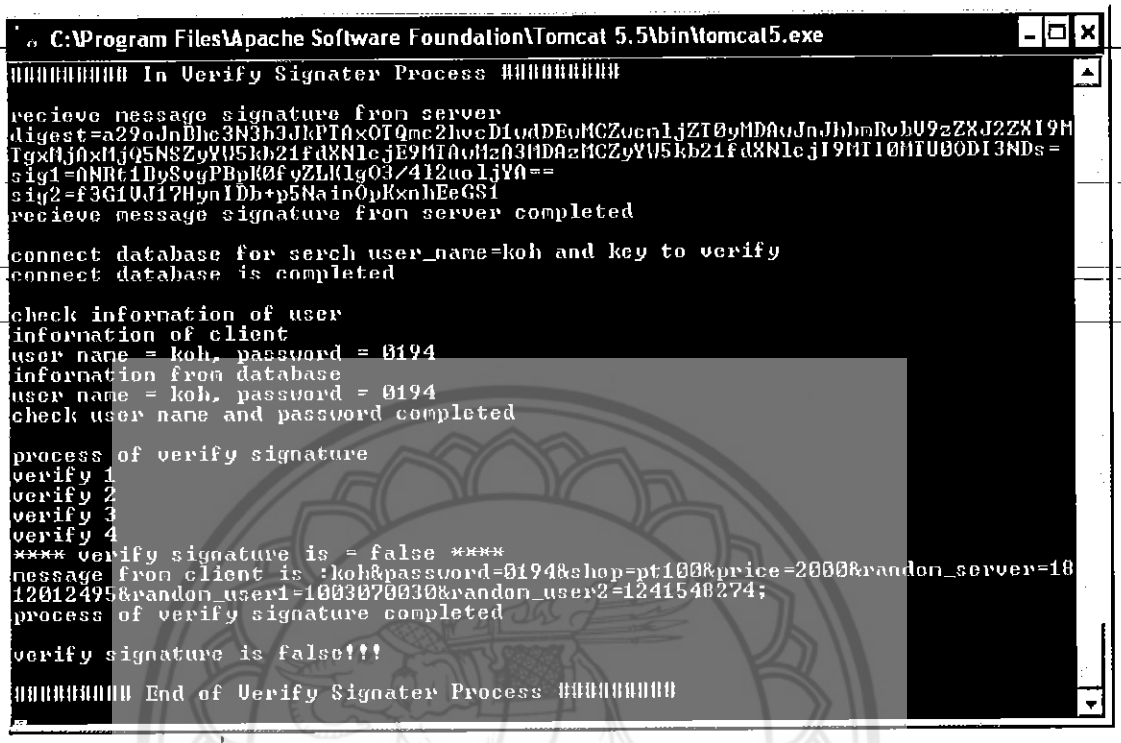
รูปที่ 4.24 Server ทำการตรวจสอบ Request การขอจ่ายเงิน

ทาง User ใช้ User name ที่ชื่อ koh แต่ใช้ Private Key ของ User name ที่ชื่อ keng ในการสร้าง Signature เมื่อเสร็จแล้วส่ง Signature ไปยัง Server ซึ่งแสดงในรูปที่ 4.25



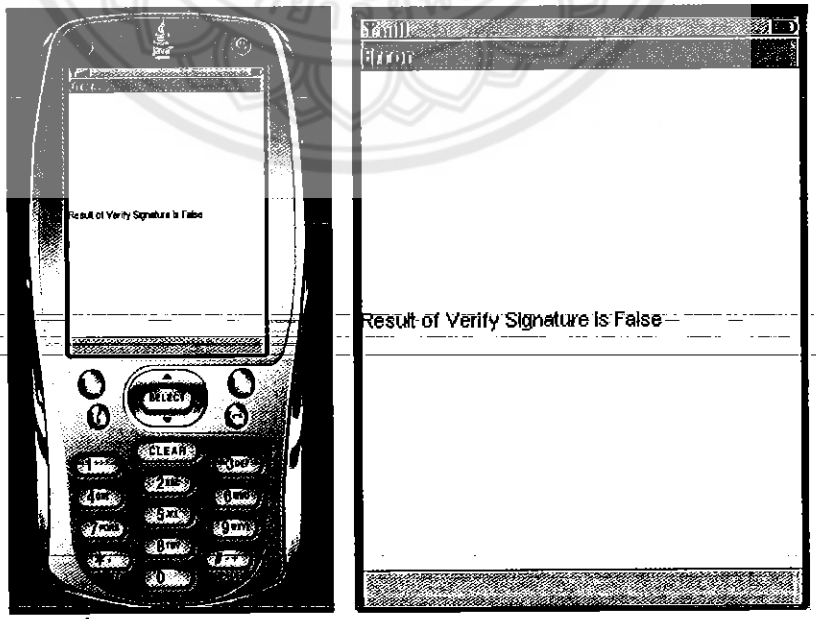
รูปที่ 4.25 ส่ง Signature ไปยัง Server

ทาง Server ทำการตรวจสอบ Signature และพบว่าการตรวจสอบ Signature ผิด แล้วจึงตอบกลับไปยัง User ว่าการตรวจสอบ Signature ผิด ซึ่งแสดงดังรูปที่ 4.26



รูปที่ 4.26 Server ตอบกลับไปยัง User ว่าการตรวจสอบ Signature ผิด

เมื่อ User ได้รับการตอบกลับจาก Server ว่าการตรวจสอบ Signature ผิดพลาดก็จะปรากฏหน้าจอดังรูปที่ 4.27



รูปที่ 4.27 Server ตอบกลับมาว่าการตรวจสอบ Signature ผิดพลาด

บทที่ 5
สรุปผล

ปัจจุบันการใช้จ่ายผ่านบัตรเครดิตเป็นที่ได้รับความนิยมมากในระดับหนึ่ง แต่บัตรเครดิตก็ยังมีปัญหาในเรื่องความปลอดภัย ผนวกกับในปัจจุบันโทรศัพท์เข้ามามีบทบาทอย่างมากในชีวิตประจำวันของทุกคน ผู้จัดทำจึงเห็นว่าน่าจะนำโทรศัพท์มือถือมาเป็นเครื่องมือในการจ่ายค่าสินค้าและบริการ อันมีความสะดวก รวดเร็ว และปลอดภัย

โครงงานนี้ได้ทำการพัฒนาระบบจ่ายเงินอิเล็คทรอนิกส์ผ่านมือถือ ซึ่งในการพัฒนาจะยึดหลักสำคัญคือ ความปลอดภัย ความสะดวกสบายในการใช้งาน และมีความรวดในการประมวลผล เพื่อตอบสนองหลักการดังกล่าวผู้จัดทำจึงได้นำโปรโตคอล SSL (Secure Socket Layer) ซึ่งเป็นโปรโตคอลที่มีความปลอดภัยและได้รับการยอมรับเป็นมาตรฐานเป็นระดับสากลใช้กันอย่างแพร่หลายทั่วโลก และนำ ECDSA (Elliptic Curve Digital Signature Algorithm) มาใช้เป็น Algorithm ในการเข้ารหัสการลงลายมือชื่อดิจิตอลเพื่อให้ระบบสามารถตรวจสอบและยืนยันตัวตนที่แท้จริงได้ ทั้งนี้ ECDSA ยังสามารถป้องกันการปฏิเสธความรับผิดชอบของผู้ใช้ได้

5.1 วิเคราะห์ผลการทดลอง

จากผลการทดลองในบทที่ 4 สรุปได้เป็นกรณีต่างๆ โดยแต่ละกรณีจะตรวจสอบเป็นลำดับขั้น ดังนี้

1. **User name และ Password** : เป็นขั้นตอนการตรวจสอบว่า User กรอกข้อมูลมาถูกหรือไม่
2. **Key** : จากข้อ 1 เมื่อตรวจสอบว่า User name และ Password ถูกต้องระบบจะทำการตรวจสอบต่อไปว่าในฐานข้อมูลมี Public/Private Key ของ User เก็บไว้หรือไม่ ถ้าไม่ระบบจะเก็บใหม่ในฐานข้อมูลของ Server
3. **Check Key** : จากข้อ 2 ถ้าหากว่า User นั้นมี Public/Private Key เก็บใน Server แล้วแต่มี User อื่นพยายาม Login ระบบจะไม่ยอมรับ Key ตัวใหม่
4. **Signature** : เป็นการตรวจสอบในขั้นตอนการจ่ายเงิน ถ้า Signature ที่สร้างขึ้นไม่ได้สร้างมาจาก Private Key ของ User ตัวจริง เมื่อระบบตรวจสอบพบระบบจะปฏิเสธการจ่ายเงินนั้น

ทั้งนี้หากไม่ผ่านการตรวจสอบในขั้นตอนใดขั้นตอนหนึ่งระบบจะยกเลิกการติดต่อทันที

ตารางที่ 5.1 เปรียบเทียบผลการทดลอง

กรณีการทดลอง	User name และ Password	Key	Check Key	Signature	Result
1. กรอก user name หรือ password ผิด	ผิด	-	-	-	ไม่สำเร็จ
2. User ยังไม่มี Public/Private Key	ถูก	ไม่มี	-	-	ไม่สำเร็จ
3. user name มี Public/Private Key แล้ว แต่มีการส่ง Public/Private Key อีก	ถูก	มี	ผิด	-	ไม่สำเร็จ
4. จ่ายเงินโดย Signature ไม่ถูกต้อง	ถูก	มี	ถูก	ผิด	ไม่สำเร็จ
5. กรอกข้อมูลถูกต้องสมบูรณ์	ถูก	มี	ถูก	ถูก	สำเร็จ

กรณี 1 ในการใช้งานหากผู้ใช้กรอก User name หรือ Password ไม่ถูกต้อง จะไม่สามารถเข้าสู่ระบบได้ และระบบจะให้กลับไปกรอก User name และ Password ใหม่

กรณี 2 เมื่อ User กรอก User name และ Password ถูกต้องแต่ในฐานข้อมูลไม่มี Public/Private Key ระบบจะให้ User ทำการสร้าง Public/Private Key แล้วเก็บ Private Key ไว้ที่มือถือ และส่ง Public Key ไปเก็บไว้ที่ Server

กรณี 3 เมื่อในฐานข้อมูลมี Public Key อยู่แล้วแต่มี User อื่นพยายาม Login เข้าสู่ระบบเพื่อทำการสร้าง Key ใหม่อีกครั้ง ระบบจะไม่ยอมรับ Key ที่สร้างขึ้น

กรณี 4 เมื่อมีการ Login ด้วย User name และ Password ที่ถูกต้องและในฐานข้อมูลของเซิร์ฟเวอร์มี Public Key อยู่แล้ว แต่ในขั้นตอนการจ่ายเงิน ถ้า Signature ที่สร้างขึ้นไม่ได้สร้างมาจาก Private Key ของ User เอง เมื่อระบบตรวจสอบพบระบบจะปฏิเสธการจ่ายเงินนั้น

กรณี 5 เมื่อมีการ Login ด้วย User name และ Password ที่ถูกต้อง ในฐานข้อมูลของเซิร์ฟเวอร์มี Public Key อยู่แล้ว และมีการสร้าง Signature จาก Private ที่ถูกต้อง ระบบจะยอมรับและดำเนินการตามคำร้องขอของ User นั้นๆ

จากทั้ง 5 กรณี จะเห็นได้ว่าในระบบจ่ายเงินผ่านมือถือที่ได้พัฒนาขึ้นนี้ ในทุกขั้นตอนของการจ่ายเงินมีการตรวจสอบตลอดเวลาว่าข้อมูลที่ใช้ทุกอย่างเป็นข้อมูลที่แท้จริงจากผู้ใช้หรือไม่ หากไม่ใช่ระบบก็จะปฏิเสธการทำงาน อีกทั้งในระหว่างการส่งข้อมูลระหว่างมือถือกับเซิร์ฟเวอร์ ระบบก็ใช้โปรโตคอลในการรับส่งที่ปลอดภัยเป็นมาตรฐานสากล คือ ใช้โปรโตคอล SSL และในแต่ละขั้นตอนของการส่งข้อมูล ข้อมูลที่ถูกส่งผ่านระหว่างมือถือกับเซิร์ฟเวอร์ก็มีการเข้ารหัสโดยอัลกอริทึมการลงลายมือชื่อดิจิตอล ECDSA อันเป็นมาตรฐานที่ยอมรับเป็นมาตรฐานสากลทั่วโลก

และเป็นอัลกอริทึมที่เหมาะสมอย่างมากที่จะใช้ในการลงลายมือชื่อดิจิตอลผ่านทางโทรศัพท์มือถือ เนื่องจากในขนาดความปลอดภัยที่เทียบเท่ากันนั้นความยาวคีย์ของ ECDSA มีขนาดน้อยกว่ามาก (จากตารางที่ 5.2) อันเป็นการทำให้เครื่องโทรศัพท์มือถือไม่ต้องใช้เวลาในการคำนวณมาก ตอบสนองความต้องการของผู้ใช้ที่ต้องการความสะดวก รวดเร็วและความปลอดภัยในการใช้งานได้เป็นอย่างดี

ตารางที่ 5.2 เปรียบเทียบความยาวของ Key และ CA's Signature (bytes) ตามมาตรฐาน ANSI X9.62

	RSA	DSA	ECDSA
User's Public Key	128	128	21
CA's Signature	128	40	41
Total	256	168	62

ตารางที่ 5.3 เปรียบเทียบข้อดีและความปลอดภัยระหว่างระบบจ่ายเงินผ่านมือถือกับบัตรเครดิต

ระบบจ่ายเงินผ่านมือถือ	บัตรเครดิต
1. เมื่อมือถือหาย ผู้ที่เก็บได้ไม่สามารถนำไปจ่ายเงินผ่านระบบจ่ายเงินได้ เนื่องจากไม่ทราบ User name กับ Password	1. เมื่อบัตรเครดิตหาย ผู้ที่เก็บได้สามารถโกงได้หากยังไม่มีการแจ้งอายัด
2. ในการจ่ายเงินจะใช้โทรศัพท์มือถือซึ่งคนส่วนใหญ่คนพกพาติดตัวเป็นปกติอยู่แล้ว	2. ในการจ่ายเงินจำเป็นต้องพกบัตรเครดิตติดตัวทุกครั้ง
3. ร้านค้าที่เป็นสามาชิกระบบจ่ายเงินผ่านมือถือมีเพียงโทรศัพท์มือถือก็สามารถใช้ระบบจ่ายเงินได้	3. ร้านค้าที่เป็นสมาชิก จำเป็นต้องมีเครื่องรูดบัตรติดตั้งในร้าน ซึ่งเป็นการสิ้นเปลืองในแง่การลงทุน

จากตารางที่ 5.3 แสดงถึงข้อดีของระบบจ่ายเงินผ่านมือถือในแง่การใช้งานต่างๆ ทำให้เห็นว่าการนำระบบจ่ายเงินผ่านมือถือไปใช้นั้น มีความสะดวกและปลอดภัยกว่าการใช้บัตรเครดิต

5.2 ปัญหาและอุปสรรค

ในระหว่างการทำโครงงานนี้ได้ประสบปัญหาทางด้านการออกแบบระบบเป็นส่วนใหญ่ ได้แก่

5.2.1 ระบบจ่ายเงินอิเล็กทรอนิกส์โดยใช้โทรศัพท์มือถือเป็นระบบที่มีผู้คิดค้นอยู่แล้วในปัจจุบันซึ่ง การที่จะออกแบบระบบขึ้นมาใหม่จำเป็นต้องมีคุณสมบัติหรือข้อดีที่แตกต่างไปจากระบบเดิมทำให้เกิดความยากในการออกแบบระบบ

5.2.2 เมื่อทำการคิดระบบขึ้นมาใหม่โดยเน้นถึงด้านความปลอดภัยแล้ว ปรากฏว่าระบบที่ได้จะมีการใช้เวลาประมวลผลนานเกินไป ทำให้ไม่สะดวกต่อการใช้งาน แต่เมื่อได้ทำการปรับเน้นทางด้านความสะดวกรวดเร็วของการประมวลผล ปรากฏว่าระบบที่ได้นั้นไม่ปลอดภัยเท่าที่ควร ทำให้ระบบมีโอกาสถูกโจมตีจากทางด้านอื่นๆ

5.3 แนวทางแก้ไขปัญหา

5.3.1 จากปัญหาในข้อ 5.2.1 มีวิธีแก้ปัญหา โดยจากการศึกษาและเปรียบเทียบถึงข้อดีและข้อเสียของระบบที่ได้ออกแบบกับระบบที่ใช้กันจริง ซึ่งจะทำให้เห็นแนวทางที่จะนำ Algorithm ที่มีในปัจจุบันมาใช้ในการเพิ่มหรือลดข้อเสียเพื่อให้ระบบมีข้อดีที่มากขึ้น

5.3.2 จากปัญหาในข้อ 5.2.2 มีวิธีแก้ปัญหา โดยการนำ Algorithm หนึ่งมาใช้ ซึ่งจากการศึกษาเปรียบเทียบ Algorithm แล้วพบว่าในระดับความปลอดภัยระดับเดียวกันแล้ว Algorithm ECDSA ใช้จำนวน Key ที่น้อยกว่า จึงทำให้ใช้เวลาในการประมวลผลน้อยกว่า ดังนั้นจึงตัดสินใจใช้ Algorithm ECDSA

5.4 ข้อเสนอแนะ

จากการพัฒนาระบบจ่ายเงินอิเล็กทรอนิกส์โดยใช้โทรศัพท์มือถือนั้น ได้มีการพัฒนาโดยใช้ ภาษา JSP, J2ME, ฐานข้อมูล MySQL และ Apache Tomcat Server ซึ่งเป็นภาษาและเครื่องมือที่เป็นที่นิยมของโปรแกรมเมอร์จึงง่ายต่อการศึกษาค้นคว้าเนื่องจากมีข้อมูลอยู่ทั่วไปในอินเตอร์เน็ต การใช้งานเครื่องมือพัฒนาเหล่านี้ทำให้ระบบที่ได้สะดวกต่อการปรับปรุงแก้ไข และเข้าใจได้ง่าย สามารถทำให้ผู้อื่นที่สนใจโครงงานนี้นำแนวคิดไปพัฒนาระบบจ่ายเงินอิเล็กทรอนิกส์แบบอื่นๆ ต่อได้ ซึ่งอาจพัฒนาให้เป็นระบบจ่ายเงินที่มีขนาดใหญ่ มีความซับซ้อน สะดวก และปลอดภัยมากยิ่งขึ้น โดยในส่วนของระบบที่ควรจะมีการพัฒนาเพิ่มเติมเพื่อให้ระบบมีความสมบูรณ์ ได้แก่

5.4.1 ในส่วนของระบบในการติดต่อกับธนาคาร โดยการพัฒนาให้มีส่วนของการติดต่อกับธนาคาร เช่น สามารถตรวจสอบยอดเงินคงเหลือในบัญชีได้ มีการตรวจสอบยอดเงินคงเหลือในบัญชีว่าพอที่จะทำรายการหรือไม่

5.4.2 การ Generate Piblic/Private Key ใหม่ในกรณีที่มีการเปลี่ยนเครื่องโทรศัพท์มือถือ เนื่องจากระบบที่ได้พัฒนาขึ้นนี้มีการเก็บ Private Key ไว้ในเครื่องซึ่งยังไม่สามารถย้ายได้ ระบบที่พัฒนาขึ้นโทรศัพท์มือถือ 1 เครื่อง สามารถรองรับสมาชิกได้เพียง 1 สมาชิกเท่านั้น

5.4.3 ควรพัฒนาให้มีการรองรับสมาชิกได้หลายๆ สมาชิก (ID) เพื่อรองรับในกรณีที่มีการใช้บริการหลายๆ ผู้ให้บริการ (Server)

เอกสารอ้างอิง

[1] กาญจนา ตันวิสุทธิ์. **"เขียนเกมและโปรแกรมบนมือถือ J2ME."** นนทบุรี : ไอดีซีฯ. 2547.

[2] ทรงเกียรติ ภาวดี. **"เก่ง J2ME ให้ครบสูตร."** กรุงเทพฯ : บริษัท วิตตี้ กรุ๊ป จำกัด. 2546.

[3] Sun Microsystem. **"Java 2 Platform, Micro Edition (J2ME)."** [Online]. Available: http://Java.sun.com/j2me/index.jsp. 2548.

[4] Thai Programmer Web. **"Mobile J2ME."** [Online]. Available: http://www.thai-programmer.com/?DPage=90700103#jme003. 2548.

[5] 9'M. **"บทที่ 2 Introduction to J2ME."** [Online]. Available : http://www.sourcecode.in.th/j2me/lession2.asp. 2548.

[6] ศูนย์พัฒนาพานิชย์อิเล็กทรอนิกส์ (Electronic Commerce Resorce Center). **"3. การรักษาความปลอดภัย (ภัยคุกคาม และเทคโนโลยีการป้องกัน)."** [Online]. Available : http://www.ecommerce.or.th/faqs/faq3-1.html#1. 2548.

[7] สิริพร จิตต์เจริญธรรม, เสาวภา ปานจันทร์ม, เลอศักดิ์ ลิ้มวิวัฒน์กุล. **"ความรู้เบื้องต้นเกี่ยวกับการพิสูจน์ตัวตน."** [Online]. Available :. http://thaicert.nectec.or.th/paper/authen/authentication_guide.php#define. 2548.

[8] ศูนย์พัฒนาพานิชย์อิเล็กทรอนิกส์ (Electronic Commerce Resorce Center). **"ระบบชำระเงินยุคใหม่ของไทย."** [Online]. Available : http://www.ecommerce.or.th/nceb2002/paper/38-thai-payment.pdf. 2548.

[9] อนุชิต อนุชิตานุกูล, สมเกียรติ ตั้งกิจวานิชย์. **"เงินอิเล็กทรอนิกส์กับนโยบายการเงินและการฟอกเงิน."** [Online]. Availale : http://www.info.tdri.or.th/reports/published/a106/chapter1.pdf. 2548.

[10] ณรงค์ชัย นิมิตบุญอนันต์. **Computer Security for E-Commerce.** กรุงเทพฯ : SUM SYSTEM COMPANY LIMITED. 2542.

[11] มณีโชติ สมานไทย. **"คู่มือการออกแบบฐานข้อมูลและภาษา SQL ฉบับผู้เริ่มต้น".** กรุงเทพมหานคร: โนสุทธาการพิมพ์. 2546.

[12] Y.Jaruwan. **"ความรู้ทั่วไปเกี่ยวกับระบบฐานข้อมูล."** [Online]. Available: http://www.chandra.ac.thofficeictdocumentitit04page01.html. 2549.

[13] ทีมงานชาวลำปาง. **"การพัฒนาโปรแกรมด้วย JSP."** [Online]. Available: http://www.thaiall.com/internet/internet09.htm. 2549.

เอกสารอ้างอิง (ต่อ)

[14] itmelody. **"JSP - Java Server Page."** [Online]. Available: http://www.itmelody.com/tu/introjsp.htm. 2549.

[15] ทินกร วัฒนเกษมสกุล. **"คัมภีร์ JSP."** นนทบุรี : เคทีพี. 2547.

[16] Apache Tomcat. **"The Apache Software Foundation."** [Online]. Available: http://tomcat.apache.org. 2549.

[17] Don D. Johnson. **"ECC, Future Resiliency and High Security Systems."** [Online]. Available: http://www.comms.scitech.susx.ac.uk/fft/crypto/ECCFut.pdf. 2549.

[18] Don D. Johnson and Alfred Menezes and Scott Vanstone. **"The Elliptic Curve Digital Signature Algorithm (ECDSA)."** [Online]. Available: http://www.comms.scitech.susx.ac.uk/fft/crypto/ecdsa.pdf. 2549.

ประวัติผู้เขียนโครงงาน

ชื่อ นายปัญญา แต่งงาม

ภูมิลำเนา 566 หมู่ 15 ตำบลบางระกำ อำเภอบางระกำ
จังหวัดพิษณุโลก 65140

ประวัติการศึกษา

- จบมัธยมศึกษาจากโรงเรียนบางระกำวิทยศึกษา จ.พิษณุโลก
- ปัจจุบันกำลังศึกษาอยู่ระดับปริญญาตรีชั้นปีที่ 4
  สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์
  มหาวิทยาลัยนเรศวร

E-mail: **panya_m2001@hotmail.com**

ชื่อ นายสุเมธ สงแผน

ภูมิลำเนา 176/2 หมู่ 7 ตำบลท่านางงาม อำเภอบางระกำ
จังหวัดพิษณุโลก 65140

ประวัติการศึกษา

- จบมัธยมศึกษาจากโรงเรียนจ่านกร้อง จ.พิษณุโลก
- ปัจจุบันกำลังศึกษาอยู่ระดับปริญญาตรีชั้นปีที่ 4
  สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์
  มหาวิทยาลัยนเรศวร

E-mail: **koh_cpe@hotmail.com**

ชื่อ นายเอนกพงษ์ อินไชยเทพ

ภูมิลำเนา 6/26 ถนนธรรมบูชา ตำบลในเมือง อำเภอเมือง
จังหวัดพิษณุโลก 65000

ประวัติการศึกษา

- จบมัธยมศึกษาจากโรงเรียนจ่านกร้อง จ.พิษณุโลก
- ปัจจุบันกำลังศึกษาอยู่ระดับปริญญาตรีชั้นปีที่ 4
  สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์
  มหาวิทยาลัยนเรศวร

E-mail: **kobbi_555@hotmail.com**