โปรแกรมปลั๊กอินบน NetBeans และ Eclipse เพื่อตรวจสอบและแก้ไขโค้ดพีเอชพีที่เป็นช่องโหว่จากการโจมตีแบบ SQL Injection

A Plug-in Program on NetBeans and Eclipse to Detect and Correct SQL Injection Code for PHP

วรมันต์ อาชะเมตรา

ณัฏฐ์ ฑีฑิวรชาติ

โครงงานนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต

สาขาวิชาวิทยาการคอมพิวเตอร์

คณะวิทยาศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2556

โปรแกรมปลั๊กอินบน NetBeans และ Eclipse เพื่อตรวจสอบและแก้ไขโค้ดพีเอชพีที่เป็นอันตรายต่อการโจมตีแบบ SQL Injection

A Plug-in Program on NetBeans and Eclipse to Detect and Correct SQL Injection Code for PHP

วรมันต์ อชะเมตรา
สัณห์ สิทธิวรชาติ

โครงงานพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต
สาขาวิชาวิทยาการคอมพิวเตอร์
คณะวิทยาศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2556

A Plug-in Program on NetBeans and Eclipse to Detect and Correct SQL Injection Code for PHP

WARAMUN ACHAMETRA

SUN SITTIWORRACHAT

A SPECIAL PROJECT SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF BACHELOR OF SCIENCE

IN COMPUTER SCIENCE

FACULTY OF SCIENCE

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

ACADEMIC YEAR 2013

หัวข้อปัญหาพิเศษ	โปรแกรมปลั๊กอินบน Netbeans และ Eclipse เพื่อตรวจสอบและแก้ไขโค้ดพีเอชพีที่เป็นอันตรายต่อการโจมตีแบบ SQL Injection A Plug-in Program on Netbeans and Eclipse to Detect and Correct SQL Injection Code for PHP
ชื่อนักศึกษา	นายวรมันต์ อชะเมตรา 53051066 นายสัณห์ สิทธิวรชาติ 53051101
ปริญญา	วิทยาศาสตรบัณฑิต
สาขาวิชา	วิทยาการคอมพิวเตอร์
อาจารย์ที่ปรึกษา	ดร.รุ่งรัตน์ เวียงศรีพนาวัลย์

คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง อนุมัติให้โครงงานพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์ ประจำปีการศึกษา 2556

คณะกรรมการสอบ	ลายมือชื่อ
ผศ.ดร.นันทิกา เบญจเทพานันท์ (ประธานกรรมการ)	
ผศ.สิริลักษณ์ อนันต์สถิตย์สิน (กรรมการ)	
ดร.รุ่งรัตน์ เวียงศรีพนาวัลย์ (กรรมการและอาจารย์ที่ปรึกษา)	

หัวข้อปัญหาพิเศษ	โปรแกรมปลั๊กอินบน NetBeans และ Eclipse เพื่อตรวจสอบและแก้ไขโค้ดพีเอชพีที่เป็นอันตรายต่อการโจมตีแบบ SQL Injection A Plug-in Program on NetBeans and Eclipse to Detect and Correct SQL Injection Code for PHP
ชื่อนักศึกษา	นายวรมันต์ อชะเมตรา 53051066 นายสัณห์ สิทธิวรชาติ 53051101
ปริญญา	วิทยาศาสตรบัณฑิต
สาขาวิชา	วิทยาการคอมพิวเตอร์
ปีการศึกษา	2556
อาจารย์ที่ปรึกษา	ดร.รุ่งรัตน์ เวียงศรีพนาวัลย์

บทคัดย่อ

โครงงานพิเศษนี้ได้ทำการพัฒนาโปรแกรมปลั๊กอินใน NetBeans และ Eclipse เพื่อช่วยผู้พัฒนาโปรแกรม PHP ในการตรวจสอบโค้ดที่เป็นอันตรายต่อการโจมตีประเภท SQL Injection ซึ่งเป็นการโจมตีที่ได้รับความนิยมในอันดับแรกๆ ของการโจมตีบนเว็บแอพพลิเคชัน โดยโปรแกรมแบ่งออกเป็นสามส่วนหลัก คือ ส่วนการตรวจสอบซอร์สโค้ด ส่วนการแก้ไขโค้ด และส่วนการแนะนำเนื้อหา ในส่วนของการตรวจสอบซอร์สโค้ดโปรแกรมใช้ Pixy (โปรแกรมวิเคราะห์โค้ดแบบ static) ในการตรวจสอบโค้ดของไฟล์ที่เข้าถึงฐานข้อมูล MySQL และ ใช้ส่วนของโปรแกรมที่เพิ่มเข้าไปในการตรวจสอบโค้ดอันตรายของไฟล์ประเภท PHP Data Object (PDO) ในส่วนของการแก้ไขโค้ด เมื่อพบโค้ดที่มีช่องโหว่โปรแกรมจะทำการแก้ไขโค้ดที่มีช่องโหว่นั้นโดยสร้างไฟล์ใหม่ให้ ผู้พัฒนาโปรแกรมจึงไม่ต้องแก้ไขโค้ดด้วยตนเอง และในส่วนของการแนะนำนั้น เนื้อหาประกอบไปด้วย ความหมาย ประเภท และ การป้องกัน SQL Injection จากการทดสอบกับไฟล์ตัวอย่าง พบว่าโปรแกรมปลั๊กอินสามารถตรวจสอบโค้ดที่มีการโจมตีโดยการเพิ่ม quote หรือ สัญลักษณ์การแสดงความคิดเห็น (comment และ inline comment) ลงไปในชุดคำสั่งได้

Title	A Plug-in Program on NetBeans and Eclipse to Detect and Correct SQL Injection Code for PHP		
Students	Mr. Waramun	Achametra	53051066
	Mr. Sun	Sittiworrachat	53051101
Degree	Bachelor of Science		
Major Program	Computer Science		
Academic Year	2013		
Advisor	Dr. Rungrat Wiangsripanawan		

ABSTRACT

The aim of this special project is to develop a plug-in on NetBeans and Eclipse to detect SQL injection's codes on PHP files. SQL Injection is always one of the top web application attacks. The major cause is that web developers lack knowledge about this attack and its preventions. The program consists of three main parts: the Scan part, the Correction part and the Help part. The Scan part not only uses Pixy, a static code analysis on PHP, to detect MySQL vulnerable codes but also adds the new feature to detect PHP Data Object (PDO) vulnerable codes. The Correction part comes together with the Scan part to fix the malicious codes on the new file so that the web programmers have no need to fix the code themselves. The Help part provides the SQL Injection knowledge: meaning, types, causes and preventions. The testing result showed that our program can detect and correct the SQL Injection attacks that caused by inserting quote, comment and inline comment.

กิตติกรรมประกาศ

โครงงานพิเศษเล่มนี้สำเร็จลุล่วงไปได้ด้วยดี เนื่องจากผู้จัดทำได้รับความช่วยเหลือจากบุคคลผู้มีพระคุณหลายท่าน ดังนี้

ขอขอบพระคุณ ดร.รุ่งรัตน์ เวียงศรีพนาวัลย์ อาจารย์ที่ปรึกษาโครงงานพิเศษที่คอยให้คำแนะนำ ให้คำปรึกษาอย่างใกล้ชิด และเสนอแนะแนวทางแก้ปัญหา รวมทั้งตรวจแก้โครงงานพิเศษฉบับนี้ให้มีความสมบูรณ์ยิ่งขึ้น

ขอขอบพระคุณ ผศ.ดร.นันทิกา เบญจเทพานันท์ และผศ.สิริลักษณ์ อนันต์สถิตย์สิน ประธานกรรมการ และกรรมการโครงงานพิเศษ ที่ให้คำแนะนำและชี้จุดบกพร่องที่ควรแก้ไขของโปรแกรม ช่วยตรวจสอบเพิ่มความสมบูรณ์ให้กับโครงงานพิเศษฉบับนี้

สุดท้ายนี้ผู้จัดทำ ขอขอบพระคุณบิดา มารดา และบุคคลในครอบครัว รวมทั้งเพื่อนๆ ที่ให้ความช่วยเหลือ คอยสนับสนุนและให้กำลังใจตลอดในการทำโครงงานพิเศษ จนโครงงานพิเศษนี้สำเร็จลุล่วงไปด้วยดี

นายวรมันต์ อชะเมตรา

นายสัณห์ สิทธิวรชาติ

สารบัญ

	หน้า
บทคัดย่อภาษาไทย	**I**
บทคัดย่อภาษาอังกฤษ	**II**
กิตติกรรมประกาศ	**III**
สารบัญ	**IV**
สารบัญตาราง	**VII**
สารบัญรูป	**VIII**
บทที่ 1 บทนำ	**1**
1.1 ความเป็นมาและความสำคัญ	1
1.2 วัตถุประสงค์ของโครงงานพิเศษ	2
1.3 เป้าหมายและขอบเขตของโครงงานพิเศษ	2
1.4 ประโยชน์ที่คาดว่าจะได้รับ	3
1.5 ขั้นตอนการดำเนินงาน	3
1.6 อุปกรณ์ที่ใช้ในการทำปัญหาพิเศษ	3
บทที่ 2 ทฤษฎีที่เกี่ยวข้อง	**4**
2.1 ความหมายและการใช้งานเว็บแอพพลิเคชัน	4
2.1.1 สถาปัตยกรรมแบบง่ายของเว็บแอพพลิเคชัน	4
2.1.2 ภาษาที่นิยมใช้เขียนแอพพลิเคชัน	5
2.1.3 PHP	5
2.1.4 คุณสมบัติ ของ PHP	6
2.1.5 สรุปลักษณะเด่นของ PHP	7
2.1.6 เวอร์ชันของ PHP	7
2.2 MySQL	8
2.2.1 สถาปัตยกรรมของ MySQL	8
2.2.2 คำสั่ง SQL ที่ใช้ใน PHP	9
2.2.3 การเชื่อมต่อกับฐานข้อมูลแบบ PDO	10
2.3 ภัยคุกคามของเว็บแอพพลิเคชัน	11
2.3.1 สาเหตุที่พบเป็นประจำของช่องโหว่ที่เกิดในเว็บแอพพลิเคชัน	11

สารบัญ (ต่อ)

	หน้า
2.3.2 ประเภทของการโจมตีผ่านเว็บแอพพลิเคชัน	11
2.3.3 SQL Injection	13
2.3.4 ประเภทของ SQL injection	15
2.3.5 สาเหตุที่ทำให้เกิด SQL Injection	18
2.3.6 การเกิด SQL Injection ใน PHP	19
2.4 การป้องกัน SQL Injection โดยการเขียนโค้ดที่ปลอดภัย	19
2.4.1 การทำ whitelist และ blacklist	19
2.4.2 Parameterized Statement หรือ Bind Variable	20
2.4.3 Stored Procedures	21
2.4.4 ยกเลิกการใช้งานตัวอักษรทั่วไป (Disabling literals)	22
2.4.5 การกำหนดระดับความสำคัญในการใช้งาน (Security Privileges)	22
2.4.6 การป้องกันโดยใช้ฟังก์ชัน mysql_real_escape_string( )	22
2.4.7 การป้องกันโดยการเขียน Prepared Statement ในการเชื่อมต่อฐานข้อมูลแบบ PDO	24
2.5 การป้องกัน SQL Injection โดยใช้โปรแกรมตรวจสอบโค้ด	25
2.5.1 โปรแกรมตรวจสอบโค้ดแบบ Static	25
2.5.2 โปรแกรมตรวจสอบโค้ดแบบ Dynamic	27
2.6 Editor ที่ใช้เขียนเว็บแอพพลิเคชัน	31
2.6.1 Netbeans	31
2.6.2 Eclipse	31
บทที่ 3 การวิเคราะห์และออกแบบ	**33**
3.1 การวิเคราะห์ระบบ	33
3.1.1 วัตถุประสงค์ในการวิเคราะห์ระบบ	33
3.1.2 วิธีการปลั๊กอิน Pixy ลงบน NetBeans และ Eclipse	33
3.1.3 การใช้ Pixy ทดสอบไฟล์ตัวอย่าง	46
3.1.4 สรุปผลการวิเคราะห์ระบบ	57
3.1.5 ขอบเขตของระบบ	57
3.2 การออกแบบระบบ	58
3.2.1 Use Case ของโปรแกรมปลั๊กอิน	58

สารบัญ (ต่อ)

	หน้า
3.2.2 Activity Diagram	58
3.2.3 การออกแบบส่วนติดต่อผู้ใช้และการทำงาน	63
3.3 การพัฒนาโปรแกรม	68
บทที่ 4 ผลการดำเนินงานและการทดสอบระบบ	**71**
4.1 การสแกนไฟล์และแก้ไขโค้ดที่มีช่องโหว่	72
4.1.1 การสแกนไฟล์และแก้ไขโค้ดที่มีช่องโหว่บน NetBeans	72
4.1.2 การสแกนไฟล์และแก้ไขโค้ดที่มีช่องโหว่บน Eclipse	77
4.2 ส่วนช่วยเหลือผู้ใช้งาน (Help)	78
4.3 ผลการทดสอบโปรแกรม	88
4.3.1 โค้ดที่มีการเชื่อมต่อกับฐานข้อมูลแบบ MySQL	88
4.3.2 โค้ดที่ทำการเชื่อมต่อกับฐานข้อมูลแบบ PDO	95
4.4 ความแตกต่างระหว่าง Pixy และโปรแกรมของผู้พัฒนา	96
4.5 ความสามารถในการป้องกันการโจมตีแบบ SQL Injection ประเภทต่างๆ	
บทที่ 5 บทสรุปและข้อเสนอแนะ	**100**
5.1 สรุปผลการดำเนินงาน	100
5.2 ข้อจำกัดในการพัฒนา	100
5.3 ข้อเสนอแนะในการพัฒนาโปรแกรม	101
เอกสารอ้างอิง	**102**
ภาคผนวก ก.การติดตั้งโปรแกรมปลั๊กอินบน Netbeans	**107**
ภาคผนวก ข.การติดตั้งโปรแกรมปลั๊กอินบน Eclipse	**119**

สารบัญตาราง

ตารางที่	หน้า
2.1 คุณสมบัติของ Analyzer Tools	28
2.1 คุณสมบัติของ Analyzer Tools (ต่อ)	29
2.1 คุณสมบัติของ Analyzer Tools (ต่อ)	30
2.2 รุ่นของ Eclipse	32
3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy	46
3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)	47
3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)	48
3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)	49
3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)	50
3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)	51
3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)	52
3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)	53
3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)	54
3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)	55
3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)	56
4.1 เปรียบเทียบผลการสแกนของโปรแกรมปลั๊กอินกับ Pixy	88
4.1 เปรียบเทียบผลการสแกนของโปรแกรมปลั๊กอินกับ Pixy (ต่อ)	89
4.1 เปรียบเทียบผลการสแกนของโปรแกรมปลั๊กอินกับ Pixy (ต่อ)	90
4.1 เปรียบเทียบผลการสแกนของโปรแกรมปลั๊กอินกับ Pixy (ต่อ)	91
4.1 เปรียบเทียบผลการสแกนของโปรแกรมปลั๊กอินกับ Pixy (ต่อ)	92
4.1 เปรียบเทียบผลการสแกนของโปรแกรมปลั๊กอินกับ Pixy (ต่อ)	93
4.1 เปรียบเทียบผลการสแกนของโปรแกรมปลั๊กอินกับ Pixy (ต่อ)	94
4.1 เปรียบเทียบผลการสแกนของโปรแกรมปลั๊กอินกับ Pixy (ต่อ)	95
4.2 เปรียบเทียบผลการสแกนไฟล์ PDO ของโปรแกรมปลั๊กอินกับ Pixy	95
4.3 สรุปความสามารถของโปรแกรมปลั๊กอิน	99

สารบัญรูป

รูปที่	หน้า
2.1 สถาปัตยกรรมแบบง่ายของเว็บแอพพลิเคชัน	5
2.2 อันดับภาษาที่นิยมในการเขียนเว็บแอพพลิเคชันประจำปี 2013 จาก builtwith.com	5
2.3 คำสั่งการสร้างฐานข้อมูล	9
2.4 คำสั่งการลบฐานข้อมูล	9
2.5 คำสั่งการคิวรี่ฐานข้อมูลในกรณีที่ได้มีการเลือกฐานข้อมูลแล้ว	9
2.6 คำสั่งการคิวรี่กรณีที่ไม่ได้มีการเลือกฐานข้อมูลหรือต้องการเลือกฐานข้อมูลอื่น	9
2.7 การยกเลิกการเชื่อมต่อฐานข้อมูล	10
2.8 การดึงข้อมูลโดยกำหนดจำนวนที่ต้องการ	10
2.9 คำสั่งการเพิ่มข้อมูล	10
2.10 ประเภทของการโจมตีที่ได้รับความนิยมประจำปี 2013 จาก builtwith.com	13
2.11 10 อันดับการโจมตีที่นิยมในปี 2013 จากการจัดอันดับของ www.owasp.org	14
2.12 ตัวอย่างหน้าต่างการเข้าสู่ระบบ	16
2.13 ตัวอย่างการใช้ฟังก์ชัน mysql_real_escape_string()	23
3.1 หน้าจอการเปิด Project บน NetBeans	33
3.2 หน้าต่างการเลือก ProjectSqlInjection บน NetBeans	34
3.3 หน้าต่าง ProjectSqlInjection บน NetBeans	35
3.4 หน้าต่างแสดงเปิดไฟล์ Scanner.java ของ Pixy บน NetBeans	35
3.5 ตั้ง filePath เป็นที่อยู่ของไฟล์ PHP ที่ต้องการสแกนบน NetBeans	36
3.6 ตัวอย่างผลการสแกนของ Pixy บน NetBeans	37
3.7 หน้าจอแสดงการสร้าง New Java Project บน Eclipse	38
3.8 หน้าต่างการตั้งชื่อ Eclipse Project	39
3.9 การ disable "build automatically" ในเมนู Project บน Eclipse	40
3.10 หน้าจอแสดงการ Import Project บน Eclipse	41
3.11 หน้าจอแสดงการ Import Project Pixy บน Eclipse	42
3.12 หน้าต่างแสดงการเปิดไฟล์ Scanner.java ของ Pixy บน Eclipse	43
3.13 การตั้ง filePath เป็นที่อยู่ของไฟล์ PHP ที่ต้องการสแกนบน Eclipse	44
3.14 หน้าต่างแสดงผลการสแกนของ Pixy บน Eclipse	45
3.15 Use Case ของโปรแกรม	58
3.16 Activity Diagram ของระบบสแกนและแก้ไข	59

สารบัญรูป(ต่อ)

รูปที่	หน้า
3.17 Activity Diagram ของ Help	60
3.18 Activity Diagram ของสาเหตุที่ทำให้เกิด SQL Injection	61
3.19 Activity Diagram ของวิธีป้องกัน SQL Injection	62
3.20 หน้าจอการเลือกการใช้งานโปรแกรมในการ Generate Code	63
3.21 หน้าจอการเลือกประเภทของการ Gen Code	64
3.22 หน้าจอของเมนู Scan now	65
3.23 หน้าจอของเมนู Setting text	66
3.24 Help Menu ของโปรแกรม	67
3.25 ผังงาน (Flowchart) ของโปรแกรม	68
3.26 ผังงาน (Flowchart) ย่อยของโปรแกรมในส่วนที่ 1 (แก้ไขไฟล์ที่มีโค้ด PDO)	69
3.27 ผังงาน (Flowchart) ย่อยของโปรแกรมในส่วนที่ 2 (แก้ไขโค้ด SQL)	70
4.1 หน้าจอการสแกนไฟล์บน NetBeans	72
4.2 หน้าจอแสดงผลการสแกนไฟล์	73
4.3 หน้าจอแสดงผลการใช้งานฟังก์ชัน Edit All ในการแก้ไขช่องโหว่	74
4.4 หน้าจอแสดงผลการแก้ไขไฟล์กรณีไม่พบช่องโหว่	75
4.5 หน้าจอแสดงผลลัพธ์ของปุ่ม Full Text	76
4.6 หน้าจอการสแกนไฟล์บน Eclipse	77
4.7 หน้าจอ Help: ความหมายของ SQL Injection	78
4.8 หน้าจอ Help: สาเหตุของการเกิด SQL Injection	79
4.9 หน้าจอ Help: การกรอง Escape Characters ที่ผิดพลาด	80
4.10 หน้าจอ Help: การกำหนดชนิดข้อมูลผิดพลาด	81
4.11 หน้าจอ Help: Blind SQL Injection	82
4.12 หน้าจอ Help: การป้องกันโดยใช้ Parameterized Statement	83
4.13 หน้าจอ Help: การป้องกันโดยใช้ Security Privileges	84
4.14 หน้าจอ Help: การป้องกันโดยใช้ Stored Procedures	85
4.15 หน้าจอ Help: การป้องกันโดยการทำ Whitelist และ Blacklist	86
4.16 หน้าจอ Help: การป้องกันโดยยกเลิกการใช้งานตัวอักษรทั่วไป	87
4.17 หน้าจอแสดงผลการสแกนไฟล์ของ Pixy	96
4.18 หน้าจอแสดงผลการสแกนด้วยโปรแกรมปลั๊กอิน	97

สารบัญรูป(ต่อ)

รูปที่	หน้า
4.19 หน้าจอแสดงผลการสแกนไฟล์ PDO ด้วยโปรแกรมปลั๊กอิน	98
ก.1 หน้าจอเลือกเมนู File > New Project… บน NetBeans	107
ก.2 หน้าต่างแสดงการเลือก Module	108
ก.3 หน้าต่างแสดงการตั้งค่า Name and Location	109
ก.4 หน้าต่างแสดงการตั้งค่า Basic Module Configuration	110
ก.5 หน้าจอการสร้างตัวติดตั้งโปรแกรมปลั๊กอินบน NetBeans	111
ก.6 หน้าจอเลือกเมนู Tools > Plugins บน NetBeans	112
ก.7 หน้าต่างการเลือก org-sunwaramun-projectsqlinjection.nbm	113
ก.8 หน้าต่างการเลือก ProjectSqlInjection	114
ก.9 หน้าต่าง Install	115
ก.10 หน้าต่างแสดงการยอมรับข้อตกลง	116
ก.11 หน้าต่างแสดงการเสร็จสิ้นการ Install	117
ก.12 หน้าจอเมนูของโปรแกรมปลั๊กอินเมื่อติดตั้งแล้ว	118
ข.1 หน้าจอแสดงการเลือกเมนู File > New > Other…	119
ข.2 หน้าต่างแสดงการเลือก Plug-in Project	120
ข.3 หน้าต่างแสดงการตั้งชื่อ Plug-in Project	121
ข.4 หน้าต่างแสดงการตั้งค่า Plug-in Project	122
ข.5 หน้าต่างแสดงเลือก Templates Plug-in Project	123
ข.6 หน้าจอโปรแกรม SQLInjectionScanner	124
ข.7 หน้าจอแสดงการเลือกเมนู File > New > Other…	125
ข.8 หน้าต่างแสดงการเลือก Feature Project	126
ข.9 หน้าต่างแสดงการตั้งชื่อ Project name	127
ข.10 หน้าต่างแสดงการเลือก SQLInjectionScanner (1.0.0.qualifier)	128
ข.11 หน้าจอการเลือก New > Other…	129
ข.12 หน้าต่าง New แสดงการเลือก Update Site Project	130
ข.13 หน้าต่างการตั้งชื่อ Project name	131
ข.14 หน้าจอแสดงการ Add Feature	132
ข.15 หน้าต่าง Feature Selection	133
ข.16 หน้าจอแสดงการ Build All	134

สารบัญรูป(ต่อ)

รูปที่	หน้า
ข.17 หน้าจอแสดงตัวติดตั้งปลั๊กอินเป็นไฟล์ .jar	135
ข.18 หน้าจอแสดงการ Install New Software	136
ข.19 หน้าต่างการ Install	137
ข.20 หน้าต่าง Add Repository	138
ข.21 หน้าต่างแสดงการเลือก content.jar	138
ข.22 หน้าต่าง Add Repository	139
ข.23 หน้าต่าง Install แสดงเลือก SQLInjectionScannerFeature	139
ข.24 หน้าต่างแสดงการ Install	140
ข.25 หน้าต่างแสดงการยอมรับข้อตกลง	141
ข.26 หน้าต่าง Restart Eclipse	142
ข.27 หน้าจอแสดงเมนูของโปรแกรมปลั๊กอินบน Eclipse	142

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญ

ปัจจุบันระบบเครือข่ายอินเทอร์เน็ตได้เข้ามาเป็นส่วนหนึ่งในชีวิตประจำวันไม่ว่าจะเป็นในการติดต่อสื่อสาร การเผยแพร่ข่าวสาร การซื้อขายสินค้า และการทำธุรกรรมทางการเงิน ซึ่งทำให้เว็บแอพพลิเคชันได้รับความนิยมเป็นอย่างสูงเนื่องจากผู้ใช้สามารถทำกิจกรรมต่างๆ บนระบบเครือข่ายอินเทอร์เน็ตได้อย่างสะดวกผ่านทางโปรแกรมเบราว์เซอร์โดยไม่ต้องทำการติดตั้งโปรแกรมใดเพิ่ม แต่ปัญหาที่พบมากคือผู้พัฒนาเว็บแอพพลิเคชันส่วนใหญ่พัฒนาโดยขาดความรู้หรือละเลยวิธีการเขียนโค้ดอย่างปลอดภัยทำให้การโจมตีระบบเว็บแอพพลิเคชันทำได้ค่อนข้างง่าย ภัยร้ายที่เกิดจากช่องโหว่ความปลอดภัยของเว็บแอพพลิเคชันจึงมีจำนวนเพิ่มขึ้นทุกปี และเป็นภัยที่ทุกองค์กรควรให้ความสนใจในระดับต้นๆ

หนึ่งในรูปแบบการโจมตีเว็บแอพพลิเคชันที่ได้รับความนิยมมากที่สุดในปัจจุบันคือ SQL Injection ซึ่งผู้โจมตีระบบใช้ประโยชน์จากการที่แอพพลิเคชันต้องรับข้อมูลจากผู้ใช้มาสร้างคำสั่ง SQL เพื่อค้นหาหรือจัดการกับฐานข้อมูล ถ้าผู้พัฒนาโปรแกรมเขียนโค้ดแบบไม่ระวังผู้โจมตีสามารถป้อนข้อมูลที่เมื่อระบบได้รับแล้วสามารถเปิดเผยหรือทำลายข้อมูลที่เป็นความลับของระบบได้ เช่น ทำให้สามารถล่วงรู้ข้อมูลต่าง ๆ ในฐานข้อมูล รวมไปถึงการทำให้ระบบอนุญาตให้เข้าใช้โดยไม่จำเป็นต้องรู้ชื่อผู้ใช้งานและรหัสผ่าน ช่องโหว่เหล่านี้ส่วนใหญ่เกิดจากการที่ผู้พัฒนาเป็นจำนวนมากขาดความรู้หรือพัฒนาระบบเว็บแอพพลิเคชันโดยไม่ได้คำนึงถึงความปลอดภัยเนื่องจากไม่ตระหนักถึงผลร้ายที่เกิดขึ้น อีกทั้งเครื่องมือที่ใช้ในการช่วยตรวจสอบมีการใช้งานค่อนข้างยาก และมักถูกเขียนแยกขึ้นมากับโปรแกรมที่ใช้ในการพัฒนาแอพพลิเคชัน ทำให้ผู้ใช้ไม่ทราบว่ามีเครื่องมือลักษณะนี้อยู่

ด้วยเหตุนี้ผู้จัดทำโครงงานพิเศษจึงเกิดแนวคิดในการพัฒนาโปรแกรมปลั๊กอินบน NetBeans และ Eclipse ซึ่งเป็น Editor ที่ผู้พัฒนาเว็บแอพพลิเคชันใช้กันอย่างแพร่หลาย เพื่อใช้ในการสแกนโค้ดที่พัฒนาด้วยภาษา PHP (PHP Hypertext Preprocessor) โดยนอกจากสามารถสแกนโค้ด PHP เพื่อหาช่องโหว่ที่ทำให้เกิดการโจมตีประเภท SQL Injection ได้แล้ว โปรแกรมสามารถแนะนำวิธีการแก้ไขโค้ด PHP ที่ทำให้เกิดช่องโหว่นั้น และสามารถแก้ไขโค้ดให้ด้วย อีกทั้งยังมีส่วนแนะนำความรู้ (Help) ที่ช่วยแนะนำผู้ใช้งานให้รู้จักกับ SQL Injection สาเหตุที่เกิด และการป้องกันอย่างถูกวิธี ซึ่งในส่วนของการตรวจสอบโค้ดนั้นได้นำโปรแกรม Pixy ซึ่งเป็นโปรแกรมวิเคราะห์โค้ดอันตรายบน PHP มาใช้ และประยุกต์ให้สามารถแก้ไขช่องโหว่แบบ SQL Injection ของไฟล์ที่เชื่อมต่อฐานข้อมูลแบบ PHP Data Object (PDO) ได้ สำหรับสาเหตุที่เลือกพัฒนาให้ใช้กับภาษา PHP เนื่องจากภาษา PHP เป็นภาษาที่เป็นโอเพนซอร์สและได้รับความนิยม

1.2 วัตถุประสงค์ของโครงงานพิเศษ

แอพพลิเคชัน

1) สแกนไฟล์ PHP เพื่อหาช่องโหว่ที่ทำให้เกิดการโจมตีประเภท SQL Injection
2) แนะนำการแก้ไขโค้ด PHP ที่เป็นช่องโหว่ให้ปลอดภัย
3) แก้ไขโค้ด PHP ที่เป็นช่องโหว่ของไฟล์ .php ทุกช่องโหว่โดยสร้างเป็นไฟล์ใหม่
4) มี Help เพื่อให้ความรู้เกี่ยวกับ SQL Injection ให้ได้ศึกษาถึง ที่มา สาเหตุ และวิธีการป้องกัน SQL Injection

ผู้พัฒนา

1) เพื่อศึกษาวิธีการทำ SQL Injection รูปแบบต่างๆที่ใช้โจมตีระบบ
2) พัฒนาโปรแกรมปลั๊กอินที่ทำงานบน NetBeans และ Eclipse
3) พัฒนาเครื่องมือที่ใช้ช่วยในการตรวจสอบโค้ด PHP เพื่อหาช่องโหว่การโจมตีประเภท SQL Injection
4) พัฒนาส่วนช่วยเหลือผู้ใช้งาน (Help) เพื่อให้ความรู้เบื้องต้นเกี่ยวกับ SQL Injection แก่ผู้ใช้งาน

1.3 เป้าหมายและขอบเขตของโครงงานพิเศษ

ฝั่งผู้พัฒนา

1) ศึกษาการโจมตีของ SQL Injection รูปแบบต่าง ๆ ที่มีต่อเว็บแอพพลอเคชัน
2) พัฒนาปลั๊กอินบน NetBeans และ Eclipse
3) พัฒนาโปรแกรมปลั๊กอินเพื่อช่วยโปรแกรมเมอร์ในการเขียนโค้ด PHP ได้อย่างปลอดภัยต่อ การโจมตีประเภท SQL Injection

ฝั่งผู้ใช้งาน

1) สามารถตรวจสอบโค้ด PHP ที่เขียนได้ว่ามีส่วนไหนบ้างที่ทำให้เกิด SQL Injection โดยใช้การวิเคราะห์แบบ Static
2) สามารถใช้โปรแกรมปลั๊กอินในการแก้ไขโค้ดที่มีช่องโหว่จากโค้ด PHP เท่านั้น
3) สามารถอ่านรายละเอียด SQL Injection จากส่วนช่วยเหลือผู้ใช้งาน (Help) ของโปรแกรมได้
4) สามารถใช้โปรแกรมปลั๊กอินได้บน NetBeans และ Eclipse
5) ปลั๊กอินสามารถสแกนหาช่องโหว่ที่ทำให้เกิด SQL Injection บน PHP ได้ 2 รูปแบบ คือการติดต่อกับฐานข้อมูลโดยใช้ฟังก์ชัน mysql_query( ) และแบบ PDO (PHP Data Objects)

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1) ผู้พัฒนาโปรแกรมเว็บแอพพลิเคชันบน PHP สามารถใช้ปลั๊กอินนี้ช่วยในการเขียนโค้ดให้ปลอดภัยจาก SQL Injection
2) ผู้จัดทำมีความสามารถในการเขียนโปรแกรมปลั๊กอินบน NetBeans และ Eclipse
3) โปรแกรมปลั๊กอินจะป้องกันการโจมตี SQL Injection ที่ต้นเหตุ (ป้องกันตั้งแต่การเขียน โค้ด)
4) โปรแกรมช่วยเขียนที่ผู้จัดทำได้ทำขึ้นสามารถนำไปพัฒนาต่อได้

1.5 ขั้นตอนการดำเนินงาน

1) ศึกษาทำความเข้าใจโครงสร้างภาษา PHP
2) ศึกษารูปแบบการเกิดการโจมตีแบบ SQL Injection
3) ศึกษาการเขียนโค้ดในภาษา PHP ที่เป็นช่องโหว่ทำให้เกิด SQL Injection ได้
4) ศึกษา Pixy เพื่อนำมาพัฒนาเป็นปลั๊กอินบน NetBeans และ Eclipse
5) ศึกษาการสร้างปลั๊กอินบน NetBeans และ Eclipse
6) เป็น ทดสอบ Pixy กับ โค้ดที่มีช่องโหว่ SQL Injection จำนวน 20 ไฟล์
7) ออกแบบโปรแกรมปลั๊กอิน
8) พัฒนาโปรแกรมตามที่ออกแบบ
9) ทดสอบโปรแกรม
10) จัดทำเอกสารและคู่มือการใช้งาน

1.6 อุปกรณ์ที่ใช้ในการทำปัญหาพิเศษ

ฮาร์ดแวร์

1) โน้ตบุ๊ก

หน่วยประมวลผล : Intel Core i7-2630QM

การ์ดจอ : AMD Radeon HD 6550M (1GB GDDR3)

หน่วยความจำ : 4 GB DDR3 Memory Bus 1333 MHz

ฮาร์ดดิสก์ : 750 GB 5400 RPM

ซอฟต์แวร์

1) Pixy
2) Eclipse เวอร์ชัน 4.3.2 SR2
3) NetBeans IDE เวอร์ชัน 8.0

บทที่ 2
ความรู้พื้นฐานและทฤษฎีที่เกี่ยวข้อง

2.1 ความหมายและการใช้งานเว็บแอพพลิเคชัน

เว็บแอพพลิเคชัน (Web application) คือโปรแกรมประยุกต์บนเว็บ เข้าถึงได้ด้วยโปรแกรมเบราว์เซอร์ผ่านเครือข่ายคอมพิวเตอร์ อินเทอร์เน็ต หรืออินทราเน็ต ปัจจุบันเว็บแอพพลิเคชันเป็นที่นิยมเนื่องจากการอัพเดท และดูแลสามารถทำได้โดยไม่ต้องติดตั้งซอฟต์แวร์บนเครื่องผู้ใช้ ตัวอย่างเว็บแอพพลิเคชันได้แก่ เว็บเมล์ เว็บไซต์ประเภทพาณิชย์อิเล็กทรอนิกส์ หรือที่รู้จักกันในนามของ อีคอมเมิร์ซ (e-Commerce) กระดานสนทนา บล็อก วิกิ เป็นต้น [1]

เว็บแอพพลิเคชันประกอบด้วย 3 ส่วน คือ

1. ส่วนติดต่อกับผู้ใช้งาน (เว็บเบราว์เซอร์)
2. ส่วนของเว็บเซิร์ฟเวอร์ (การเขียนโปรแกรมภาษา เช่น C #, ASP,. NET, PHP, JSP, ฯลฯ)
3. ส่วนของการจัดเก็บข้อมูล (ฐานข้อมูล เช่น Microsoft SQL Server, MySQL, Oracle, ฯลฯ)

2.1.1 สถาปัตยกรรมแบบง่ายของเว็บแอพพลิเคชัน

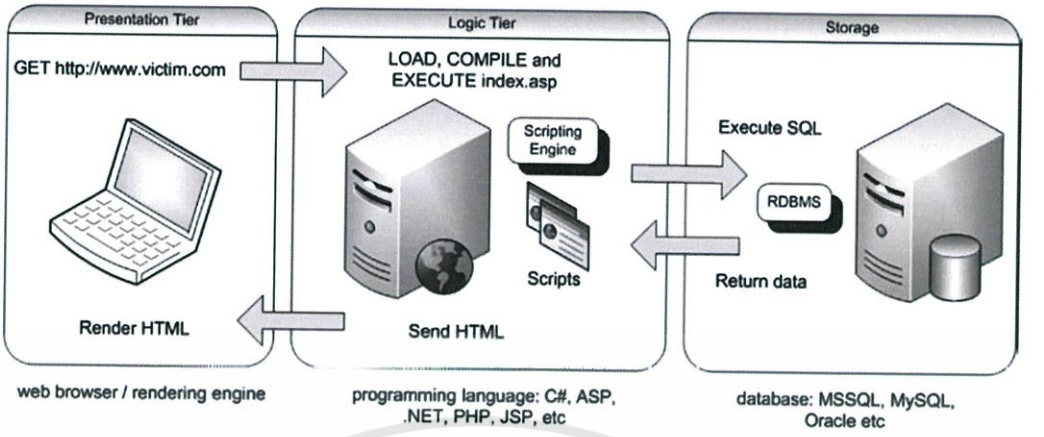
เว็บแอพพลิเคชันที่ใช้ฐานข้อมูลโดยทั่วไปจะประกอบด้วยสามส่วน คือ ส่วนที่ติดต่อกับผู้ใช้งาน ส่วนของเว็บเซิร์ฟเวอร์ และส่วนของการจัดเก็บข้อมูล [2]

1. ส่วนติดต่อกับผู้ใช้งาน จะแสดงข้อมูลที่เกี่ยวข้องกับเว็บแอพพลิเคชันนั้นๆ เช่น เว็บแอพพลิเคชันร้านค้าออนไลน์จะมีข้อมูลการเรียกดูสินค้า การจัดซื้อ เนื้อหาตะกร้าสินค้า โดยจะแสดงผลไปยังเว็บเบราว์เซอร์

2. ส่วนของเว็บเซิร์ฟเวอร์ มีหน้าที่ควบคุมการทำงานของเว็บแอพพลิเคชัน โดยการดำเนินการและประมวลผลคำสั่งต่างๆ ที่ได้รับมาจากเว็บเบราว์เซอร์

3. ส่วนของการจัดเก็บข้อมูล ส่วนนี้จะทำการเก็บรวบรวมข้อมูลต่างๆที่อยู่ในเว็บแอพพลิเคชัน

รูปที่ 2.1 แสดงสถาปัตยกรรมแบบง่ายของเว็บแอพพลิเคชัน เมื่อผู้ใช้เปิดเว็บเบราว์เซอร์และเชื่อมต่อกับ http://www.victim.com ส่วนติดต่อกับผู้ใช้งานจะแสดงผลบนเว็บเบราว์เซอร์ โดยมีเว็บเซิร์ฟเวอร์เป็นตัวควบคุมการแสดงผลและดึงข้อมูลจากส่วนของฐานข้อมูลมาแสดงบนเว็บเบราว์เซอร์



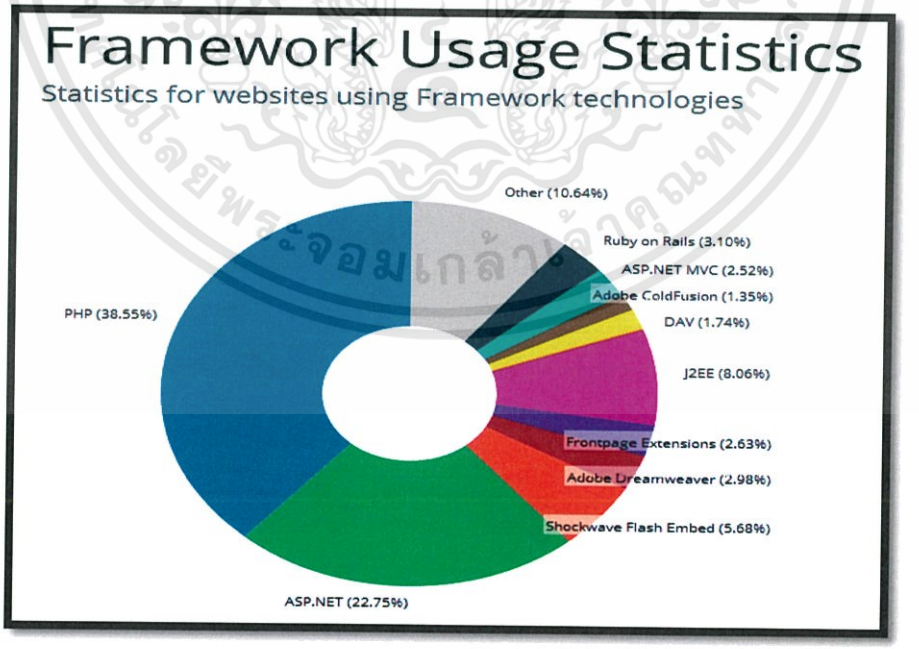
รูปที่ 2.1 สถาปัตยกรรมแบบง่ายของเว็บแอพพลิเคชัน

2.1.2 ภาษาที่นิยมใช้เขียนเว็บแอพพลิเคชัน

เว็บไซต์ builtwith.com [3] ทำการสำรวจภาษาที่นิยมใช้เขียนเว็บแอพพลิเคชัน พบว่าภาษาที่ได้รับความนิยมมากคือ PHP ตามด้วย ASP.NET และ J2EE

2.1.3 PHP

PHP ย่อมาจาก PHP Hypertext Preprocessor [4] เดิมย่อมาจาก Personal Home Page Tools เป็นภาษาประเภทสคริปต์ซึ่งคำสั่งต่างๆ จะถูกเก็บอยู่ในไฟล์ประเภทสคริปต์ (script) และเวลาใช้งานต้องอาศัยการแปลชุดคำสั่ง ตัวอย่างของภาษาสคริปต์เช่น JavaScript และ Perl เป็นต้น



รูปที่ 2.2 อันดับภาษาที่นิยมในการเขียนเว็บแอพพลิเคชันประจำปี 2013 จาก builtwith.com

PHP เป็นภาษาสคริปต์ที่ไม่ได้เป็นส่วนหนึ่งของเว็บเซิร์ฟเวอร์ดังนั้นก่อนเลือกใช้ PHP ผู้ใช้ควรตรวจสอบก่อนว่าเว็บเซิร์ฟเวอร์นั้นสามารถใช้สคริปต์ PHP ได้หรือไม่ ซึ่งเว็บเซิร์ฟเวอร์ที่สามารถใช้ภาษา PHP ได้ เช่น Apache Web Server [5] และ Personal Web Server (PWP) [6] Apache สามารถใช้ PHP ได้สองแบบคือ แบบ Apache Module หรือแบบ CGI[1] ซึ่งถ้าใช้แบบแรก PHP จะถูกรวมเป็นส่วนหนึ่งหรือเป็นส่วนขยายของ Apache ซึ่งจะทำงานได้เร็วกว่าแบบ Apache ต้องเรียกขึ้นมาทำงานทุกครั้งที่ต้องการใช้ PHP ดังนั้นในการใช้ PHP แบบที่เป็นโมดูลหนึ่งของ Apache จะทำงานได้มีประสิทธิภาพมากกว่าแบบ CGI

2.1.4 คุณสมบัติ ของ PHP

การแสดงผลของไฟล์ PHP จะปรากฏในรูปแบบของ HTML โดยจะไม่แสดงคำสั่งที่ผู้ใช้เขียน ซึ่งเป็นลักษณะเด่นที่ภาษา PHP แตกต่างจากภาษาในลักษณะเซิร์ฟเวอร์ไซต์-สคริปต์ภาษาอื่น นอกจากนี้ PHP เป็นภาษาที่เรียนรู้และเริ่มต้นได้ไม่ยาก มีเครื่องมือช่วยเหลือที่สามารถหาอ่านได้ฟรีบนอินเทอร์เน็ตเป็นจำนวนมาก

ความสามารถหลักของ PHP ได้แก่ การสร้างเนื้อหาอัตโนมัติในการจัดการคำสั่ง การอ่านและประมวลผลข้อมูลจากผู้ใช้ การอ่านข้อมูลจากฐานข้อมูล การจัดการกับคุกกี้ นอกจากนี้ยังทำงานเช่นเดียวกับโปรแกรม CGI เช่น ประมวลผลตามบรรทัดคำสั่ง (command line scripting) ทำให้ผู้เขียนโปรแกรมสร้างสคริปต์ PHP ทำงานผ่าน PHP พาร์เซอร์ (PHP parser) โดยไม่ต้องผ่านเซิร์ฟเวอร์หรือเบราว์เซอร์ ซึ่งมีลักษณะเหมือนกับ Cron (ใน ยูนิกซ์หรือลีนุกซ์) หรือ Task Scheduler (ในวินโดวส์) สคริปต์เหล่านี้สามารถนำไปใช้ในแบบ Simple text processing tasks ได้

ในการแสดงผลนอกจาก PHP สามารถแสดงผลแบบ HTML ซึ่งเป็นจุดประสงค์หลักได้แล้ว PHP สามารถสร้าง XHTML หรือ XML สามารถทำงานร่วมกับคำสั่งเสริมต่างๆ ซึ่งแสดงผลข้อมูลเป็น PDF flash (โดยใช้ libswf และ Ming) สามารถทำงานประมวลผลข้อความจาก POSIX Extended หรือ รูปแบบ Perl ทั่วไปเพื่อแปลงเป็นเอกสาร XML ซึ่ง PHP รองรับมาตรฐาน Simple API for XML (SAX) [7] และ Document Object Model (DOM) [8] ที่ใช้รูปแบบ Extensible Stylesheet Language Transformations (XSLT) [9]

การสร้างโปรแกรม PHP สามารถสร้างผ่านทางโปรแกรมแก้ไขข้อความทั่วไป เช่น Notepad โดยเมื่อเขียนคำสั่งแล้วนำมาประมวลผลบนเว็บเซิร์ฟเวอร์ซึ่งเว็บเซิร์ฟเวอร์ที่สามารถใช้งาน PHP ได้เช่น Apache Microsoft Internet Information Services (IIS) [10] Personal Web Server [11]

---

[1] CGI เป็นโปรแกรมที่ทำงานอยู่บนเว็บเซิร์ฟเวอร์เพื่อทำหน้าที่โต้ตอบ (interact) กับเว็บเบราว์เซอร์ CGI program จะถูกสั่งให้ทำงานขณะมีการเรียกใช้งานนั้น ดังนั้นจึงทำให้มันสามารถ แสดงข้อมูลผลลัพธ์ในลักษณะของ Dynamic information ได้

Netscape [12] iPlanet servers [13] Oreilly Website Pro server [14] Caudium [15] Xitami [16] OmniHTTPd [17]

PHP สามารถทำงานร่วมกับฐานข้อมูลได้หลายชนิด ซึ่งฐานข้อมูลส่วนหนึ่งที่รองรับได้แก่ Oracle [18] dBase [19] PostgreSQL [20] IBM DB2 [21] MySQL [22] Informix [23] เนื่องจากโครงสร้างของฐานข้อมูลแบบ DBX อนุญาตให้ PHP ใช้งานได้กับฐานข้อมูลที่รองรับ และ PHP สามารถรองรับ Open Database Connection (ODBC) ซึ่งเป็นมาตรฐานการเชื่อมต่อฐานข้อมูลที่ใช้กันแพร่หลายได้อีกด้วย

PHP รองรับการสื่อสารกับหลายโปรโตคอล เช่น LDAP IMAP SNMP NNTP POP3 HTTP เป็นต้น ผู้ใช้งานสามารถสร้าง Socket บนเครือข่ายได้ทุกโปรโตคอล นอกจากนี้ PHP มีการรองรับการแลกเปลี่ยนข้อมูลแบบ WDDX Complex (Web Distributed Data eXchange) [25] กับภาษาโปรแกรมอื่นๆ ทั่วไปได้ ในส่วนของ Interconnection PHP มีการอนุญาตให้ผู้ใช้งานเปลี่ยน Java objects เป็น PHP Object แล้วใช้งานได้

2.1.5 สรุปลักษณะเด่นของ PHP ได้ดังนี้

- ไม่เสียค่าใช้จ่าย (Open Source)
- เป็นภาษาประเภทเซิร์ฟเวอร์ไซด์สคริปต์ทำงานบนเว็บเซิร์ฟเวอร์ที่มีขีดความสามารถไม่จำกัด
- สามารถทำงานได้บนหลายระบบปฏิบัติการ เช่น UNIX, Linux และ Windows
- เรียนรู้ง่าย เนื่องจาก PHP ถูกฝังเข้าไปใน HTML และเป็นภาษาที่ใช้โครงสร้างและไวยากรณ์ภาษาแบบง่าย
- เร็วและมีประสิทธิภาพ โดยเฉพาะเมื่อใช้กับ Apache Server เพราะไม่ต้องใช้โปรแกรมจากภายนอกและใช้ร่วมกับ XML ได้ทันที
- ใช้กับระบบแฟ้มข้อมูล ข้อมูลตัวอักษรได้อย่างมีประสิทธิภาพ
- ใช้กับโครงสร้างข้อมูลแบบ Scalar Array Associative array และการประมวลผลภาพได้

2.1.6 เวอร์ชันของ PHP

ปัจจุบัน PHP พัฒนามาถึงเวอร์ชัน 5 โดยเวอร์ชันล่าสุดที่ใช้งานคือเวอร์ชัน 5.5.12 โดยสามารถใช้งานได้ตั้งแต่ 1 พฤษภาคม 2557 เป็นต้น

2.2 MySQL

MySQL [57] จัดเป็นระบบจัดการฐานข้อมูลเชิงสัมพันธ์ (RDBMS: Relational Database Management System) ซึ่งเป็นที่นิยมใช้กันมากในปัจจุบัน โดยเฉพาะอย่างยิ่งในโลกอินเทอร์เน็ต เนื่องจาก

- MySQL เป็นซอฟต์แวร์โอเพนซอร์สทางด้านฐานข้อมูลที่มีประสิทธิภาพสูง
- นักพัฒนาฐานข้อมูลที่เคยใช้ MySQL ต่างยอมรับในความรวดเร็ว การรองรับจำนวนผู้ใช้ และขนาดของข้อมูลจำนวนมหาศาล
- สนับสนุนการใช้งานบนระบบปฏิบัติการได้หลายประเภท เช่น UNIX OS/2, MAC OS X และ Windows เป็นต้น
- สามารถใช้งานร่วมกับ Web Development platform เช่น C, C++, Java, Perl, PHP, Python, TCL (Tool Command Language) หรือ ASP (Active Server Page)
- ได้รับความนิยมในปัจจุบัน และมีแนวโน้มสูงมากขึ้นในอนาคต

MySQL เป็นซอฟต์แวร์ประเภทโอเพนซอร์ส สามารถดาวน์โหลดซอร์สโค้ดต้นฉบับได้โดยไม่เสียค่าใช้จ่าย MySQL ยึดถือสิทธิบัตรตาม GPL (GNU General Public License) [26] ซึ่งเป็นข้อกำหนดของซอฟต์แวร์โอเพนซอร์สประเภทนี้ MySQL ถูกนำไปใช้ในระบบต่างๆ มากมาย ทั้งระบบเล็กที่มีจำนวนตารางข้อมูลน้อย เช่น ระบบฐานข้อมูลของแผนกที่มีขนาดเล็ก ไปจนถึงระบบฐานข้อมูลขนาดใหญ่ เช่น ระบบบัญชีเงินเดือน ปัจจุบันนิยมใช้ MySQL เป็น Database Server สำหรับฐานข้อมูลบนเว็บเป็นจำนวนมาก

2.2.1 สถาปัตยกรรมของ MySQL

โครงสร้างการทำงานของ MySQL เป็นลักษณะการทำงานแบบ Client/Server ซึ่งประกอบด้วย 2 ส่วนหลักคือ ส่วนของผู้ให้บริการ (Server) และ ส่วนของผู้ใช้บริการ (Client) โดยในแต่ละส่วนก็จะมีโปรแกรมสำหรับการทำงานตามหน้าที่ของตน

ส่วนของผู้ให้บริการ (Server) คือส่วนที่ทำหน้าที่บริหารจัดการระบบฐานข้อมูล และเป็นที่จัดเก็บข้อมูลทั้งหมด ซึ่งคือ MySQL server นั่นเอง

ส่วนของผู้ใช้บริการ (Client) คือส่วนของผู้ใช้นั่นเอง โปรแกรมใช้งานในส่วนนี้ได้แก่ MySQL client, Access, Web development platform ต่างๆ เช่น Java, Perl, PHP, ASP

2.2.2 คำสั่ง SQL ที่ใช้ใน PHP

1) การสร้างฐานข้อมูล

```
<?
$c= mysql_connect("localhost","username","password");
mysql_create_db("dbname",$c);

```

รูปที่ 2.3 คำสั่งการสร้างฐานข้อมูล

2) การลบฐานข้อมูล

```
<?
$c= mysql_connect("localhost","username","password");
mysql_drop_db("dbname",$c);

```

รูปที่ 2.4 คำสั่งการลบฐานข้อมูล

3) การคิวรี่ฐานข้อมูลในกรณีที่ได้มีการเลือกฐานข้อมูลไว้แล้ว

```
<?
$query = "SELECT * from table_name"; // เครื่องหมาย * เป็นการเลือกข้อมูลทั้งหมด
mysql_query($query);

```

รูปที่ 2.5 คำสั่งการคิวรี่ฐานข้อมูลในกรณีที่ได้มีการเลือกฐานข้อมูลแล้ว

4) สำหรับกรณีที่ไม่ได้มีการเลือกฐานข้อมูลหรือต้องการเลือกฐานข้อมูลอื่น

```
<?
$new_dbname = "xxx"; // ชื่อฐานข้อมูลที่ต้องการติดต่อ
$query = "SELECT field1, field2 from table_name";
mysql_db_query($new_dbname,$query);

```

รูปที่ 2.6 คำสั่งการคิวรี่กรณีที่ไม่ได้มีการเลือกฐานข้อมูลหรือต้องการเลือกฐานข้อมูลอื่น

5) การยกเลิกการเชื่อมต่อฐานข้อมูล

```
<?
$c= mysql_connect("localhost","username","password");
...
...
...
mysql_close($c);
```

รูปที่ 2.7 การยกเลิกการเชื่อมต่อฐานข้อมูล

6) การดึงข้อมูลโดยกำหนดจำนวนที่ต้องการ (SELECT)

```
<?
select * from table_name limit 0,1
```

รูปที่ 2.8 การดึงข้อมูลโดยกำหนดจำนวนที่ต้องการ

7) การเพิ่มข้อมูล (INSERT)

```
<?
insert into table_name (field_name1, field_name2) values ('$data1','$data2')
```

รูปที่ 2.9 คำสั่งการเพิ่มข้อมูล

2.2.3 การเชื่อมต่อกับฐานข้อมูลแบบ PDO

PHP Data Objects (PDO) [37] [38] คือการติดต่อกับฐานข้อมูลอีกรูปแบบหนึ่งของ PHP โดย PDO นั้นสามารถเขียนให้ติดต่อกับฐานข้อมูลได้หลายชนิด สาเหตุที่ต้องใช้ PDO เพราะฟังก์ชัน MySQL เริ่มที่จะเก่าและไม่สามารถติดต่อกับฐานข้อมูลใหม่ได้ ตัวอย่างการเชื่อมต่อกับฐานข้อมูลแบบ PDO มีดังนี้

1) การเชื่อมต่อกับฐานข้อมูล

```
<?php
$db = new PDO('mysql:host=localhost;dbname=testdb;charset=utf8', 'username', 'password');
```

2) การ SELECT

```
<?php
$stmt = $db->query('SELECT * FROM table');
```

3) การ UPDATE

```
<?php
$result = $db->exec("UPDATE table SET field='value'");
```

4) การ DELETE

```
<?php
$result = $db->exec("DELETE FROM categories WHERE name = :name");
```

5) การ INSERT

```
<?php
$result = $db->exec("INSERT INTO table(firstname, lastname) VAULES('John', 'Doe')");
```

2.3 ภัยคุกคามของเว็บแอพพลิเคชัน

ปัจจุบันพบว่าจำนวนของช่องโหว่ในส่วนของเว็บแอพพลิเคชัน มีจำนวนมากกว่าช่องโหว่ของระบบปฏิบัติการ (Operating System) เนื่องจากการใช้งานแอพพลิเคชันแบบ web based มีเทคโนโลยีที่ช่วยในการบริหารจัดการและบำรุงรักษาที่ ง่าย รวดเร็ว และทั่วถึงกว่าแอพพลิเคชัน Client/Server แบบเดิม แต่ในขณะเดียวกันการที่เปิดกว้างทำให้เกิดการโจมตีจากภายนอกได้ง่ายเช่นเดียวกัน [28]

2.3.1 สาเหตุที่พบเป็นประจำของช่องโหว่ที่เกิดในเว็บแอพพลิเคชัน

- การเขียนโปรแกรมที่ไม่รัดกุม หรือ ไม่ได้คำนึงถึงการรักษาความปลอดภัยอย่างเพียงพอ (Secure Coding) เช่น Input validation [29] ที่ไม่เหมาะสม การไม่มี Session tracking [30] หรือ Cookie [31] , integrity check [32] เป็นต้น
- การพบช่องโหว่ หรือ ข้อผิดพลาด ในระบบที่ทำงานร่วมกันเป็นเว็บแอพพลิเคชัน เช่น ช่องโหว่ที่แอพพลิเคชันเซิร์ฟเวอร์ ช่องโหว่ที่เว็บเซิร์ฟเวอร์ หรือ ช่องโหว่ของระบบปฏิบัติการที่ระบบต่างๆ ทำงานอยู่
- การกำหนดค่า Configuration ที่ไม่เหมาะสม หรือ ไม่ปลอดภัยในขณะที่ทำการพัฒนาระบบ หรือการใช้ค่า default ในระหว่างติดตั้ง

2.3.2 ประเภทของการโจมตีผ่านเว็บแอพพลิเคชัน

OWASP (The Open Web Application Security Project) [33] ซึ่งเป็นองค์กรไม่แสวงหากำไรที่เกิดจากการรวมตัวของผู้ชำนาญการทางสายต่างๆ เพื่อวิเคราะห์ เฝ้าระวัง และให้ความรู้แก่บุคคลทั่วไปในเรื่องภัยคุกคามบนเว็บแอพพลิเคชัน ได้จัดทำ OWASP Top 10 [34] ซึ่งรวบรวมความเสี่ยงสูงสุด 10 อันดับแรก ซึ่งการจัดลำดับครั้งล่าสุดได้ทำขึ้นในปี 2013 (พ.ศ. 2556) โดย OWASP Top 10

Release Candidate 1 ได้จัดอันดับความเสี่ยง (ประเภท) ต่อการโจมตีเว็บแอพพลิเคชันไว้ดังรูปที่ 2.11 โดย 4 อันดับแรกมีรายละเอียดดังนี้

1. Injection

เป็นวิธีการโจมตีโดยใช้ช่องโหว่ของแอพพลิเคชันที่เขียนมาไม่รัดกุมหรือปลอดภัย โดยใส่โค้ดประเภท SQL คำสั่งของระบบปฏิบัติการหรือ LDAP Injection [35] โดยข้อมูลเหล่านี้จะถูกนำไปประมวลผลหรือใช้เพื่อเข้าถึงข้อมูลที่ต้องการได้ เช่น list directory หรือ login เข้าสู่ระบบโดยไม่ต้องทราบ username และ password เป็นต้น

2. Cross Site Scripting (XSS)

การโจมตีแบบ XSS เกิดขึ้นจากการที่เว็บเซิร์ฟเวอร์รับข้อมูลที่ไม่ประสงค์ดีไว้ เช่นการที่ให้ผู้ใช้โพสต์ความคิดเห็นโดยไม่มีการตรวจสอบสิ่งที่ผู้ใช้โพสต์ผู้ใช้จึงโพสต์สคริปต์ที่อันตรายแทนความคิดเห็น และสคริปต์นั้นจะถูกส่งไปยังเบราว์เซอร์โดยไม่มีการตรวจสอบข้อมูลก่อน ซึ่งทำให้ผู้โจมตีสามารถสั่งให้สคริปต์ทำงานบนเครื่องของเหยื่อ ซึ่งสามารถใช้เปลี่ยนแปลงหน้าเว็บ หรือ ขโมย session (Session Hijacking) หรือ นำเหยื่อไปยังหน้าเว็บที่ไม่ต้องการได้

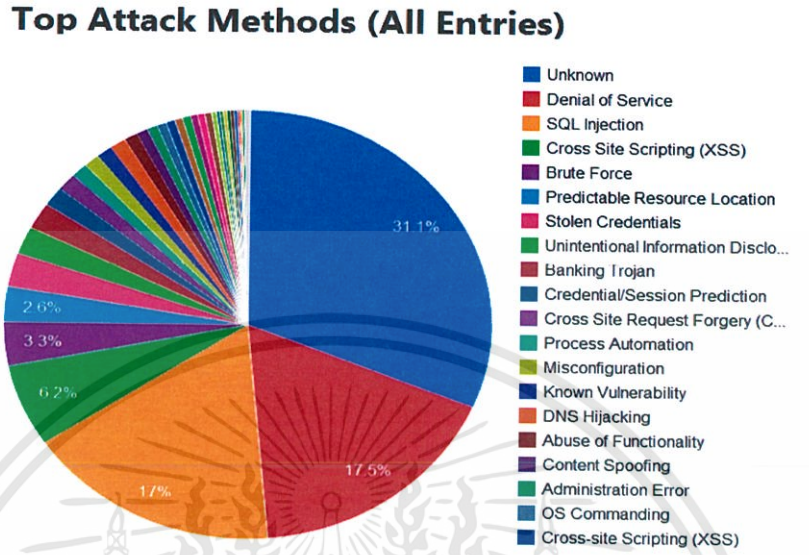
3. Broken Authentication and Session Management

ในบางครั้งการพัฒนาระบบในส่วนของการทำ Authentication อาจมีข้อผิดพลาดที่ทำให้ผู้ไม่ประสงค์ดี สามารถสวมรอยเป็นบุคคลอื่นได้

4. Insecure Direct Object References

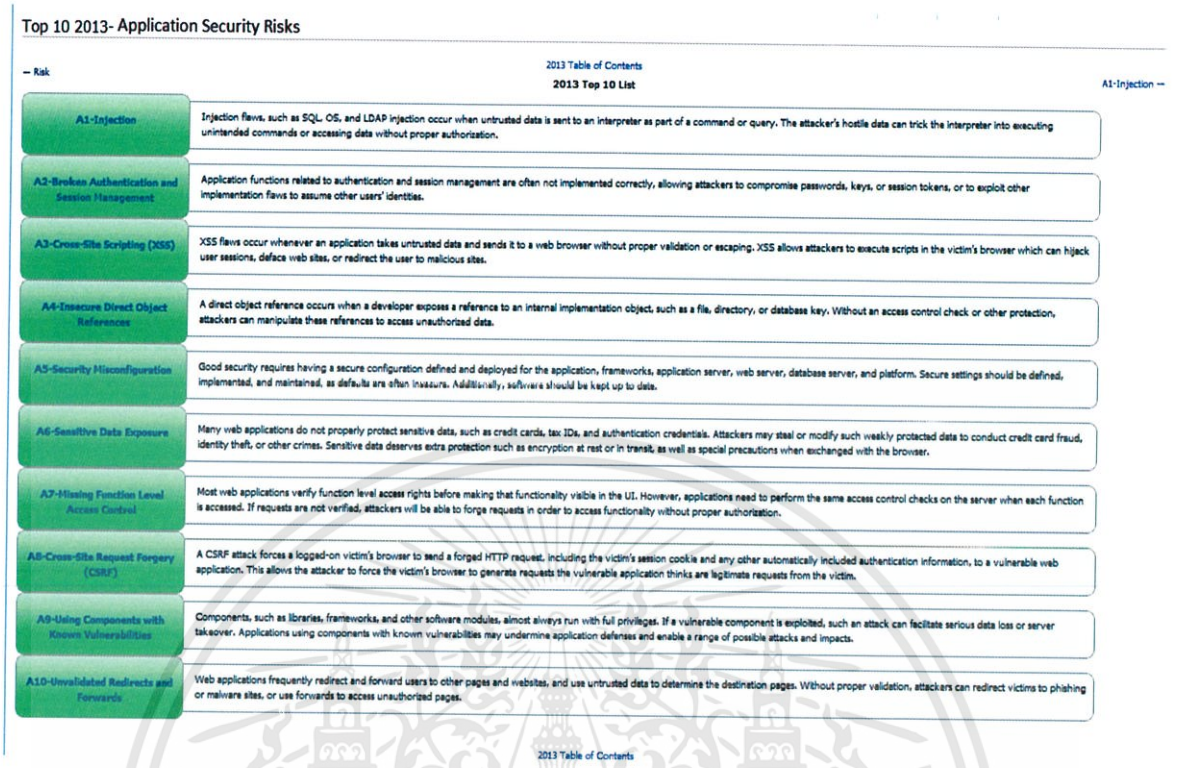
เกิดจากการอ้างอิงถึง Object ที่ใช้ภายใน เช่น ไฟล์ ไดเร็กทอรี่ หรือคีย์ของฐานข้อมูล โดยไม่มีการตรวจสอบสิทธิ์หรือควบคุมการเข้าถึง ซึ่งทำให้ผู้โจมตีสามารถใช้การอ้างอิงนี้เข้าถึงข้อมูลที่ไม่ได้รับอนุญาตได้

2.3.3 SQL Injection



รูปที่ 2.10 ประเภทของการโจมตีที่ได้รับความนิยมประจำปี 2013 จาก builtwith.com

SQL Injection [2] เป็นหนึ่งในช่องโหว่ที่ร้ายแรงที่สุดที่ส่งผลกระทบต่อธุรกิจทำให้ข้อมูลสำคัญที่เก็บไว้ในฐานข้อมูลของโปรแกรมประยุกต์ รวมถึงข้อมูลที่มีประโยชน์ เช่น ชื่อผู้ใช้ รหัสผ่าน ที่อยู่ หมายเลขโทรศัพท์ และรายละเอียดของบัตรเครดิตมีความเสี่ยงในการรั่วไหล ถูกแก้ไข หรือถูกทำลาย ซึ่งมีสาเหตุมาจากการที่ผู้โจมตีสามารถแก้ไข เปลี่ยนแปลงคำสั่ง SQL ที่ทางเว็บเซิร์ฟเวอร์ใช้ส่งไปยังฐานข้อมูล back-end โดยใช้ประโยชน์จากไวยากรณ์และความสามารถของภาษา SQL โดย SQL Injection ไม่ได้เป็นช่องโหว่ที่มีผลกระทบเฉพาะเว็บแอพพลิเคชันเท่านั้น โปรแกรมที่รับข้อมูลจากแหล่งที่ไม่น่าเชื่อถือ แล้วส่งอินพุตในรูปแบบ dynamic SQL statements ไปที่ฐานข้อมูลถือว่าเป็นช่องโหว่ได้ทั้งหมด (เช่นการใช้งาน " fatclient " ในสถาปัตยกรรมไคลเอ็นท์ / เซิร์ฟเวอร์) [36] ในอดีต SQL Injection มักถูกใช้ในการโจมตีกับฐานข้อมูลฝั่งเซิร์ฟเวอร์ แต่ในปัจจุบันด้วยข้อกำหนด HTML5 ผู้โจมตีสามารถรัน JavaScript หรือโค้ดอื่น ๆ เพื่อโต้ตอบกับฐานข้อมูลฝั่งผู้ใช้ (Client Side Database) เพื่อจะขโมยข้อมูลได้



รูปที่ 2.11 10 อันดับการโจมตีที่นิยมในปี 2013 จากการจัดอันดับของ www.owasp.org

ในการโจมตีด้วย SQL Injection ผู้โจมตีสามารถทำการแทรกหรือต่อท้ายโค้ดลงไปในแอพพลิเคชันผ่านทางค่าที่ผู้ใช้ป้อนข้อมูลเข้ามา (User input parameters) ซึ่งค่านี้จะถูกส่งผ่านไปยังเว็บเซิร์ฟเวอร์ เพื่อให้เว็บเซิร์ฟเวอร์นำค่านี้ไปใช้ในการสร้างคำสั่ง SQL ซึ่งอาจเป็นช่องโหว่ให้ผู้ไม่หวังดีเข้าถึงข้อมูลได้ โค้ดที่เป็นอันตรายจะถูกดำเนินการเมื่อโปรแกรมประยุกต์บนเว็บล้มเหลวในการตรวจจับอย่างถูกต้อง ค่าที่ถูกป้อนเข้ามาจะถูกส่งผ่านไปยังคำสั่ง SQLที่ถูกสร้างแบบไดนามิก (ในหนึ่งประโยคของ SQL มีการนำสตริงที่สร้างไว้มารวมกับค่าข้อมูลที่ผู้ใช้ป้อนเข้ามา) ซึ่งทำให้ผู้โจมตีสามารถเปลี่ยนประโยค SQL ให้เป็นไปอย่างที่ต้องการได้

ตัวอย่างเช่น ถ้าแอพพลิเคชันออนไลน์ต้องการดูสินค้าทั้งหมดภายในร้านค้าที่มีราคาน้อยกว่า $100 โดยใช้ URL ต่อไปนี้: products.php?val=100 ถ้าผู้โจมตีป้อนคำสั่ง SQL ต่อท้ายให้กับพารามิเตอร์ Val โดยการผนวกสตริง 'OR '1' = '1 ไปยัง URL: products.php?val=100' OR '1'='1

เนื่องจากคำสั่ง SQL ในส่วนของการจัดการค้นหาข้อมูลในฐานข้อมูลคือ

SELECT * FROM ProductsTbl WHERE Price < '100.00' OR '1' = '1' ORDER BY ProductDescription;

(ตัวอักษรสีน้ำเงินคือค่าของสตริง และสีแดงคือค่าที่ขึ้นอยู่กับค่าที่ผู้ใช้ป้อนเข้ามา)

ถ้าผู้โจมตีป้อนข้อมูลดังกล่าว คำสั่ง SQL ที่สคริปต์ของการคิวรี่จะถูกเปลี่ยนเป็น

```
SELECT *
FROM ProductsTbl
WHERE Price < '100.00' OR '1' = '1'
ORDER BY ProductDescription;
```

การต่อท้ายคำสั่งด้วย OR การดำเนินการของการคิวรี่จะให้ผลเป็นจริงเสมอ นั่นคือ 1 จะเท่ากับ 1 ดังนั้นกรณีนี้ไม่ว่าค่าของราคาจะเป็นเท่าใดข้อมูลจะแสดงออกมาทั้งหมด

การใช้ประโยชน์จากช่องโหว่ SQL Injection มีมากมายหลายรูปแบบ ซึ่งในบางครั้งผู้โจมตีต้องใช้ทักษะและความพยายามในการลองข้อมูลหลายวิธี

2.3.4 ประเภทของ SQL injection

1) Basic SQL Injection

ลักษณะการทำงานของ SQL Injection คือการแทรกโค้ดเข้าไปจากพื้นที่ที่ชุดคำสั่งอนุญาตให้เขียนได้ เช่น คำสั่ง SQL ชุดหนึ่งที่เขียนด้วยภาษา PHP ใช้ในการตรวจสอบการเข้าถึงของระบบ โดย ชื่อผู้ใช้และรหัสผ่านซึ่งเขียนได้ดังนี้

“SELECT * FROM `users` WHERE username=’$user’ AND password=’$pass’ ”

จากชุดคำสั่ง SQL ข้างต้นอธิบายได้ดังนี้

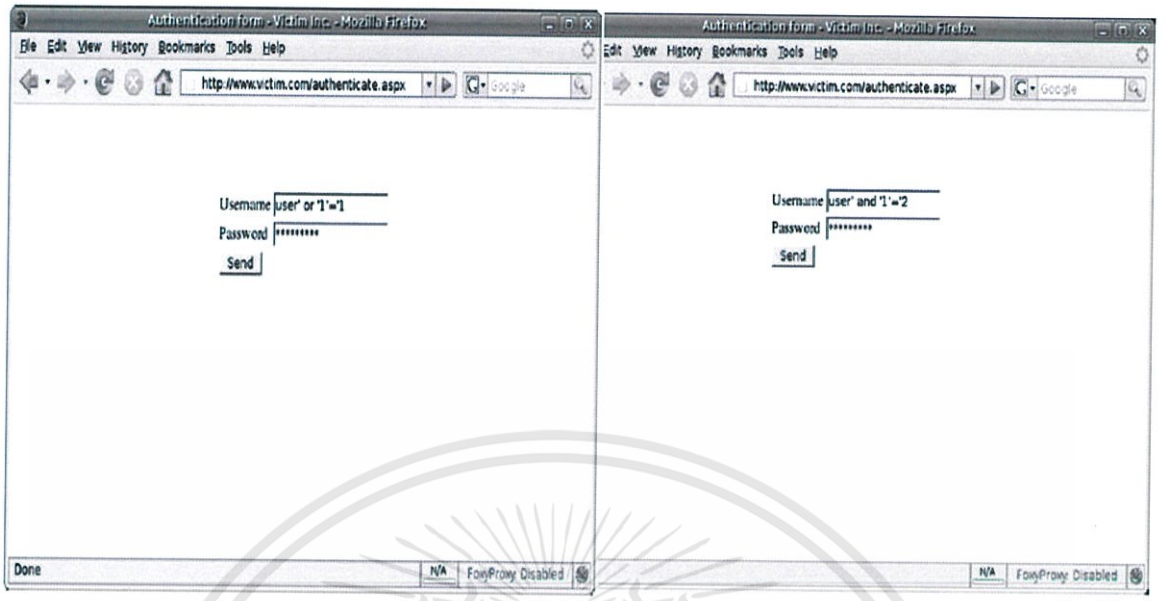
- “ ” คือ Double Quote เป็นการบอกว่าข้อมูลภายใน “ ” เป็น String
- ตัวอักษรตัวใหญ่ คือ Reserve word ในการเขียนคำสั่ง SQL
- * สัญลักษณ์บอกว่า ทั้งหมด
- การแทรกตัวแปรเพื่อไปต่อสตริงวิธีการคือใช้ ‘ ’ (Single Quote) ครอบสตริงที่จะแทรก เช่น ’$user’

ในส่วนข้อมูลนำเข้า จะอยู่ในตัวแปรของ $user และ $pass

หากเรากำหนดให้ $user = admin และ $pass = 123 ชุดคำสั่งของ SQL จะเป็นดังนี้

“SELECT * FROM `users` WHERE username=’admin’ AND password=’123’ ”

ซึ่งชุดคำสั่งนี้ SQL Engine จะไปค้นหา record ในตาราง users ที่มี username เป็น admin และ password เป็น 123 หากค้นหาเจอจะยืนยันตัวตนได้ว่ามี username ที่ใช้ password นี้อยู่จริง แต่ในการโจมตี ผู้โจมตีจะให้ SQL นี้คืนค่าออกมาเป็นจริงเสมอโดยเติม ‘or’x’=’x ลงไปในช่อง Password เสมือนกับว่ายืนยันตัวตนได้ถูกต้องทั้งที่ไม่ทราบพาสเวิร์ดดังตัวอย่างต่อไปนี้



รูปที่ 2.12 ตัวอย่างหน้าต่างการเข้าสู่ระบบ

"SELECT * FROM `users` WHERE username='' AND password=''or'x'='x' "

ผลลัพธ์ที่เกิดจากการ Injection นี้ จะเปลี่ยน statement การทำงานของเงื่อนไข โดยขั้นตอนเป็นดังนี้

username='' AND password='' OR 'x'='x' : Overall

username='' AND password='' OR 'x'='x' : Check 1st Statement, True or False

(username='' AND password='') OR 'x'='x' : Check 2nd Statement, True Only

ซึ่งเมื่อมีการเทียบกับ record แต่ละ record แล้ว ไม่ว่าเงื่อนไขแรกจะเป็นจริงหรือไม่ แต่ เงื่อนไขสุดท้ายจะเป็นจริงเสมอ ดังนั้นแม้ไม่ทราบพาสเวิร์ดแต่จะสามารถเข้าระบบได้

2) SQL Injection ที่เกิดจากการใช้ Line Comment

เป็นลักษณะการใช้งานคอมเม้นท์ (comment) ในบรรทัดของชุดคำสั่ง SQL เพื่อกันคำสั่งที่ไม่ต้องการออกหลังใช้คอมเม้นท์นี้ สัญลักษณ์ที่ใช้แสดงคอมเม้นท์มักจะมีอยู่สองรูปแบบดังนี้

1. -- ใช้กับ MySQL เป็นขีดสองตัว แล้วตามด้วย whitespace อย่างน้อย 1 อักขระ
2. # ใช้กับ MySQL

ตัวอย่างนี้จะเป็นวิธีการได้สิทธิ์ของผู้ใช้งานหรือผู้ดูแลระบบในการเข้าใช้งานระบบด้วยวิธี SQL Injection แบบ Line comments โดยชุดคำสั่งที่ใช้เป็นชุดคำสั่งเดียวกับหัวข้อข้างต้นคือ

SELECT * FROM `users` WHERE username='$user' AND password='$pass'

การ bypass พาสเวิร์ดสามารถทำได้ดังนี้

ในช่อง user ให้ใส่เป็น ชื่อ user'-- (ตามด้วยเว้นวรรค) หรือ ชื่อ user'# ชุดคำสั่งจะเปลี่ยนเป็น

SELECT * FROM `users` WHERE username='admin'-- ' AND password=''

ซึ่งคำสั่งนี้คือการสั่งให้ฐานข้อมูลแสดงค่า record ที่มี username ผู้โจมตีสามารถสร้างให้เงื่อนไขเป็นจริงเสมอได้ด้วยการใส่ข้อมูลลงในช่อง user เป็น ' or 'x' = 'x'-- (ตามด้วยเว้นวรรค) หรือ ' or 'x' = 'x'# จะได้เป็น

SELECT * FROM `users` WHERE username='' or 'x'='x'-- ' AND password=''

3) SQL Injection ที่เกิดจากการใช้ Inline comment

เป็นลักษณะการใช้งานคอมเม้นท์ในบรรทัดของชุดคำสั่ง SQL กันคำสั่งที่ไม่ ต้องการที่อยู่ระหว่างสัญลักษณ์ของคอมเม้นท์ออก และมีวิธีการใช้ดังนี้

/* This is Comments */

ตัวอย่างนี้จะเป็นวิธีการได้สิทธิ์ของผู้ใช้งาน หรือผู้ดูแลระบบในการเข้าใช้งานระบบ ด้วยวิธี SQL Injection แบบ Inline comment ซึ่ง bypass พาสเวิร์ดได้โดย

ในช่อง user ให้ป้อนข้อมูลเป็น ชื่อuser'/*

ในช่อง pass ให้ป้อนข้อมูลเป็น */ or 'x' = 'y

ชุดคำสั่งจะเปลี่ยนเป็น

SELECT * FROM `users` WHERE username='admin'/*' AND password='*/ or 'x' = 'y'

ซึ่ง คำสั่งนี้จะไปค้นหาข้อมูลที่มี username เป็น admin มาให้

เนื่องจากเงื่อนไขหลังจะเป็นเท็จเสมอ การเชื่อมด้วย OR จึงสามารถให้เป็นจริงได้ด้วยเงื่อนไขแรกเท่านั้นดังนั้นในกรณีที่ไม่ทราบ user ผู้โจมตีสามารถทำให้เงื่อนไขในส่วนนี้เป็นจริงเสมอได้ด้วยการใส่ Inline comment ดังนี้

ในช่อง user ให้ป้อน '/* ในช่อง pass ให้ป้อน */ or 'x' = 'x ซึ่งชุดคำสั่งจะเปลี่ยนเป็น

SELECT * FROM `users` WHERE username=''/*' AND password='*/ or 'x' = 'y'

4) Union SQL Injections

เป็นลักษณะการใช้งานคำสั่ง Union เพื่อใช้ในการประมวลผลข้ามตารางผลลัพธ์ของข้อมูลแถวที่ได้จะรวมตารางที่เราใช้ Union เข้าไปด้วย เช่น จากเดิมคำสั่งคือ

SELECT header, txt FROM news

แทรกคำสั่ง Union ต่อท้ายจะเป็น

SELECT header, txt FROM news UNION ALL SELECT name, pass FROM members

ผลลัพธ์ของการคิวรี่คำสั่งนี้จะเป็นการนำข้อมูลจากตาราง news และตาราง members มาเป็นผลลัพธ์ของการค้นหา

2.3.5 สาเหตุที่ทำให้เกิด SQL Injection

1) การกรอง (Filter) Escape Characters ที่ผิดพลาด

รูปแบบของ SQL Injection ลักษณะนี้จะเกิดขึ้นเมื่อข้อมูลที่ผู้ใช้ป้อนเข้าไปไม่ได้ผ่านการกรอง Escape Character ก่อนที่จะนำไปรวมกับชุดคำสั่ง SQL ดังนั้นช่องโหว่นี้เปิดโอกาสให้ผู้ไม่หวังดี ทำการป้อนข้อมูลที่มีการใช้ประโยชน์ของ Escape Characters ต่างๆ เช่น single quote (') comment (--) และ inline comment (/* */) ได้ (Escape Characters คือ ตัวอักษร หรืออักขระที่เมื่อโปรแกรมคอมพิวเตอร์แปลความหมายของตัวอักขระที่ตามหลังตัวอักขระนี้จะไม่แปลความหมายตามปกติ แต่จะแปลความหมายในลักษณะของ metacharacter เช่น \n ในคำสั่ง printf ใน ภาษาซีจะถูกแปลความหมายเป็นให้เว้นหนึ่งบรรทัด แทนค่า n จริง) บรรทัดต่อไปนี้แสดงตัวอย่างโค้ดที่มีช่องโหว่

```
statement := "SELECT * FROM users WHERE name = '" + userName + "';"
```

โค้ด SQL นี้ถูกออกแบบเพื่อดึงค่า record ของตาราง usersโดยเจาะจงชื่อผู้ใช้ (username) ให้เป็นค่าที่ป้อนเข้ามา ดังนั้นถ้า username ที่รับมามีค่าเป็น 'a' or 't'='t' จะได้คำสั่ง SQL ที่ส่งไปฐานข้อมูลดังนี้:

```
SELECT * FROM users WHERE name = 'a' or 't'='t';
```

ซึ่งถ้าโค้ดนี้ถูกนำไปใช้ในการตรวจสอบสิทธิ์ Authentication Procedure ถ้าไม่มีการกรองข้อมูลที่ป้อนเข้ามาอาจจะส่งผลให้เกิดการยืนยันสิทธิ์ที่ผิดพลาด เนื่องจากค่าของ 't' = 't' เป็นจริงเสมอ นอกจากนี้การที่ SQL Servers จำนวนหนึ่งเช่น MSSQL Server อนุญาตให้เรียกใช้ชุดคำสั่ง SQL หลายชุดคำสั่งในครั้งเดียวได้ ทำให้ผู้โจมตีสามารถใช้ช่องโหว่นี้ในการป้อนข้อมูลลักษณะดังต่อไปนี้ได้

```
'a';DROP TABLE users; SELECT * FROM data WHERE name LIKE '%'
```

ซึ่งข้อมูลที่ป้อนเข้าไปนี้จะถูกนำไปแทนค่า userName ทำให้ได้รูปแบบคำสั่ง SQLในขั้นสุดท้ายดังนี้

```
SELECT * FROM users WHERE name = 'a';DROP TABLE users; SELECT * FROM data WHERE
name LIKE '%';
```

ซึ่งผลลัพธ์คือ ตาราง users จะถูกลบทิ้งไปซึ่งอาจส่งผลให้แอพพลิเคชันใช้งานไม่ได้

2) ความผิดพลาดในการกำหนดชนิดข้อมูล (Incorrect type handling)

SQL Injection รูปแบบนี้เกิดขึ้นเมื่อส่วนรับข้อมูลจากผู้ใช้ไม่ได้กำหนดชนิดข้อมูลที่รับเข้ามาอย่างเหมาะสมหรือไม่ได้ตรวจสอบข้อกำหนดเกี่ยวกับชนิดข้อมูล เช่น ในคำสั่ง SQL นั้นชนิดของข้อมูลที่ต้องการคือตัวเลขแต่ผู้เขียนโปรแกรมไม่ได้ตรวจสอบว่าข้อมูลที่ผู้ใช้ป้อนเข้ามาเป็นข้อมูลชนิดตัวเลขหรือไม่ ทำให้ผู้ไม่ประสงค์ดีสามารถใส่ข้อความ (สตริง) เข้ามาได้ ตัวอย่างเช่น

```
statement := "SELECT * FROM data WHERE id = " + a_variable + ";"
```

3) ช่องโหว่แบบ Blind SQL Injection

เว็บแอพพลิเคชันบางแอพพลิเคชันจะแสดงข้อความ error ให้ผู้ใช้ทราบว่ามีความผิดพลาดอย่างไร ถ้าไวยากรณ์ (syntax) หรือการสืบค้นข้อมูลในเซิร์ฟเวอร์ของฐานข้อมูลนั้นไม่ถูกต้อง (เช่น ชื่อฟิลด์ผิด จำนวนคอลัมน์ที่สอบถามเกินกว่าจำนวนคอลัมน์ที่มีอยู่) ซึ่งผู้โจมตีจะใช้ประโยชน์จากข้อความ error เหล่านี้โดยนำไปวิเคราะห์หาคุณสมบัติหรือองค์ประกอบของข้อมูลในฐานข้อมูลที่ต้องการ วิธีการของการโจมตีนี้คือจะส่งอินพุตในลักษณะคำถามที่ฐานข้อมูลจะให้ผลลัพธ์เป็นคำตอบว่าจริงหรือเท็จ ผู้โจมตีจะถามจนกว่าจะได้คำตอบที่ต้องการ มักใช้กับเว็บแอพพลิเคชันที่ถูก configure ให้แสดงข้อความ error

2.3.6 การเกิด SQL Injection ใน PHP

ในภาษา PHP การเกิด SQL Injection เกิดขึ้นจากช่องโหว่ของคำสั่งในการติดต่อกับฐานข้อมูล คือฟังก์ชัน mysql_query( ) ในการสร้างสตริง เนื่องจากฟังก์ชัน mysql_query( ) นี้ไม่ได้กรอง Escape Characters จึงเป็นช่องโหว่ให้ผู้โจมตีสามารถแทรกโค้ด SQL ที่มี Escape Characters ต่างๆ เช่น ‘ ’ (single quote) และ “ ” (double quote) เพื่อทำ SQL Injection ดังตัวอย่างในบทที่ 2.3.4 ได้

2.4 การป้องกัน SQL Injectionโดยการเขียนโค้ดที่ปลอดภัย

2.4.1 การทำ Whitelist และ Blacklist

1) ป้องกันการใช้คำสั่งมากกว่าหนึ่งคำสั่ง (Preventing multi-statement attacks)

วิธีหนึ่งในการป้องกันคือไม่อนุญาตให้มีการเรียกใช้คำสั่ง SQL มากกว่าหนึ่งคำสั่งในการร้องขอหนึ่งครั้งซึ่งจะป้องการการโจมตีที่ทำการเพิ่มคำสั่งลงไปในข้อมูลที่ป้อนเข้ามา เช่น การเพิ่มโดยเติม (;) เช่น a';DROP TABLE users; แต่วิธีการนี้จะใช้ได้เฉพาะการแทรกชุดคำสั่งเท่านั้นไม่สามารถป้องกันได้ทุกกรณีเช่น ถ้าในหน้าเว็บมีการแสดงรายการสิ่งของ ของชื่อผู้ใช้งาน(username) ที่ได้รับมาซึ่งสามารถเขียนคำสั่ง การร้องขอ SQL ได้ดังนี้

```
SELECT * from items where username='$username';
```

ถ้าผู้โจมตีใส่ชื่อผู้ใช้งานในส่วนของ username ($username) ดังนี้

```
' or username is not null or username='
```

คำสั่ง SQL ที่ได้คือ

```
SELECT * from items where username='' or username is not null or username='';
```

ซึ่งในกรณีนี้ถ้า table items มีข้อมูล username is not null จะเป็นจริงเสมอ

2) การป้องกันโดยการกำจัด single quote

ตัวอย่างในข้อ 1) Basic SQL Injection จะป้องกันได้ถ้ามีการกำจัด single quote ออกจากสิ่งที่ผู้ใช้อินป้อนข้อมูลมา แต่การกำจัด single quote ไม่สามารถป้องกันความเสี่ยงจาก SQL injection ได้อย่างสมบูรณ์ เช่น ถ้ารูปแบบการร้องขอ SQL เป็นดังนี้

SELECT * from items where userid=$userid;

ถ้าไม่มีการตรวจสอบชนิดข้อมูลของ $userid ว่าควรเป็นตัวเลขเท่านั้น การให้ค่าแก่ $userid ดังต่อไปนี้ สามารถเปิดเผยรายชื่อสิ่งของทั้งหมดของผู้ใช้งานทุกคนได้

33 or userid is not null or userid=44

ซึ่งจะพบว่าไม่มีการใส่ single quote ในข้อมูลที่ป้อนเข้ามา แต่ฐานข้อมูลสามารถถูกโจมตีได้ด้วยคำสั่งดังนี้

SELECT * from items where userid=33 or userid is not null or userid=44;

ดังนั้นวิธีที่ดีที่สุดในการป้องกันนอกจากการการทำบัญชีดำ (Blacklist) (กำหนดว่าสิ่งใดไม่อนุญาตให้รับเข้าสู่ระบบ) สำหรับข้อมูลที่อาจก่อให้เกิดความผิดพลาดแล้ว ควรอนุญาตเฉพาะข้อมูลที่ให้ผลลัพธ์ที่ถูกต้อง (Whitelist) (มีการตรวจสอบชนิดของข้อมูลด้วยและจะทำงานเมื่อชนิดของข้อมูลถูกต้องเท่านั้น) เช่น ในตัวอย่างนี้ จำเป็นต้องมีการตรวจสอบด้วยว่าตัวแปร $userid นั้น เป็นข้อมูลประเภทตัวเลขจริงก่อนนำไปใช้ในคำสั่ง SQL

2.4.2 Parameterized Statement หรือ Bind Variable

สาเหตุหลักของการเกิด SQL Injection คือการที่เว็บเซิร์ฟเวอร์สร้างชุด คำสั่ง SQL เก็บไว้ในตัวแปรชนิดสตริงแล้วจึงส่งค่าสตริงนี้ไปให้ฐานข้อมูล ซึ่งค่าของชุดคำสั่ง จะขึ้นอยู่กับค่าที่ผู้ใช้ป้อนเข้ามา (การสร้างสตริงในลักษณะนี้มีชื่อเรียกว่า dynamic string building หรือ dynamic SQL) เพื่อความปลอดภัยภาษาโปรแกรมหลายภาษาจึงหลีกเลี่ยงปัญหาที่เกิดจากการสร้างสตริงแบบไดนามิก โดยการเก็บค่าต่างๆ ที่ผู้ใช้ป้อนเข้ามาไว้ที่ที่เก็บชั่วคราว (placeholder) หรือผูกไว้กับตัวแปร (bind variable) แล้วส่งค่านี้เป็นพารามิเตอร์ไปที่ประโยค SQL หรือที่รู้จักกันในนาม Parameterized Statement

การใช้ placeholder ทำให้การผูกข้อมูลที่ได้รับมาลงไปในคำสั่ง SQL กับการเขียนคำสั่ง SQL ถูกแยกออกจากกันดังนั้นผู้เขียนโปรแกรมไม่ต้องคอยกังวลในการกรองข้อมูลสำหรับทุกๆ ค่าที่จะได้รับมาจากผู้ใช้งาน

ฐานข้อมูลแต่ละชนิดจะมีการเรียกใช้ placeholders โดยใช้ฟังก์ชันที่แตกต่างกัน ยกตัวอย่างเช่น mysql_stmt_bind_param สำหรับ MySQL หรือ oci_bind_by_name สำหรับ Oracle

การเข้าถึงฐานข้อมูลด้วยภาษา PHP ทำได้หลายวิธี หนึ่งในวิธีที่นิยมมากคือการใช้แพคเกจ mysqli ซึ่งจะใช้สัญลักษณ์เครื่องหมายคำถาม “?” เป็น placeholder ในการผูกกับตัวแปรเวลาสร้างคำสั่ง SQL ในตัวอย่างต่อไปนี้ถ้าไม่มีการใช้พารามิเตอร์ คำสั่ง SQL ถูกสร้างแบบวิธีไดนามิกด้วยคำสั่งต่อไปนี้

```
Sql = "SELECT * FROM users WHERE username='" + Username + "' AND password='"+ Password
+ """
```

ซึ่งถ้ามีการใช้ placcholdcr จะเขียนใหม่ได้ดังนี้

```
$sql = "SELECT * FROM users WHERE username=? AND password=?";
$cmd = $con->prepare($sql);
// Add parameters to SQL query
$cmd->bind_param("ss", $username, $password);
// bind parameters as strings
$cmd->execute();
```

ซึ่งอธิบายได้ว่าให้นำค่าที่ผู้ใช้ป้อนเข้ามาไปใส่ตัวแปรชื่อ $username และ $password ตามลำดับซึ่งทั้งสองตัวแปรมีชนิดเป็นสตริง ("ss" บอกถึงชนิดของตัวแปรโดย "s" ตัวแรกคือชนิดของตัวแปร $username และ "s" คัวที่สองคือชนิดตัวแปร $password) และในคำสั่ง $sql ให้นำค่าจาก ตัวแปร $username และ $password มาใส่แทนเครื่องหมาย ? ตัวที่หนึ่งและตัวที่สองตามลำดับ ซึ่งการผูกตัวแปรจะเป็นการให้ค่ากับพารามิเตอร์ตัวนั้นๆ โดยไม่มีการนำค่าของตัวแปรนั้นไปแปลความหมายเสมือนเป็นส่วนหนึ่งของสตริงของคำสั่ง SQL [59]

2.4.3 Stored Procedures

Stored Procedure [24] คือ procedure ชุดคำสั่ง SQL บนฐานข้อมูลซึ่งผู้พัฒนาฐานข้อมูลเขียนไว้เพื่อให้แอพพลิเคชันที่เรียกใช้ฐานข้อมูลนั้นสามารถเข้าถึงข้อมูลได้โดยไม่จำเป็นต้องเขียนชุดคำสั่ง SQL แต่จะเรียกใช้โดยทำการ call procedure และส่งพารามิเตอร์ของค่าที่ต้องการไปแทน [1] ดังนั้นการที่แอพพลิเคชันจะเรียกใช้ procedure เหล่านี้ได้นั้นทางฐานข้อมูลจะต้องสร้าง procedure เหล่านี้ขึ้นมาก่อน ส่วนแอพพลิเคชันที่เรียกใช้งานทราบเพียงแค่ชื่อ procedure และพารามิเตอร์ที่ต้องส่งเท่านั้น

stored procedure มีข้อได้เปรียบในด้านความปลอดภัยคือ การเรียกใช้งานในรูปแบบของพารามิเตอร์ และการบังคับชนิดข้อมูลของข้อมูลที่ได้รับ ทำให้การกรองข้อมูลที่ได้รับมาจากผู้ใช้ทำได้อย่างมีประสิทธิภาพ นอกจากนั้นฐานข้อมูลยังสามารถกำหนดสิทธิในการรันคำสั่งใน stored

procedures ให้แตกต่างกันสำหรับแต่ละผู้ใช้งานฐานข้อมูล ยกตัวอย่างเช่น แอพพลิเคชันหนึ่งสามารถรันข้อมูลจาก stored procedures แต่ไม่สามารถเข้าถึงข้อมูลในตารางโดยตรงได้ ซึ่งเป็นการจำกัดไม่ให้แอพพลิเคชันสามารถทำสิ่งที่นอกเหนือจากที่กำหนดไว้ใน stored procedures ได้ อย่างไรก็ตาม store procedure ไม่ได้ป้องกัน SQL Injection ได้ทั้งหมด

2.4.4 ยกเลิกการใช้งานตัวอักษรทั่วไป (Disabling literals)

ปัญหา SQL Injection จะสามารถแก้ไขได้ถ้าฐานข้อมูลนั้นๆ สนับสนุนการทำงานที่เรียกว่า disabling literals คือการที่ฐานข้อมูลจะไม่อนุญาตให้ข้อมูลในรูปแบบข้อความและรูปแบบตัวเลขเป็นส่วนหนึ่งของรูปแบบคำสั่ง SQL จะอนุญาตเฉพาะข้อมูลในลักษณะตัวแทน (placeholder) เท่านั้น ดังนั้นรูปแบบคำสั่ง SQL ดังต่อไปนี้

```
SELECT * FROM USER WHERE NAME='Smith'
SELECT * FROM ITEMS WHERE USERID=2
```

จะไม่ได้รับอนุญาตให้ทำงานได้ (ฐานข้อมูลจะส่งข้อผิดพลาดกลับคืนมา) ดังนั้นรูปแบบคำสั่งที่ถูกต้องจะต้องเป็นในลักษณะนี้

```
SELECT * FROM USER WHERE NAME=?
SELECT * FROM ITEMS WHERE USERID=?
```

การทำงานแบบ disabling literals จะบังคับให้ต้องใช้ placeholders ในการเขียนคำสั่ง SQL สำหรับทุกข้อมูลที่ได้รับมาจากผู้ใช้ ดังนั้นการโจมตีด้วย SQL Injection ในรูปแบบที่กล่าวไปข้างต้นจึงไม่สามารถเป็นไปได้ ปัจจุบันมีเพียงแค่ H2 database engine เท่านั้นที่สนับสนุนการใช้งาน disabling literals อย่างไรก็ตามเทคโนโลยีนี้ไม่ได้มีการจดสิทธิบัตรจึงทำให้มีโอกาสที่ฐานข้อมูลอื่นจะพัฒนาคุณสมบัตินี้ได้

2.4.5 การกำหนดระดับความสำคัญในการใช้งาน (Security Privileges)

การกำหนดระดับความสำคัญในการใช้งานฐานข้อมูลเท่าที่จำเป็นอนุญาตให้เข้าถึงข้อมูลเฉพาะที่ผู้ใช้มีสิทธิ์เป็นวิธีการป้องกันพื้นฐานวิธีหนึ่ง ดังนั้นจึงไม่ควรอนุญาตให้ผู้ใช้ที่มี Privilege ต่ำสามารถเพิ่มหรือลบตารางหรือฐานข้อมูลได้ อย่างไรก็ดีวิธีการนี้เพียงแค่สามารถลดความเสียหายจากการถูกโจมตีด้วย SQL Injection ได้เท่านั้นไม่สามารถแก้ปัญหา SQL Injection ได้

หัวข้อ 2.4.6 และ 2.4.7 แสดงวิธีแก้ไขเฉพาะของ PHP

2.4.6 การป้องกันโดยใช้ฟังก์ชัน mysql_real_escape_string( )

เป็นฟังก์ชันสำหรับเลี่ยงการใช้ตัวอักขระพิเศษเช่น เครื่องหมาย ' ในคำสั่ง SQL เพื่อให้ได้คำสั่ง SQL ที่ปลอดภัยสำหรับการ query หรือปลอดภัยจากการเรียกใช้ฟังก์ชัน mysql_query()

```
<?
$item = "Zak's and Derick's Laptop";
$escaped_item = mysql_real_escape_string($item);
?>
```

รูปที่ 2.13 ตัวอย่างการใช้ฟังก์ชัน mysql_real_escape_string()

จากตัวอย่าง สิ่งที่ตัวแปร $item เก็บจะมีเครื่องหมาย single qoute (') ซึ่งเป็นอักขระพิเศษเมื่อมีการเรียกใช้ฟังก์ชัน mysql_real_escape_string() แล้วเก็บค่าไว้ที่ตัวแปร $escaped_item ซึ่งค่าที่ตัวแปร $escaped_item เก็บจะมีค่าเป็น

"Zak\'s and Derick's Laptop"

โดยจะมีการนำเครื่องหมาย \ นำหน้าอักขระพิเศษ

ถ้าคำสั่ง SQL สำหรับตรวจสอบการล็อกอินคือ

$q="SELECT * FROM member WHERE username='$username' and password='$password' ";

หากผู้ใช้ป้อนข้อมูลดังนี้ :

username : admin

password : ' or '1' = '1

คำสั่ง SQL ที่ได้จะกลายเป็น :

$q="SELECT * FROM user WHERE username='admin' and password='' or '1' = '1' ";

ซึ่งจะทำให้ผู้โจมตีสามารถเข้าระบบล็อกอินได้ เพราะว่า password จะกลายเป็น or 1 = 1 ซึ่งเป็นการทำให้เงื่อนไขเป็นจริง แต่ถ้าใช้ฟังก์ชัน mysql_real_escape_string() เข้ามาช่วย เช่น

$q="SELECT * FROM member WHERE username='$username' and password='$password' ";

$q=mysql_real_escape_string($q);

ค่า $q จะเป็นดังนี้

$q="SELECT * FROM user WHERE username='admin' and password= '\' or \'1\' = \'1'";

การ query คำสั่ง SQL ปลอดภัยเพราะ \ จะเป็นตัว escape ' ทำให้ค่าของ password จะเก็บเป็น \' or \'1\' = \'1

สำหรับในกรณีที่ผู้ใช้มีชื่อประกอบด้วยตัวอักขระที่เป็น escape character เช่น O'Reilly ในแอพพลิเคชันปัจจุบันจึงมีการใช้คำสั่ง mysql_real_escape_string( ) ตั้งแต่ขั้นตอนการเก็บข้อมูลลงฐานข้อมูล

2.4.7 การป้องกันโดยการเขียน Prepared Statement ในการเชื่อมต่อฐานข้อมูลแบบ PDO

Prepared Statements [40] เป็นฟีเจอร์ API ในการทำงานร่วมกันของภาษาโปรแกรมกับฐานข้อมูลโดยตรง การเรียก mysql_query( ) แบบเดิมจะรวมทั้งคำสั่ง SQL กับข้อมูลที่ผู้ใช้คิวรี่ไว้ในตัวแปรสตริงค่าเดียว จึงสามารถโดนการโจมตีแบบ SQL Injection ได้

แต่ Prepared Statement จะแบ่งคำสั่ง(Command) กับ query แยกกันโดยจะคอมไพล์ส่วนของคำสั่งไว้ก่อน แล้ว ค่อย bind query เข้าไปทีหลัง

วิธีใช้งาน Prepared Statements ในการเชื่อมต่อกับฐานข้อมูลแบบ PDO

1) **การ SELECT ข้อมูลแบบ Prepared Statement**

```
<?php
$stmt = $db->prepare("SELECT * FROM table WHERE id=? AND name=?");
$stmt->execute(array($id, $name));
```

2) **การ INSERT ข้อมูลแบบ Prepared Statement**

```
<?php
$stmt = $db->prepare("INSERT INTO table(field1,field2,field3,field4,field5)
VALUES(:field1,:field2,:field3,:field4,:field5)");
$stmt->execute(array(':field1' => $field1, ':field2' => $field2, ':field3' => $field3, ':field4' => $field4,
':field5' => $field5));
```

3) **การ DELETE ข้อมูลแบบ Prepared Statement**

```
<?php
$stmt = $db->prepare("DELETE FROM table WHERE id=:id");
$stmt->bindValue(':id', $id, PDO::PARAM_STR);
$stmt->execute();
```

4) **การ UPDATE ข้อมูลแบบ Prepared Statement**

```
<?php
$stmt = $db->prepare("UPDATE table SET name=? WHERE id=?");
$stmt->execute(array($name, $id));
```

2.5 การป้องกัน SQL Injection โดยใช้โปรแกรมตรวจสอบโค้ด

2.5.1. โปรแกรมตรวจสอบโค้ดแบบ Static

เป็นการใช้ Static Analysis Tool เข้ามาช่วยในการตรวจสอบโค้ดที่อาจมีช่องโหว่ที่ทำให้ผู้โจมตีสามารถเพิ่มโค้ด SQL เพื่อทำ SQL Injection เข้ามาได้ โดยเมื่อพบช่องโหว่ต่างๆแล้วเว็บแอพพลิเคชันโปรแกรมเมอร์สามารถแก้ไขช่องโหว่ได้อย่างทันที ซึ่ง Static Analysis Tool ที่ผู้จัดทำได้รวบรวมมามีดังนี้

1) Graudit

Graudit [42] เป็น simple shell script และเก็บเซตของรูปแบบที่ช่วยให้สามารถค้นหาข้อบกพร่องในการรักษาความปลอดภัยที่อาจเกิดขึ้นในซอร์สโค้ดโดยใช้ utility GNU grep ผู้ใช้สามารถเขียนไฟล์ Configuration และค่าต่างๆในการตรวจสอบ Regular Expressions ของแต่ละภาษาเองได้ มี technical requirements ต่ำทำให้โปรแกรมมีความยืดหยุ่น

2) Yet Another Source Code Analyzer (YASCA)

YASCA [43] เป็นโปรแกรมประเภทโอเพ่นซอร์สที่มองหาช่องโหว่ในความปลอดภัยและปัญหาจาก code quality จาก source codes ในโปรแกรม สามารถใช้ได้กับ PHP, Java, C / C + +, และ JavaScript สำหรับการรักษาความปลอดภัยและปัญหาจาก code quality

3) Pixy

Pixy [44] เป็นโปรแกรม Java แบบไม่มีค่าใช้จ่ายที่ดำเนินการสแกนโค้ดของ PHP โดยอัตโนมัติ ใช้สำหรับตรวจสอบ cross-site scripting (XSS) และช่องโหว่ SQL injection ของ PHP โดยเฉพาะ

4) AppCodeScan

AppCodeScan [45] เป็นเครื่องมือที่ใช้เพื่อสแกนซอร์สโค้ดสำหรับค้นหาช่องโหว่หลายรูปแบบซึ่งหนึ่งในนั้นคือ SQL Injection จะใช้การจับคู่สตริงเพื่อระบุฟังก์ชันและสตริงที่อาจเป็นอันตรายจาก code base เครื่องมือนี้ไม่ได้ระบุที่อยู่ของช่องโหว่ จะระบุเพียงการใช้งานของฟังก์ชันและสตริงที่สามารถนำไปสู่การเกิดขึ้นของช่องโหว่เท่านั้น ทำงานบน .NET Framework ซึ่งง่ายต่อการเขียนและแก้ไขเป็นที่ชื่นชอบสำหรับผู้ที่ต้องการทำงานใน GUI ในขณะนี้เป็นรุ่นเบตาเท่านั้น

5) OWASP LAPSE+ Project

LAPSE + [46] เป็นสแกนเนอร์ด้านความปลอดภัยสำหรับการตรวจสอบช่องโหว่โดยเฉพาะการ Injection ของข้อมูลที่ไม่น่าเชื่อถือในส่วนโปรแกรมภาษา Java EE ซึ่งได้รับการพัฒนาเป็นปลั๊กอินสำหรับโปรแกรม Eclipse (www.eclipse.org) ทำงานเฉพาะกับ Eclipse Helios และ Java 1.6 หรือสูง

กว่า LAPSE + ซอฟต์แวร์พัฒนาโดย Benjamin Livshits เป็นส่วนหนึ่งของ Griffin Software Security Project

6) Microsoft Source Code Analyzer for SQL Injection

Microsoft Source Code Analyzer for SQL Injection tool [47] ใช้เพื่อหาช่องโหว่ SQL Injection โค้ดของ Active Server Pages (ASP) เป็นเครื่องมือสำหรับ classic ASP code ไม่ใช่ .NET code ถูกเขียนใน VBScript

7) Microsoft Code Analysis Tool .NET (CAT.NET)

CAT.NET [48] เป็น binary code analysis tool ในการระบุตัวแปรทั่วไปของช่องโหว่ XSS และ SQL Injection โดย CAT.NET เป็น snap-in ใน Visual Studio 2005 หรือ 2008 ที่ระบุข้อบกพร่องด้านความปลอดภัยภายในการจัดการ code (C #, Visual Basic. NET code, J #) เท่านั้น

8) RIPS—A Static Source Code Analyzer for Vulnerabilities in PHP Scripts

RIPS [49] เป็นเครื่องมือที่เขียนใน PHP ที่สามารถใช้วิธีการ Static Code Analysis เพื่อหาช่องโหว่ใน PHP applications เป็นเครื่องมือที่ต้องการเว็บเซิร์ฟเวอร์ในการทดสอบ

9) CodePro AnalytiX

CodePro Analytix [50] สามารถทำงานได้ดีกับ Eclipse โดยใช้ automated source code analysis เพื่อระบุปัญหาและช่องโหว่ด้านความปลอดภัย และมีกฎที่สามารถนำมาใช้เพื่อหาการกระทำที่อาจเกิดขึ้นจากต้นทางไปยังปลายทาง

10) Teachable Static Analysis Workbench

Teachable Static Analysis Workbench (TeSA) [51] คือ เครื่องมือในการตรวจสอบจาวาเว็บแอพพลิเคชันเพื่อหาช่องโหว่ด้านความปลอดภัยที่เกี่ยวข้องกับตรวจสอบการป้อนข้อมูลที่ไม่เหมาะสม TESA จะสอนการกำหนดค่าเครื่องมือในการหาช่องโหว่ทั้งหมดโดยแสดงใน data flows
รุ่นปัจจุบันของ TESA จะสนับสนุน servlets และ Java Server เท่านั้นไม่ได้สนับสนุนในส่วนของเว็บแอพพลิเคชัน

11) Fortify Source Code Analyzer

Fortify Source Code Analyzer [52] เป็น Source code analyzer ที่ประมวลผลและพยายามที่จะหาช่องโหว่ ใช้สร้างเครื่องมือที่ทำงานบนไฟล์ซอร์สโค้ด หรือชุดของไฟล์และแปลงเป็นไฟล์

12) Rational AppScan Source Edition

AppScan Source Edition [53] คือ Static Analysis Tool ที่ระบุช่องโหว่ ผ่าน reviewing data

13) Klocwork Solo

Klocwork Solo [54] คือ โปรแกรมสำหรับ Java Developers มุ่งเน้นไปที่การพัฒนา mobile และเว็บแอพพลิเคชันโดยเป็นปลั๊กอินบน Eclipse ทำงานอัตโนมัติในการค้นหาโอกาสในการโดนโจมตี

2.5.2 โปรแกรมตรวจสอบโค้ดแบบ Dynamic

เป็นโปรแกรมที่จำเป็นต้องมีการจำลองทรัพยากรพื้นฐานของระบบขึ้นมาก่อนที่จะทำการตรวจสอบจริง ในขั้นตอนการตรวจสอบนั้นจะเป็นการทดสอบการโจมตีด้วยการโจมตีที่เกิดขึ้นจริงต่อระบบทุกรูปแบบที่เป็นไปได้ ตัวอย่างโปรแกรมตรวจสอบโค้ดแบบ Dynamic เช่น Acunetix [55] เป็น Dynamic Tool ใช้สำหรับสแกนเว็บไซต์ หรือ เว็บเซิร์ฟเวอร์ เพื่อตรวจดูว่ามีช่องโหว่ใดๆที่เป็นอันตรายต่อเซิร์ฟเวอร์หรือไม่

ตารางที่ 2.1 คุณสมบัติของ Analyzer Tools

ชื่อเครื่องมือ	ประเภท	ภาษา	แพลตฟอร์ม	IDE	ลักษณะ	ราคา	เว็บไซต์ที่มา
Graudit	N/A	ASP, JSP, Perl, PHP และ Python	Windows, Linux, และ OS X	N/A	open source	Free	www.justanotherhacker.com/projects/graudit.html
YASCA	Static	Any language	Windows, Linux	N/A	open source	Free	http://www.scovetta.com/yasca.html
Pixy	Static	PHP	Windows, Linux	N/A	open source	Free	https://github.com/oliverklee/pixy
AppCodeScan	Static	Any language	Windows	N/A	free software	Free	www.blueinfy.com
LAPSE+	Static	Java, J2EE	Windows, Linux	Eclipse	open source	Free	https://www.owasp.org/index.php/OWASP_LAPSE_Project
Microsoft Source Code Analyzer for SQLInjection	Static	ASP	Windows	N/A	N/A	Free	http://support.microsoft.com/kb/954476

ตารางที่ 2.1 คุณสมบัติของ Analyzer Tools (ต่อ)

ชื่อเครื่องมือ	ประเภท	ภาษา	แพลตฟอร์ม	IDE	ลักษณะ	ราคา	เว็บไซต์ที่มา
CAT.NET	N/A	C#, Visual Basic .NET, and J#	Windows	Visual Studio	free software	Free	www.microsoft.com/download/en/details.aspx?id=19968
RIPS	Static	PHP	Windows, Linux	N/A	open source	Free	http://rips-scanner.sourceforge.net/
Code Pro AnalytiX	Static	Java, JSP, JSF, Struts และ XML	Windows, Linux, และ OS X	Eclipse	free software	Free	https://developers.google.com/java-dev-tools/codepro/doc/
TeSA	N/A	Java	Windows, Linux	Eclipse	open source	Free	https://code.google.com/p/teachablesa/
Fortify Source Code Analyzer	Static	Any language	Windows, Mac, Solaris, Linux, AIX, และHP-UX	Visual Studio, Eclipse	Commer-cial	สอบถามทางเว็บไซต์	http://www8.hp.com/us/en/software-solutions/application-security/index.html

ตารางที่ 2.1 คุณสมบัติของ Analyzer Tools (ต่อ)

ชื่อเครื่องมือ	ประเภท	ภาษา	แพลตฟอร์ม	IDE	ลักษณะ	ราคา	เว็บไซต์ที่มา
Rational AppScan Source Edition	Static	Windows, Solaris, และ Linux	Microsoft Visual Studio, Eclipse	Microsoft Visual Studio, Eclipse	Commer -cial	สอบถามทางเว็บไซต์	www.ibm.com/software/rational/products/appscan/source/
Klocwork Solo	N/A	Java	Windows 32 bit	Eclipse	Commer -cial	สอบถามทางเว็บไซต์	www.klocwork.com/products/solo/
Acunetix	Dynamic	Any language	On Web , Base OS	Stand Alone	Commer -cial	สอบถามทางเว็บไซต์	http://www.acunetix.com/

2.6 Editor ที่ใช้เขียนเว็บแอพพลิเคชัน

2.6.1 Netbeans

NetBeans [39] คือ เครื่องมือสำหรับโปรแกรมเมอร์ที่จะใช้พัฒนาแอพพลิเคชันด้วยภาษา Java เป็นโปรแกรมประเภทโอเพ่นซอร์สซึ่งผู้ใช้งานไม่จำเป็นต้องเสียเงินในการใช้งาน และเปิดเผยซอร์สโค้ดให้ผู้สนใจและนักพัฒนานำไปดัดแปลง แก้ไข ตามกฎ ของโอเพ่นซอร์สโดยมี Sun Micro System เป็นผู้สนับสนุนโครงการ

ปัจจุบัน NetBeans ได้รับความนิยมมากยิ่งขึ้นและได้รับการพัฒนาให้มีความสามารถเพิ่มสูงยิ่งขึ้นเรื่อยๆ นอกจากจะใช้ในการพัฒนาแอพพลิเคชันด้วยภาษาจาวาแล้ว ยังสามารถใช้ในการพัฒนาสิ่งอื่นๆได้อีกหลากหลายโดยติดตั้งโปรแกรมเสริม (Add-on)ได้จาก เว็บไซต์หรืออัปเดตเซนเตอร์ (Update Center) ของ NetBeans เช่น ภาษา C/C++, Ruby, UML, SOA, Web Application Java EE Mobility(Java ME), Java FX, Java Script และ PHP เป็นต้น ในเวอร์ชัน 6.0 เป็นต้นไปมีการรวมโปรแกรมเสริมต่างๆ ที่สำคัญในตัวติดตั้งของ NetBeans มาให้โดยสามารถเลือกติดตั้งได้ภายหลัง

โปรแกรม NetBeans นั้นทำงานแยกส่วนออกจากกันเป็นโมดูล (Module) จึงทำให้สามารถนำ Module ต่างๆที่มีผู้ที่ได้พัฒนาต่อเติมมาติดตั้งเพิ่มเติมในภายหลังได้ ใช้งานได้กับระบบปฏิบัติการวินโดวส์ ลีนุกซ์ MAC OS X และ Solaris

2.6.2 Eclipse

Eclipse เป็นเครื่องมือที่เรียกว่า integrated development environment (IDE) สำหรับพัฒนา applications โดยใช้ java หรือภาษาอื่น ๆ เช่น C/C++, Python, PERL และ Ruby ฯลฯ [58]

Eclipse สามารถรองรับปลั๊กอินได้หลากหลาย ผู้พัฒนาที่ใช้ภาษา Java ในการพัฒนาแอพพลิเคชันของภาษาต่างๆ สามารถใช้ Eclipse ในการพัฒนาได้ โดยตัว Eclipse มีสภาวะแวดล้อมที่สมบูรณ์ คือมีเครื่องมือต่างๆให้ใช้พร้อม นอกจากนี้ Eclipse สามารถใช้พัฒนาโปรแกรมภาษาอื่น ๆ ได้ถ้ามีตัวปลั๊กอินนั้นอยู่ เช่น ถ้าต้องการพัฒนาแอพพลิเคชันโดยใช้ภาษา PHP ถ้า Eclipse มีปลั๊กอินภาษา PHP ผู้ใช้สามารถใช้ Eclipse ในการพัฒนาได้ Eclipse และ ปลั๊กอินต่าง ๆ ของ Eclipse พัฒนาอยู่ภายใต้ Eclipse Public License (EPL) เพื่อให้ Eclipse สามารถดาวน์โหลด และติดตั้งได้ฟรี นอกจากนี้ยังสามารถปรับปรุงแก้ไขและนำไปจัดจำหน่ายได้

ตารางที่ 2.2 รุ่นของ Eclipse

Codename	Year	Platform Version
Callisto	2006	3.2
Europa	2007	3.3
Ganymede	2008	3.4
Galileo	2009	3.5
Helios	2010	3.6
Indigo	2011	3.7
Juno	2012	3.8 and 4.2
Kepler	2013 (planned)	4.3

บทที่ 3
การวิเคราะห์และออกแบบ

3.1 การวิเคราะห์ระบบ

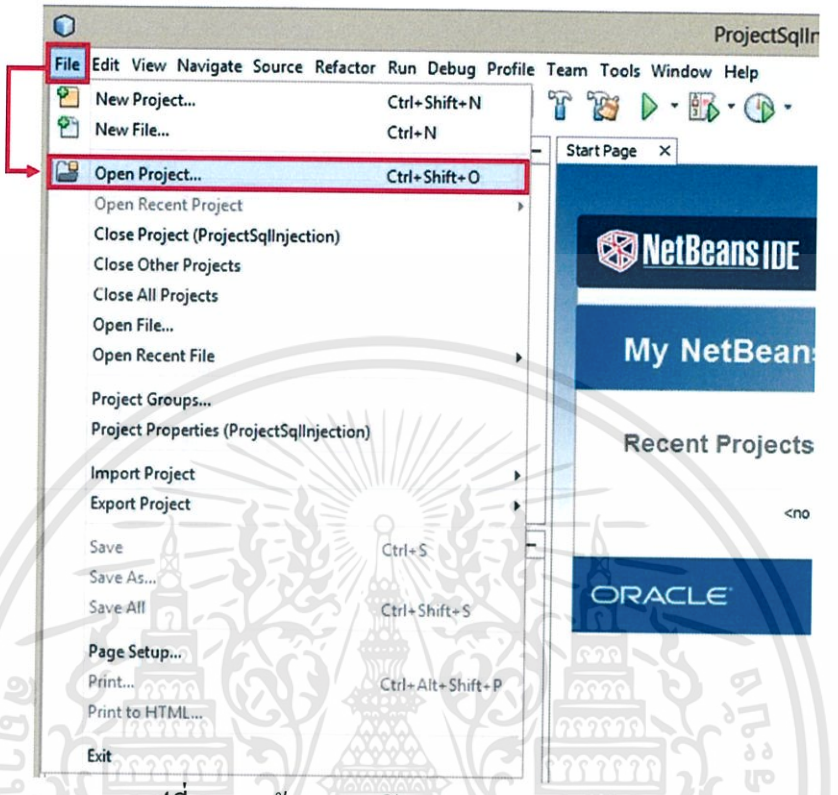
3.1.1 วัตถุประสงค์ในการวิเคราะห์ระบบ

เนื่องจากวัตถุประสงค์หลักของโครงงานพิเศษนี้ คือ การทำโปรแกรมปลั๊กอินเพื่อตรวจสอบโค้ด PHP ที่มีช่องโหว่ต่อการโจมตีแบบ SQL Injection ซึ่งมีสองแนวทางคือ ทำการพัฒนาเครื่องมือ (tool) ขึ้นมาใหม่ หรือนำเครื่องมือที่มีอยู่แล้วมาพัฒนาต่อ จากการศึกษาพบว่ามีเครื่องมือในการตรวจสอบอยู่สองประเภทหลัก คือ เครื่องมือที่ทำการตรวจสอบโค้ดแบบ static (Static Analysis Tool) ซึ่งทำการสแกนที่ซอร์สโค้ด และ เครื่องมือที่ทำการตรวจสอบการทำงานของโปรแกรม (Dynamic Tool) ซึ่งจะจำลองสิ่งแวดล้อมเสมือนให้โปรแกรมทำงาน ซึ่งเครื่องมือในการตรวจสอบโค้ดแบบ static บน PHP ที่สามารถนำมาใช้งานได้มี 2 โปรแกรมคือ Pixy และ RIPS ทางผู้จัดทำเลือก Pixy เนื่องจากไม่ต้องใช้เว็บเซิร์ฟเวอร์ในการทดสอบโปรแกรมและสามารถรันโค้ดแล้วแจ้งเตือนบรรทัดที่มีโค้ดอันตรายได้ จากนั้นจึงทำการวิเคราะห์ความสามารถของ Pixy ในการตรวจสอบโค้ด PHP ที่เป็นอันตรายต่อการโจมตีแบบ SQL Injection โดยตรวจสอบกับไฟล์ 20 ไฟล์ ที่รวบรวมมาจาก 1800 php scripts web developer [56] เมื่อได้ผลการทดสอบแล้ว จึงนำมาวิเคราะห์หาวิธีแก้ไข เนื่องจากการรัน Pixy นั้นปกติรันใน Text mode เพื่อความสะดวกในการตรวจสอบ ผู้จัดทำจึงทำการปลั๊กอิน Pixy เข้ากับทั้ง NetBeans และ Eclipse เพื่อให้สามารถเรียกใช้งาน Pixy บน NetBeans และ Eclipse ได้ ซึ่งวิธีการปลั๊กอินทำได้ดังนี้

3.1.2 วิธีการปลั๊กอิน Pixy ลงบน NetBeans และ Eclipse

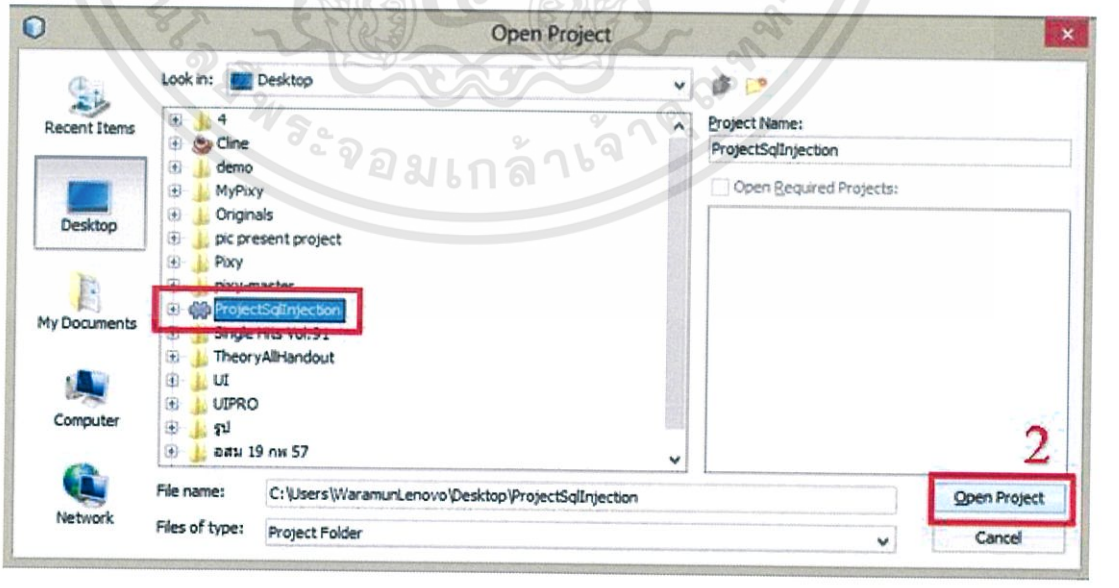
- การปลั๊กอิน Pixy ลงบน NetBeans

1) บนเมนูบาร์ของ NetBeans เลือก File > Open Project… ดังรูปที่ 3.1



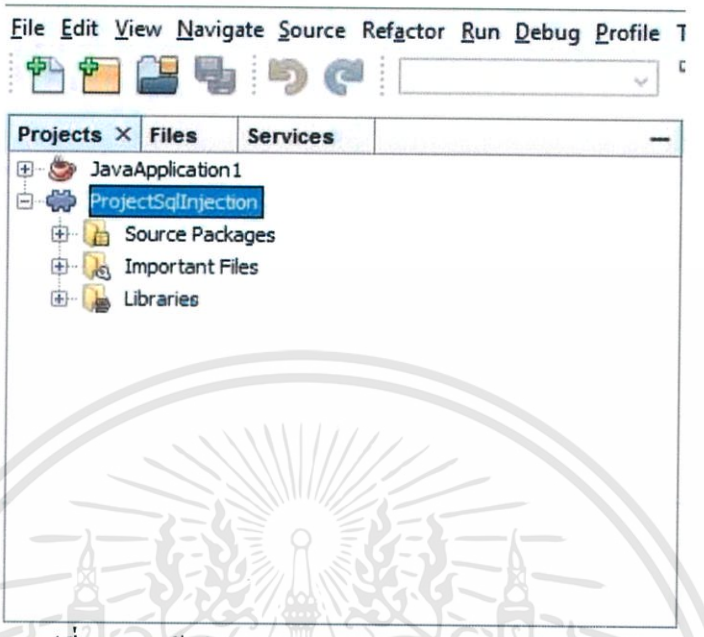
รูปที่ 3.1 หน้าจอการเปิด Project บน NetBeans

2) ที่หน้าต่าง Open Project เลือก ProjectSqlInjection แล้วกด Open Project



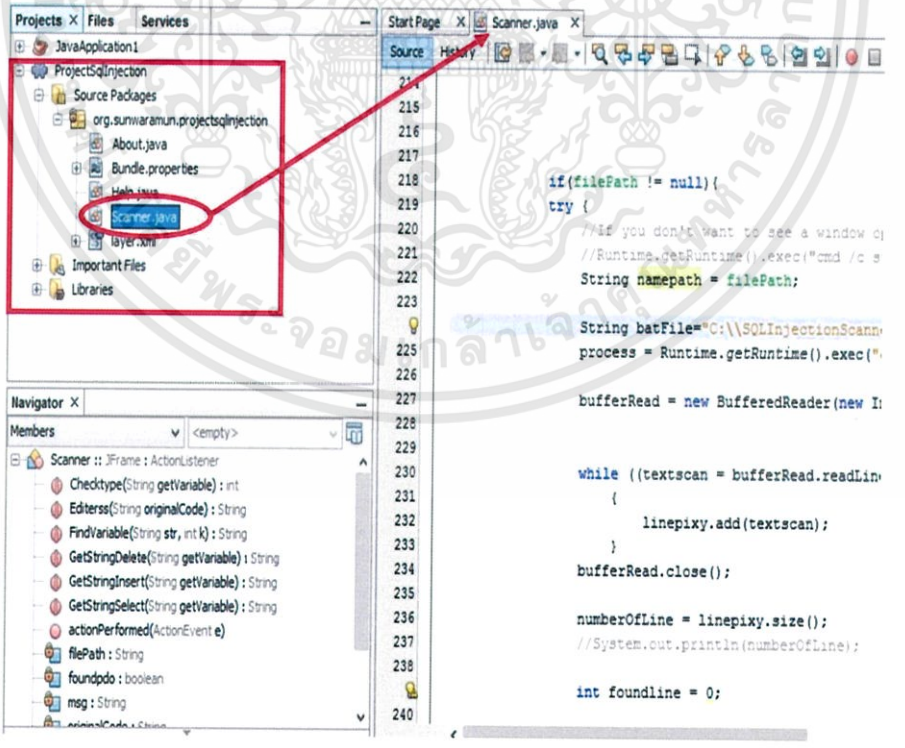
รูปที่ 3.2 หน้าต่างการเลือก ProjectSqlInjection บน NetBeans

3) ProjectSqlInjection ที่เลือกมาจะปรากฏบนหน้าต่าง Projects ของ NetBeans ดังรูปที่ 3.3



รูปที่ 3.3 หน้าต่าง ProjectSqlInjection บน NetBeans

4) ที่หน้าต่าง Projects ของ NetBeans ให้เปิดไฟล์ Scanner.java ดังรูปที่ 3.4



รูปที่ 3.4 หน้าต่างแสดงเปิดไฟล์ Scanner.java ของ Pixy บน NetBeans

5) ในบรรทัดที่ 218 และ 222 ของไฟล์ Scanner.java ให้เปลี่ยน filePath ให้เป็นที่อยู่ (path) ของไฟล์ PHP ที่ต้องการสแกนจากนั้นกด Run  หรือ F6 เพื่อสั่งให้ Pixy ทำการสแกน

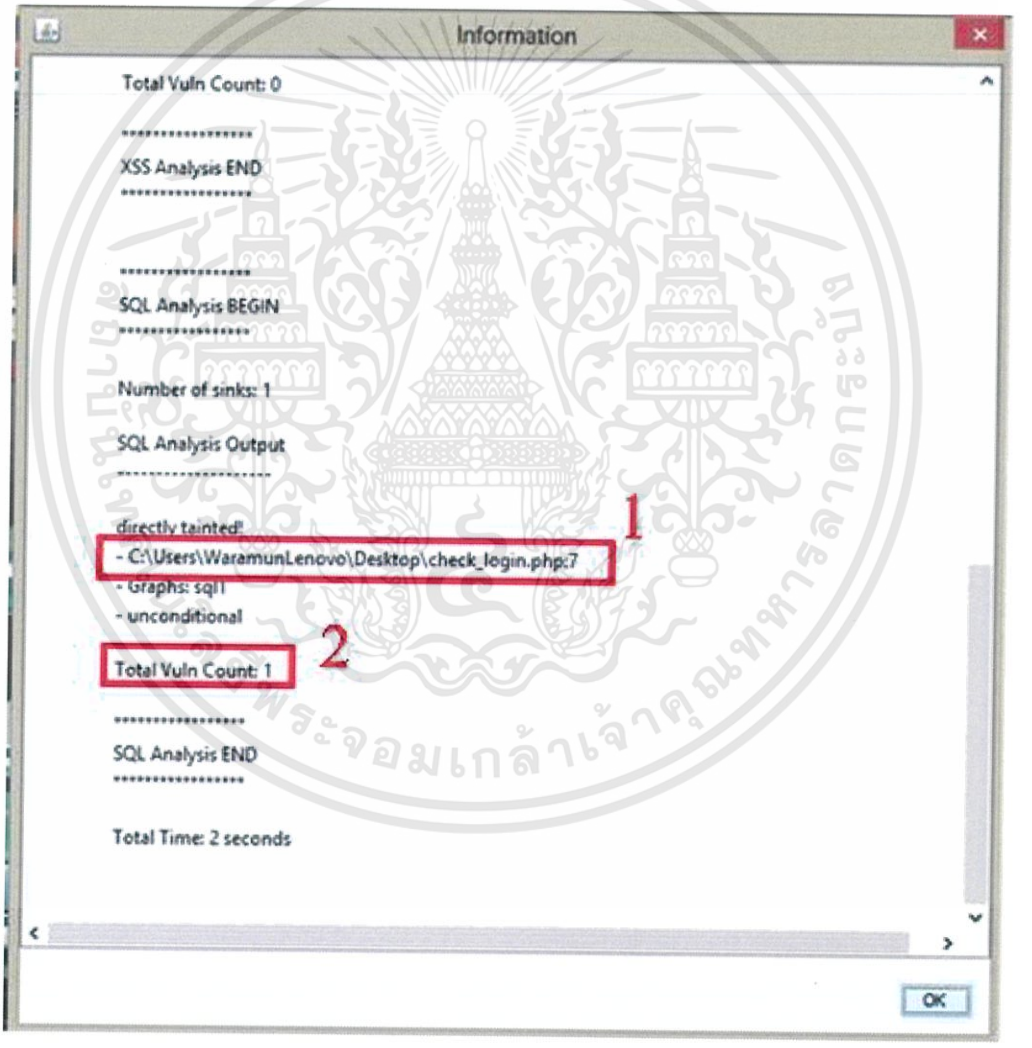
```
218     if (filePath != null){
219     try {
220         //If you don't want to see a window open
221         //Runtime.getRuntime().exec("cmd /c start /MIN mybatch.bat");
222         String namepath = filePath;
223
224         String batFile="C:\\SQLInjectionScanner\\pixy-master\\run-all "+namepath;
225         process = Runtime.getRuntime().exec("cmd /c "+batFile);//start
226
227         bufferRead = new BufferedReader(new InputStreamReader(process.getInputStream()));
228
```

รูปที่ 3.5 ตั้ง filePath เป็นที่อยู่ของไฟล์ PHP ที่ต้องการสแกนบน NetBeans

6) เมื่อ Pixy ทำการสแกนเสร็จเรียบร้อยแล้วจะแสดงผลการสแกนดังหน้าต่างในรูปที่ 3.6 ซึ่งประกอบไปด้วย

1. ชื่อไฟล์และหมายเลขบรรทัดที่มีช่องโหว่
2. จำนวนช่องโหว่ที่พบ

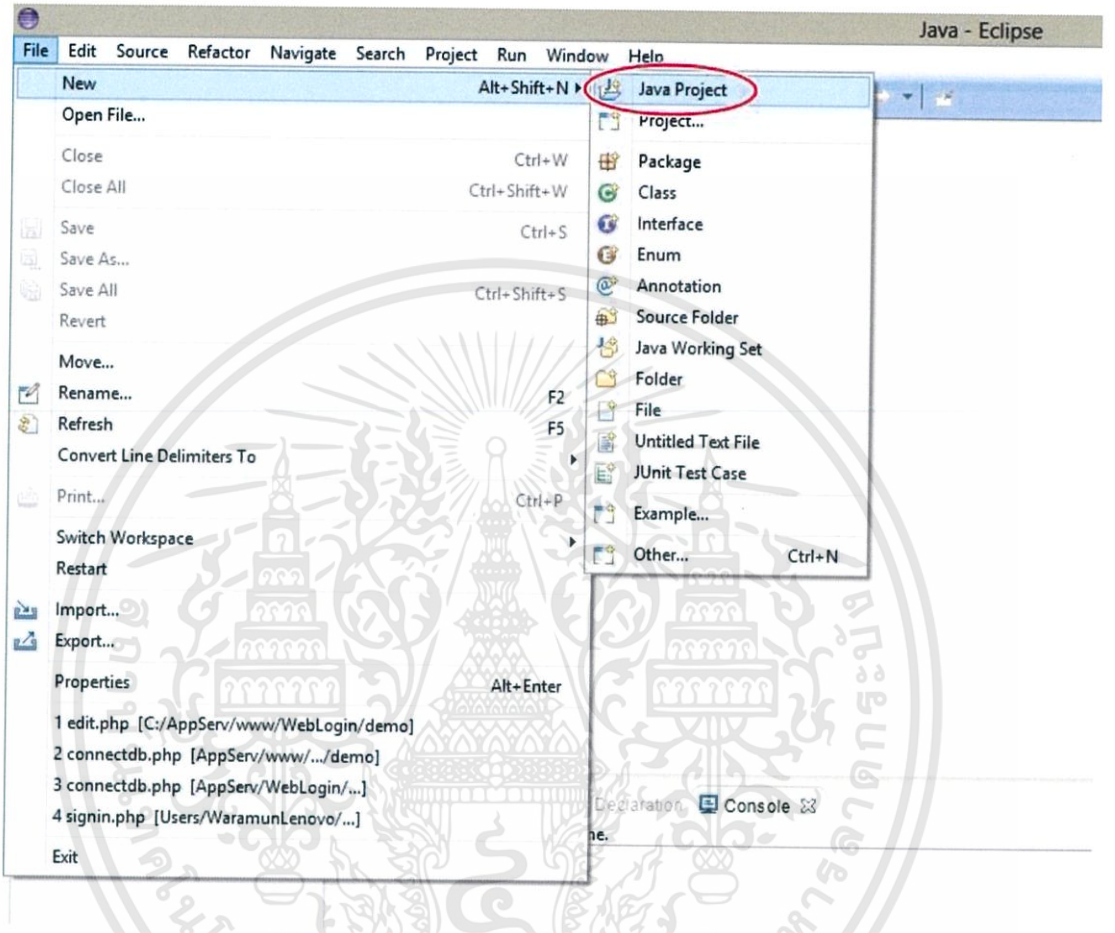
หมายเหตุ Pixy นอกจากสแกนช่องโหว่ของโค้ดที่อันตรายต่อการโจมตีแบบ SQL Injection ได้แล้ว Pixy สามารถสแกนช่องโหว่ที่อันตรายต่อการโจมตีประเภท Cross-site Scripting (XSS) ได้ด้วย แต่ในโครงงานพิเศษนี้จะสนใจแต่โค้ดที่ก่อให้เกิด SQL Injection เท่านั้น ซึ่งจะเริ่มที่บรรทัด SQL Analysis Output



รูปที่ 3.6 ตัวอย่างผลการสแกนของ Pixy บน NetBeans

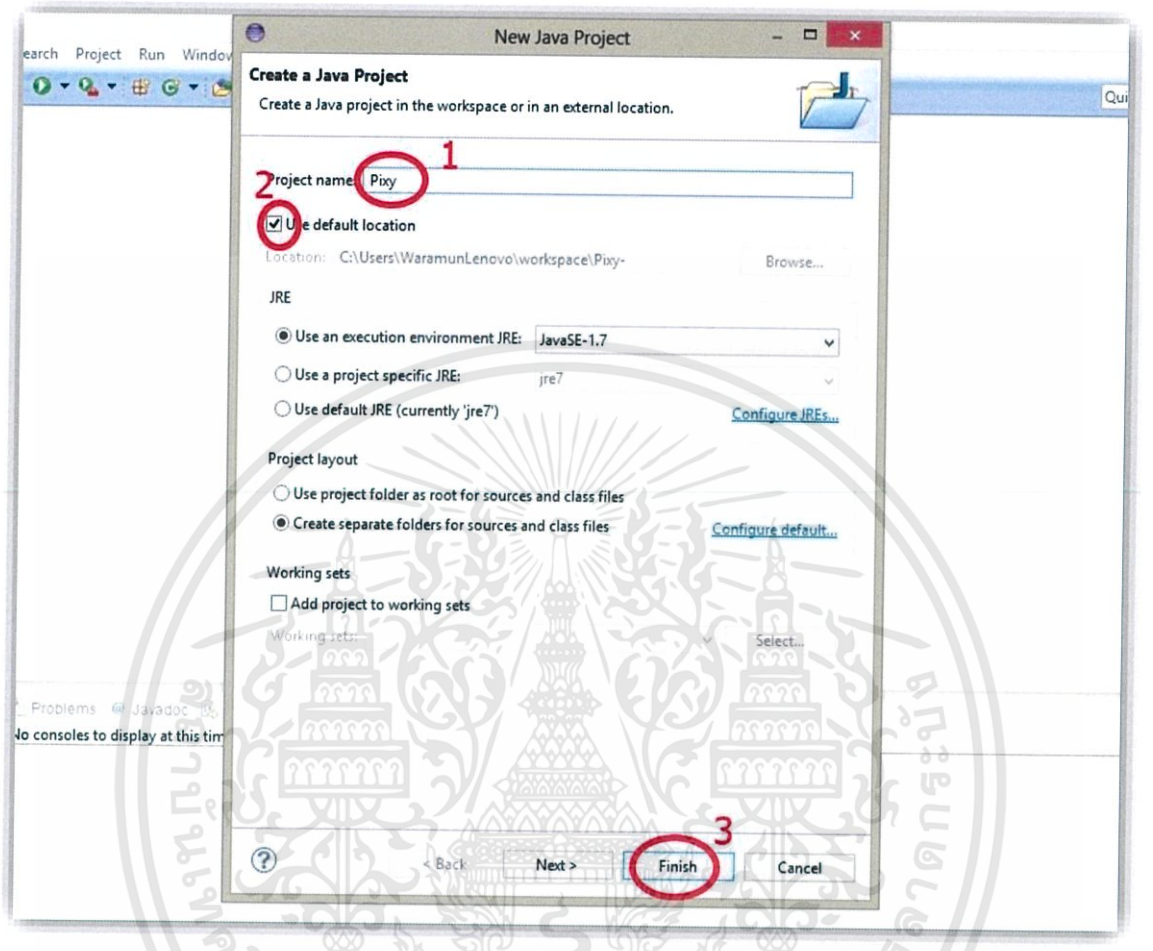
- การปลั๊กอิน Pixy บน Eclipse

1) ทำการสร้าง New Java Project ขึ้นมา โดยเลือก File > New > Java Project ดังรูปที่ 3.7



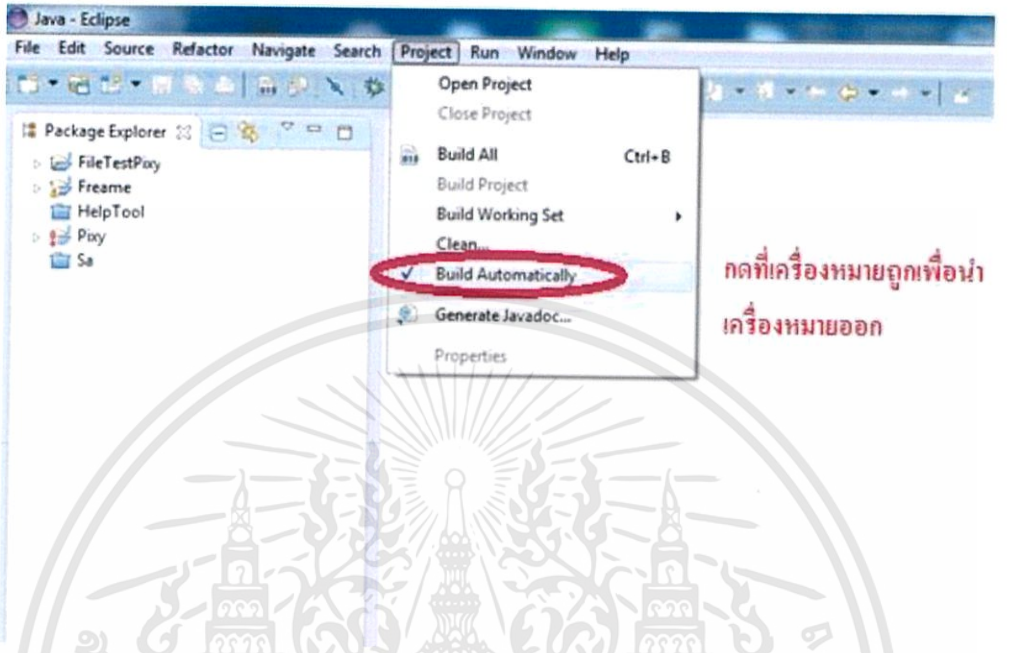
รูปที่ 3.7 หน้าจอแสดงการสร้าง New Java Project บน Eclipse

2) ตั้งชื่อ Project เป็น Pixy จากนั้นเลือก Use default location แล้วกด Finish ดังรูปที่ 3.8



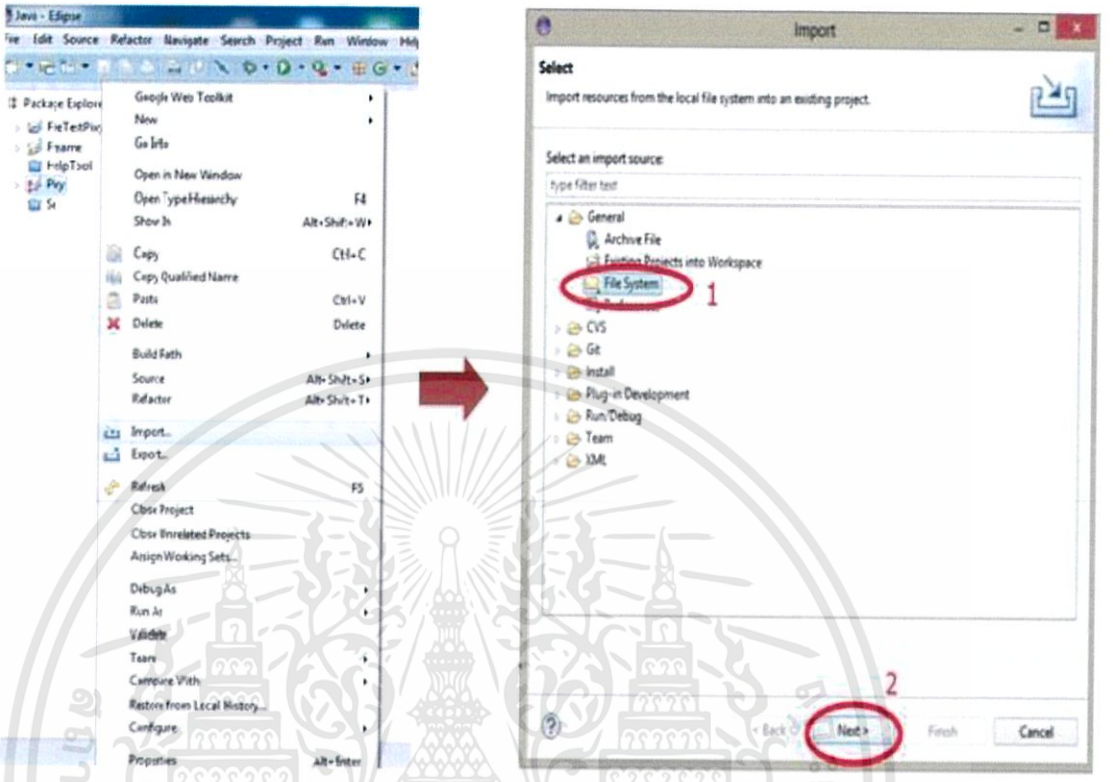
รูปที่ 3.8 หน้าต่างการตั้งชื่อ Eclipse Project

3) ทำการ disable "build automatically" จากเมนู Project โดยไม่ให้มีเครื่องหมายถูกที่อยู่หน้าข้อความ ดังรูปที่ 3.9



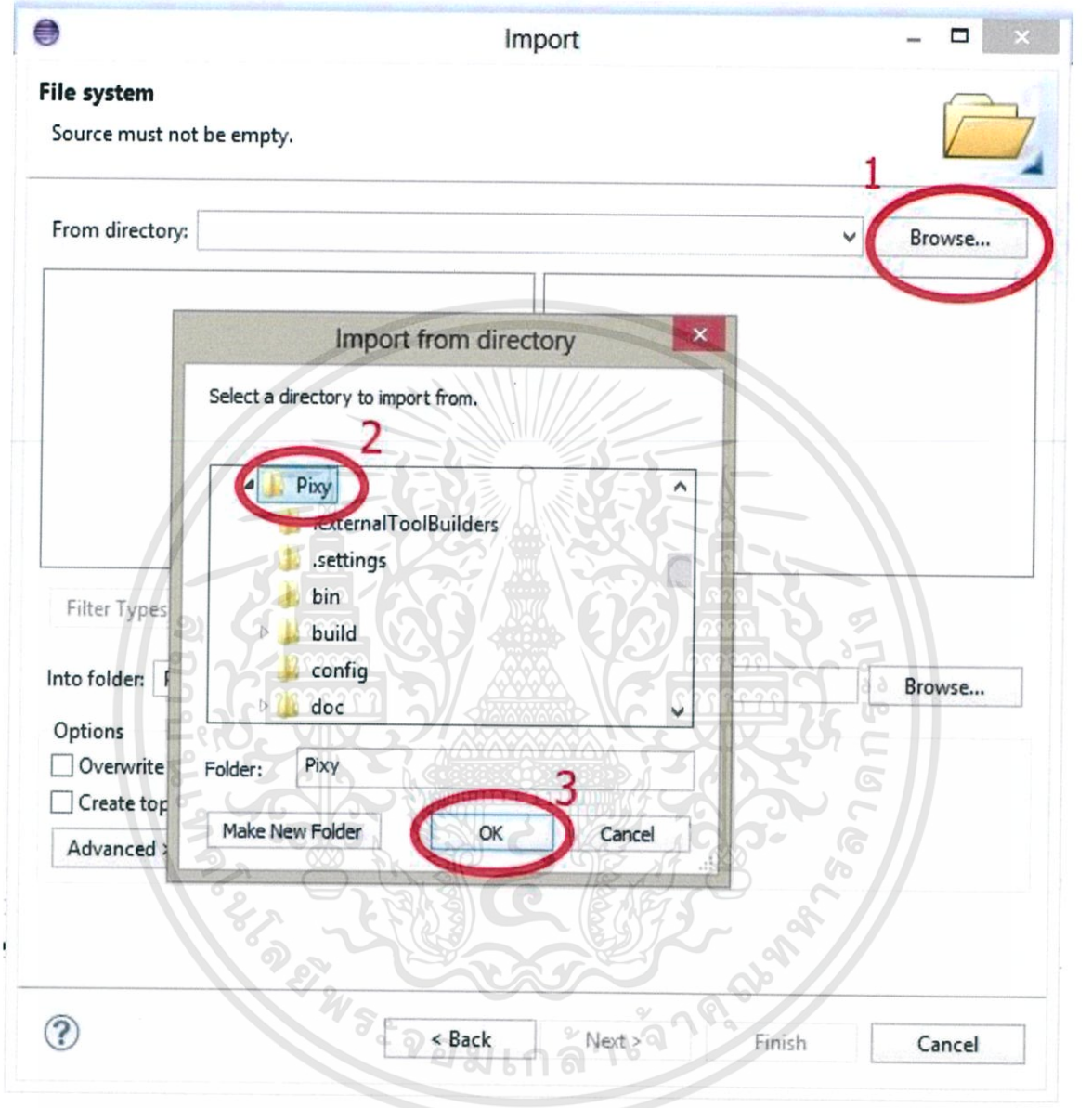
รูปที่ 3.9 การ disable "build automatically" ในเมนู Project บน Eclipse

4) คลิกขวาที่ Pixy เลือก Import…. > File System แล้วกด Next



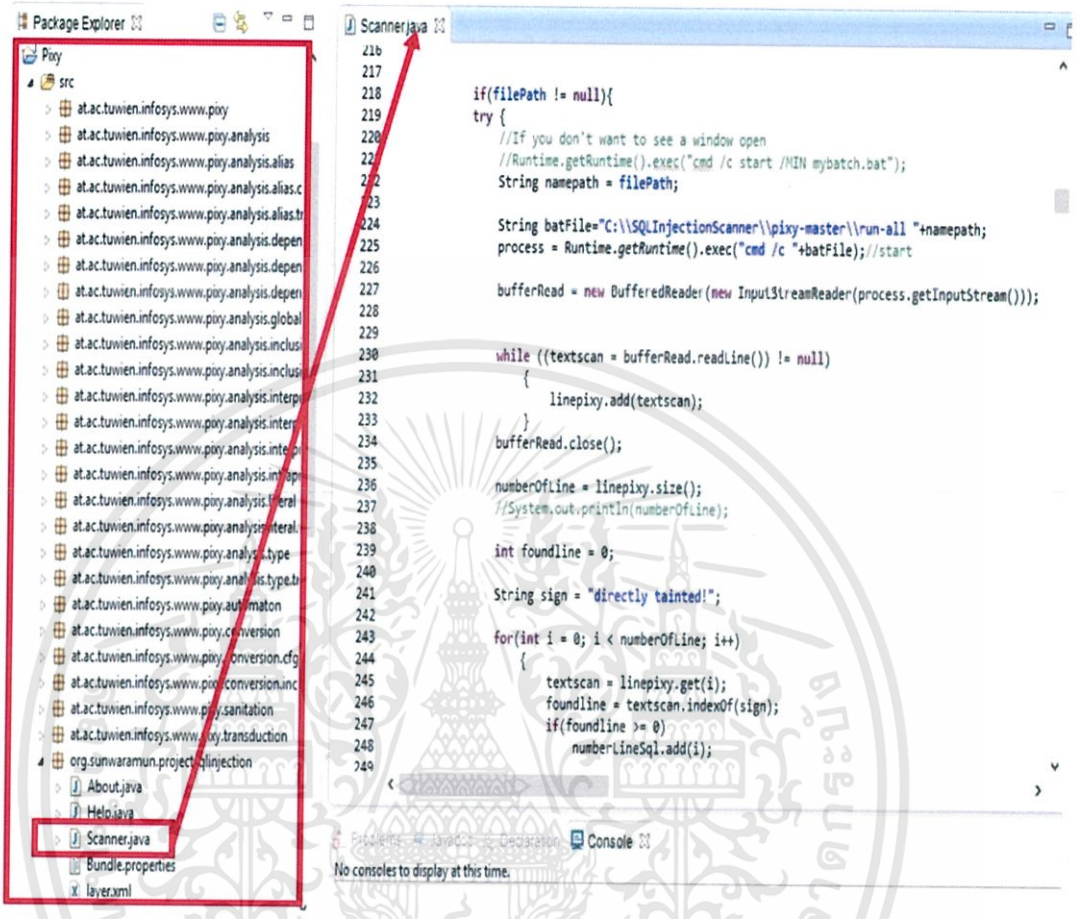
รูปที่ 3.10 หน้าจอแสดงการ Import Project บน Eclipse

5) เลือก Browse… > Pixy > OK ดังรูปที่ 3.11



รูปที่ 3.11 หน้าจอแสดงการ Import Project Pixy บน Eclipse

6) ที่หน้าต่าง Projects ของ Eclipse ให้เปิดไฟล์ Scanner.java ดังรูปที่ 3.12



รูปที่ 3.12 หน้าต่างแสดงการเปิดไฟล์ Scanner.java ของ Pixy บน Eclipse

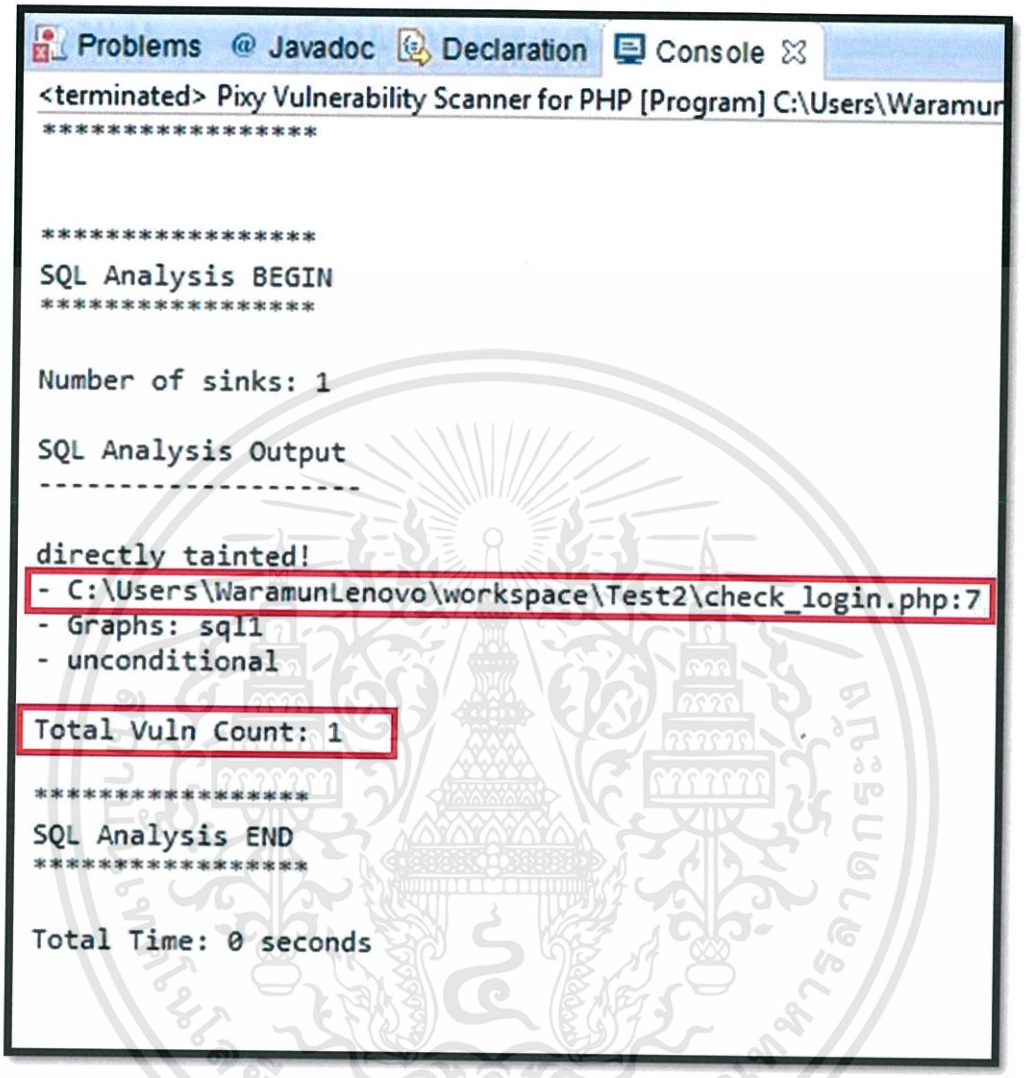
7) ในบรรทัดที่ 218 และ 222 ของไฟล์ Scanner.java ให้เปลี่ยน filePath ให้เป็นที่อยู่ (path) ของไฟล์ PHP ที่ต้องการสแกนจากนั้นกด Run  เพื่อสั่งให้ Pixy ทำการสแกน

Scanner.java

```
216
217
218         if filePath != null){
219         try {
220             //If you don't want to see a window open
221             //Runtime.getRuntime().exec("cmd /c start /MIN mybatch.bat");
222             String namepath = filePath;
223
224             String batFile="C:\\SQLInjectionScanner\\pixy-master\\run-all "+namepath;
225             process = Runtime.getRuntime().exec("cmd /c "+batFile);//start
226
227             bufferRead = new BufferedReader(new InputStreamReader(process.getInputStream()));
228
229
```

รูปที่ 3.13 การตั้ง filePath เป็นที่อยู่ของไฟล์ PHP ที่ต้องการสแกนบน Eclipse

8) เมื่อกด Run แล้ว Pixy จะแสดงผลการสแกนออกมาเป็นหน้าต่างใหม่ดังรูปที่ 3.14



รูปที่ 3.14 หน้าต่างแสดงผลการสแกนของ Pixy บน Eclipse

3.1.3 การใช้ Pixy ทดสอบไฟล์ตัวอย่าง

ตารางที่ 3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	โค้ดที่โดนโจมตี
1	Ad Management\fad\ctchrefscr.php	$get_ref = @mysql_query("SELECT * FROM `referrals` WHERE `id`='$id'");	SELECT * FROM `referrals` WHERE `id`='$id'
		$update_ref = @mysql_query("UPDATE `referrals` SET `hits`='$hits' WHERE `id`='$id'");	UPDATE `referrals` SET `hits`='$hits' WHERE `id`='$id'
		$update_cookie = @mysql_query("UPDATE `ref_cookie`_list SET `ip`='$ip' WHERE `id`=$id'");	UPDATE `ref_cookie`_list SET `ip`='$ip' WHERE `id`=$id'
		$insert_cookie = @mysql_query("INSERT INTO `ref_cookie_list` (`id`,`ip`) VALUES ('$id','$ip')");	INSERT INTO `ref_cookie_list` (`id`,`ip`) VALUES ('$id','$ip')
2	Blog\aoblogger\ppppp.php	mysql_query("INSERT INTO blog SET title = '$title', message ='$message', time = '$time'") or die(mysql_error());	INSERT INTO blog SET title = '$title', message ='$message', time = '$time'

ตารางที่ 3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	โค้ดที่โดนโจมตี
3	Blog\blog\admin\addarticle.php	mysql_query($createblog) or die("Could not create blog");	$createblog="INSERT into bl_blog(author,blogtitle,thetime,realtime,shortblurb,maincontent,allowcomments,month,year,catparent) values('$blogadmin','$title','$thetime','$realtime','$short','$long','$allow','$month','$year','$thecat')";
4	Blog\blog\admin\addchoice.php	mysql_query($makeanswer) or die(mysql_error());	$makeanswer="INSERT into bl_pollchoices (answer) values('$pollchoice')";
5	News Publishing\aWeb News\aWebNews\feedback.php	$result = mysql_query($query);	$query = "SELECT id, username, password FROM users WHERE username='$user123'";
6	News Publishing\aWeb News\aWebNews\changep.php	$result = mysql_query($query);	$query = "SELECT id, username, password FROM users WHERE username='$user123'";
		$result12 = mysql_query($query12);	$query12 = "UPDATE users SET password='$_POST[password1]' WHERE username='$_SESSION[Username]'";
7	Blog\blog\admin\ban.php	mysql_query($insertip) or die("Could not insert ip");	$insertip="INSERT into bl_banip (banip) values ('$ip')";

ตารางที่ 3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	โค้ดที่โดนโจมตี
8	News Publishing\aWeb News\aWebNews\ post.php	$result = mysql_query($query);	$query = "SELECT id, username, password FROM users WHERE username='$user123'";
		mysql_query($query);	$query = "INSERT INTO news(cid, category, title, author, shorta, longa, datetime, eauthor) VALUES ('$_POST[cid]','$_POST[category]','$_POST[title]','$_SESSION[Username]','$_POST[shorta]','$_POST[longa]','$_POST[datetime]','$_POST[eauthor]')";
9	Advance\viewme.php	$banner = @mysql_query("SELECT * FROM banners WHERE `id`='$id'");	SELECT * FROM banners WHERE `id`='$id'
10	Advance\catchadd.php	$banner_stat = @mysql_fetch_array(@mysql_query("SELECT * FROM stats WHERE `id`='$id'"));	SELECT * FROM stats WHERE `id`='$id'
		$update_banner = @mysql_query("UPDATE stats SET `hits`='$hits' , `uni_hits`='$uni_hits' WHERE id='$id'");	UPDATE stats SET `hits`='$hits' , `uni_hits`='$uni_hits' WHERE id='$id'

ตารางที่ 3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	โค้ดที่โดนโจมตี
11	Advance\editme.php	$get_banner = @mysql_query("SELECT * FROM banners WHERE `id`='$id'");	SELECT * FROM banners WHERE `id`='$id'
		$update = @mysql_query("UPDATE banners SET `name`='$name' , `mouseover`='$mouse_over', `location`='$location',`urlto`='$urlto',`stopit`='$stopit',`java_status`='$java_statusbar',`openin`='$openin' WHERE `id`='$id'");	UPDATE banners SET `name`='$name' , `mouseover`='$mouse_over', `location`='$location',`urlto`='$urlto',`stopit`='$stopit',`java_status`='$java_statusbar',`openin`='$openin' WHERE `id`='$id'
		$update = @mysql_query("UPDATE banners SET `name`='$name' , `mouseover`='$mouse_over', `urlto`='$urlto',`stopit`='$stopit',`java_status`='$java_statusbar',`openin`='$openin' WHERE `id`='$id'");	UPDATE banners SET `name`='$name' , `mouseover`='$mouse_over', `urlto`='$urlto',`stopit`='$stopit',`java_status`='$java_statusbar',`openin`='$openin' WHERE `id`='$id'

ตารางที่ 3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	โค้ดที่โดนโจมตี
11	Advance\editme.php	$resultf = @mysql_query($queryf);	$queryf = "SELECT * FROM stats WHERE `id`='$id'";
		$update = @mysql_query("UPDATE stats SET `made`='$made',`file_location`='$location',`width`='$width',`length`='$height',`hits`='$hits',`uni_hits`='$hits',`views`='$views',`uni_views`='$uni_views' WHERE `id`='$id'");	UPDATE stats SET `made`='$made',`file_location`='$location',`width`='$width',`length`='$height',`hits`='$hits',`uni_hits`='$hits',`views`='$views',`uni_views`='$uni_views' WHERE `id`='$id'
		$update = @mysql_query("UPDATE stats SET `made`='$made',`width`='$width',`length`='$height',`hits`='$hits',`uni_hits`='$hits',`views`='$views',`uni_views`='$uni_views' WHERE `id`='$id'");	UPDATE stats SET `made`='$made',`width`='$width',`length`='$height',`hits`='$hits',`uni_hits`='$hits',`views`='$views',`uni_views`='$uni_views' WHERE `id`='$id'
		$stat = @mysql_fetch_array(@mysql_query("SELECT * FROM stats WHERE id='$id'"));	@mysql_query("SELECT * FROM stats WHERE id='$id'")

ตารางที่ 3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	โค้ดที่โดนโจมตี
12	Advance\remove .php	$banner_info = @mysql_fetch_array(@mysql_query("SELECT * FROM `banners` WHERE `id`='$delete'"));	@mysql_query("SELECT * FROM `banners` WHERE `id`='$delete'")
		$remove_banner = @mysql_query("DELETE FROM `banners` WHERE `id`='$delete'");	DELETE FROM `banners` WHERE `id`='$delete'
		$remove_stat = @mysql_query("DELETE FROM `stats` WHERE `id`='$delete'");	DELETE FROM `stats` WHERE `id`='$delete'
13	Ad Management \banner_ad\banner_ad\ran.php	MySQL_Query($Query);	$Query = "update banner set shown = 0 where ID = $OldShown";
		MySQL_Query($Query);	$Query = "update banner set shown = 1 where ID = $NewShown";
14	Ad Management \banner_ad\banner_ad\ban_install .php	$result = mysql_query($query);	$query = "INSERT INTO `banner` (`ID` , `banner` , `link` , `shown`) VALUES ('', '$banner', '$link', '$shown')";
		$result = mysql_query($query);	$query = "delete from banner where ID = '$ID'";

ตารางที่ 3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	โค้ดที่โดนโจมตี
15	Ad Management \azbanner\admin\ stats.php	if($action==loeschen){$loesche=mysql_query("DELETE FROM $ad_table WHERE ID=$ID");}	$loesche=mysql_query("DELETE FROM $ad_table WHERE ID=$ID");
		$stats=mysql_query("UPDATE $ad_table SET IMPRESSIONS=0 WHERE ID=$ID");	"UPDATE $ad_table SET IMPRESSIONS=0 WHERE ID=$ID"
		$stats=mysql_query("UPDATE $ad_table SET KLICKS=0 WHERE ID=$ID");	"UPDATE $ad_table SET KLICKS=0 WHERE ID=$ID"
		if($action==partnernew){ $new=mysql_query("INSERT INTO $ad_table (ID, ART, PARTNER, CODE, IMPRESSIONS) VALUES (",1,'$BEZEICHNUNG','$CODE',0)");}	$new=mysql_query("INSERT INTO $ad_table (ID, ART, PARTNER, CODE, IMPRESSIONS) VALUES (",1,'$BEZEICHNUNG','$CODE',0)");
		if($bannermodify==true){ $aendern=mysql_query(" UPDATE $ad_table Set URL = '$URL' WHERE ID = '$ID'");	$aendern=mysql_query("UPDATE $ad_table Set URL = '$URL' WHERE ID = '$ID'");

ตารางที่ 3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	โค้ดที่โดนโจมตี
15	Ad Management \azbanner\admin\ stats.php	if($action==bannernew){ $new=mysql_query("INS ERT INTO $ad_table (ID, ART, BILD, URL, IMPRESSIONS, KLICKS) VALUES (',0,'$BANNER','$URL',0 ,0)");}	$new=mysql_query("INSERT INTO $ad_table (ID, ART, BILD, URL, IMPRESSIONS, KLICKS) VALUES (',0,'$BANNER','$URL',0,0)");
		$aendern=mysql_query(" UPDATE $ad_table Set BILD = '$BILD' WHERE ID = '$ID'");	$aendern=mysql_query("UPDATE $ad_table Set BILD = '$BILD' WHERE ID = '$ID'");
		if($partnermodify==true){ $aendern=mysql_query(" UPDATE $ad_table Set PARTNER = '$PARTNER' WHERE ID = '$ID'");	$aendern=mysql_query("UPDATE $ad_table Set PARTNER = '$PARTNER' WHERE ID = '$ID'");
		$aendern=mysql_query(" UPDATE $ad_table Set CODE = '$CODE' WHERE ID = '$ID'");	$aendern=mysql_query("UPDATE $ad_table Set CODE = '$CODE' WHERE ID = '$ID'");
		$abfrage=mysql_query("S ELECT * FROM $ad_table WHERE ART=0 ORDER BY ID");	$abfrage=mysql_query("SELECT * FROM $ad_table WHERE ART=0 ORDER BY ID");

ตารางที่ 3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	โค้ดที่โดนโจมตี
15	Ad Management \azbanner\admin\ stats.php	$abfrage=mysql_query("S ELECT * FROM $ad_table WHERE ART=1 ORDER BY ID");	$abfrage=mysql_query("SELECT * FROM $ad_table WHERE ART=1 ORDER BY ID");
16	Affiliate Programs\oversto ck_dfs\feeds \upload.php	$queryp2=mysql_query(" LOAD DATA LOCAL INFILE '$file' REPLACE INTO TABLE $table FIELDS TERMINATED BY ',' ENCLOSED BY '\"' LINES TERMINATED BY '\n' IGNORE 1 LINES");	$queryp2=mysql_query("LOAD DATA LOCAL INFILE '$file' REPLACE INTO TABLE $table FIELDS TERMINATED BY ',' ENCLOSED BY '\"' LINES TERMINATED BY '\n' IGNORE 1 LINES");
		$mysqlresult = mysql_query("update $table set rndnumber =1000000*rand() ");	"update $table set rndnumber=1000000*rand() ";
17	Affiliate Programs\oversto ck_dfs \category.php	$querypcatname=mysql_q uery("Select * from categories where id='$_GET[cat]' ");	$querypcatname=mysql_query("Select * from categories where id='$_GET[cat]' ");
		$queryp1=mysql_query(" Select * from $table where category='$catslash' ");	$queryp1=mysql_query("Select * from $table where category='$catslash' ");

ตารางที่ 3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	โค้ดที่โดนโจมตี
17	Affiliate Programs\oversto ck_dfs \categoiy.php	$queryp=mysql_query("S elect * from $table where category='$catslash' limit $start,$num_per_page");	$queryp=mysql_query("Select * from $table where category='$catslash' limit $start,$num_per_page");
18	Affiliate Programs\oversto ck_dfs \dbsetup.php	if (mysql_query($sql)) {	$sql="insert INTO admin VALUES ('1','$_POST[username]','$pass','MSCA xU4Bdrc');";
19	Click Tracking \counter\counter\ counter.php	mysql_query("UPDATE sites SET clicks = '$row [1]', users ='$udusers' WHERE id='$row[0]'", $db)or die(mysql_error());	UPDATE sites SET clicks='$row[1]', users='$udusers' WHERE id='$row[0]'
		$result = mysql_query($sql) or die(mysql_error());	$sql = "INSERT INTO sites (id, clicks, url, users) VALUES ('$id', '$clicks', '$xrl', '$userid')" or die(mysql_error());
		mysql_query("DELETE FROM sites WHERE id=$id");	DELETE FROM sites WHERE id=$id
		$result = mysql_query("SELECT id, clicks, url, users FROM sites WHERE id=$id", $db) or die();	SELECT id, clicks, url, users FROM sites WHERE id=$id

ตารางที่ 3.1 ผลการสแกนไฟล์ PHP โดยใช้ Pixy (ต่อ)

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	โค้ดที่โดนโจมตี
20	Classified Ads\PHPRentals\ phprentals\admin \passwordchange .php	mysql_query("UPDATE users SET pword = md5('$newpass') WHERE email = '$user'");	UPDATE users SET pword = md5('$newpass') WHERE email = '$user'
		mysql_query("UPDATE users SET pword = md5('$newpass') WHERE email = '$user'");	UPDATE users SET pword = md5('$newpass') WHERE email = '$user'

ในตารางที่ 3.1 ผู้จัดทำได้ใช้ไฟล์เว็บแอพพลิเคชันส่วนหนึ่งจาก 1800 PHP scripts web developers mega pack ในการทดสอบหาช่องโหว่ SQL Injection โดยจากการสแกนไฟล์ทั้งหมด พบว่าโค้ดที่เป็นช่องโหว่นั้น มีการใช้คำสั่ง mysql_query () ซึ่งเป็นคำสั่งที่ใช้ในการ query ข้อมูลจากฐานข้อมูลโดยข้อมูลที่อยู่ในวงเล็บใน ( ) หลังคำสั่งนั้นมีด้วยกัน 2 รูปแบบคือ

1) ตัวแปรประเภทสตริง ซึ่งมีรูปแบบการเขียนคือ mysql_query (ตัวแปร); เช่น

- mysql_query ($query);
- $objQuery = mysql_query($strSQL);

โดยต้องมีการกำหนดค่าของตัวแปรที่ใช้ก่อน

2) สตริงคำสั่ง SQL ซึ่งมีรูปแบบการเขียนคือ mysql_query (คำสั่ง SQL); เช่น

- $banner = @mysql_query("SELECT * FROM banners WHERE `id`='$id');
- mysql_query("DROP TABLE $table_ads;"); เป็นต้น

นอกจากนี้ mysql_query สามารถใช้ได้กับ คำสั่ง sprintf โดยมีรูปแบบดังนี้

```
$sql = sprintf("DELETE from 'user' WHERE 'user_id' = '%s' ",
$_POST['user_id']);
```

โดยที่ %s แทนตัวแปรสตริง $_POST['user_id']

เนื่องจากการใช้ฐานข้อมูลในภาษา PHP มีการใช้ในลักษณะ PHP Data Objects (PDO) ได้ด้วย ทางผู้จัดทำจึงนำไฟล์ PHP ที่มีการเข้าถึงข้อมูลของฐานข้อมูลในลักษณะ PDO ไปทดสอบกับ Pixy พบว่า Pixy ไม่สามารถทำการตรวจสอบโค้ดได้

3.1.4 สรุปผลการวิเคราะห์ระบบ

จากผลการทดลองพบว่า Pixy สามารถสแกนไฟล์ที่มีโค้ดที่เป็นอันตรายต่อ SQL Injection ไม่ว่าจะเป็น SQL Injection ที่เกิดจากการโจมตีแบบใช้ quote (') line และ inline comment ได้ เนื่องจากโค้ดเหล่านี้มีการใช้ประโยชน์จากการสร้างคำสั่ง SQL โดยใช้คำสั่ง mysql_query ดังนั้น ถ้าแก้ไขไม่ให้เกิดช่องโหว่ที่คำสั่งนี้ได้จะป้องกันการโจมตีประเภท SQL Injection ที่กล่าวมาแล้วได้ แต่ Pixy ไม่สามารถสแกนไฟล์ PHP ที่ใช้ PDO ได้ ดังนั้นในส่วนของการสแกนและแก้ไขโค้ดที่พบช่องโหว่ แบ่งออกเป็น 2 ส่วนหลัก

1) การแก้ไขในส่วนของการใช้คำสั่ง mysql_query

2) การแก้ไขในส่วนของ PDO

3.1.5 ขอบเขตของระบบ

1) การสแกน

- ระบบสามารถตรวจสอบได้เฉพาะภาษา PHP
- โค้ดที่เขียนต้องใช้การเชื่อมต่อกับฐานข้อมูล MySQL หรือ เชื่อมต่อกับฐานข้อมูลแบบ PDO
- สามารถตรวจสอบได้กับการโจมตีที่มีการใช้ช่องโหว่ของคำสั่ง SQL ในการสร้างคำสั่งเติม quote หลังการใช้ ได้แก่ การใช้ Line และ Inline comment

2) การแก้ไข

- สามารถแก้ไขการเขียนด้วยคำสั่ง mysql_query ของฐานข้อมูล MySQL
- สามารถแก้ไขการเขียนด้วยคำสั่ง ->query ของฐานข้อมูลแบบ PDO
- แสดงการแก้ไขทีละช่องโหว่หรือแก้ไขช่องโหว่ทั้งหมด

3) ส่วนช่วยเหลือผู้ใช้ (Help)

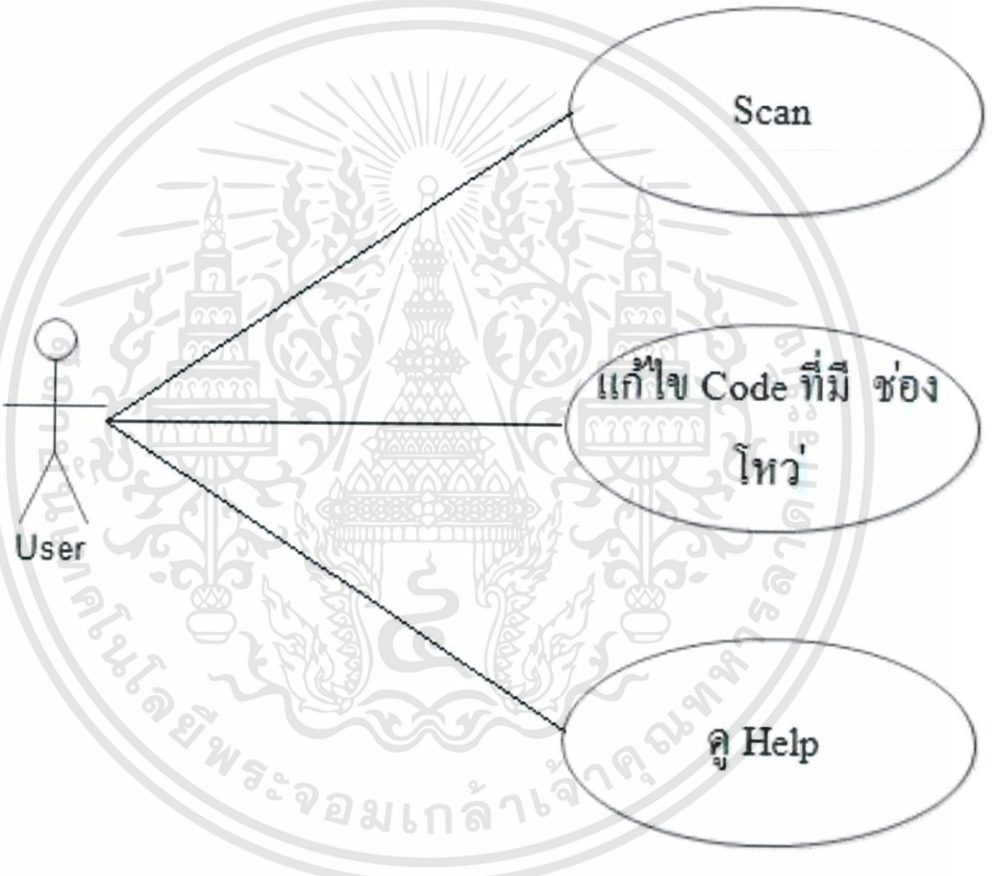
- เนื่องจากเป็นเรื่องที่ทางผู้พัฒนามีความรู้เรื่องนี้ไม่มากทางผู้จัดทำจึงได้สร้างส่วนช่วยเหลือผู้ใช้ (Help) ในการช่วยให้ความรู้ เรื่อง SQL Injection ซึ่งประกอบไปด้วย ความหมาย สาเหตุ และ วิธีการป้องกัน

3.2 การออกแบบระบบ

3.2.1 Use Case ของโปรแกรมปลั๊กอิน

การทำงานของโปรแกรมแบ่งออกเป็น 3 ส่วนคือ

1) สแกนไฟล์ PHP เพื่อหาช่องโหว่ที่ทำให้เกิด SQL Injection
2) แก้ไขโค้ดที่มีช่องโหว่ให้ปลอดภัย
3) ดู Help เพื่อช่วยให้ผู้ใช้งานทราบถึงที่มา สาเหตุ และวิธีป้องกัน SQL Injection

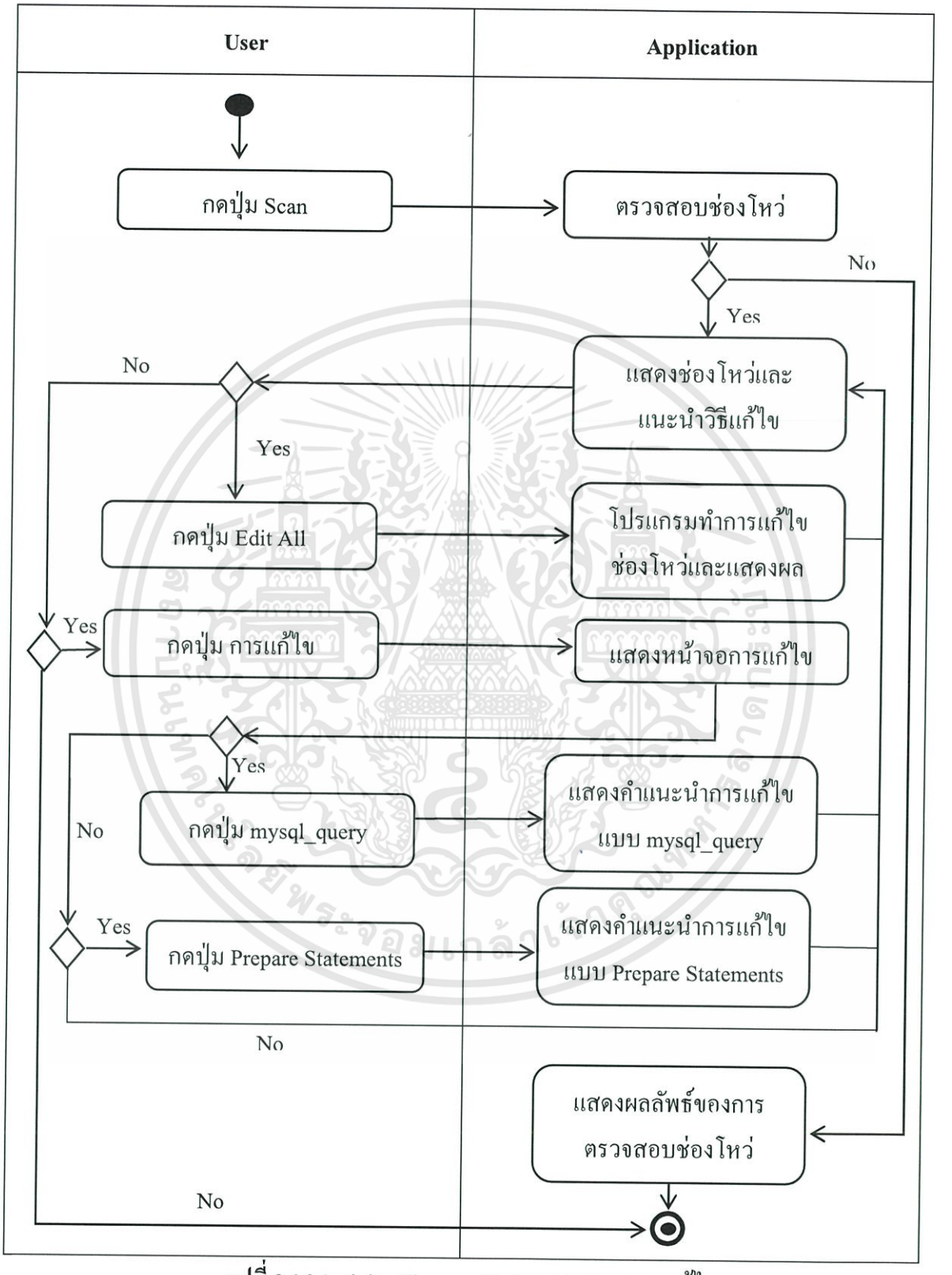


รูปที่ 3.15 Use Case ของโปรแกรม

3.2.2 Activity Diagram

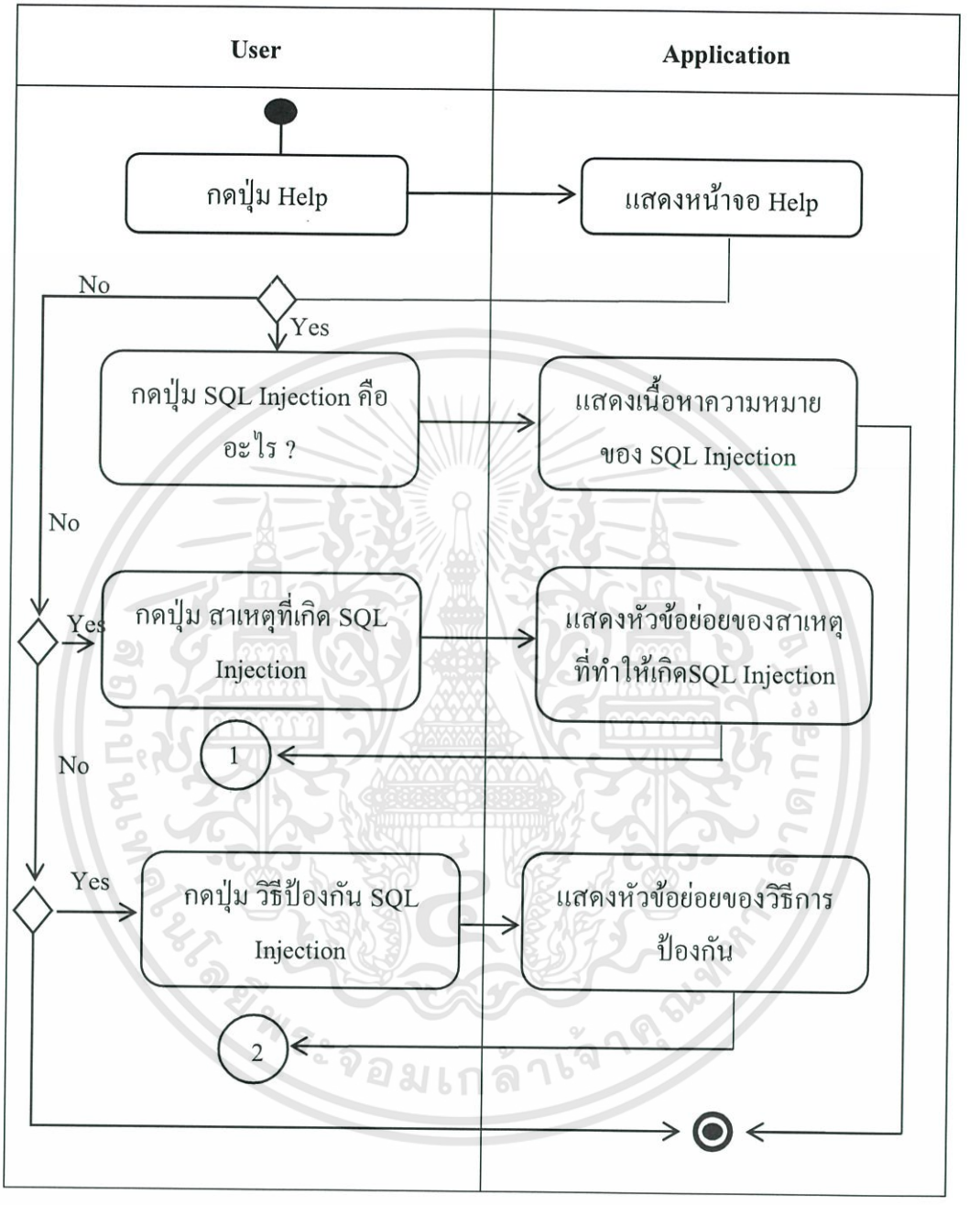
Activity Diagram แสดงการทำงานของระบบ

1) Activity Diagram ของระบบสแกนและแก้ไข



รูปที่ 3.16 Activity Diagram ของระบบสแกนและแก้ไข

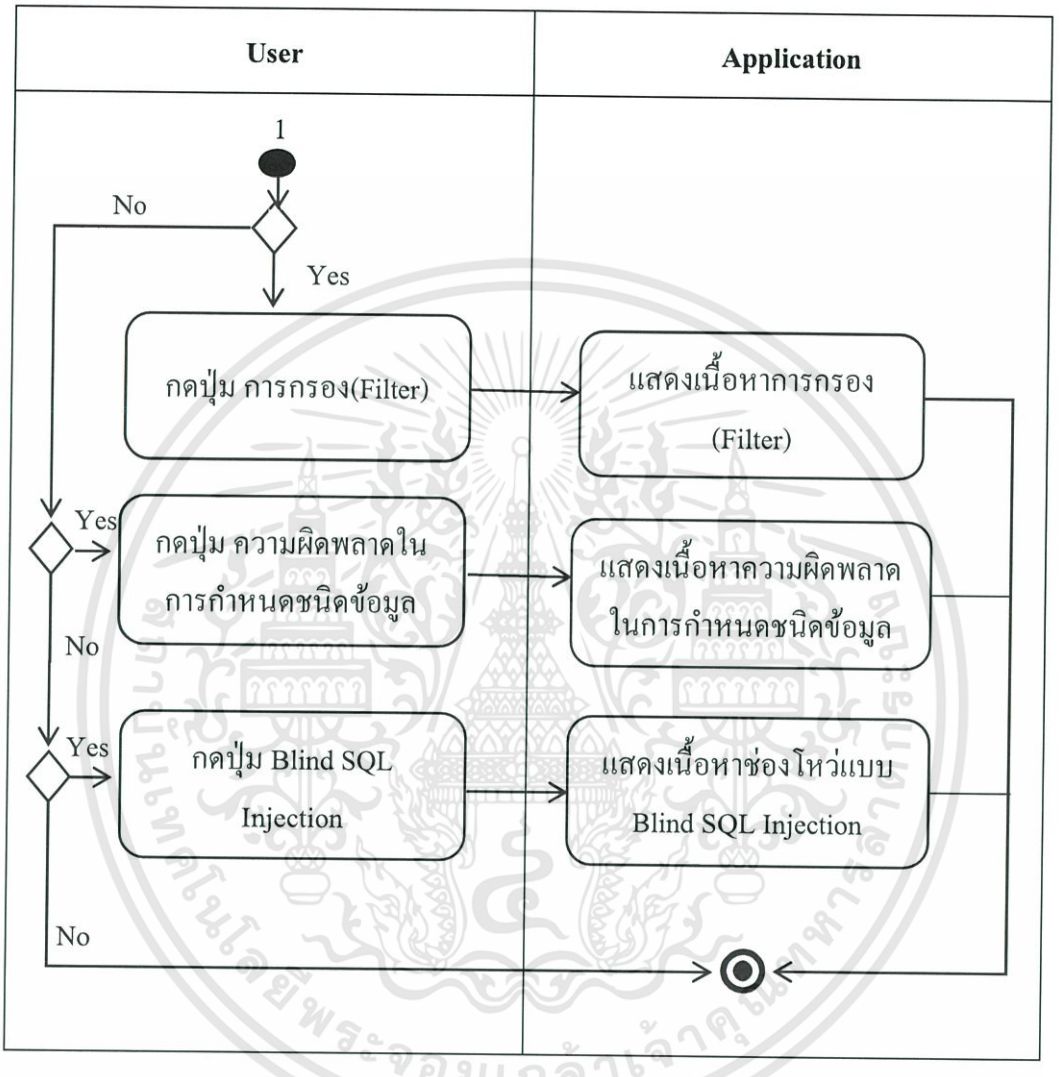
2) Activity Diagram ของระบบ Help



รูปที่ 3.17 Activity Diagram ของ Help

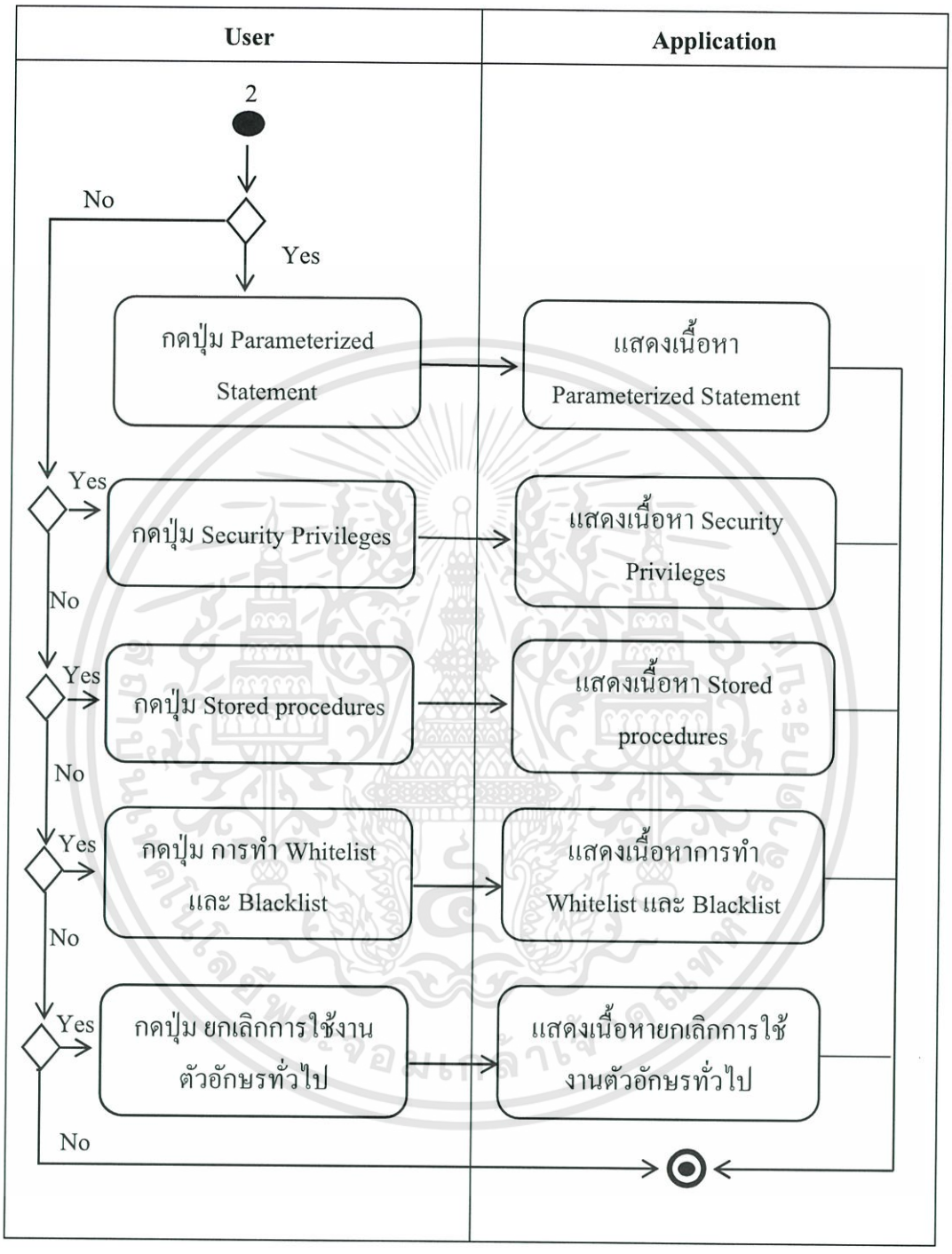
จาก Activity Diagram ของระบบ Help นั้นผู้ใช้งานสามารถใช้เมนู Help เพื่อศึกษาความรู้เพิ่มเติมเกี่ยวกับ SQL Injection โดยแบ่งออกเป็น 3 เมนูหลักคือ ความหมายของ SQL Injection สาเหตุที่ทำให้เกิด SQL Injection และวิธีการป้องกัน SQL Injection โดยในหัวข้อสาเหตุที่ทำให้เกิด SQL

Injection และวิธีการป้องกันนั้นจะมีหัวข้อย่อยลงไป ซึ่ง Activity Diagram ของสาเหตุของ SQL Injection และ วิธีการป้องกันเป็นดังรูปที่ 3.18 และ 3.19 ตามลำดับ



รูปที่ 3.18 Activity Diagram ของสาเหตุที่ทำให้เกิด SQL Injection

รูปที่ 3.18 จะเป็นเมนูย่อยสาเหตุที่ทำให้เกิด SQL Injection ของภาพที่ 3.17 โดยจะมีเมนูย่อยลงไปอีก 3 เมนูย่อยคือ เมนูการกรอง(Filter) เมนูความผิดพลาดในการกำหนดชนิดข้อมูล และเมนู Blind SQL Injection



รูปที่ 3.19 Activity Diagram ของวิธีป้องกัน SQL Injection

รูปที่ 3.19 แสดง Activity Diagram ของเมนูย่อยในวิธีป้องกัน SQL Injection ของรูปที่ 3.17 โดยจะมีเมนูย่อยลงไปอีก 5 เมนูย่อย คือ Parameterized Statement, Security Privileges, Stored procedures, การทำ Whitelist และ Blacklist และการยกเลิกการใช้งานตัวอักษรทั่วไป

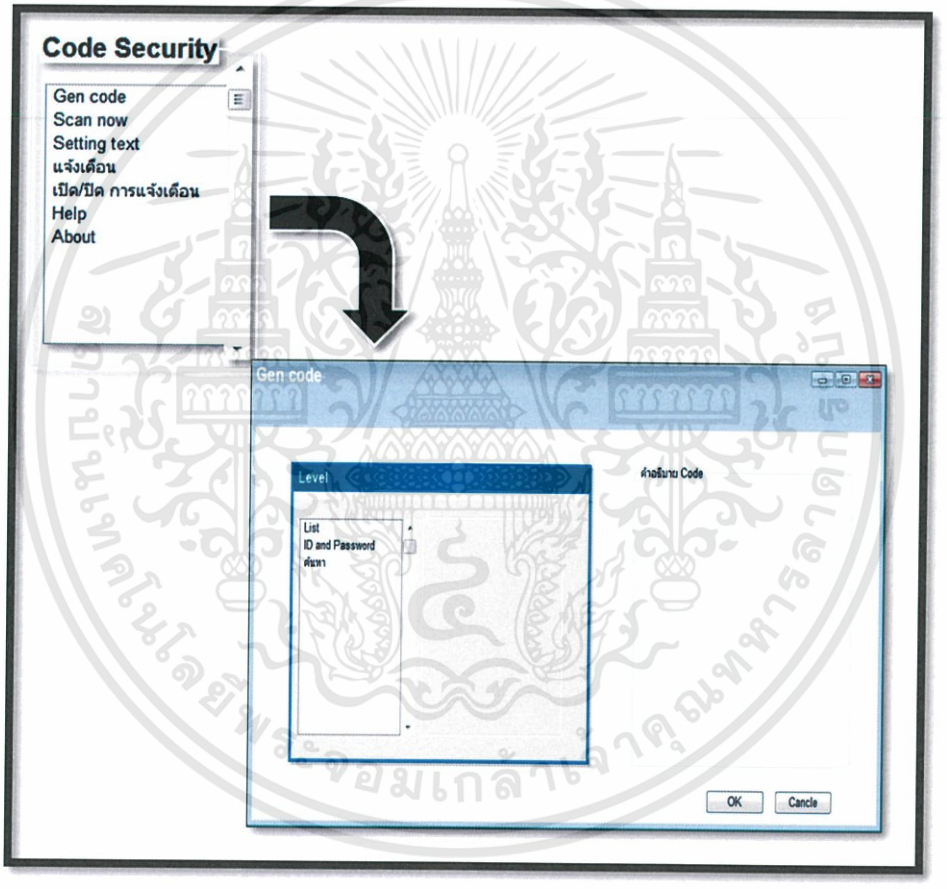
3.2.3 การออกแบบส่วนติดต่อผู้ใช้และการทำงาน

ในการออกแบบช่วงแรกนั้นผู้จัดทำได้ทำการออกแบบส่วนประกอบต่าง ๆ ดังนี้

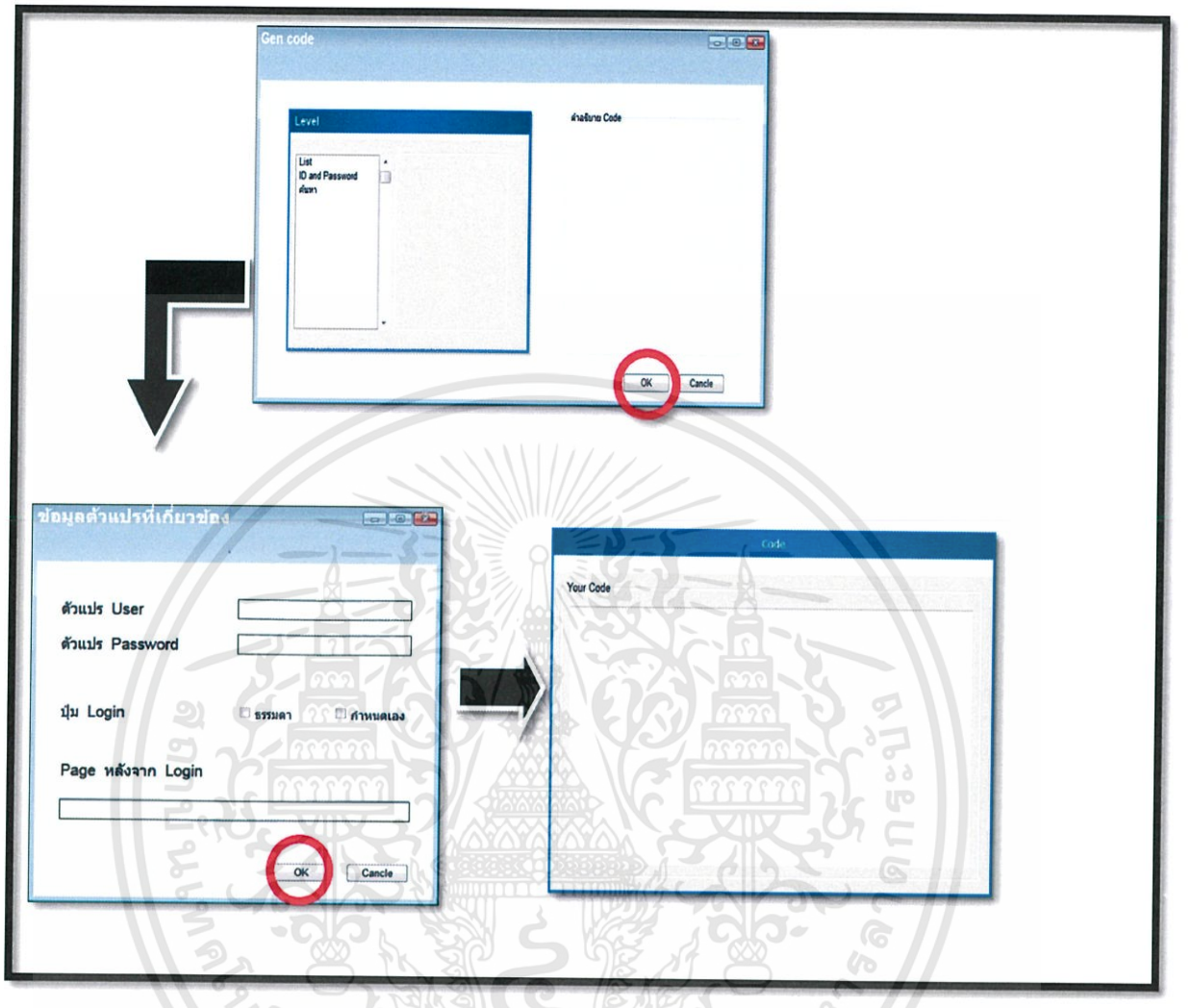
1) สร้างโค้ดที่ปลอดภัยให้อัตโนมัติ

โปรแกรมจะสร้างโค้ดที่ปลอดภัยต่อ SQL Injection ในลักษณะต่าง ๆ เช่น โค้ดการล็อกอิน โค้ดการค้นหาข้อมูล เป็นต้น

เมื่อผู้ใช้งานต้องการสร้างโค้ดที่โปรแกรมแนะนำให้เลือกเมนู Gen code จากนั้นโปรแกรมจะมีโค้ดแนะนำอยู่ 3 แบบ คือ List, ID and Password และค้นหา

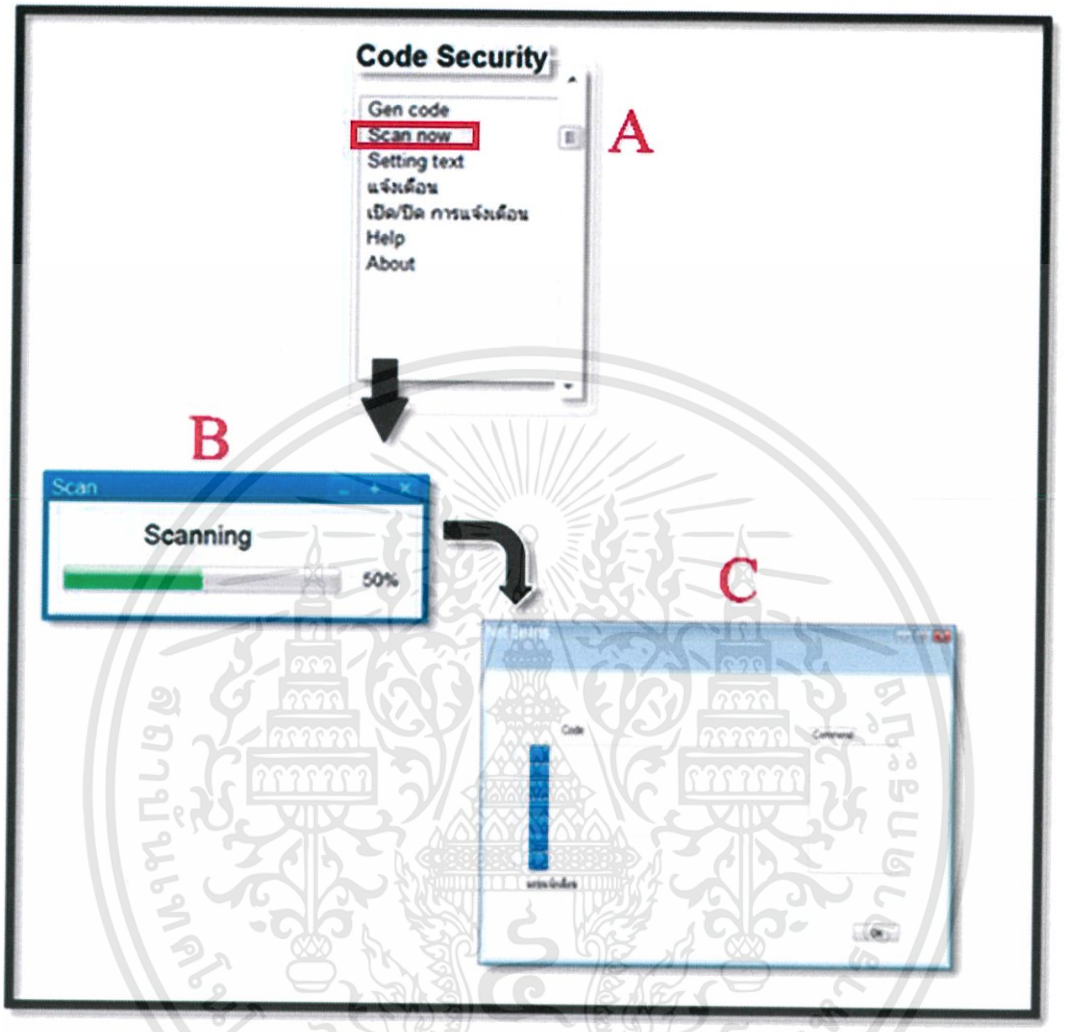


รูปที่ 3.20 หน้าจอการเลือกการใช้งานโปรแกรมในการ Generate Code



รูปที่ 3.21 หน้าจอการเลือกประเภทของการ Gen Code

2) สแกนไฟล์ PHP

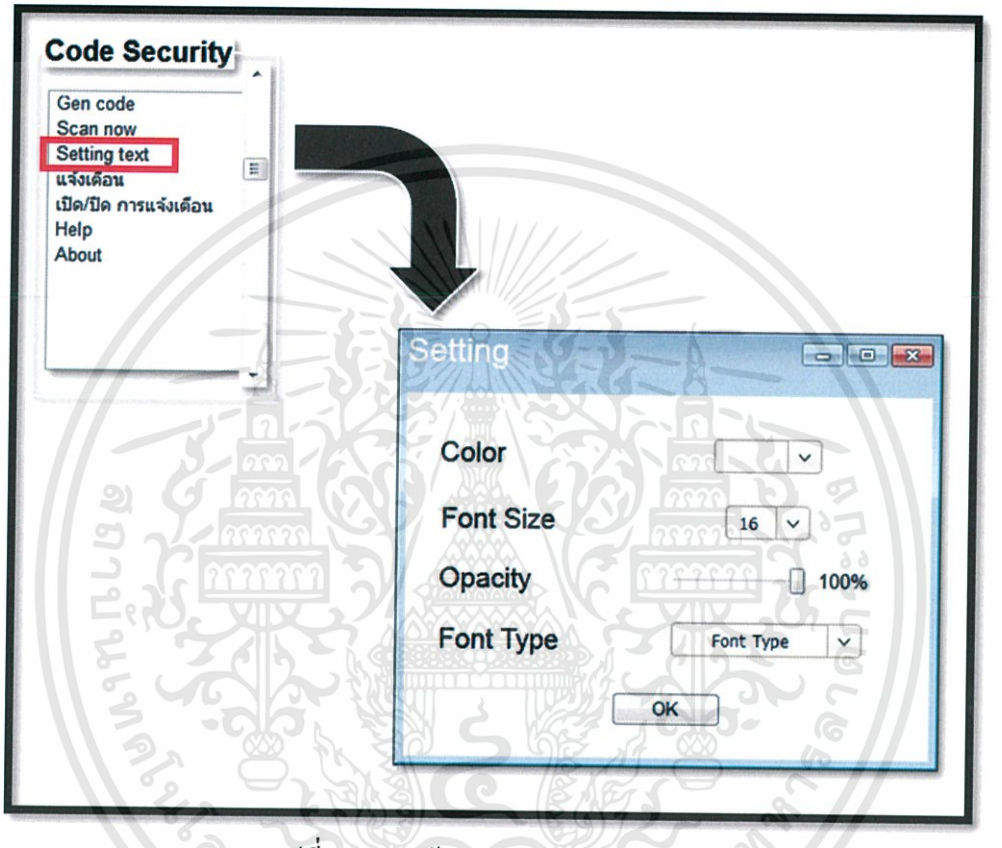


รูปที่ 3.22 หน้าจอของเมนู Scan now

จากรูปที่ 3.22 เมื่อผู้ใช้เปิดไฟล์ PHP บน NetBeans หรือ Eclipse แล้วต้องการสแกนไฟล์ ให้เลือกเมนู Scan now ของโปรแกรม(A) จากนั้นโปรแกรมจะทำการสแกนโค้ดที่ผู้ใช้งานต้องการสแกนโดยจะมีหน้าต่างแสดงความคืบหน้าในการแสดง(B) เมื่อสแกนเรียบร้อยแล้วโปรแกรมจะแสดงหน้าต่างผลการสแกน(C)

3) ตั้งค่าการแสดงผลของตัวอักษร

ผู้ใช้งานสามารถกำหนดลักษณะรูปแบบของการแสดงผลของโค้ดที่เป็นช่องโหว่ หรือโค้ดที่ถูกแก้ไขได้โดยเลือกเมนู Setting text จากนั้นจะมีหน้าต่างในการ Setting แบบอักษรในการแสดงผลดังรูปที่ 3.23



รูปที่ 3.23 หน้าจอของเมนู Setting text

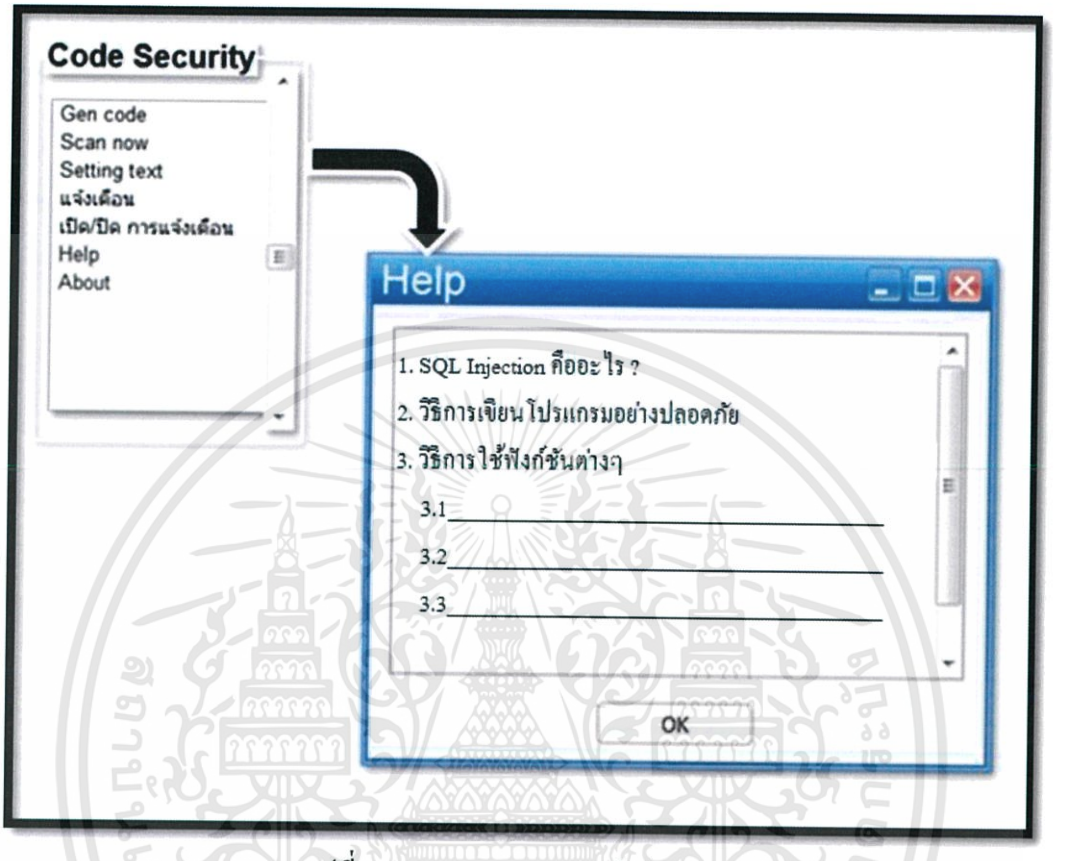
4) แจ้งเตือน

เป็นการแจ้งเตือนผลลัพธ์ของการสแกนแบบเรียลไทม์

5) เปิด/ปิด การแจ้งเตือน

ผู้ใช้งานสามารถเปิด/ปิดการแจ้งเตือนแบบเรียลไทม์ได้

6) Help



รูปที่ 3.24 Help Menu ของโปรแกรม

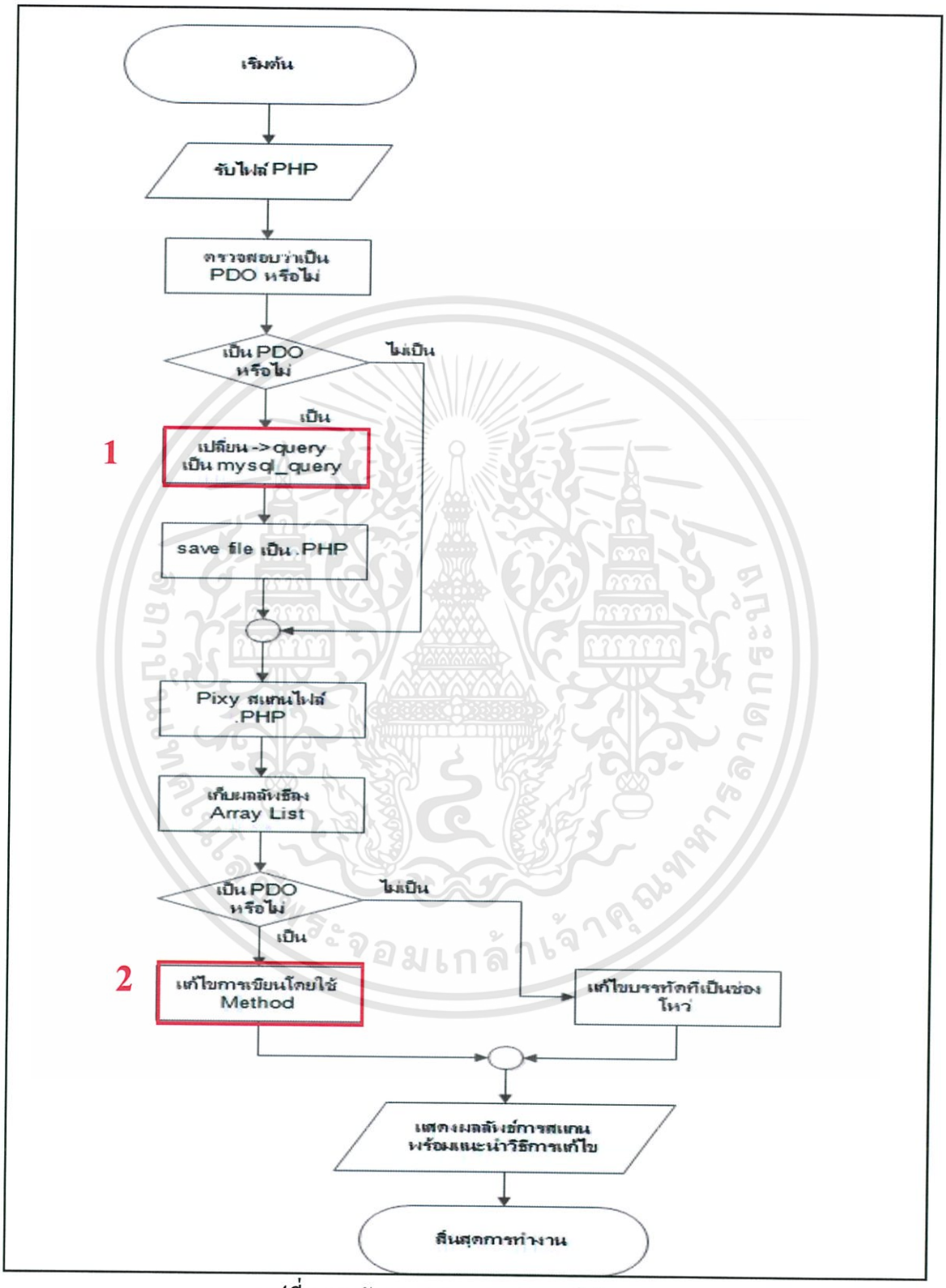
จากรูปที่ 3.24 เมื่อผู้ใช้งานต้องการศึกษาข้อมูลเกี่ยวกับ SQL Injection เพิ่มเติม สามารถเลือกเมนู Help โดย Help จะอธิบายเกี่ยวกับ SQL Injection คืออะไร วิธีการเขียนโปรแกรมอย่างปลอดภัย และวิธีการใช้งานฟังก์ชัน SQL ต่าง ๆ เป็นต้น

สรุปเมื่อทำการพัฒนาแล้ว ไม่สามารถพัฒนาเมนูต่อไปนี้ได้

1) สร้างโค้ดที่ปลอดภัยให้อัตโนมัติ เนื่องจากการเขียนโค้ด PHP ให้ปลอดภัยสามารถเขียนได้หลากหลาย ผู้จัดทำจึงไม่สามารถสร้างโค้ดที่เหมาะสมกับระบบต่างๆ ได้
2) ตั้งค่าการแสดงผลของตัวอักษร จำเป็นต้องอาศัยความเข้าใจ API ในการแสดงผลของ NetBeans และ Eclipse ซึ่งการศึกษาต้องใช้เวลา และ เกินกว่าขอบเขตงานที่วางแผนไว้
3) แจ้งเตือน
4) เปิด/ปิด การแจ้งเตือน

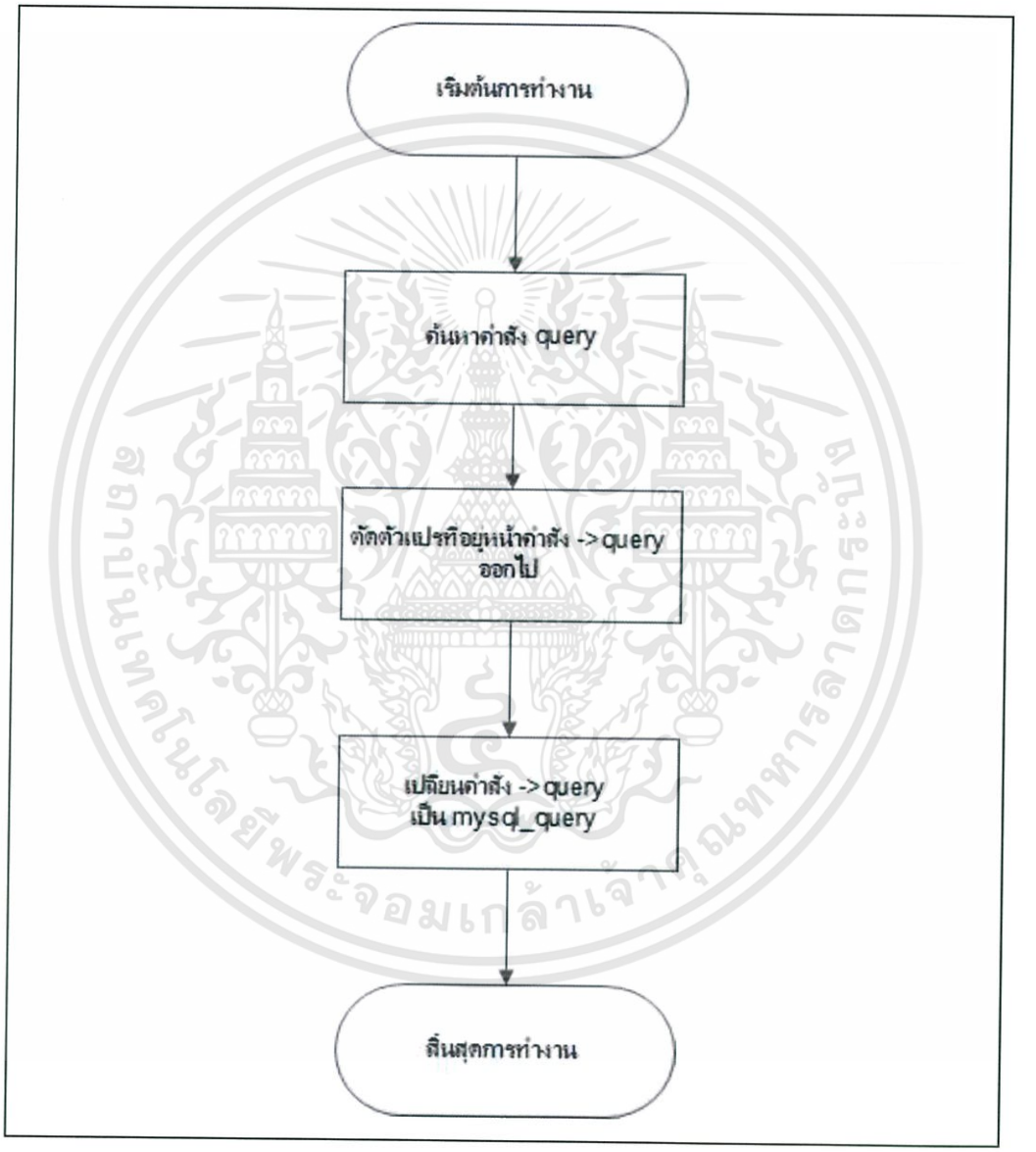
ในข้อสามและข้อสี่นั้น เนื่องจากการสแกนจำเป็นต้องใช้ไฟล์ที่เขียนโค้ดเสร็จแล้วทำให้ผู้จัดทำไม่สามารถทำการแจ้งเตือนได้

3.3 การพัฒนาโปรแกรม



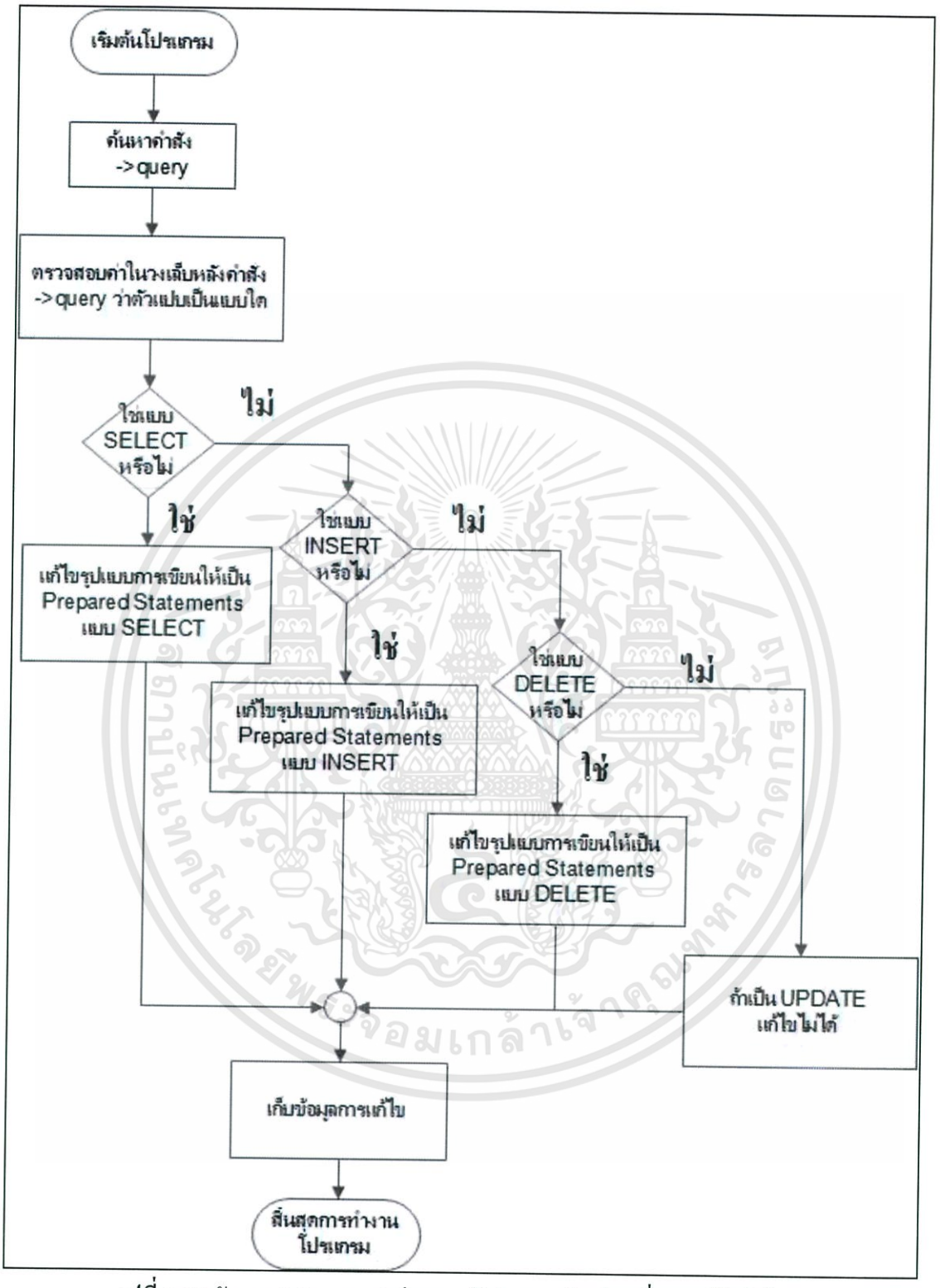
รูปที่ 3.25 ผังงาน (Flowchart) ของโปรแกรม

รูปที่ 3.25 อธิบายการทำงานของโปรแกรม เมื่อผู้เปิดไฟล์ PHP บน NetBeans หรือ Eclipse แล้วกดปุ่ม “Scan” โปรแกรมทำการสแกนไฟล์ PHP ที่ถูกเปิดใช้งานอยู่จากนั้นจะแสดงผลการสแกนว่าพบหรือไม่พบช่องโหว่ของ SQL Injection ผู้ใช้งานสามารถเลือกได้ว่าจะให้โปรแกรมทำการแก้ไขโค้ดในส่วนที่มี error ให้ทั้งหมดหรือเลือกดูคำแนะนำในการแก้ไขโค้ดใน error แต่ละบรรทัดแล้วจบการทำงานของโปรแกรม



รูปที่ 3.26 ผังงาน (Flowchart) ย่อยของโปรแกรมในส่วนที่ 1 (แก้ไขไฟล์ที่มีโค้ด PDO)

จากรูปที่ 3.26 แสดงถึงขั้นตอนการทำแปลงคำสั่ง query ข้อมูลลักษณะ PDO ให้เป็นแบบที่ใช้กับ MySQL เพื่ออาศัยการทำงานของ Pixy ในการสแกนไฟล์ PHP



รูปที่ 3.27 ผังงาน (Flowchart) ย่อยของโปรแกรมในส่วนที่ 2 (แก้ไขโค้ด SQL)

จากรูปที่ 3.27 แสดงถึงขั้นตอนในการแก้ไขคำสั่ง query ข้อมูลลักษณะ PDO จากแบบปกติให้เป็นแบบ Prepared Statementที่เข้ากับคำสั่ง SQL

บทที่ 4
ผลการดำเนินงานและการทดสอบระบบ

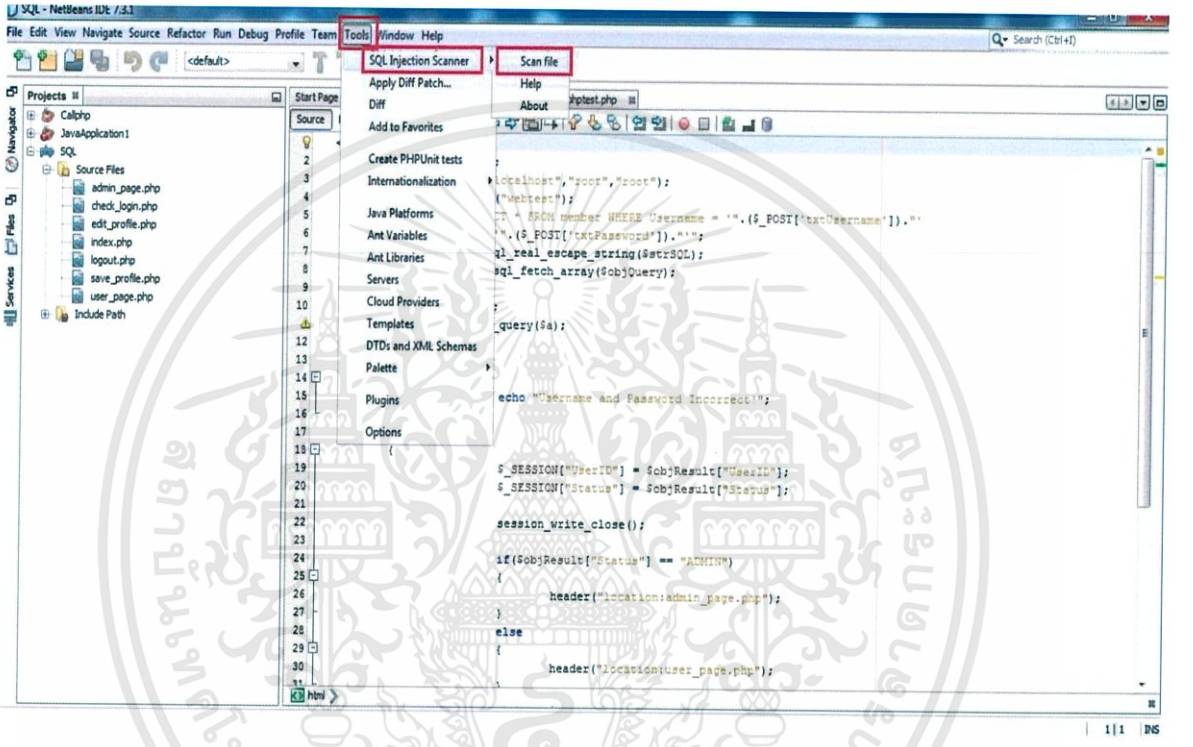
ผลลัพธ์ที่ได้จากการพัฒนาระบบคือ โปรแกรมปลั๊กอินบน NetBeans และ Eclipse ที่สามารถสแกนและแก้ไขช่องโหว่ของการโจมตีเว็บแอพพลิเคชันประเภท SQL Injection บนโปรแกรมเซิร์ฟเวอร์ไซต์ที่เขียนด้วยภาษา PHP ได้ ซึ่งไฟล์ที่สามารถสแกนได้คือไฟล์ที่ประกอบไปด้วยโค้ดคำสั่งที่มีการใช้ฟังก์ชัน mysql_query() และ โค้ดที่มีการติดต่อกับฐานข้อมูลแบบ PDO อีกทั้งโปรแกรมนี้มีส่วนที่ให้ความรู้ในการแนะนำความหมาย ประเภท และ การป้องกัน SQL Injection ให้กับผู้พัฒนาเว็บอีกด้วย

ในการใช้งานโปรแกรมนั้น ผู้ใช้ต้องทำการติดตั้งโปรแกรมปลั๊กอินซึ่งเป็นผลลัพธ์ของโครงงานพิเศษนี้ใน NetBeans และ Eclipse ก่อนซึ่งรายละเอียดการลงโปรแกรมสามารถดูได้จากภาคผนวก ก. การติดตั้งโปรแกรมปลั๊กอินบน NetBeans และ ภาคผนวก ข. การติดตั้งโปรแกรมปลั๊กอินบน Eclipse ผลการดำเนินงานของโปรแกรมจะแบ่งออกเป็น 2 ส่วนหลักคือ ส่วนการสแกนไฟล์และการแก้ไขโค้ดที่มีช่องโหว่ และส่วนช่วยเหลือผู้ใช้งาน (Help) ซึ่งแต่ละส่วนนั้นมีรายละเอียดดังต่อไปนี้

4.1 การสแกนไฟล์และแก้ไขโค้ดที่มีช่องโหว่

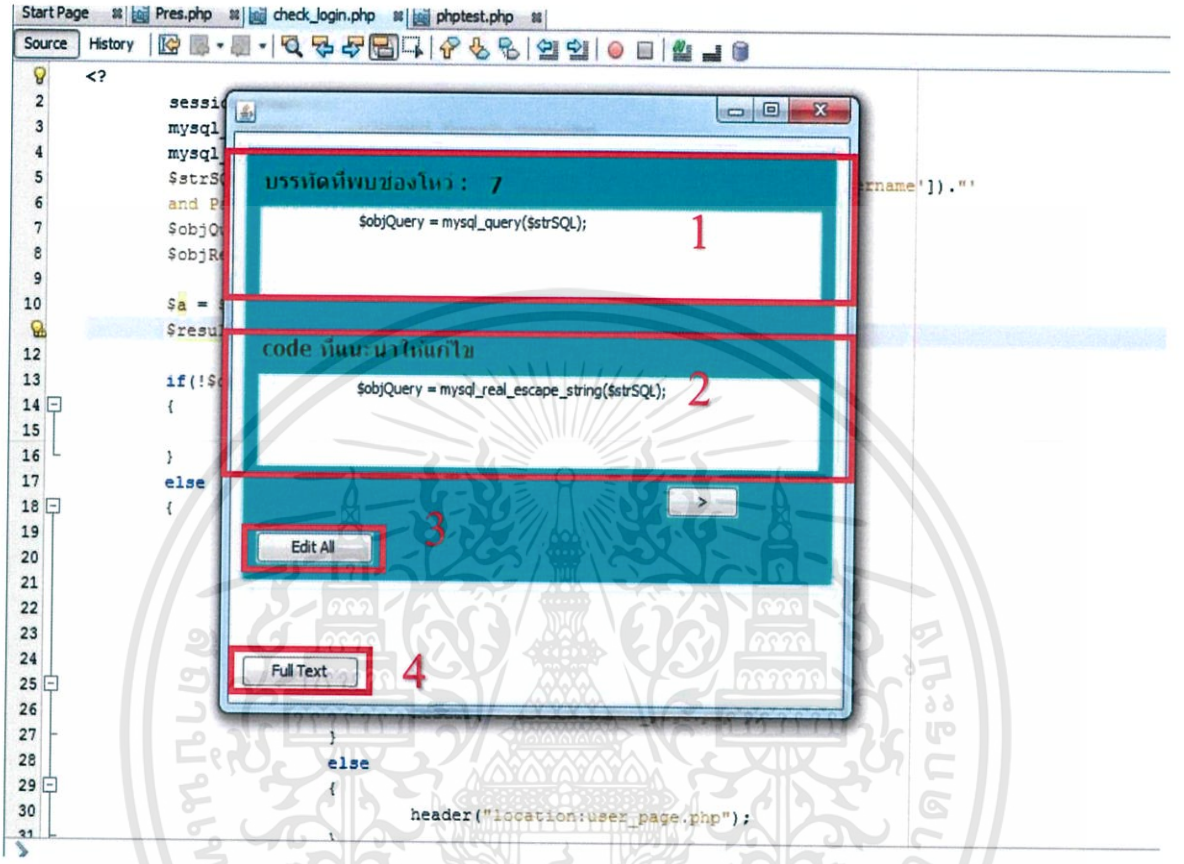
4.1.1 การสแกนไฟล์และแก้ไขโค้ดที่มีช่องโหว่บน NetBeans

1) เมื่อผู้พัฒนาโปรแกรมเว็บต้องการตรวจสอบโปรแกรมให้เปิดไฟล์ที่ต้องการสแกนบน NetBeans จากนั้นเลือก Tools บน Menu bar แล้วกด SQL Injection Scanner > Scan file เพื่อทำการสแกนไฟล์ ดังรูปที่ 4.1



รูปที่ 4.1 หน้าจอการสแกนไฟล์บน NetBeans

2) จากนั้นจะมีหน้าต่างขึ้นมาแสดงผลลัพธ์ของการสแกนซึ่งประกอบด้วย บรรทัดที่พบช่องโหว่ โค้ดที่แนะนำให้แก้ไข ปุ่ม Edit All และ ปุ่ม Full Text

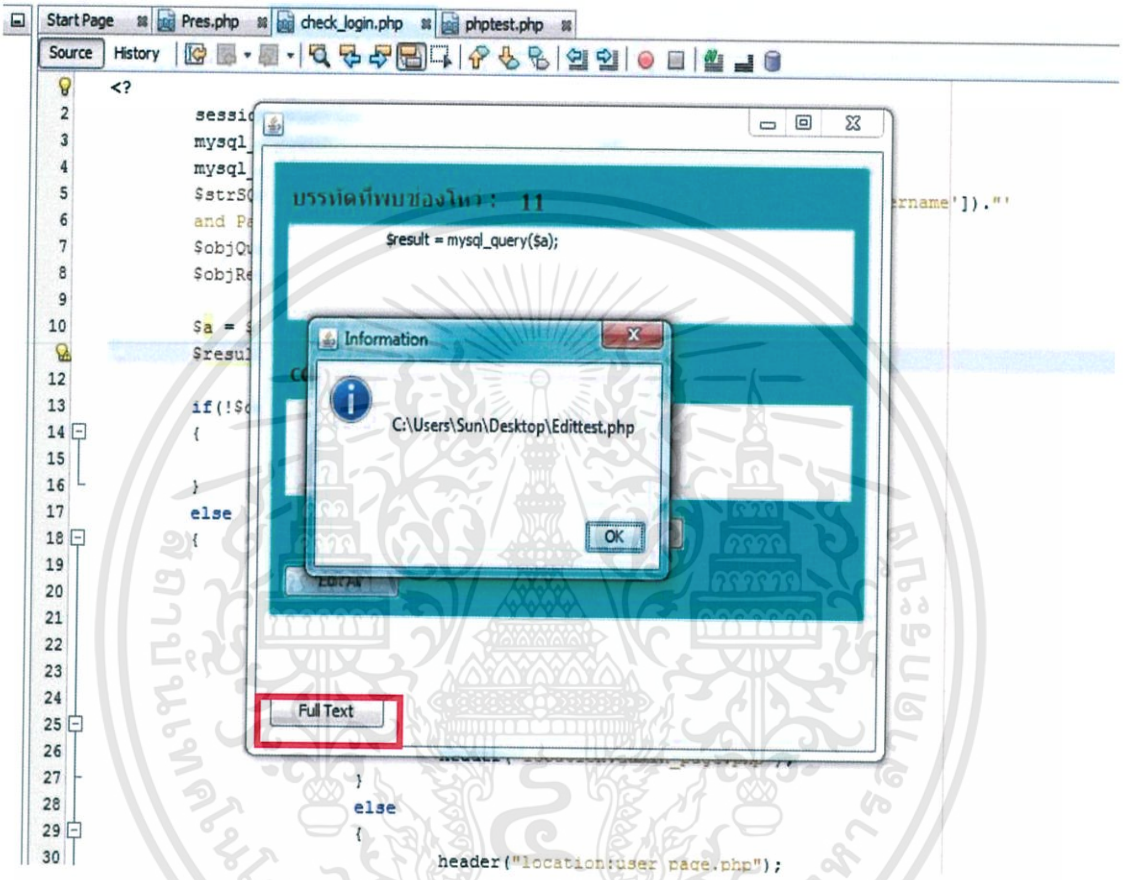


รูปที่ 4.2 หน้าจอแสดงผลการสแกนไฟล์

2.1) บรรทัดที่พบช่องโหว่ : ในส่วนนี้โปรแกรมจะแจ้งบรรทัดที่พบช่องโหว่ (ในตัวอย่างคือบรรทัดที่ 7) และคำสั่งที่พบช่องโหว่ ซึ่งโปรแกรมจะแสดงผลลัพธ์ทีละช่องโหว่และจะมีปุ่ม > สำหรับดูช่องโหว่ถัดไป

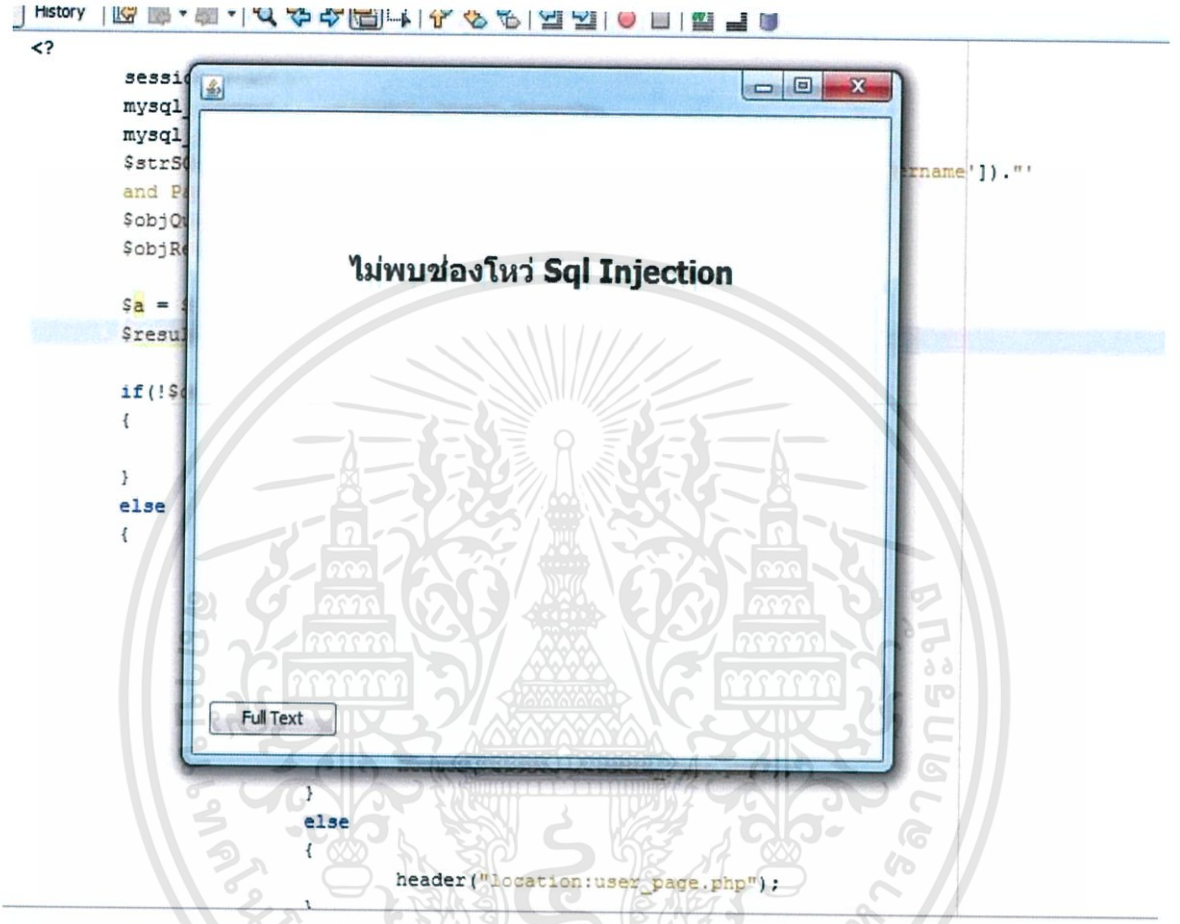
2.2) โค้ดที่แนะนำให้แก้ไข: ในส่วนนี้โปรแกรมจะแสดงโค้ดที่ผู้ใช้ควรแก้ไขเพื่อไม่ให้เกิดช่องโหว่

2.3) ปุ่ม Edit All : คือ ปุ่มแก้ไขโค้ดทั้งหมดให้โดยอัตโนมัติซึ่งเมื่อผู้ใช้งานกดปุ่มนี้ โปรแกรมจะทำการแก้ไขโค้ดที่มีช่องโหว่ทั้งหมดให้โดยอัตโนมัติและจะทำการบันทึกโค้ดที่แก้ไขแล้วลงในไฟล์ใหม่ เมื่อแก้ไขเรียบร้อยแล้วโปรแกรมจะแสดงหน้าจอดังในรูปที่ 4.3



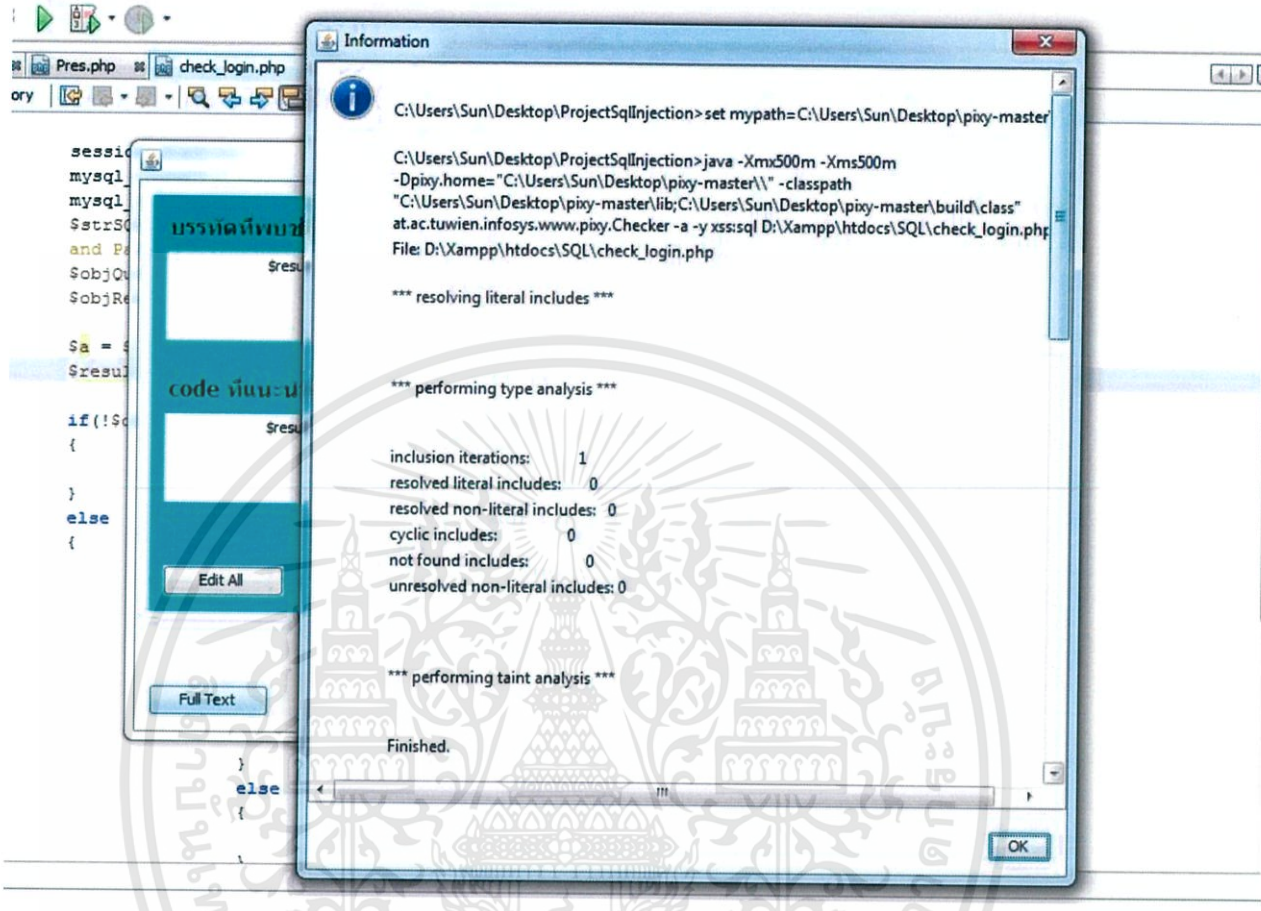
รูปที่ 4.3 หน้าจอแสดงผลการใช้งานฟังก์ชัน Edit All ในการแก้ไขช่องโหว่

ในกรณีที่โปรแกรมสแกนไฟล์ PHP แล้วไม่พบโค้ดที่มีช่องโหว่ โปรแกรมจะมีการแจ้งเตือนว่า “ไม่พบช่องโหว่ SQL Injection” ดังรูปที่ 4.4



รูปที่ 4.4 หน้าจอแสดงผลการแก้ไขไฟล์กรณีไม่พบช่องโหว่

2.4) ปุ่ม Full Text : ปุ่มนี้ใช้ในกรณีที่ผู้ใช้ต้องการดูผลการสแกนของ Pixy

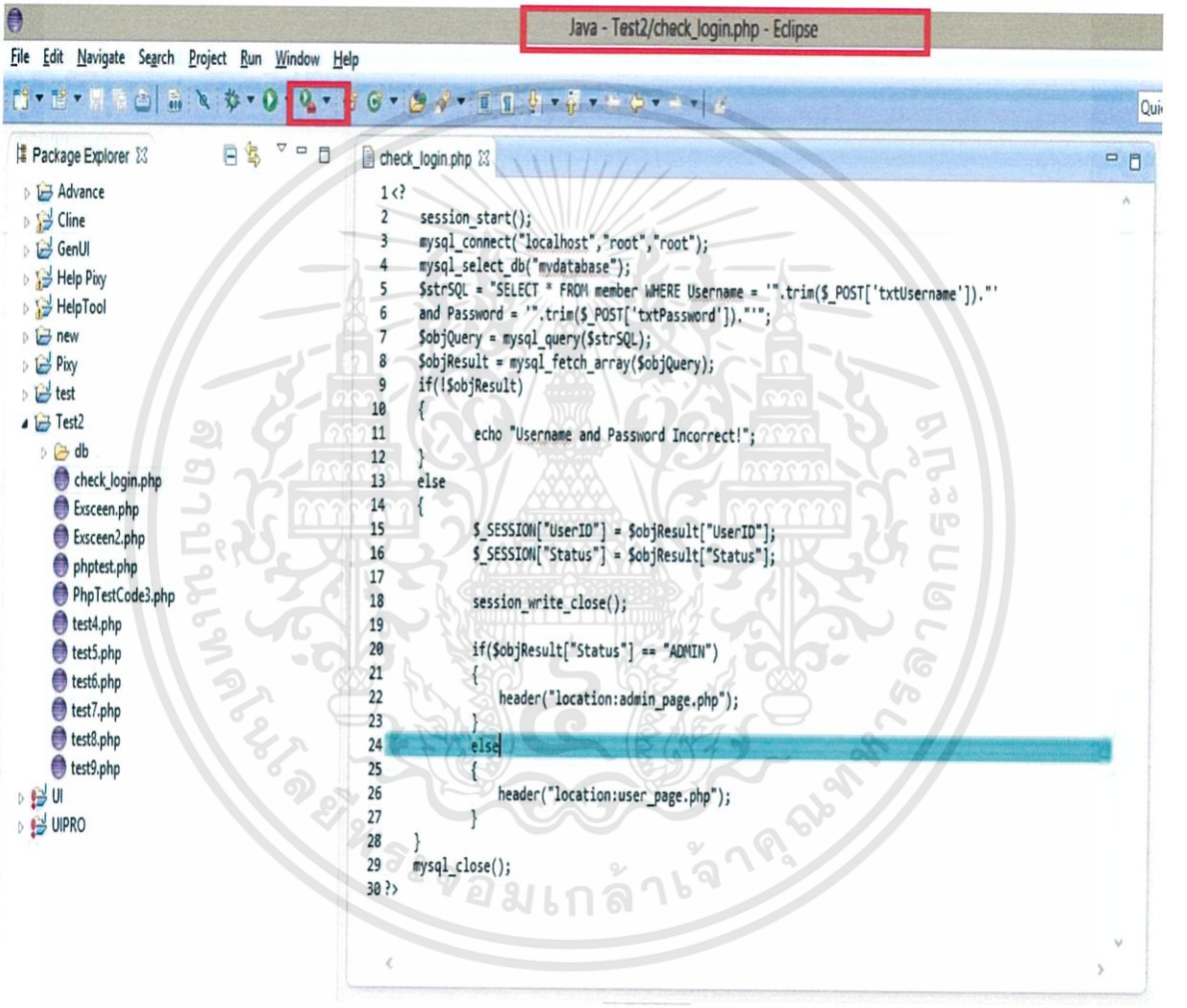


รูปที่ 4.5 หน้าจอแสดงผลลัพธ์ของปุ่ม Full Text

4.1.2 การสแกนไฟล์และแก้ไขโค้ดที่มีช่องโหว่บน Eclipse

หลังจากทำการปลั๊กอินโปรแกรมเข้ากับ Eclipse แล้ว เมื่อต้องการสแกนให้ทำดังวิธีต่อไปนี้

1) เปิดไฟล์ที่ต้องการสแกนบน Eclipse จากนั้นเลือก Tools บน Menu bar แล้วกด SQL Injection Scanner > Scan file รูปที่ 4.6 เป็นตัวอย่างการสแกนไฟล์ check_login.php



รูปที่ 4.6 หน้าจอการสแกนไฟล์บน Eclipse

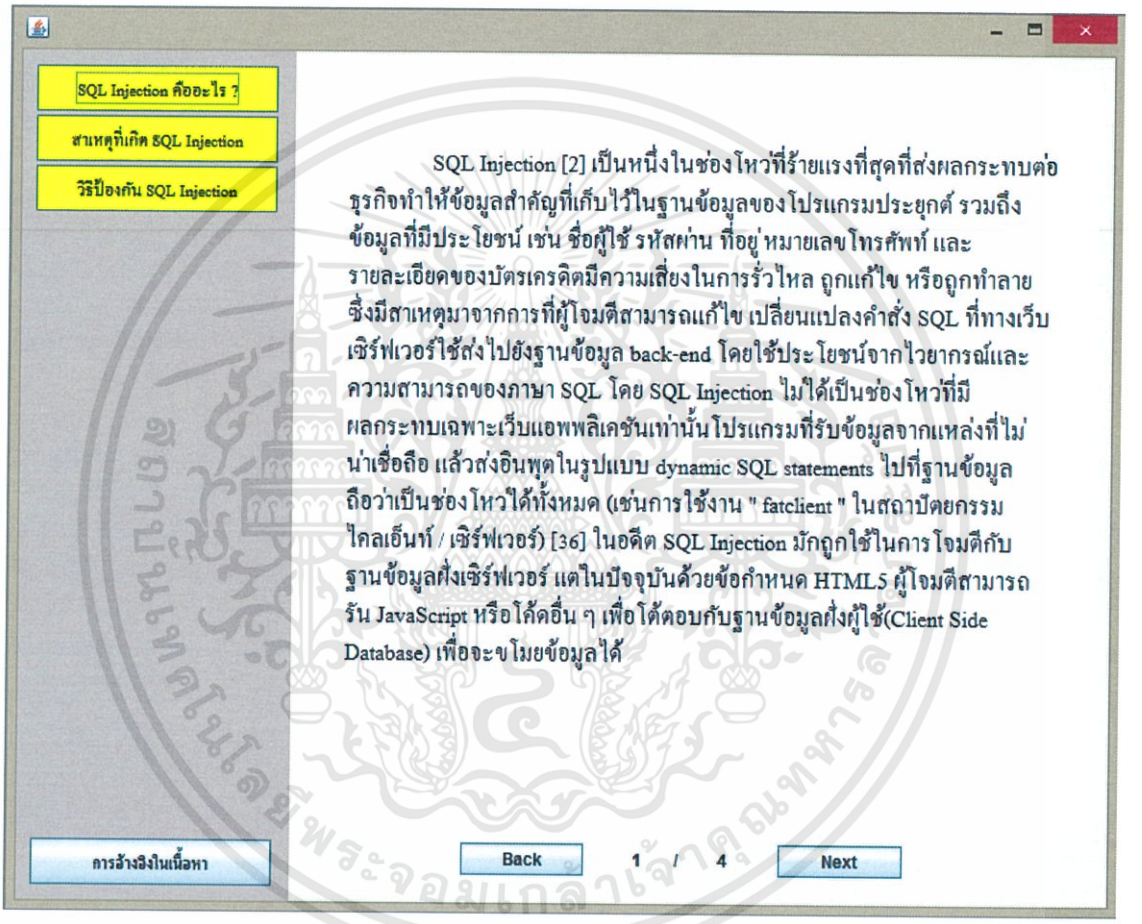
2) โปรแกรมปลั๊กอินจะแสดงผลลัพธ์ที่ได้จากการสแกนเหมือนกับการสแกนบน NetBeans ทุกประการ

4.2 ส่วนช่วยเหลือผู้ใช้งาน (Help)

Help เป็นส่วนของโปรแกรมที่จะแสดงข้อมูลให้ผู้ใช้งานสามารถศึกษาหาข้อมูลเพิ่มเติมเกี่ยวกับ SQL Injection โดยจะมีหัวข้อต่างๆ ดังใน Help

1) ความหมายของ SQL Injection

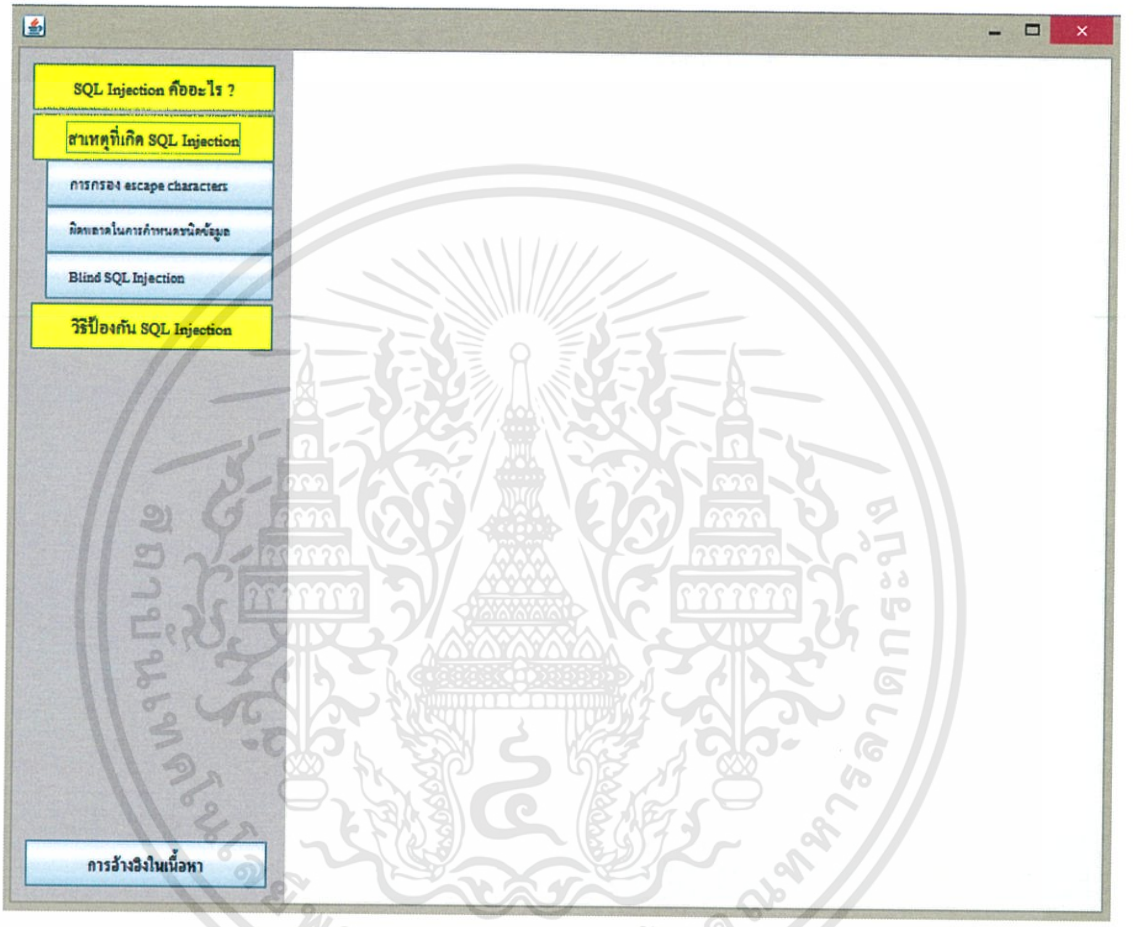
ส่วนนี้อธิบายว่า SQL Injection คืออะไรและมีวิธีการอย่างไร



รูปที่ 4.7 หน้าจอ Help: ความหมายของ SQL Injection

2) สาเหตุของการเกิด SQL Injection

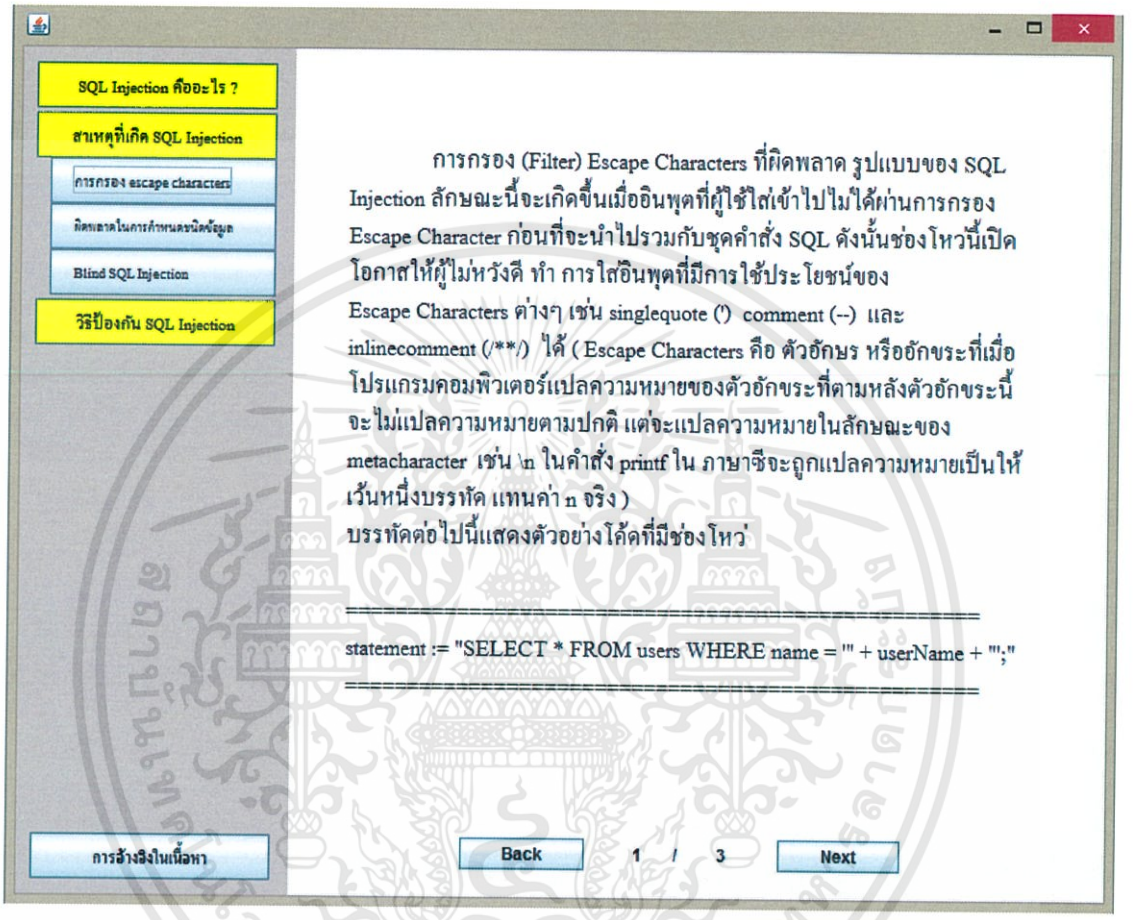
ส่วนอธิบายสาเหตุของการเกิด SQL Injection ซึ่งแบ่งออกเป็น 3 ประเภทคือ การกรอง (Filter) Escape Characters ที่ผิดพลาด ความผิดพลาดในการควบคุมชนิดข้อมูล และช่องโหว่ในฐานข้อมูล



รูปที่ 4.8 หน้าจอ Help: สาเหตุที่เกิด SQL Injection

2.1) การกรอง escape characters ที่ผิดพลาด

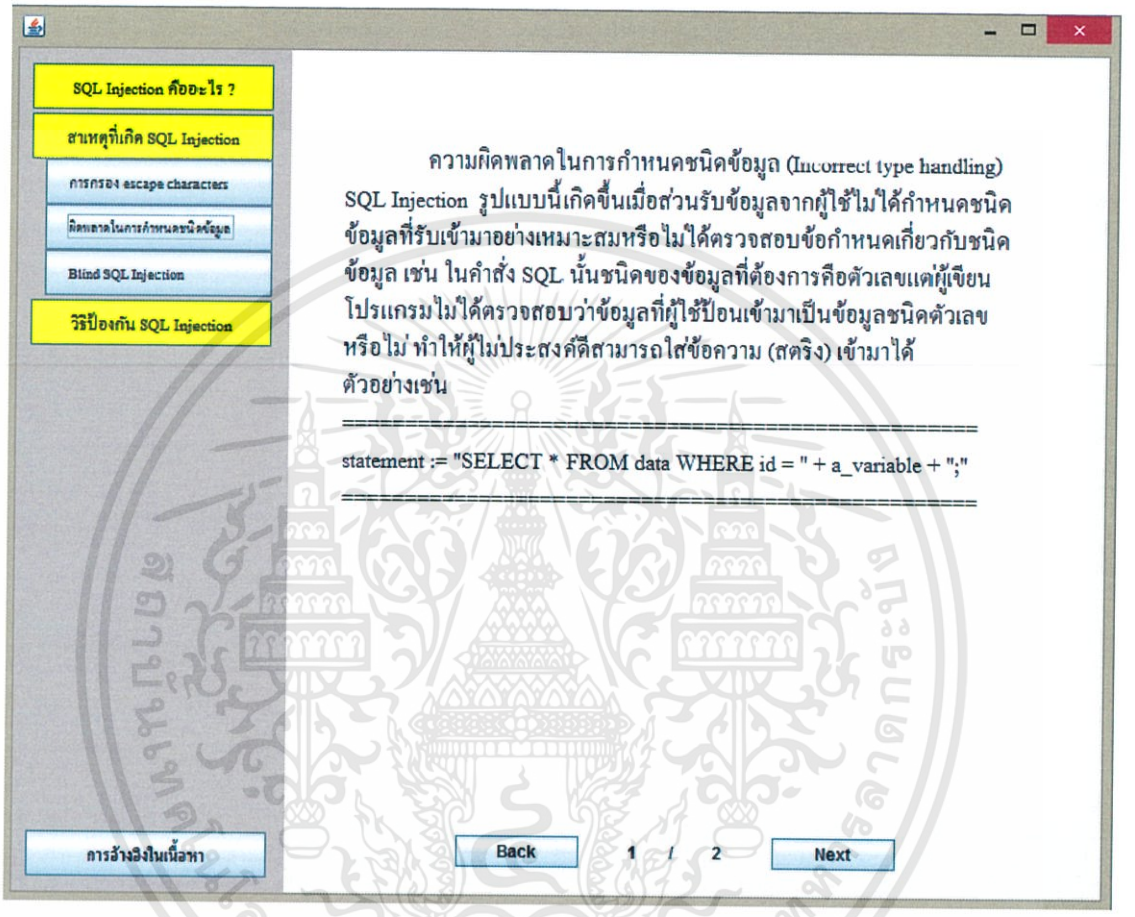
อธิบายเกี่ยวกับการกรอง (Filter) Escape Characters ที่ผิดพลาดซึ่งทำให้เกิดช่องโหว่ที่สามารถโจมตีโดยใช้ SQL Injection ได้



รูปที่ 4.9 หน้าจอ Help: การกรอง Escape Characters ที่ผิดพลาด

2.2) การกำหนดชนิดข้อมูลผิดพลาด

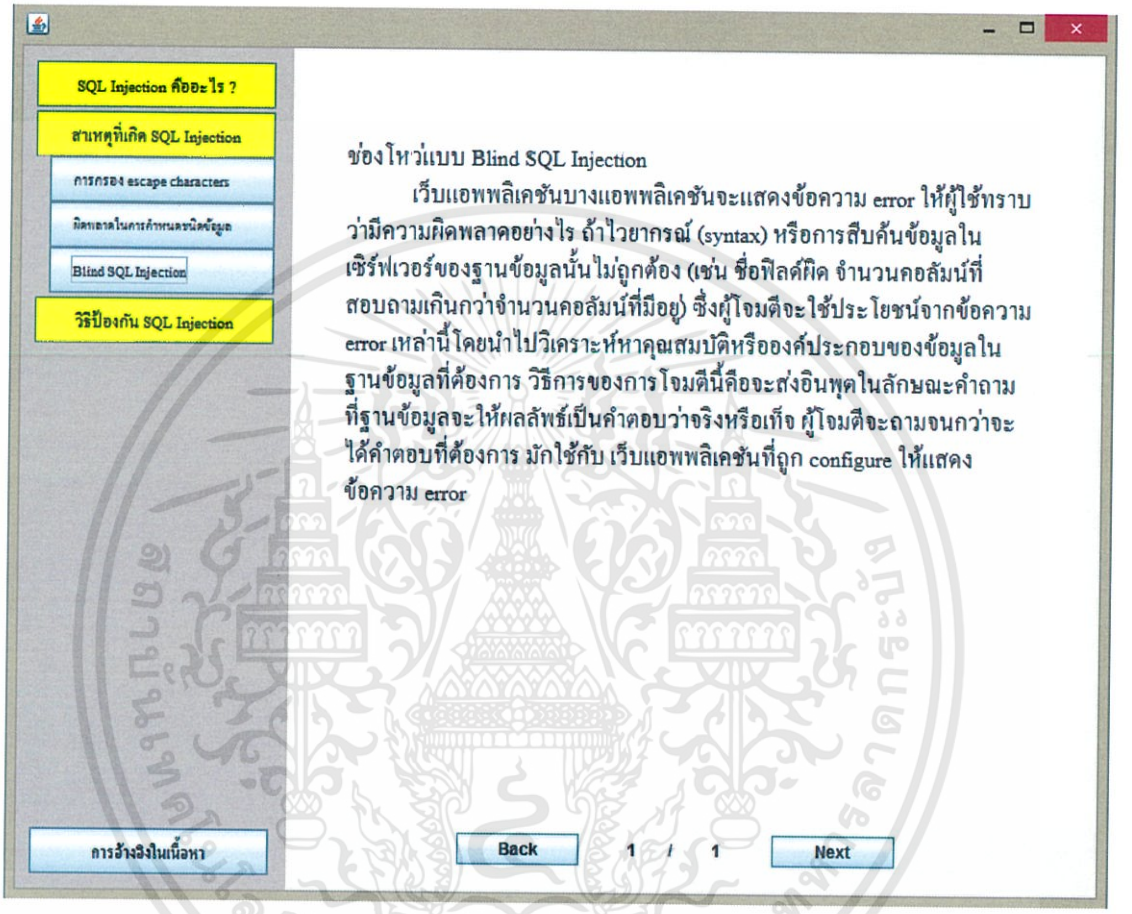
อธิบายเกี่ยวกับความผิดพลาดในการกำหนดชนิดข้อมูลซึ่งทำให้เกิดช่องโหว่ที่สามารถโจมตีโดยใช้ SQL Injection ได้



รูปที่ 4.10 หน้าจอ Help: การกำหนดชนิดข้อมูลผิดพลาด

2.3) ช่องโหว่ในฐานข้อมูล

อธิบายเกี่ยวกับช่องโหว่ในฐานข้อมูลซึ่งทำให้เกิดช่องโหว่ที่สามารถโจมตีโดยใช้ SQL Injection ได้



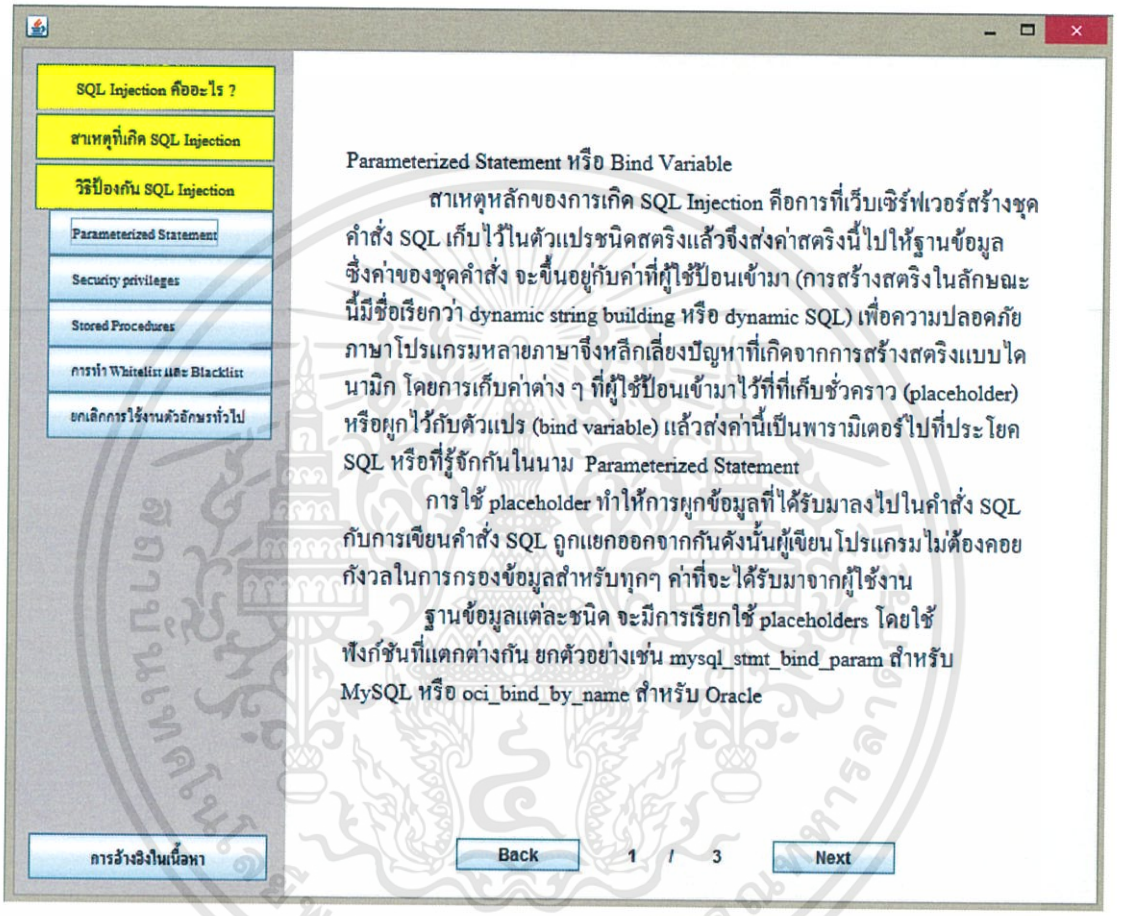
รูปที่ 4.11 หน้าจอ Help: Blind SQL Injection

3) วิธีการป้องกัน SQL Injection

ใน Help จะมีวิธีการป้องกัน SQL Injection ทั้งหมด 5 วิธีดังต่อไปนี้

3.1) วิธีการป้องกัน SQL Injection โดยใช้ Parameterized Statement

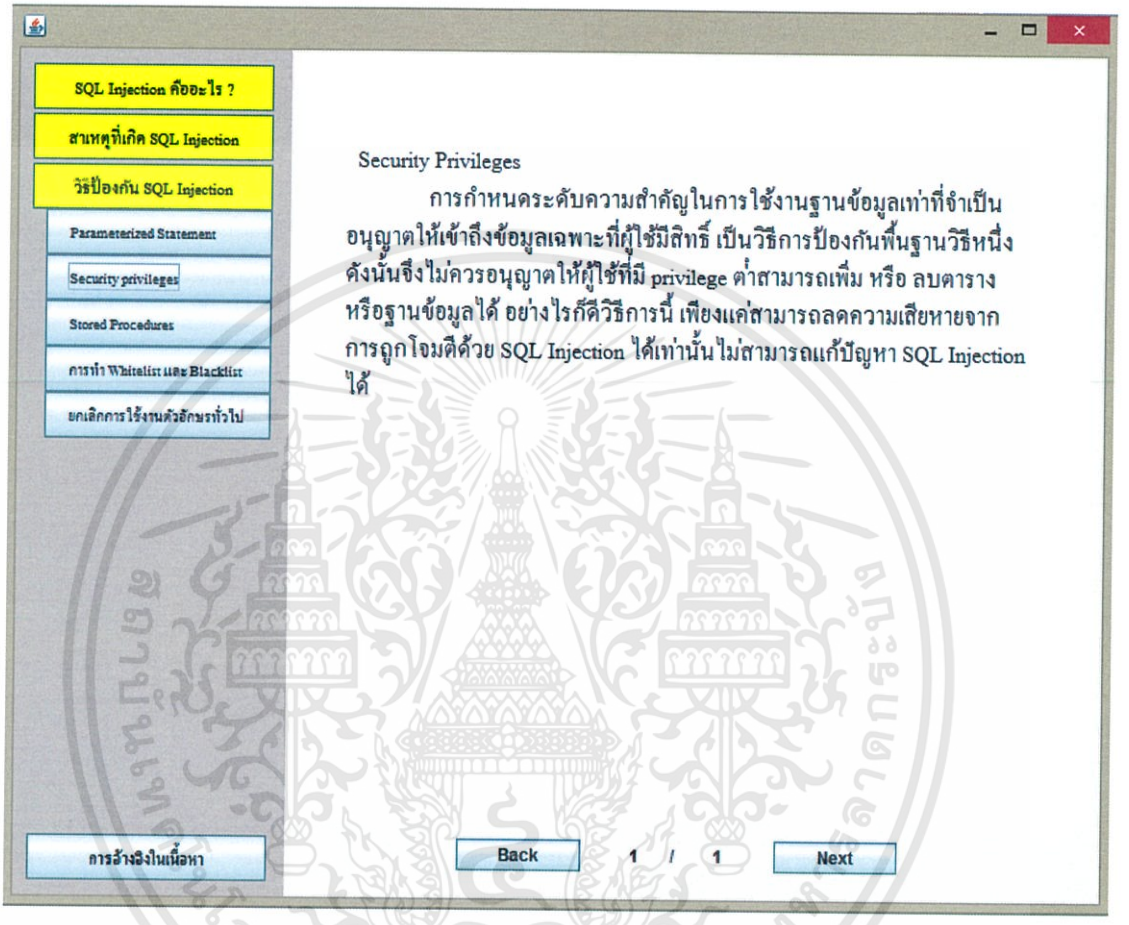
อธิบายวิธีการป้องกัน SQL Injection โดยใช้ Parameterized Statement



รูปที่ 4.12 หน้าจอ Help: การป้องกันโดยใช้ Parameterized Statement

3.2) วิธีการป้องกัน SQL Injection โดยการแก้ไขที่ฐานข้อมูล

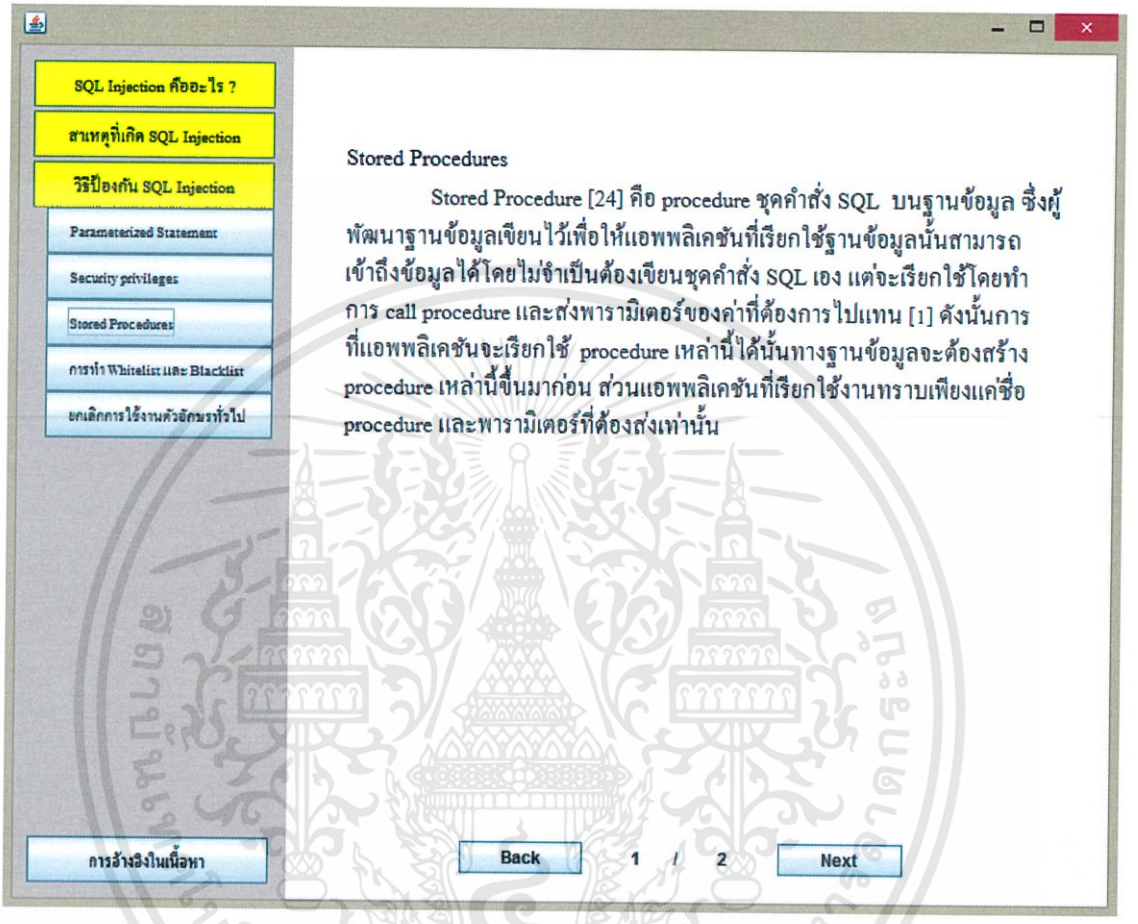
อธิบายวิธีการป้องกัน SQL Injection โดยการกำหนดระดับความสำคัญในการใช้งานฐานข้อมูลเท่าที่จำเป็น



รูปที่ 4.13 หน้าจอ Help: การป้องกันโดยใช้ Security Privileges

3.3) วิธีการป้องกัน SQL Injection โดยใช้ Stored Procedures

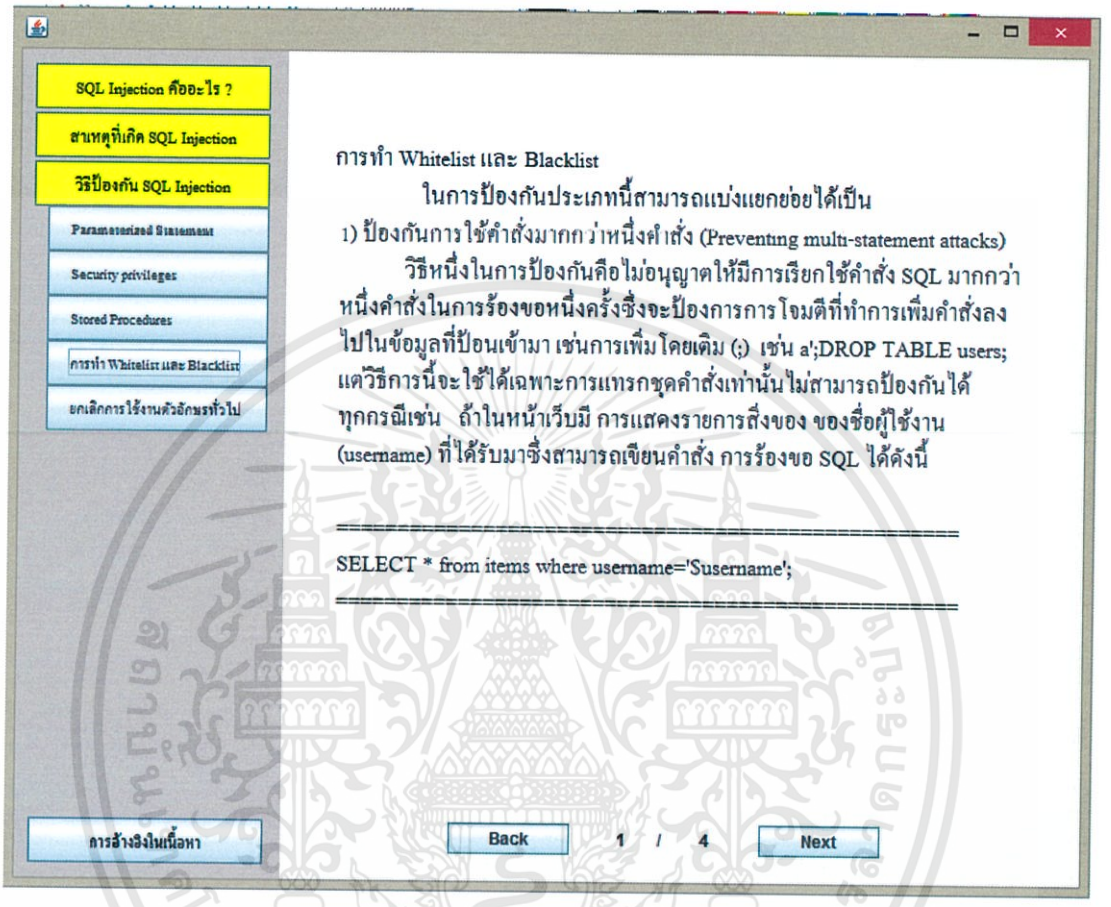
อธิบายวิธีการป้องกัน SQL Injection โดยใช้ Stored procedures



รูปที่ 4.14 หน้าจอ Help: การป้องกันโดยใช้ Stored Procedures

3.4) วิธีการป้องกัน SQL Injection โดยการทำ Whitelist และ Blacklist

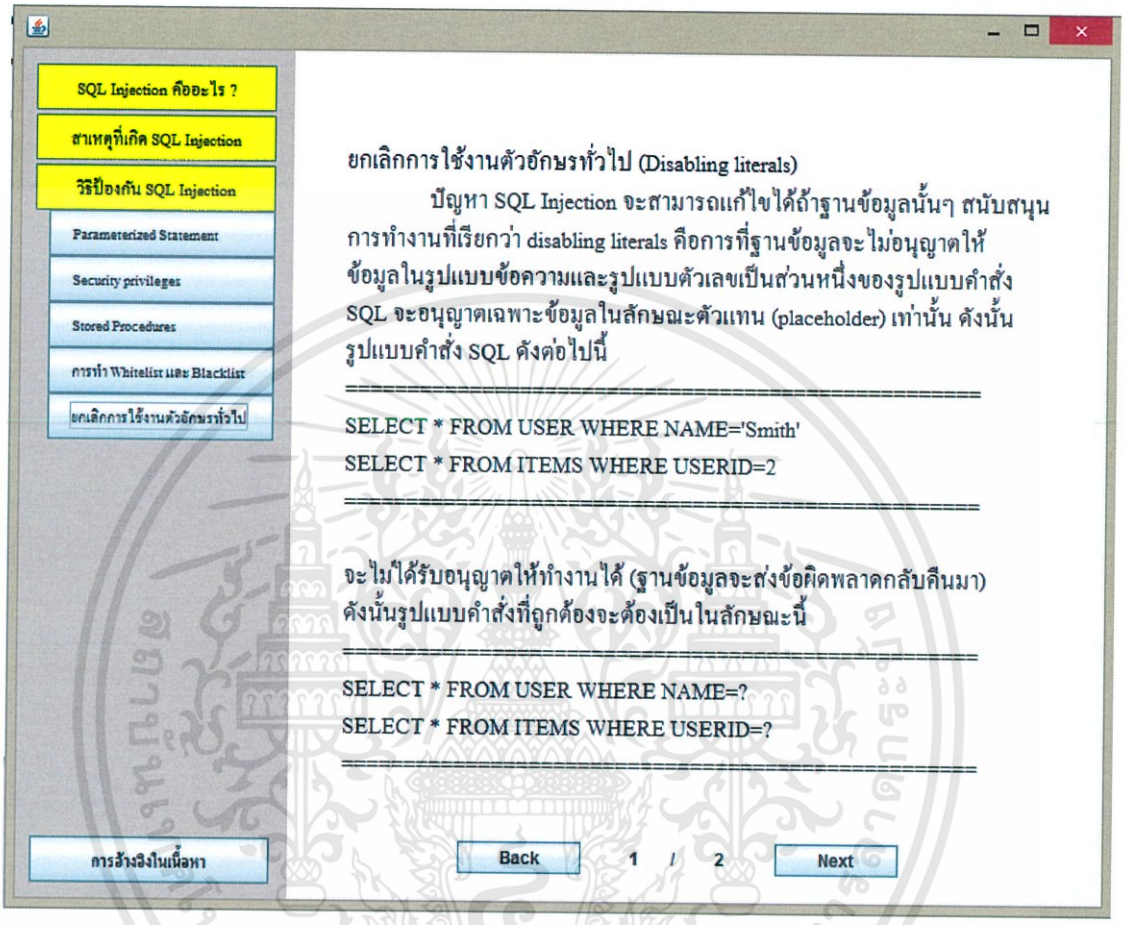
อธิบายวิธีการป้องกัน SQL Injection โดยป้องกันการทำ Whitelist และ Blacklist



รูปที่ 4.15 หน้าจอ Help: การป้องกันโดยการทำ Whitelist และ Blacklist

3.5) วิธีการป้องกัน SQL Injection โดยยกเลิกการใช้งานตัวอักษรทั่วไป

อธิบายวิธีการป้องกัน SQL Injection โดยยกเลิกการใช้งานตัวอักษรทั่วไป



รูปที่ 4.16 หน้าจอ Help: การป้องกันโดยยกเลิกการใช้งานตัวอักษรทั่วไป

4.3 ผลการทดสอบโปรแกรม

ผู้พัฒนาได้ทำการทดสอบโปรแกรมโดยทดสอบกับกลุ่มไฟล์ PHP ซึ่งมีการเชื่อมต่อกับฐานข้อมูลแบบ MySQL ซึ่งเป็นกลุ่มไฟล์เดียวกับที่ใช้ในการทดสอบกับ Pixy ในบทที่ 3 และไฟล์ที่เชื่อมต่อกับฐานข้อมูลแบบ PDO

4.3.1 โค้ดที่มีการเชื่อมต่อกับฐานข้อมูลแบบ MySQL

เนื่องจากโปรแกรมปลั๊กอินมีการนำผลลัพธ์ของ Pixy มาพัฒนาต่อ จึงทำการทดสอบความถูกต้องของโปรแกรมโดยนำไปทดสอบกับไฟล์กลุ่มเดียวกับที่ใช้ทดสอบ Pixy ในบทที่ 3 จำนวนทั้งหมด 20 ไฟล์ที่มีการเชื่อมต่อกับฐานข้อมูลแบบ MySQL ซึ่งผลที่ได้จากการทดสอบและการเปรียบเทียบกับ Pixy สรุปได้ดังตารางที่ 4.1

ตารางที่ 4.1 เปรียบเทียบผลการสแกนของโปรแกรมปลั๊กอินกับ Pixy

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	ผลลัพธ์เมื่อเทียบกับ Pixy
1	Ad Management\fad\ctchrefscr.php	$get_ref = @mysql_query("SELECT * FROM `referrals` WHERE `id`='$id'");	เหมือนกัน
		$update_ref = @mysql_query("UPDATE `referrals` SET `hits`='$hits' WHERE `id`='$id'");	เหมือนกัน
		$update_cookie = @mysql_query("UPDATE `ref_cookie`_list SET `ip`='$ip' WHERE `id`=$id'");	เหมือนกัน
		$insert_cookie = @mysql_query("INSERT INTO `ref_cookie_list` (`id`,`ip`) VALUES ('$id','$ip')");	เหมือนกัน

ตารางที่ 4.1 เปรียบเทียบผลการสแกนของโปรแกรมปลั๊กอินกับ Pixy (ต่อ)

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	ผลลัพธ์เมื่อเทียบกับ Pixy
2	Blog\aoblogger\ppppp.php	mysql_query("INSERT INTO blog SET title = '$title', message ='$message', time = '$time'") or die(mysql_error());	เหมือนกัน
3	Blog\blog\admin\addarticle.php	mysql_query($createblog)or die("Could not create blog");	เหมือนกัน
4	Blog\blog\admin\addchoice.php	mysql_query($makeanswer) or die(mysql_error());	เหมือนกัน
5	News Publishing\aWebNews\aWebNews\feedback.php	$result = mysql_query($query);	เหมือนกัน
6	News Publishing\aWebNews\aWebNews\changep.php	$result = mysql_query($query);	เหมือนกัน
		$result12 = mysql_query($query12);	เหมือนกัน
7	Blog\blog\admin\ban.php	mysql_query($insertip) or die("Could not insert ip");	เหมือนกัน
8	News Publishing\aWebNews\aWebNews\post.php	$result = mysql_query($query);	เหมือนกัน
		mysql_query($query);	เหมือนกัน
9	Advance\viewme.php	$banner = @mysql_query("SELECT * FROM banners WHERE `id`='$id'");	เหมือนกัน

ตารางที่ 4.1 เปรียบเทียบผลการสแกนของโปรแกรมปลั๊กอินกับ Pixy (ต่อ)

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	ผลลัพธ์เมื่อเทียบกับ Pixy
10	Advance\catchadd.php	$banner_stat = @mysql_fetch_array(@mysql_query("SELECT * FROM stats WHERE `id`='$id'"));	เหมือนกัน
		$update_banner = @mysql_query("UPDATE stats SET `hits`='$hits' , `uni_hits`='$uni_hits' WHERE id='$id'");	เหมือนกัน
11	Advance\editme.php	$get_banner = @mysql_query("SELECT * FROM banners WHERE `id`='$id'");	เหมือนกัน
		$update = @mysql_query("UPDATE banners SET `name`='$name' , `mouseover`='$mouse_over', `location`='$location',`urlto`='$urlto',`stopit`='$stopit',`java_status`='$java_statusbar',`openin`='$openin' WHERE `id`='$id'");	เหมือนกัน
		$update = @mysql_query("UPDATE banners SET `name`='$name' , `mouseover`='$mouse_over', `urlto`='$urlto',`stopit`='$stopit',`java_status`='$java_statusbar',`openin`='$openin' WHERE `id`='$id'");	เหมือนกัน
		$resultf = @mysql_query($queryf);	เหมือนกัน

ตารางที่ 4.1 เปรียบเทียบผลการสแกนของโปรแกรมปลั๊กอินกับ Pixy (ต่อ)

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	ผลลัพธ์เมื่อเทียบกับ Pixy
11	Advance\editme.php	$update = @mysql_query("UPDATE stats SET `made`='$made',`file_location`='$location',`width`='$width',`length`='$height',`hits`='$hits',`uni_hits`='$hits',`views`='$views',`uni_views`='$uni_views' WHERE `id`='$id'");	เหมือนกัน
		$update = @mysql_query("UPDATE stats SET `made`='$made',`width`='$width',`length`='$height',`hits`='$hits',`uni_hits`='$hits',`views`='$views',`uni_views`='$uni_views' WHERE `id`='$id'");	เหมือนกัน
		$stat = @mysql_fetch_array(@mysql_query("SELECT * FROM stats WHERE id='$id'"));	เหมือนกัน
12	Advance\remove.php	$banner_info = @mysql_fetch_array(@mysql_query("SELECT * FROM `banners` WHERE `id`='$delete'"));	เหมือนกัน
		$remove_banner = @mysql_query("DELETE FROM `banners` WHERE `id`='$delete'");	เหมือนกัน
		$remove_stat = @mysql_query("DELETE FROM `stats` WHERE `id`='$delete'");	เหมือนกัน

ตารางที่ 4.1 เปรียบเทียบผลการสแกนของโปรแกรมปลั๊กอินกับ Pixy (ต่อ)

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	ผลลัพธ์เมื่อเทียบกับ Pixy
13	Ad Management\banner_ad\banner_ad\ran.php	MySQL_Query($Query);	เหมือนกัน
14	Ad Management\banner_ad\banner_ad\ban_install.php	$result = mysql_query($query);	เหมือนกัน
		$result = mysql_query($query);	เหมือนกัน
15	Ad Management\azbanner\admin\stats.php	if($action==loeschen){$loesche=mysql_query("DELETE FROM $ad_table WHERE ID=$ID");}	เหมือนกัน
		$stats=mysql_query("UPDATE $ad_table SET IMPRESSIONS=0 WHERE ID=$ID");	เหมือนกัน
		$stats=mysql_query("UPDATE $ad_table SET KLICKS=0 WHERE ID=$ID");	เหมือนกัน
		if($action==partnernew){$new=mysql_query("INSERT INTO $ad_table (ID, ART, PARTNER, CODE, IMPRESSIONS) VALUES (",1,'$BEZEICHNUNG','$CODE',0)");}	เหมือนกัน
		if($action==bannernew){$new=mysql_query("INSERT INTO $ad_table (ID, ART, BILD, URL, IMPRESSIONS, KLICKS) VALUES (",0,'$BANNER','$URL',0,0)");}	เหมือนกัน

ตารางที่ 4.1 เปรียบเทียบผลการสแกนของโปรแกรมปลั๊กอินกับ Pixy (ต่อ)

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	ผลลัพธ์เมื่อเทียบกับ Pixy
15	Ad Management\azbanner\admin\stats.php	if($bannermodify==true){$aendern=mysql_query("UPDATE $ad_table Set URL = '$URL' WHERE ID = '$ID'");	เหมือนกัน
		$aendern=mysql_query("UPDATE $ad_table Set BILD = '$BILD' WHERE ID = '$ID'");	เหมือนกัน
		if($partnermodify==true){$aendern=mysql_query("UPDATE $ad_table Set PARTNER = '$PARTNER' WHERE ID = '$ID'");	เหมือนกัน
		$aendern=mysql_query("UPDATE $ad_table Set CODE = '$CODE' WHERE ID = '$ID'");	เหมือนกัน
		$abfrage=mysql_query("SELECT * FROM $ad_table WHERE ART=0 ORDER BY ID");	เหมือนกัน
		$abfrage=mysql_query("SELECT * FROM $ad_table WHERE ART=1 ORDER BY ID");	เหมือนกัน
16	Affiliate Programs\overstock_dfs\feeds \upload.php	$queryp2=mysql_query("LOAD DATA LOCAL INFILE '$file' REPLACE INTO TABLE $table FIELDS TERMINATED BY ',' ENCLOSED BY '\"' LINES TERMINATED BY '\n' IGNORE 1 LINES");	เหมือนกัน

ตารางที่ 4.1 เปรียบเทียบผลการสแกนของโปรแกรมปลั๊กอินกับ Pixy (ต่อ)

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	ผลลัพธ์เมื่อเทียบกับ Pixy
16	Affiliate Programs\overstock_dfs\feeds \upload.php	$mysqlresult = mysql_query("update $table set rndnumber=1000000*rand() ");	เหมือนกัน
17	Affiliate Programs\overstock_dfs \category.php	$querypcatname=mysql_query("Select * from categories where id='$_GET[cat]' ");	เหมือนกัน
		$queryp1=mysql_query("Select * from $table where category='$catslash' ");	เหมือนกัน
		$queryp=mysql_query("Select * from $table where category='$catslash' limit $start,$num_per_page");	เหมือนกัน
18	Affiliate Programs\overstock_dfs \dbsetup.php	if (mysql_query($sql)) {	เหมือนกัน
19	Click Tracking\counter\counter\counter.php	mysql_query("UPDATE sites SET clicks='$row[1]', users='$udusers' WHERE id='$row[0]'", $db)ordie(mysql_error());	เหมือนกัน
		$result = mysql_query($sql) or die(mysql_error());	เหมือนกัน
		mysql_query("DELETE FROM sites WHERE id=$id");	เหมือนกัน
		$result = mysql_query("SELECT id, clicks, url, users FROM sites WHERE id=$id", $db) or die();	เหมือนกัน

ตารางที่ 4.1 เปรียบเทียบผลการสแกนของโปรแกรมปลั๊กอินกับ Pixy (ต่อ)

ไฟล์ที่	ชื่อไฟล์	คำสั่งที่มีช่องโหว่	ผลลัพธ์เมื่อเทียบกับ Pixy
20	Classified Ads \PHPRentals\phprentals\admin \passwordchange.php	mysql_query("UPDATE users SET pword = md5('$newpass') WHERE email = '$user'");	เหมือนกัน
		mysql_query("UPDATE users SET pword = md5('$newpass') WHERE email = '$user'");	เหมือนกัน

4.3.2 โค้ดที่ทำการเชื่อมต่อกับฐานข้อมูลแบบ PDO

สำหรับผลการทดสอบกับไฟล์ PDO นั้น เป็นดังตารางที่ 4.2

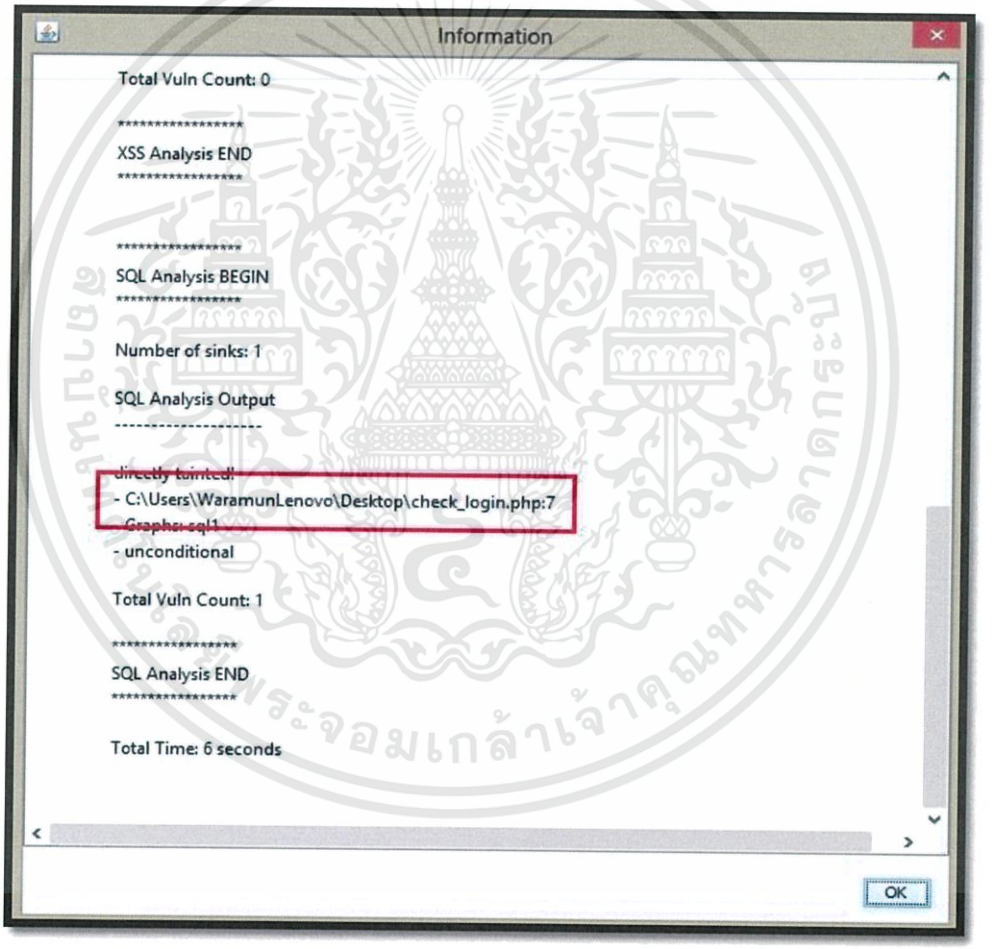
ตารางที่ 4.2 เปรียบเทียบผลการสแกนไฟล์ PDO ของโปรแกรมปลั๊กอินกับ Pixy

ไฟล์ที่	ชื่อไฟล์ PDO	ผลลัพธ์ของโปรแกรมปลั๊กอิน	ผลลัพธ์ของ Pixy
1	forum/index.php	$stmt = $db->query ("SELECT * FROM table WHERE id=:id AND name=:name");	ไม่พบช่องโหว่
2	platform/database.php	$result= $conn->query ("SELECT * FROM idname WHERE `id`='$id'");	ไม่พบช่องโหว่
3	static/storage-page-layout.php	$result = $db_conn->query($query);	ไม่พบช่องโหว่

ผลการทดสอบและเปรียบเทียบในตารางที่ 4.1 สรุปได้ว่าในการเชื่อมต่อกับฐานข้อมูล MySQL โปรแกรมปลั๊กอินและ Pixy ให้ผลลัพธ์ของจำนวนโค้ดและโค้ดที่เป็นช่องโหว่ที่เหมือนกัน ส่วนผลการทดสอบในตารางที่ 4.2 สรุปได้ว่า โปรแกรมปลั๊กอินสามารถสแกนหาช่องโหว่ของไฟล์ที่เขียนเชื่อมต่อแบบ PDO ได้ ในขณะที่ Pixy ไม่สามารถตรวจสอบโค้ดประเภทนี้ได้

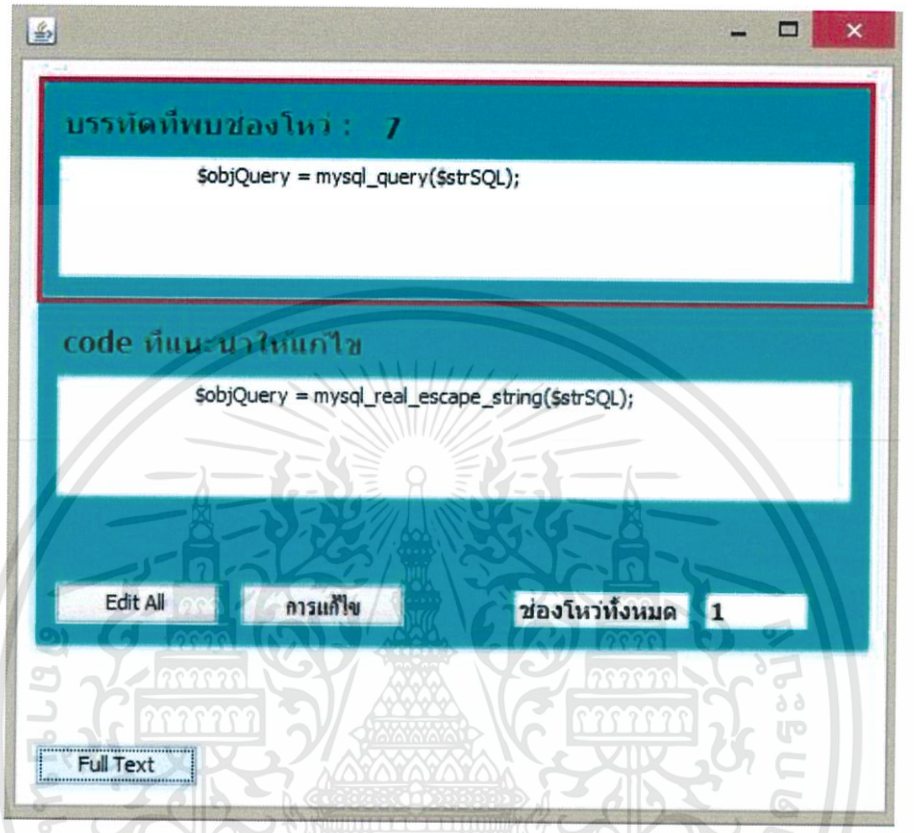
4.4 ความแตกต่างระหว่าง Pixy และโปรแกรมของผู้พัฒนา

1) Pixy แสดงผลการสแกนออกมาทุกช่องโหว่โดยจะมีการบอกบรรทัดของโค้ดที่เป็นช่องโหว่ ดังตัวอย่างในรูปที่ 4.17 มี 1 ช่องโหว่ที่บรรทัดที่ 7



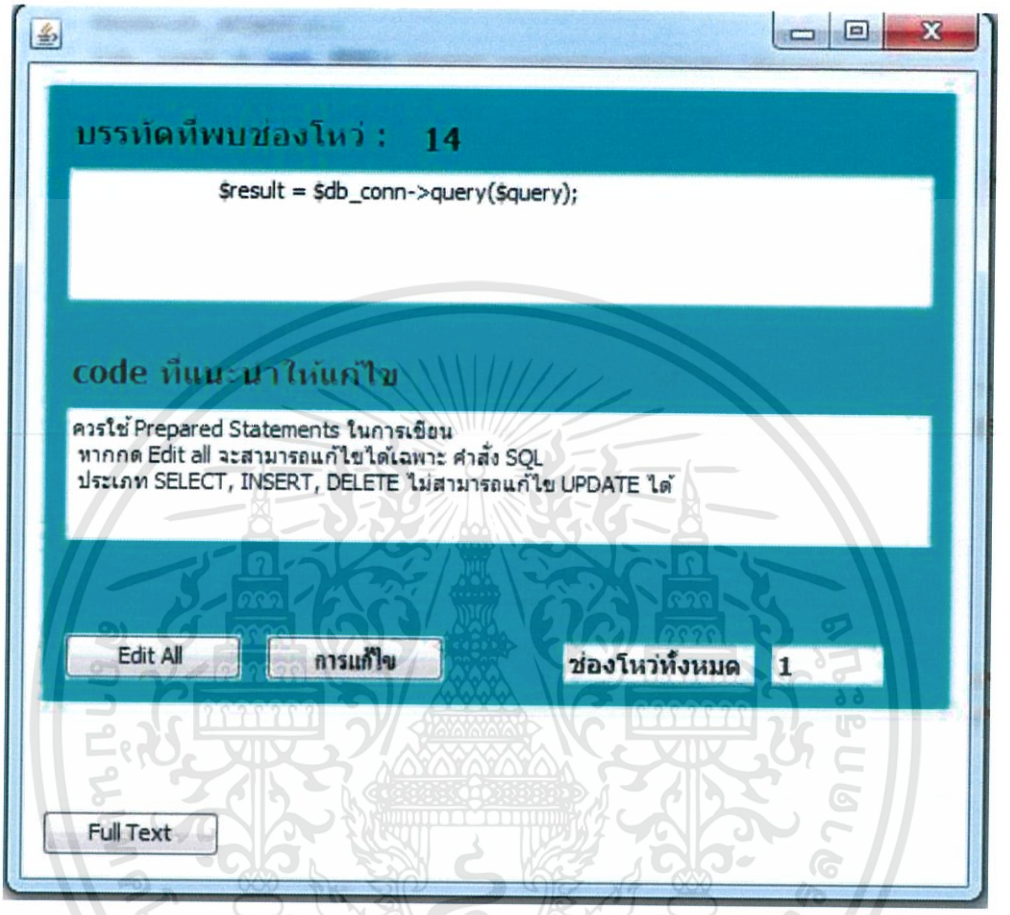
รูปที่ 4.17 หน้าจอแสดงผลการสแกนไฟล์ของ Pixy

2) โปรแกรมปลั๊กอินจะแสดงผลการสแกนทีละช่องโหว่โดยแจ้งบรรทัดที่มีช่องโหว่ พร้อมทั้งแนะนำวิธีการแก้ไขช่องโหว่นั้น และ สามารถแก้ไขช่องโหว่ทั้งหมดให้โดยอัตโนมัติได้



รูปที่ 4.18 หน้าจอแสดงผลการสแกนด้วยโปรแกรมปลั๊กอิน

3)โปรแกรมปลั๊กอินสามารถสแกนไฟล์ที่เชื่อมต่อกับฐานข้อมูลแบบ PDO ได้ แต่ Pixy ไม่สามารถทำได้



รูปที่ 4.19 หน้าจอแสดงผลการสแกนไฟล์ PDO ด้วยโปรแกรมปลั๊กอิน

4.5 ความสามารถในการป้องกันการโจมตีแบบ SQL Injection

ตารางที่ 4.3 แสดงความสามารถของโปรแกรมปลั๊กอิน ในการป้องกัน SQL Injection ประเภทต่างๆ

ตารางที่ 4.3 สรุปความสามารถของโปรแกรมปลั๊กอิน

ประเภทของ SQL Injection	ตรวจสอบได้
1. Basic SQL Injection (Escape Characters)	✓
2. Inline Comment	✓
3. Line Comment	✓
4. Union SQL Injections	✗
5. Blind SQL Injection	เฉพาะกรณีที่มีการใช้ Escape Characters Comment และ Inline Comment

บทที่ 5
บทสรุปและข้อเสนอแนะ

5.1 บทสรุปผลการดำเนินงาน

โครงงานพิเศษนี้ได้ทำการพัฒนาโปรแกรมปลั๊กอินบน NetBeans และ Eclipse เพื่อให้ผู้พัฒนาเว็บแอพพลิเคชันที่ใช้ภาษา PHP นำไปใช้ในการตรวจสอบและแก้ไขโค้ดที่เป็นอันตราย ต่อการโจมตีประเภท SQL Injection โดยในส่วนของการตรวจสอบได้นำโปรแกรม Pixy ซึ่งเป็นเครื่องมือในการวิเคราะห์โค้ดอันตรายแบบ static (static code anaylis tool) มาใช้เป็นแกนหลัก พร้อมทั้งทำการเพิ่มคุณสมบัติให้สามารถตรวจสอบโค้ดที่มีการเขียนเชื่อมต่อกับฐานข้อมูลแบบ PDO (PHP Data Object) ได้ สำหรับในส่วนของการแสดงผลนั้นนอกจากโปรแกรมจะทำการแจ้งเตือนโค้ดที่มีช่องโหว่ให้แล้ว โปรแกรมจะแนะนำวิธีการแก้ไขโค้ดนั้นหรือผู้ใช้สามารถเลือกให้โปรแกรมแก้ไขให้ได้ด้วย โดยความสามารถของโปรแกรมปลั๊กอินมีดังต่อนี้

- สามารถตรวจสอบโค้ด PHP เพื่อหาช่องโหว่ที่เป็นอันตรายต่อการโจมตีแบบ SQL Injection ในลักษณะการเขียนเชื่อมต่อกับฐานข้อมูลประเภท MySQL หรือการเขียนเชื่อมต่อกับฐานข้อมูลแบบ PDO (PHP Data Object)
- สามารถแก้ไขโค้ดที่มีช่องโหว่ที่เป็นอันตรายต่อการโจมตีแบบ SQL Injection ทั้งแบบ MySQL และ PDO
- มีส่วนช่วยเหลือผู้ใช้ (Help) เพื่อให้คำแนะนำและความรู้เพิ่มเติมในส่วนของการอธิบาย SQL Injection ประเภท และวิธีป้องกัน SQL Injection

5.2 ข้อจำกัดในการพัฒนา

- ในช่วงแรกของการพัฒนาโปรแกรม Pixy ไม่สามารถตรวจสอบโค้ดภาษา PHP ที่มีเวอร์ชันมากกว่า PHP 4 ได้
- ในการทดสอบโปรแกรม Pixy ต้องทำการเรียกใช้งานผ่าน cmd ซึ่งมีการเรียกใช้งานที่ค่อนข้างยุ่งยาก
- ในส่วนของการเขียนโค้ด PHP เชื่อมต่อกับฐานข้อมูล Pixy สามารถสแกนหาช่องโหว่ได้เฉพาะวิธีเชื่อมต่อแบบ MySQL และ PDO เท่านั้น

- ในขั้นตอนการแก้ไขโค้ด โปรแกรมไม่สามารถแก้ไขคำสั่ง SQL ประเภท UPDATE ให้เป็น Prepare Statements ได้
- โปรแกรมไม่สามารถแก้ไขการโจมตีของ SQL Injection แบบ Error Handing และ Union ได้
- ไม่สามารถพัฒนาเมนูสร้างโค้ดที่ปลอดภัยได้ เนื่องจากการเขียนโค้ด PHP ให้ปลอดภัยสามารถเขียนได้หลากหลาย ผู้จัดทำจึงไม่สามารถสร้างโค้ดที่เหมาะสมกับระบบต่างๆ ได้
- ไม่สามารถตั้งค่าการแสดงผลของตัวอักษรได้ เนื่องจากการแสดงผลของตัวอักษร จำเป็นต้องอาศัยความเข้าใจ API ในการแสดงผลของ NetBeans และ Eclipse ซึ่งการศึกษาต้องใช้เวลาและ เกินกว่าขอบเขตงานที่วางแผนไว้
- ไม่สามารถแจ้งเตือน และ เปิด/ปิด การแจ้งเตือนได้ เนื่องจากการสแกนจำเป็นต้องใช้ไฟล์ที่เขียนโค้ดเสร็จแล้วทำให้ผู้จัดทำไม่สามารถทำการแจ้งเตือนได้

5.3 ข้อเสนอแนะในการพัฒนาโปรแกรม

- ควรพัฒนาปลั๊กอินให้สามารถแก้ไข SQL Injection แบบ Error Handing และ Union
- ควรมีความรู้ความเข้าใจหรือศึกษา Object-oriented Programming เพื่อพัฒนาโปรแกรมได้อย่างรวดเร็ว
- ควรทำการศึกษา API ของ NetBeans และ Eclipse เพื่อเรียกใช้ฟังก์ชันการแสดงผลได้หลากหลาย
- เพิ่มฟังก์ชันการเพิ่มตัวอย่างโค้ดของ PHP ที่ปลอดภัยให้กับผู้ใช้
- ปรับปรุงรูปแบบการนำเสนอ Help ให้มีรูปแบบที่น่าสนใจกว่าเดิม เช่นทำเป็น Flash
- สามารถนำโปรแกรมไปประยุกต์เพื่อใช้ในการตรวจสอบช่องโหว่แบบ Cross Site Scripting (XSS)
- สามารถนำโปรแกรมไปพัฒนาบน IDE ตัวอื่นๆนอกจาก NetBeans และ Eclipse ได้
- สามารถศึกษาขั้นตอน กระบวนการตรวจสอบโค้ดของ Pixy เพื่อนำไปประยุกต์ใช้ตรวจสอบภาษาอื่นๆนอกจาก PHP ได้

เอกสารอ้างอิง

[1] วิกิพีเดีย สารานุกรมเสรี (2556). เว็บแอพพลิเคชัน. สืบค้นเมื่อ 1 กันยายน 2556, จาก http://th.wikipedia.org/wiki/โปรแกรมประยุกต์บนเว็บ

[2] Justin Clarke. 2556. SQL Injection Attacks and Defense. 2nd Edition. Justin Clarke. Waltham: Syngress.

[3] Framework Usage Statistics (2556). ภาษาที่นิยมใช้เขียนเว็บแอพพลิเคชัน. สืบค้นเมื่อ 3 สิงหาคม 2556, จาก http://trends.builtwith.com/framework

[4] วิกิพีเดีย สารานุกรมเสรี (2556). ภาษาพีเอชพี. สืบค้นเมื่อ 5 กันยายน 2556, จาก http://th.wikipedia.org/wiki/ภาษาพีเอชพี

[5] The Apache Software Foundation. Apache Web Server. สืบค้นเมื่อ 5 กันยายน 2556, จาก http://apache.org

[6] วิกิพีเดีย สารานุกรมเสรี (2556). Personal Web Server (PWP). สืบค้นเมื่อ 6 กันยายน 2556, จาก http://en.wikipedia.org/wiki/Microsoft_Personal_Web_Server

[7] วิกิพีเดีย สารานุกรมเสรี (2556). Simple API for XML (SAX). สืบค้นเมื่อ 6 กันยายน 2556, จาก http://en.wikipedia.org/wiki/Simple_API_for_XML

[8] วิกิพีเดีย สารานุกรมเสรี (2556). Document Object Model. สืบค้นเมื่อ 6 กันยายน 2556, จาก http://en.wikipedia.org/wiki/Document_Object_Model

[9] วิกิพีเดีย สารานุกรมเสรี (2556). XSLT. สืบค้นเมื่อ 6 กันยายน 2556, จาก http://en.wikipedia.org/wiki/XSLT

[10] Microsoft. Microsoft Internet Information Services (IIS) (2556). สืบค้นเมื่อ 6 กันยายน 2556, จาก http://www.microsoft.com/web/platform/server.aspx

[11] วิกิพีเดีย สารานุกรมเสรี (2556). Personal Web Server. สืบค้นเมื่อ 6 กันยายน 2556, จาก http://en.wikipedia.org/wiki/Personal_web_server

[12] วิกิพีเดีย สารานุกรมเสรี (2556). Netscape Web Server. สืบค้นเมื่อ 6 กันยายน 2556, จาก http://en.wikipedia.org/wiki/Netscape

[13] Oracle Team. iPlanet. สืบค้นเมื่อ 6 กันยายน 2556, จาก http://www.oracle.com/technetwork/middleware/webtier/overview/index.html#iWS

[14] Oreilly. Oreilly Website Pro server. สืบค้นเมื่อ 6 กันยายน 2556, จาก http://oreilly.com/software/

[15] Caudium. Caudium Server. สืบค้นเมื่อ 6 กันยายน 2556, จาก http://www.caudium.net/space/start

[16] The Xitami developer's group (2552). Xitami. สืบค้นเมื่อ 6 กันยายน 2556, จาก http://www.xitami.com/

[17] The PHP Group. OmniHTTPd. สืบค้นเมื่อ 6 กันยายน 2556, จาก http://www.php.net/manual/sr/install.windows.omnihttpd.php

[18] Oracle. Oracle Database. สืบค้นเมื่อ 6 กันยายน 2556, จาก http://www.oracle.com/us/products/database/overview/index.html

[19] dBase Team. dBASE PLUS 8.1. สืบค้นเมื่อ 6 กันยายน 2556, จาก www.dbase.com

[20] PostgreSQL Team (2556). PostgreSQL. สืบค้นเมื่อ 6 กันยายน 2556, จาก www.postgresql.org

[21] IBM. IBM DB2. สืบค้นเมื่อ 6 กันยายน 2556, จาก www.ibm.com/software/data/db2/

[22] Oracle. MySQL. สืบค้นเมื่อ 6 กันยายน 2556, จาก www.mysql.com

[23] IBM. Informix. สืบค้นเมื่อ 6 กันยายน 2556, จาก www.ibm.com/software/data/informix/

[24] Margaret Rouse (2548). Stored Procedure. สืบค้นเมื่อ 9 สิงหาคม 2556, จาก http://searchoracle.techtarget.com/definition/stored-procedure

[25] วิกิพีเดีย สารานุกรมเสรี (2556). WDDX (Web Distributed Data eXchange). สืบค้นเมื่อ 6 กันยายน 2556, จาก http://en.wikipedia.org/wiki/WDDX

[26] GNU (2555). GPL (GNU General Public License). สืบค้นเมื่อ 6 กันยายน 2556, จาก http://www.gnu.org/copyleft/gpl.html

[27] GNU (2555). The GNU Operating System. สืบค้นเมื่อ 6 กันยายน 2556, จาก http://www.gnu.org

[28] OWASP (2556). Top 10 2013-Risk. สืบค้นเมื่อ 5 กันยายน 2556, จาก https://www.owasp.org/index.php/Top_10_2013-Risk

[29] OWASP (2556). Input Validation Cheat Sheet. สืบค้นเมื่อ 5 กันยายน 2556, จาก https://www.owasp.org/index.php/Input_Validation_Cheat_Sheet

[30] StackOverFlow (2552). HTTP Session Tracking. สืบค้นเมื่อ 10 สิงหาคม 2556, จาก http://stackoverflow.com/questions/1740860/http-session-tracking

[31] วิกิพีเดีย สารานุกรมเสรี (2556). HTTP Cookie. สืบค้นเมื่อ 10 สิงหาคม 2556, จาก http://en.wikipedia.org/wiki/HTTP_cookie

[32] DaBoss (2556). Integrity Checking. สืบค้นเมื่อ 10 สิงหาคม 2556, จาก http://www.cknow.com/cms/vtutor/integrity-checking.html

[33] OWASP. The Open Web Application Security Project (OWASP). สืบค้นเมื่อ 10 สิงหาคม 2556, จาก https://www.owasp.org

[34] OWASP (2556). 2013 Top 10 List. สืบค้นเมื่อ 10 สิงหาคม 2556, จาก https://www.owasp.org/index.php/Top_10_2013-Top_10

[35] OWASP (2552). LDAP injection. สืบค้นเมื่อ 10 สิงหาคม 2556, จาก https://www.owasp.org/index.php/LDAP_injection

[36] วิกิพีเดีย สารานุกรมเสรี (2556). Fat client. สืบค้นเมื่อ 10 สิงหาคม 2556, จาก http://en.wikipedia.org/wiki/Fat_client

[37] PHP.net. The PHP Data Objects (PDO). สืบค้นเมื่อ 10 สิงหาคม 2556, จาก https://php.net/manual/en/intro.pdo.php

[38] hashPHP. PDO. สืบค้นเมื่อ 11 สิงหาคม 2556, จาก http://wiki.hashphp.org/PDO_Tutorial_for_MySQL_Developers

[39] NetBeans Team. NetBeans. สืบค้นเมื่อ 11 สิงหาคม 2556, จาก https://netbeans.org/

[40] hashPHP. Prepared Statements. สืบค้นเมื่อ 5 กันยายน 2556, จาก http://wiki.hashphp.org/PDO_Tutorial_for_MySQL_Developers

[41] Margaret Rouse (2548). Stored Procedure. สืบค้นเมื่อ 11 สิงหาคม 2556, จาก http://searchoracle.techtarget.com/definition/stored-procedure

[42] JUST ANOTHER HACKER. GRAUDIT. สืบค้นเมื่อ 11 สิงหาคม 2556, จาก www.justanotherhacker.com/projects/graudit.html

[43] Michael V. Scovetta. Yasca. สืบค้นเมื่อ 11 สิงหาคม 2556, จาก http://www.scovetta.com/yasca.html

[44] Oliverklee (2556). Pixy. สืบค้นเมื่อ 11 สิงหาคม 2556, จาก https://github.com/oliverklee/pixy

[45] Blueinfy Team. Blueinfy. สืบค้นเมื่อ 11 สิงหาคม 2556, จาก www.blueinfy.com

[46] OWASP. Lapse+. สืบค้นเมื่อ 15 สิงหาคม 2556, จาก https://www.owasp.org/index.php/OWASP_LAPSE_Project

[47] Microsoft. Microsoft Source Code Analyzer for SQL Injection. สืบค้นเมื่อ 15 สิงหาคม 2556, จาก http://support.microsoft.com/kb/954476

[48] Microsoft. CAT.NET. สืบค้นเมื่อ 15 สิงหาคม 2556, จาก ww.microsoft.com/download/en/details.aspx?id=19968

[49] Johannes.dahse. RIPS. สืบค้นเมื่อ 16 สิงหาคม 2556, จาก http://rips-scanner.sourceforge.net/

[50] Google Developers. CodePro Analytix. สืบค้นเมื่อ 16 สิงหาคม 2556, จาก https://developers.google.com/java-dev-tools/codepro/doc/

[51] Google Project Hosting. TeSA. สืบค้นเมื่อ 16 สิงหาคม 2556, จาก https://code.google.com/p/teachablesa/

[52] HP. Fortify Source Code Analyzer. สืบค้นเมื่อ 19 สิงหาคม 2556, จาก http://www8.hp.com/us/en/software-solutions/application-security/index.html

[53] IBM. Rational AppScan Source Edition. สืบค้นเมื่อ 19 สิงหาคม 2556, จาก www.ibm.com/software/rational/products/appscan/source/

[54] Klocwork Team. Klocwork Solo. สืบค้นเมื่อ 20 สิงหาคม 2556, จาก www.klocwork.com/products/solo /

[55] Acunetix Team. Acunetix. สืบค้นเมื่อ 20 สิงหาคม 2556, จาก http://www.acunetix.com/

[56] bienvenue78 (2556). 1800 PHP scripts web developers mega pack. สืบค้นเมื่อ 21 สิงหาคม 2556, จาก http://www.4shared.com/rar/W0bV391U/1800_php_scripts_web_developer.html

[57] วิกิพีเดีย สารานุกรมเสรี (2556). MySQL. สืบค้นเมื่อ 21 สิงหาคม 2556, จาก http://en.wikipedia.org/wiki/MySQL

[58] ECLIPSE KEPLER. Eclipse. สืบค้นเมื่อ 23 สิงหาคม 2556, จาก http://www.eclipse.org/

[59] Citec Club (2553). SQL Injection. สืบค้นเมื่อ 26 สิงหาคม 2556, จาก http://citecclub.org/wiki/index.php?title=SQL_Injection

ภาคผนวก

ภาคผนวก ก.

การพัฒนาปลั๊กอินบน NetBeans ประกอบไปด้วยสามขั้นตอน คือ

1. การสร้างตัวปลั๊กอิน

เป็นขั้นตอนที่โปรแกรมจัดเตรียมพื้นที่และส่วนประกอบที่จำเป็นในการพัฒนาให้ผู้พัฒนาสามารถเขียนโค้ดลงไปได้

2. การสร้างตัวติดตั้งปลั๊กอิน

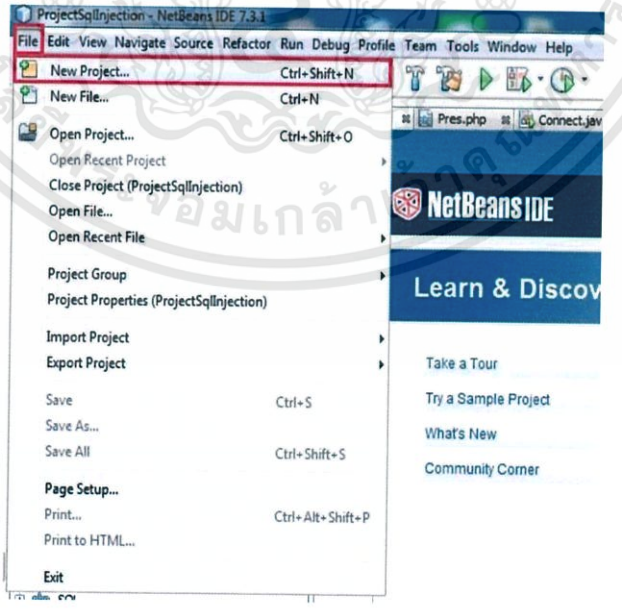
เป็นขั้นตอนที่โปรแกรมดำเนินการกับโค้ดที่ผู้พัฒนาเขียน เพื่อสร้างตัวติดตั้ง ในขั้นตอนนี้จะแยกจากขั้นตอนการสร้างตัวปลั๊กอิน เพราะในการพัฒนาปลั๊กอิน ผู้พัฒนาอาจมีการเปลี่ยนแปลงโค้ดของตัวปลั๊กอินอยู่เสมอ การสร้างตัวติดตั้งปลั๊กอินจึงเป็นการรวมโค้ดที่เขียนเสร็จแล้วเพื่อนำไปใช้ได้สะดวก

3. การติดตั้งปลั๊กอิน

เป็นขั้นตอนในการติดตั้งปลั๊กอินเพื่อใช้งานจริง

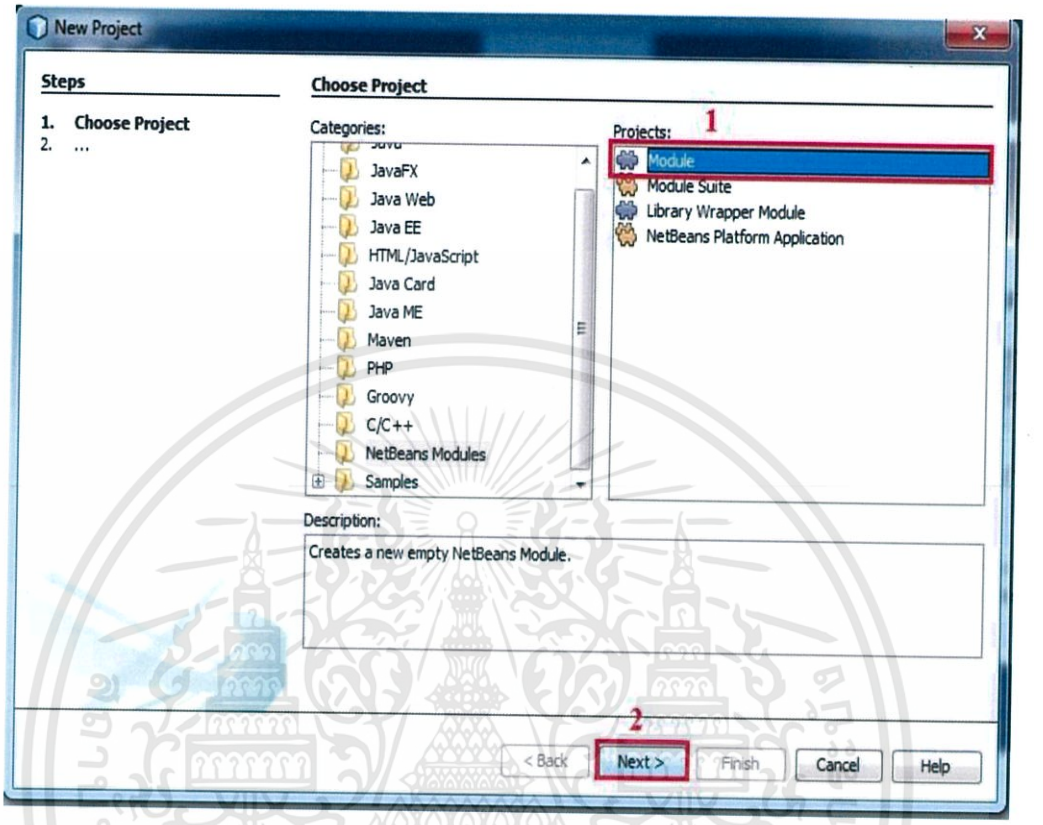
ก.1 วิธีสร้างโปรแกรมปลั๊กอินบน NetBeans

1) เลือกเมนู File > New Project… ดังรูปที่ ก.1



รูปที่ ก.1 หน้าจอเลือกเมนู File > New Project… บน NetBeans

2) ในหน้าต่าง New Project เลือก Module แล้วกดปุ่ม Next



รูปที่ ก.2 หน้าต่างแสดงการเลือก Module

3) ในหน้าต่าง New Module ตั้งค่าดังรูปที่ ก.3

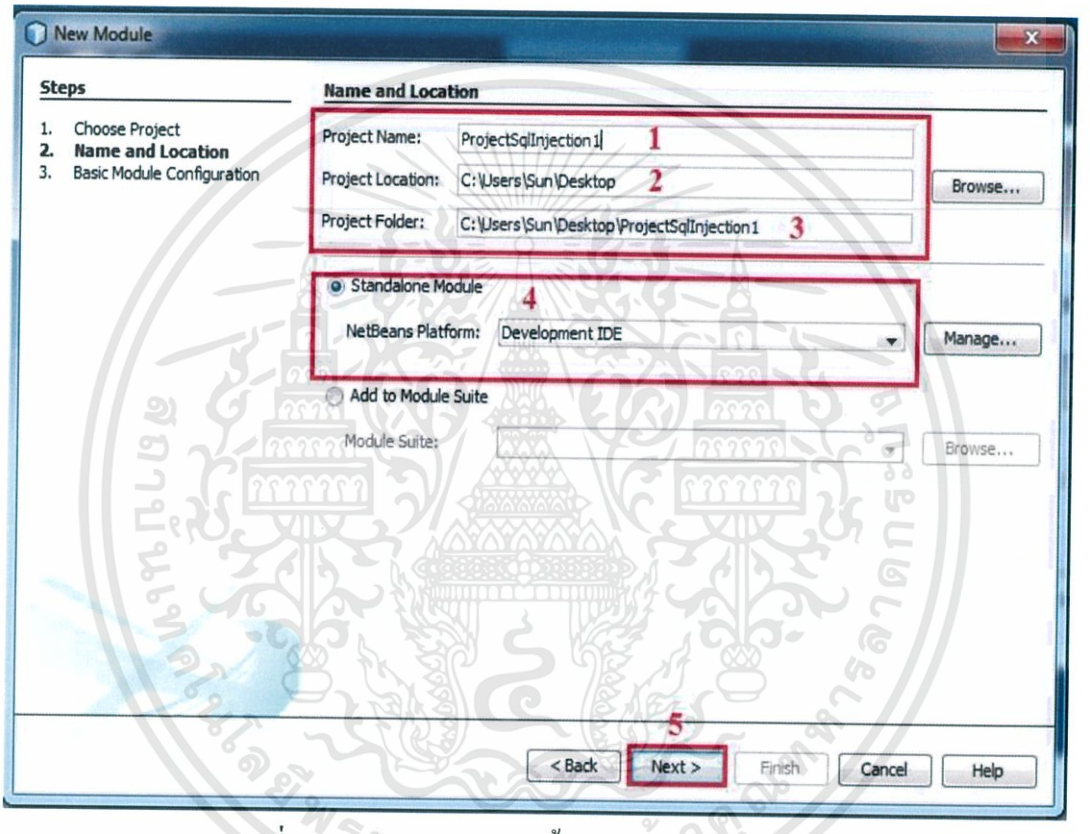
3.1) ตั้งชื่อ Project Name ตาม 1

3.2) เลือก Project Location ตาม 2

3.3) ตั้งชื่อ Project Folder ตาม 3

3.4) คลิกเลือก Standalone Module ตาม 4

3.5) กดปุ่ม Next >



รูปที่ ก.3 หน้าต่างแสดงการตั้งค่า Name and Location

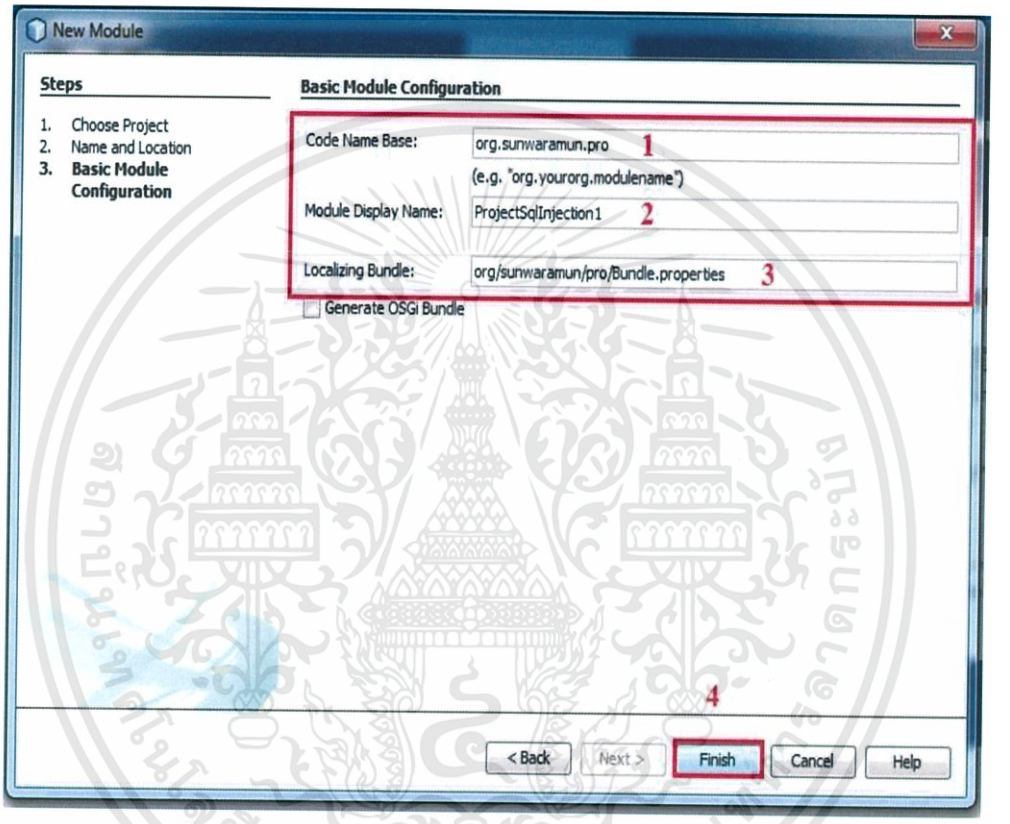
4) ในหน้า New Module ตรงส่วน Basic Module Configuration ตั้งค่าตามรูปที่ ก.4

4.1) ตั้งชื่อ Code Name Base ตาม 1

4.2) ตั้งชื่อ Module Display Name ตาม 2

4.3) ตั้งชื่อ Location Bundle ตาม 3

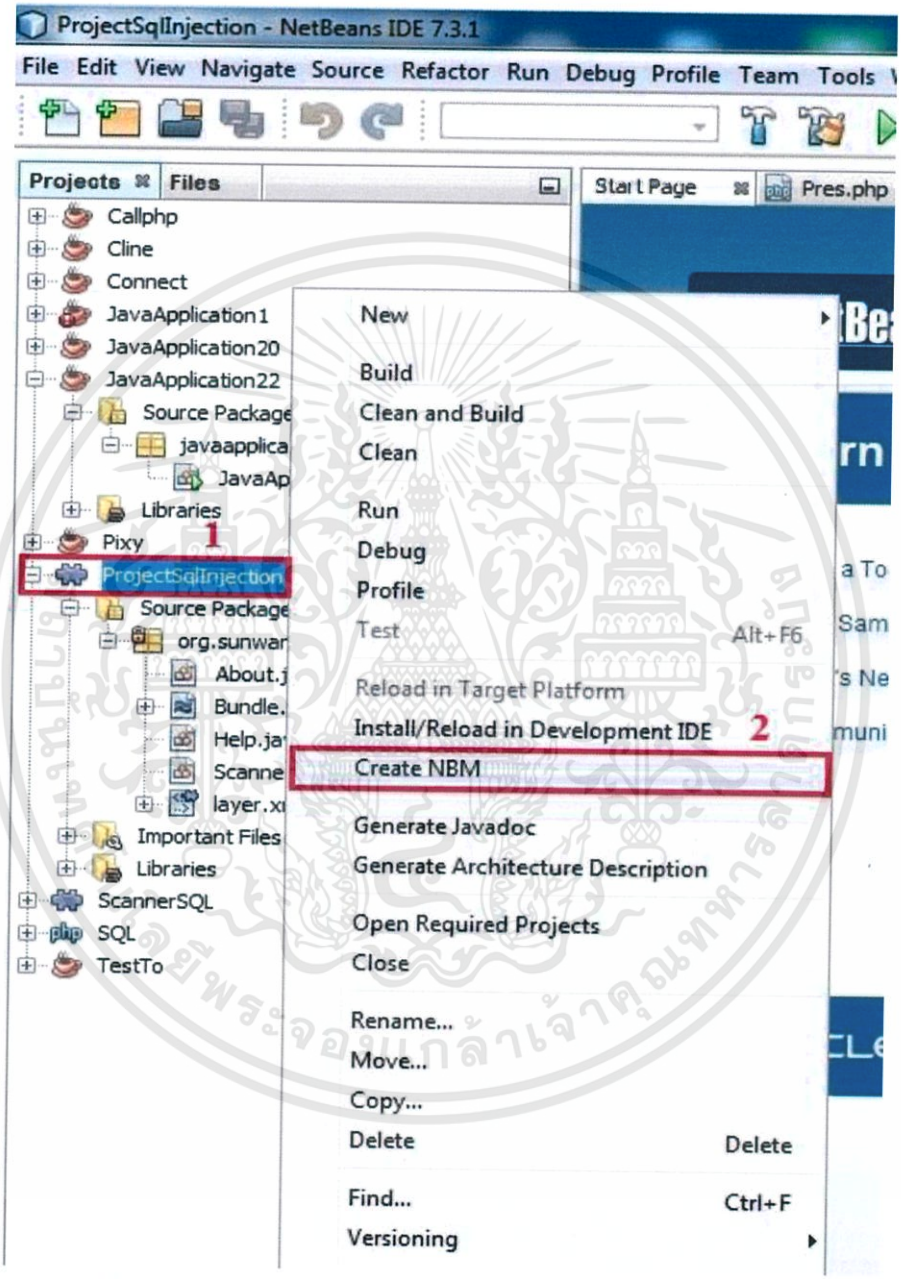
4.4) กดปุ่ม Finish เป็นอันเสร็จการสร้างตัวปลั๊กอินบน NetBeans



รูปที่ ก.4 หน้าต่างแสดงการตั้งค่า Basic Module Configuration

ก.2 วิธีสร้างตัวติดตั้ง (Install) ของโปรแกรมปลั๊กอินบน NetBeans

เมื่อทำการตั้งค่าตาม ก.1 เรียบร้อยแล้วคลิกขวาที่ ProjectSqlInjection แล้วเลือกเมนู Create NBM เป็นอันเสร็จการสร้างตัวติดตั้งโปรแกรมปลั๊กอินบน NetBeans

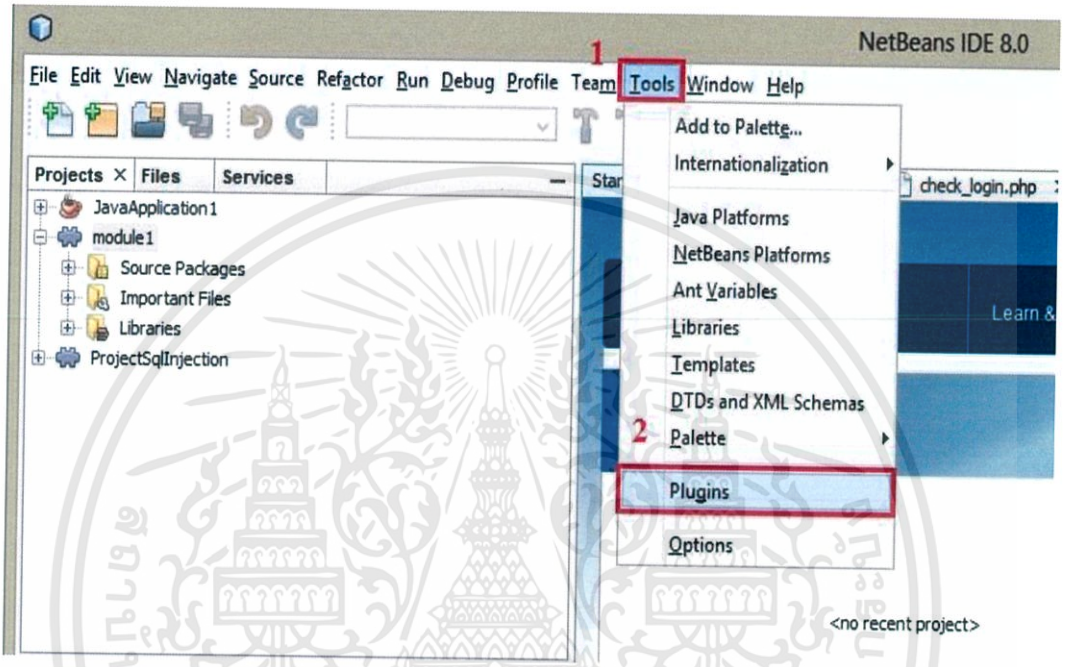


รูปที่ ก.5 หน้าจอการสร้างตัวติดตั้งโปรแกรมปลั๊กอินบน NetBeans

ก.3 การติดตั้งโปรแกรมปลั๊กอินบน NetBeans

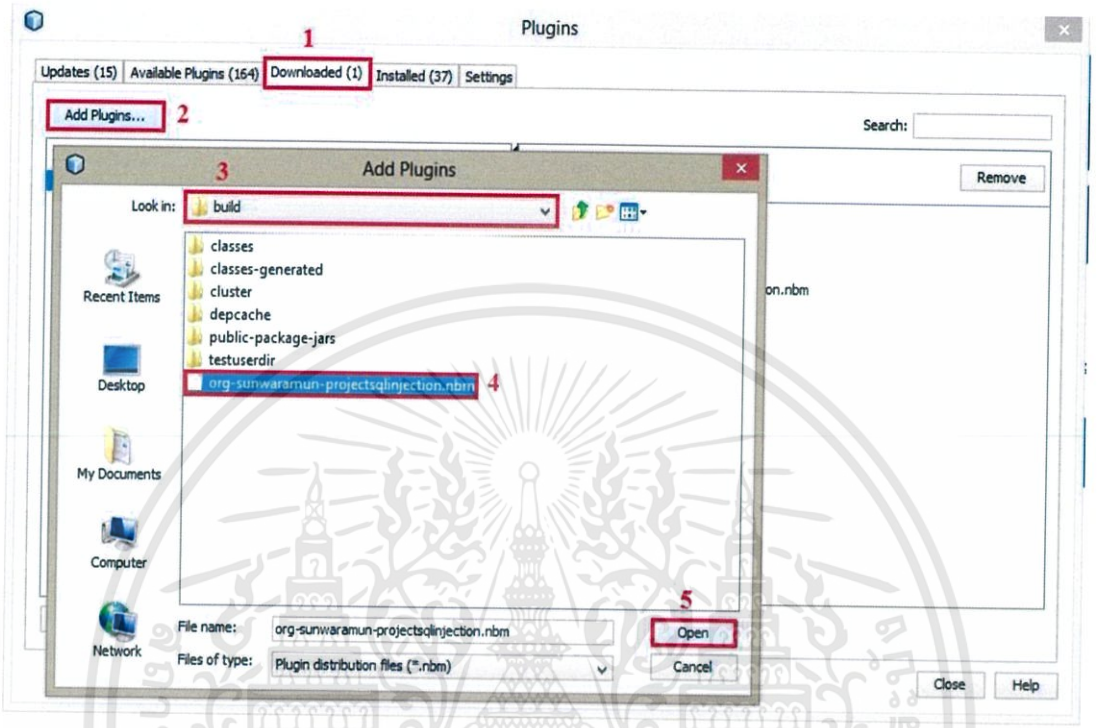
ก่อนที่ผู้ใช้งานจะสามารถใช้งานโปรแกรมปลั๊กอินได้นั้นจำเป็นต้องทำการติดตั้งลงบน NetBeans เสียก่อน โดยมีวิธีการติดตั้งดังนี้

1) เปิดโปรแกรม NetBeans ขึ้นมาแล้วเลือกเมนู Tools > Plugins ดังภาพที่ ก.6



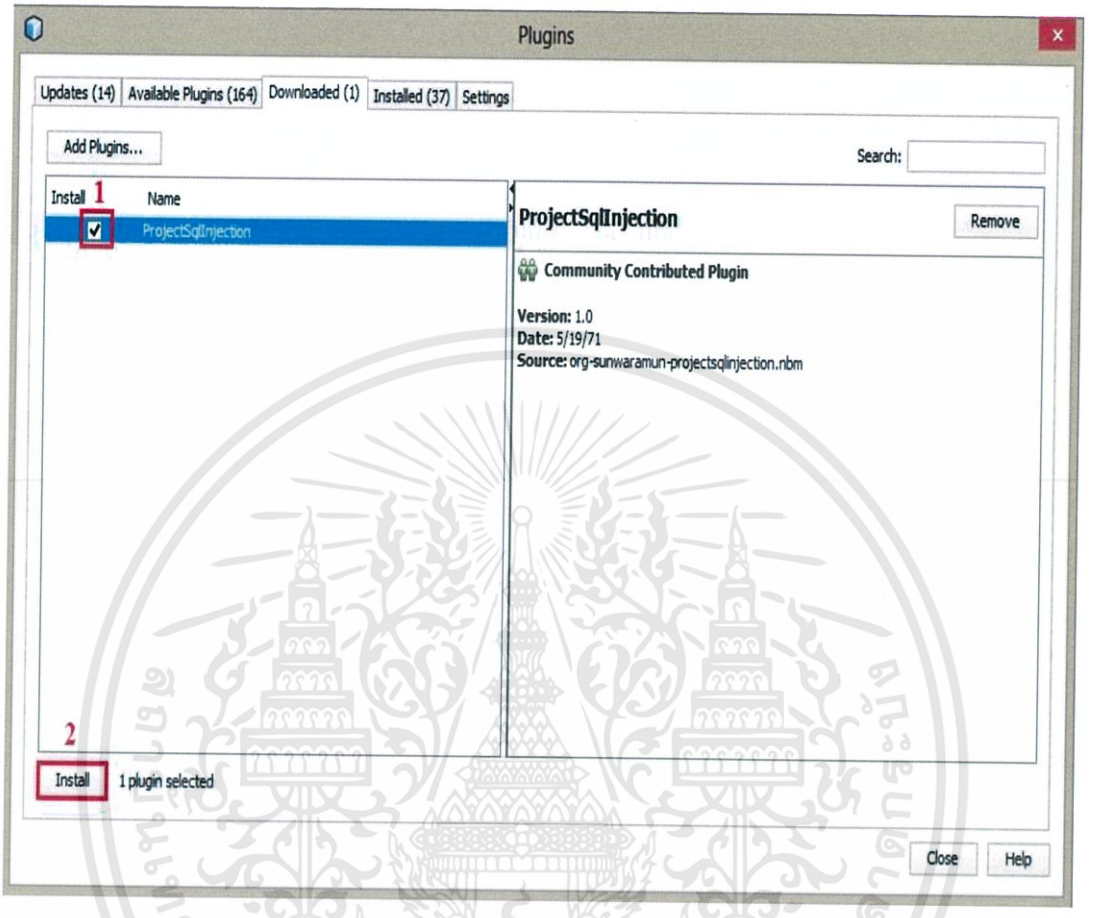
รูปที่ ก.6 หน้าจอเลือกเมนู Tools > Plugins บน NetBeans

2) บนหน้าต่าง Plugins เลือก Downloaded > Add Plugins… จากนั้นเลือกโฟลเดอร์ของโปรแกรม ตามด้วย build > org-sunwaramun-projectsqlinjection.nbm



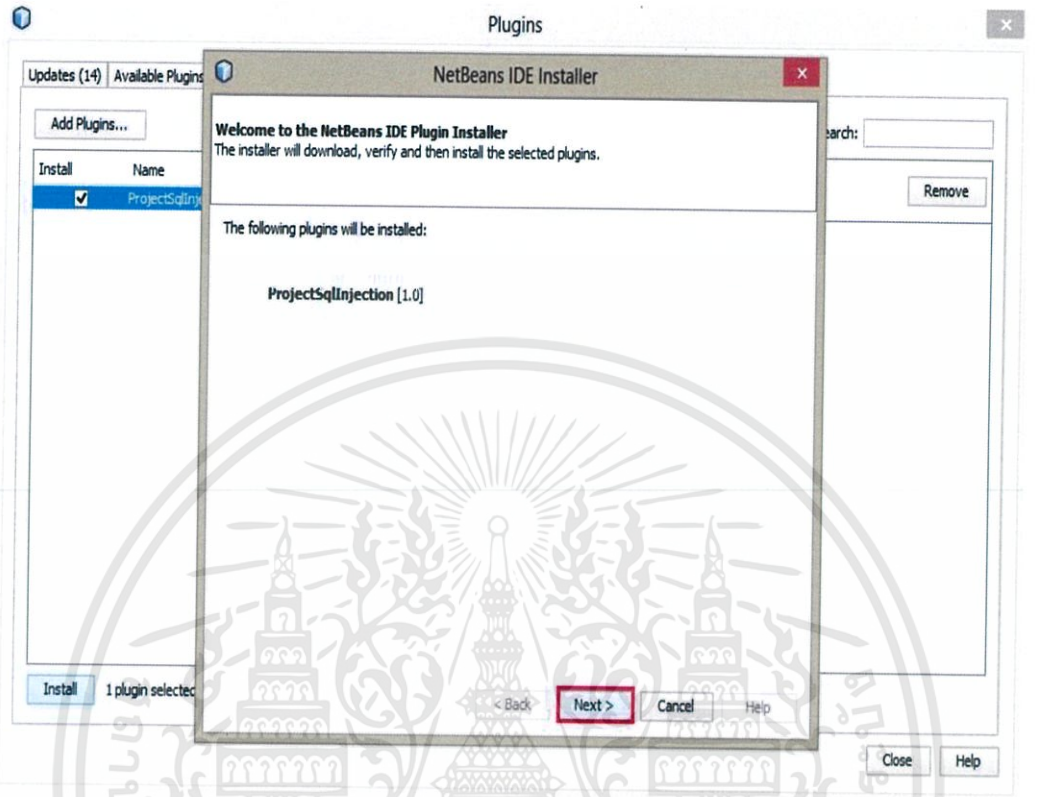
รูปที่ ก.7 หน้าต่างการเลือก org-sunwaramun-projectsqlinjection.nbm

3) เมื่อ Add Plugins แล้วให้คลิกเลือก ProjectSqlInjection แล้วกดปุ่ม Install ดังภาพที่ ก.8



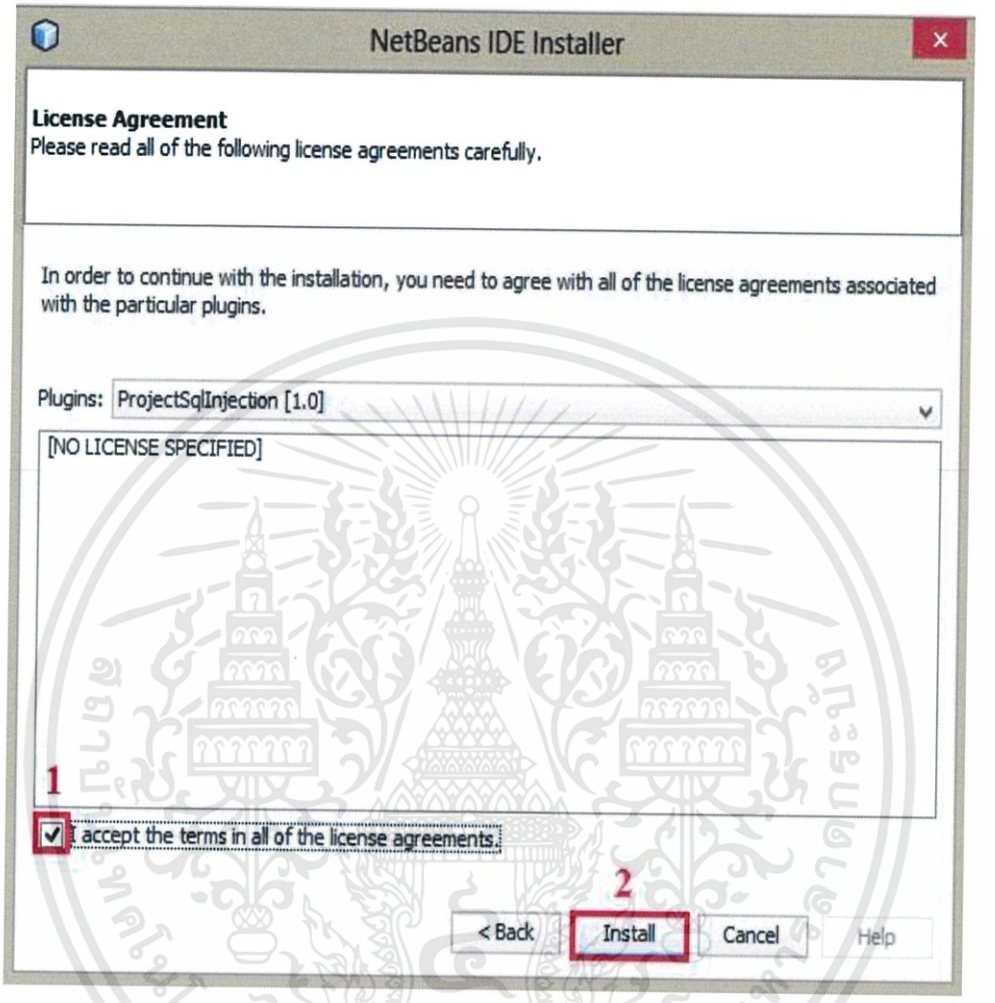
รูปที่ ก.8 หน้าต่างการเลือก ProjectSqlInjection

4) เมื่อเข้าสู่หน้าต่าง Install ให้กดปุ่ม Next



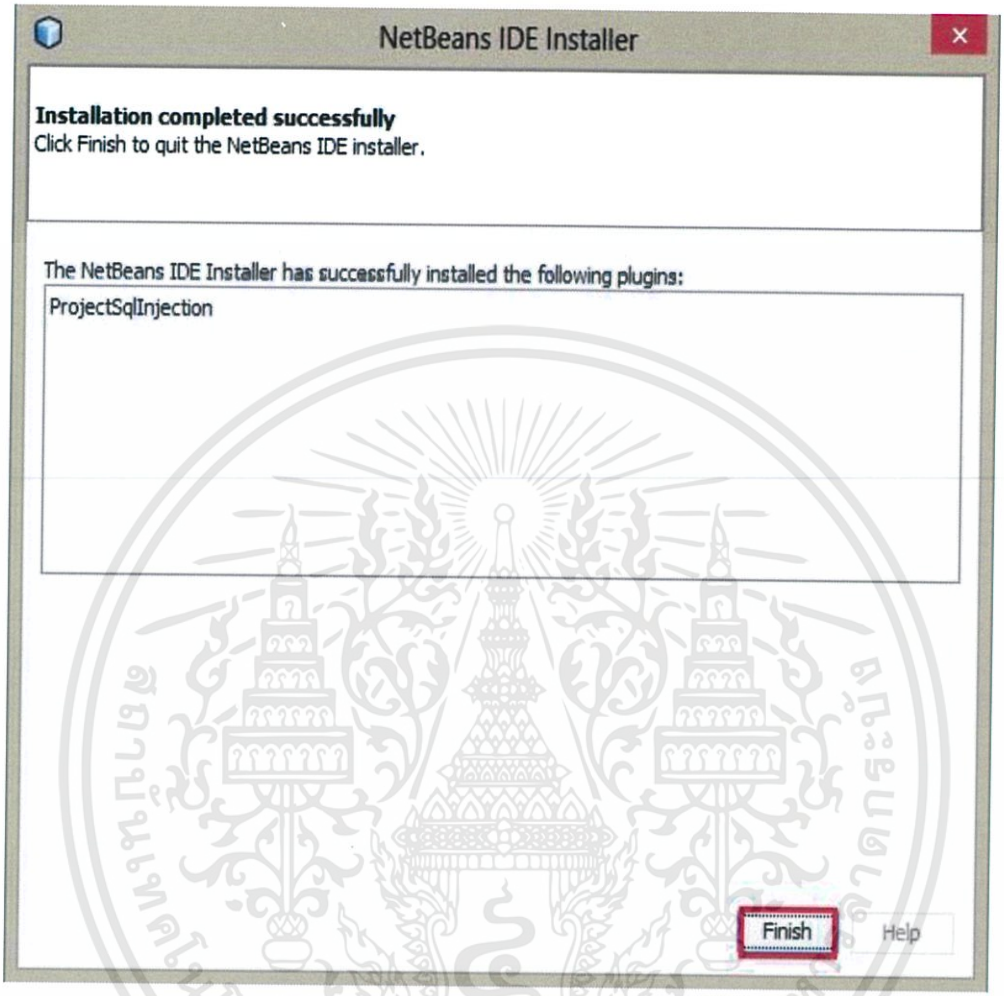
รูปที่ ก.9 หน้าต่าง Install

5) จากนั้นคลิกเลือก “I accept the terms in all of the license agreements” แล้วกดปุ่ม Install



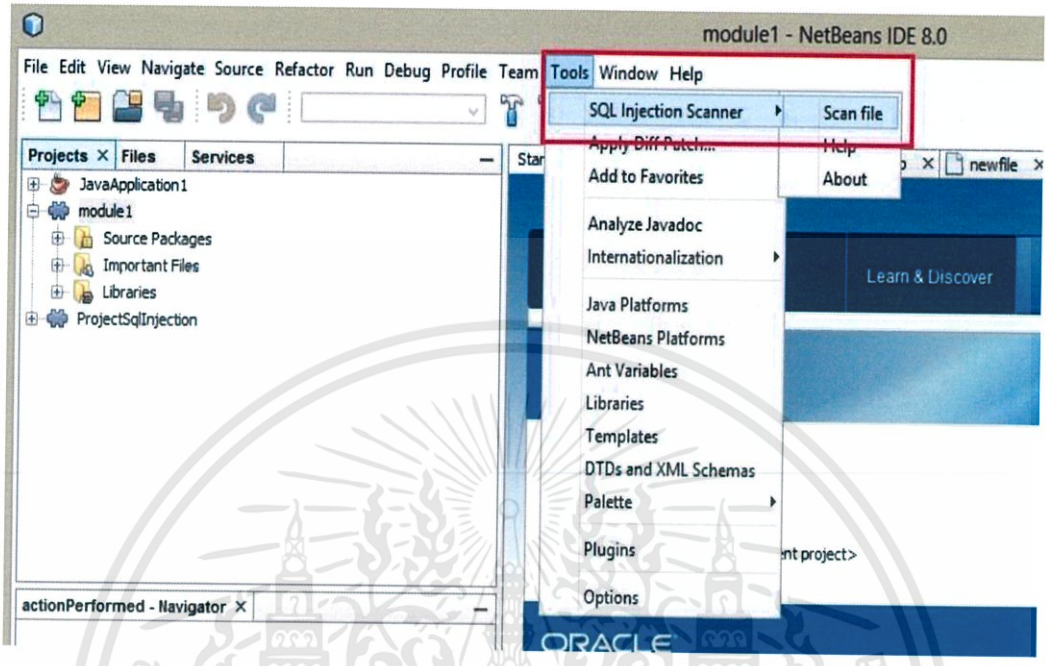
รูปที่ ก.10 หน้าต่างแสดงการยอมรับข้อตกลง

6) เมื่อโปรแกรมติดตั้งเรียบร้อยแล้วจะมีหน้าต่างขึ้นมาให้กด Finish



รูปที่ ก.11 หน้าต่างแสดงการเสร็จสิ้นการ Install

7) โปรแกรมปลั๊กอินจะแสดงอยู่บนเมนู Tools ดังภาพที่ ก.7



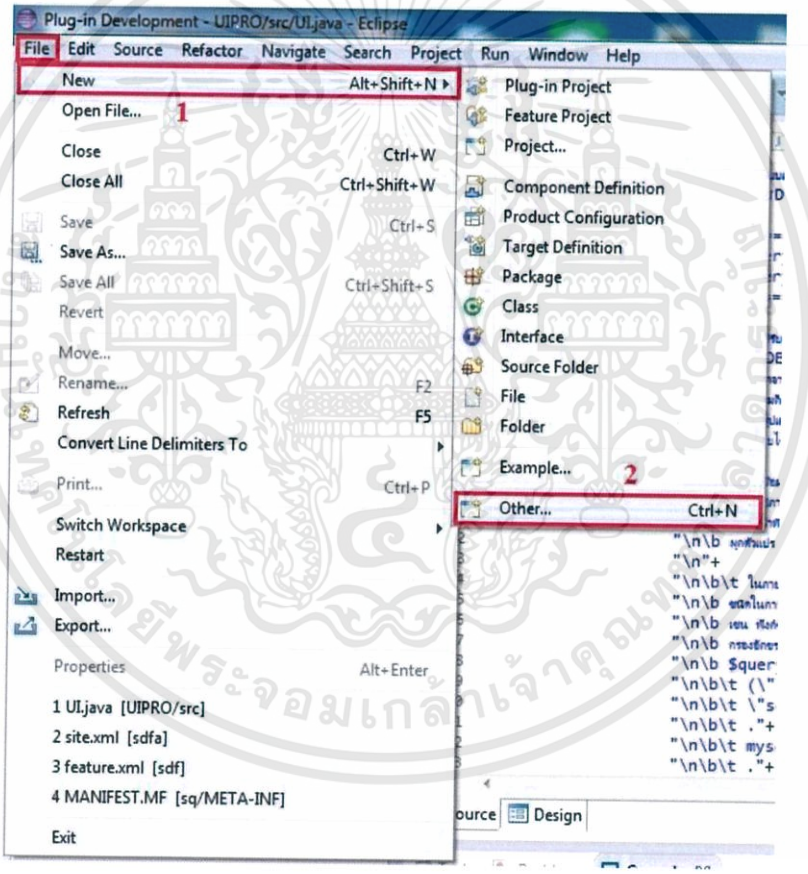
รูปที่ ก.12 หน้าจอเมนูของโปรแกรมปลั๊กอินเมื่อติดตั้งแล้ว

ภาคผนวก ข.

การพัฒนาปลั๊กอินบน Eclipse ประกอบไปด้วยสามขั้นตอน ได้แก่ การสร้างตัวปลั๊กอิน การสร้างตัวติดตั้งปลั๊กอินและการติดตั้งปลั๊กอินเช่นเดียวกับบน NetBeans

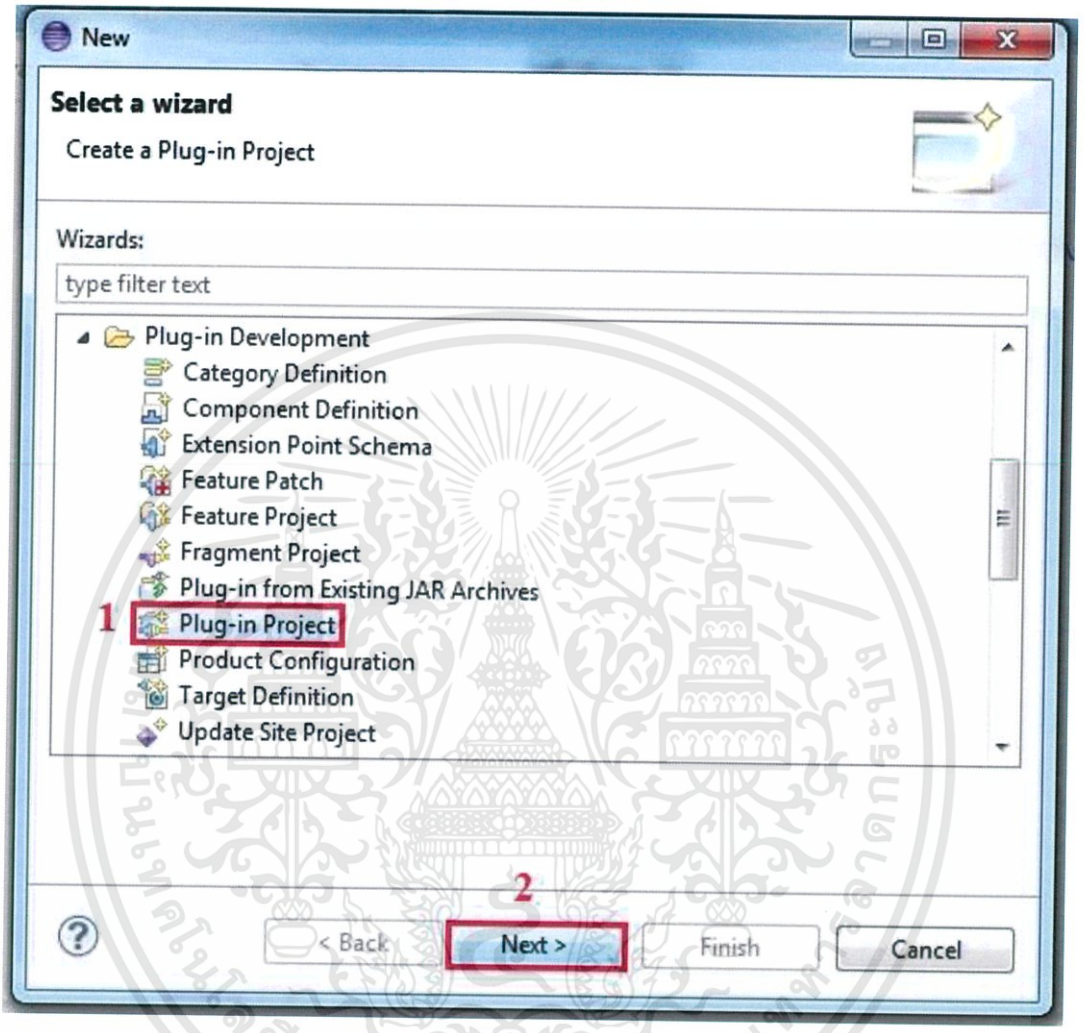
ข.1 วิธีสร้างโปรแกรมปลั๊กอินบน Eclipse

1) เลือกเมนู File > New > Other… ดังรูปที่ ข.1



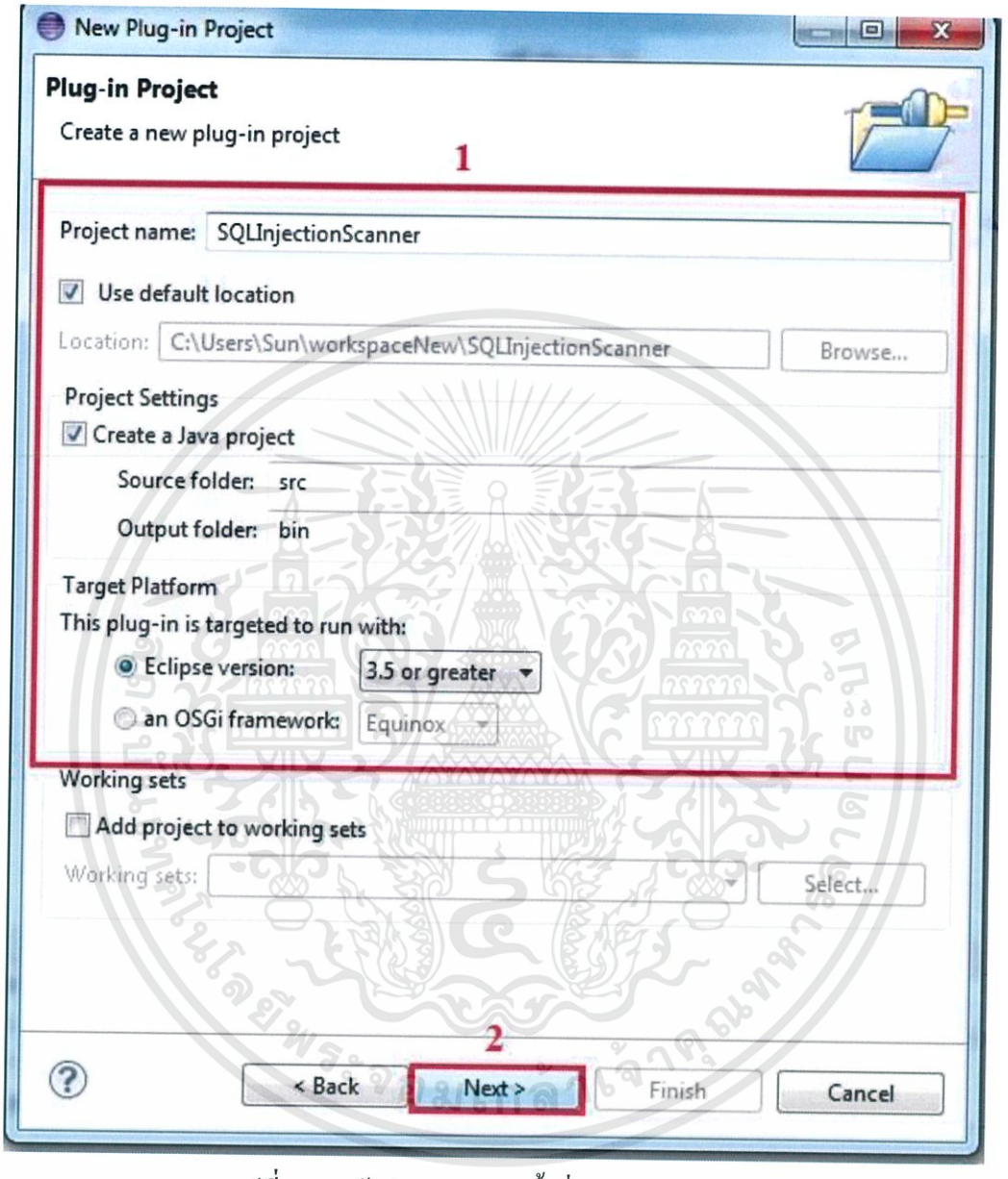
รูปที่ ข.1 หน้าจอแสดงการเลือกเมนู File > New > Other…

2) เลือก Plug-in Project แล้วกด Next



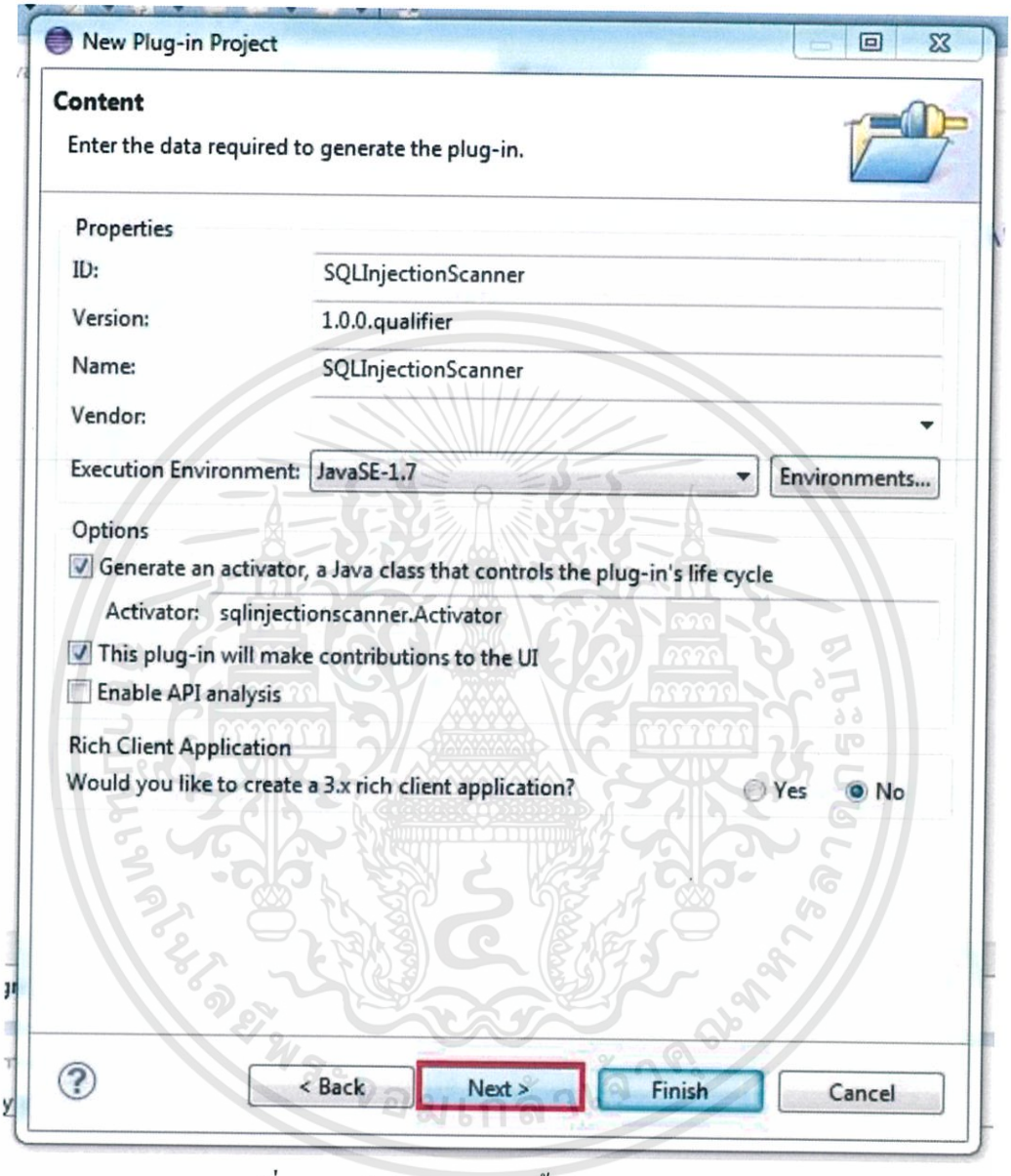
รูปที่ ข.2 หน้าต่างแสดงการเลือก Plug-in Project

3) ตั้งค่า Plug-in Project แล้วกดปุ่ม Next > ดังรูปที่ ข.3



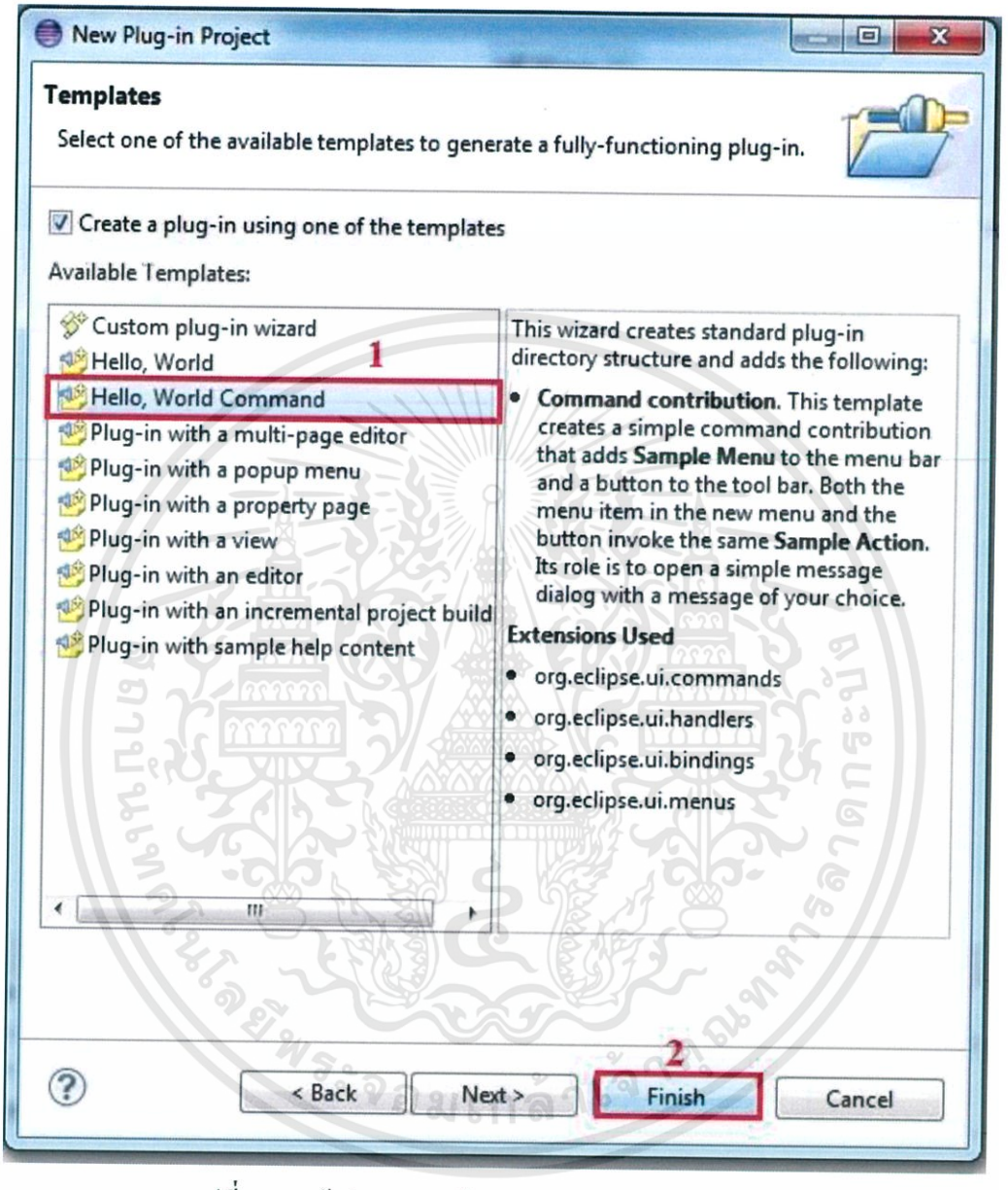
รูปที่ ข.3 หน้าต่างแสดงการตั้งชื่อ Plug-in Project

4) กดปุ่ม Next > อีกครั้ง



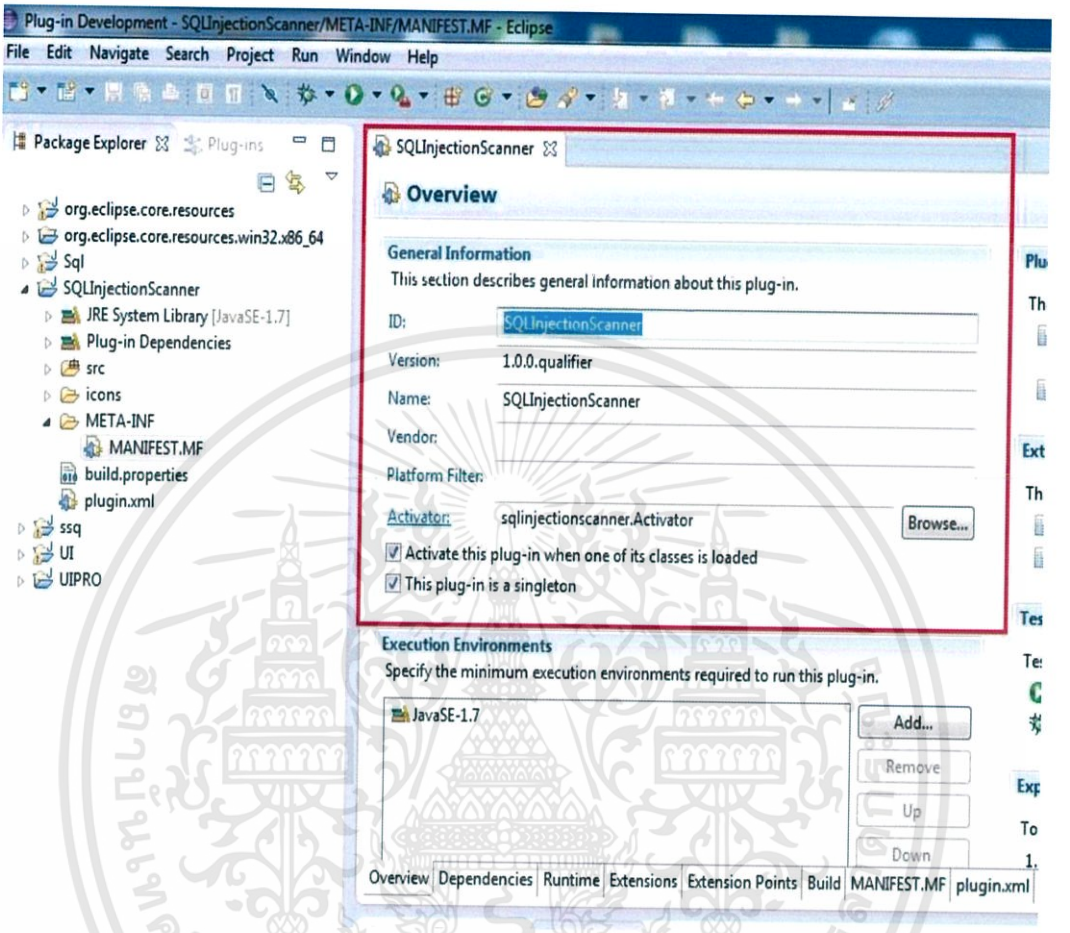
รูปที่ ข.4 หน้าต่างแสดงการตั้งค่า Plug-in Project

5) เลือก Hello, World Command แล้วกด Finish ดังรูปที่ ข.5



รูปที่ ข.5 หน้าต่างแสดงเลือก Templates Plug-in Project

6) จะได้ปลั๊กอินโปรแกรม SQLInjectionScanner

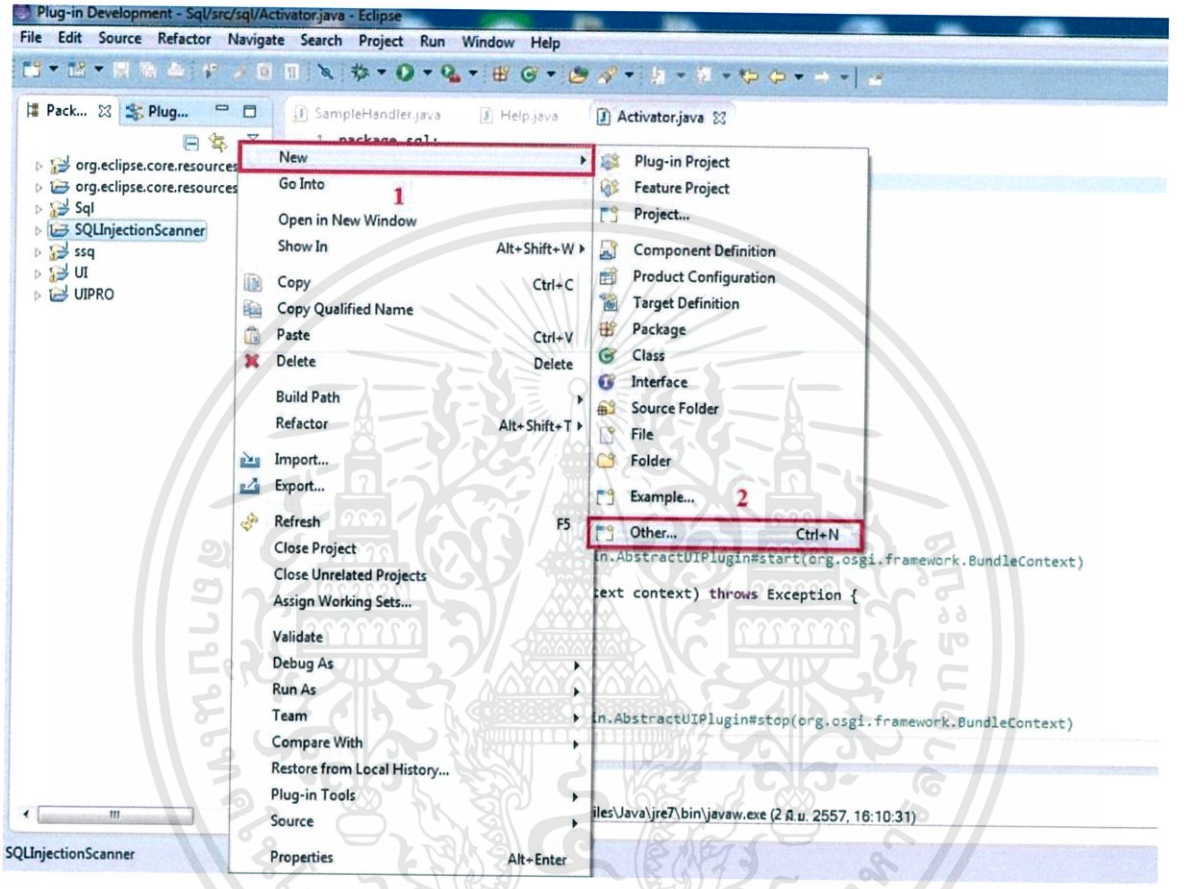


รูปที่ ข.6 หน้าจอโปรแกรม SQLInjectionScanner

ข.2 วิธีสร้างตัว ติดตั้ง (Install) ของโปรแกรมปลั๊กอินบน Eclipse

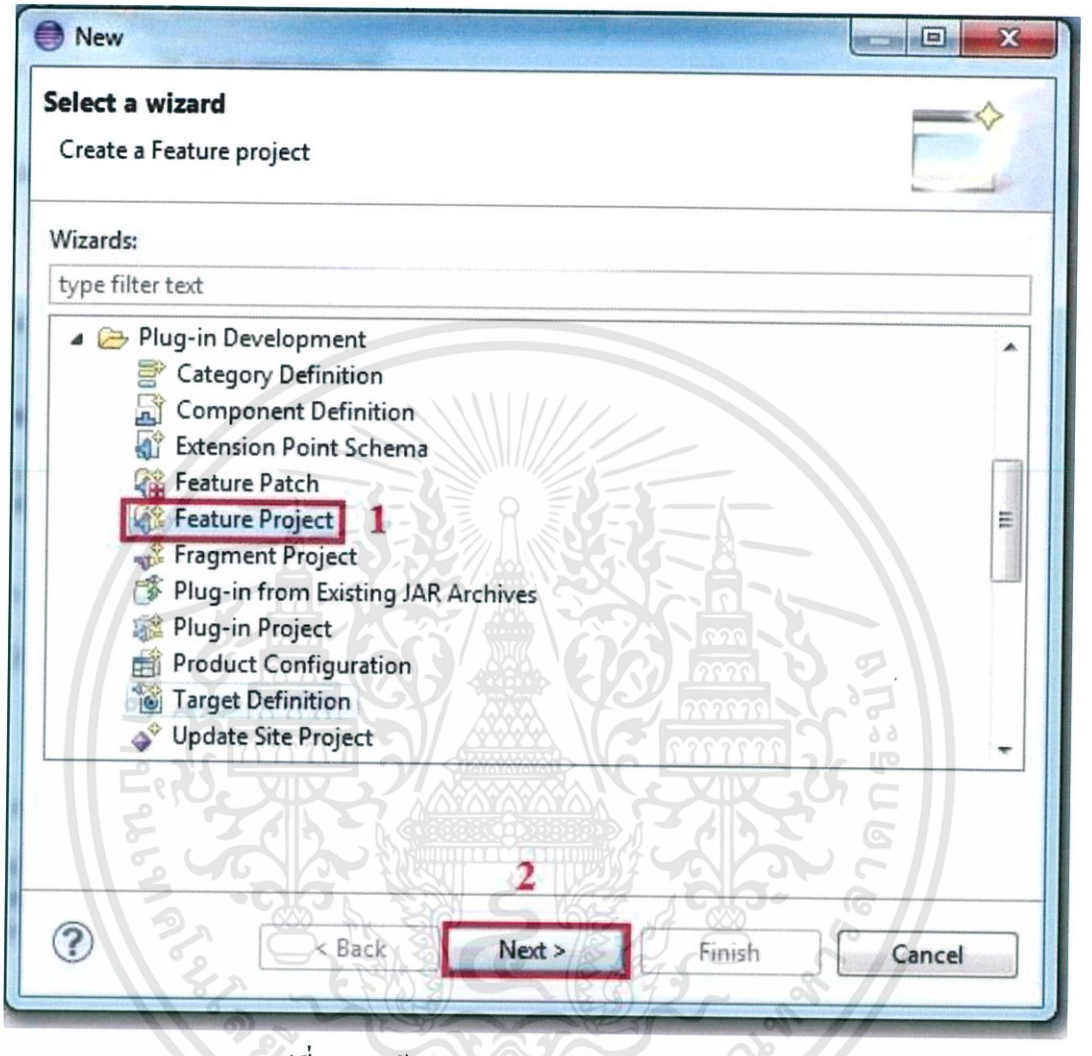
ขั้นตอนในการสร้างตัว Install ของโปรแกรมปลั๊กอินบน Eclipse

1) คลิกขวาที่โปรเจค SQLInjectionScanner แล้วเลือก New > Other… ดังรูปที่ ข.7



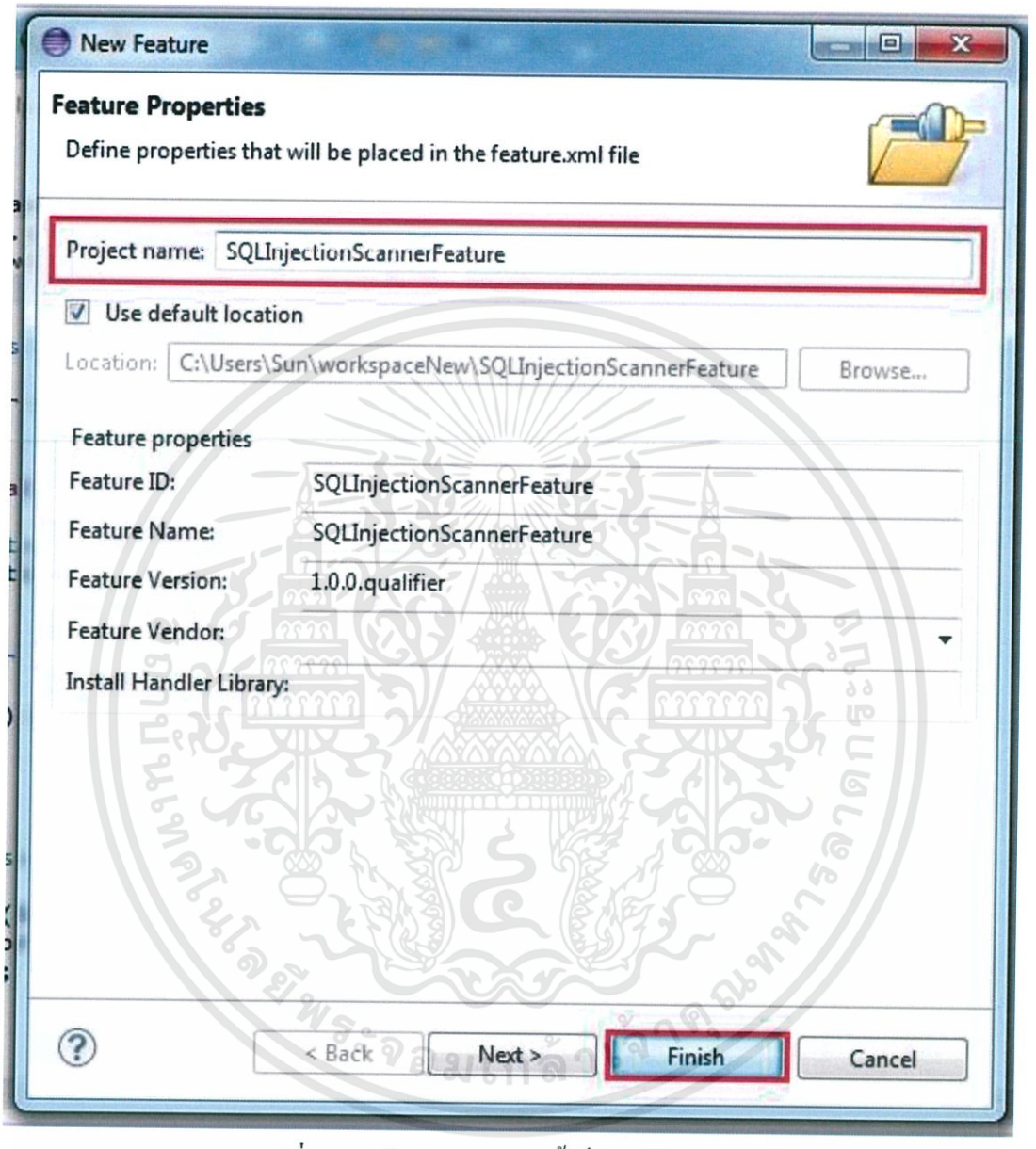
รูปที่ ข.7 หน้าจอแสดงการเลือกเมนู File > New > Other…

2) เลือก Feature Project แล้วกด Next



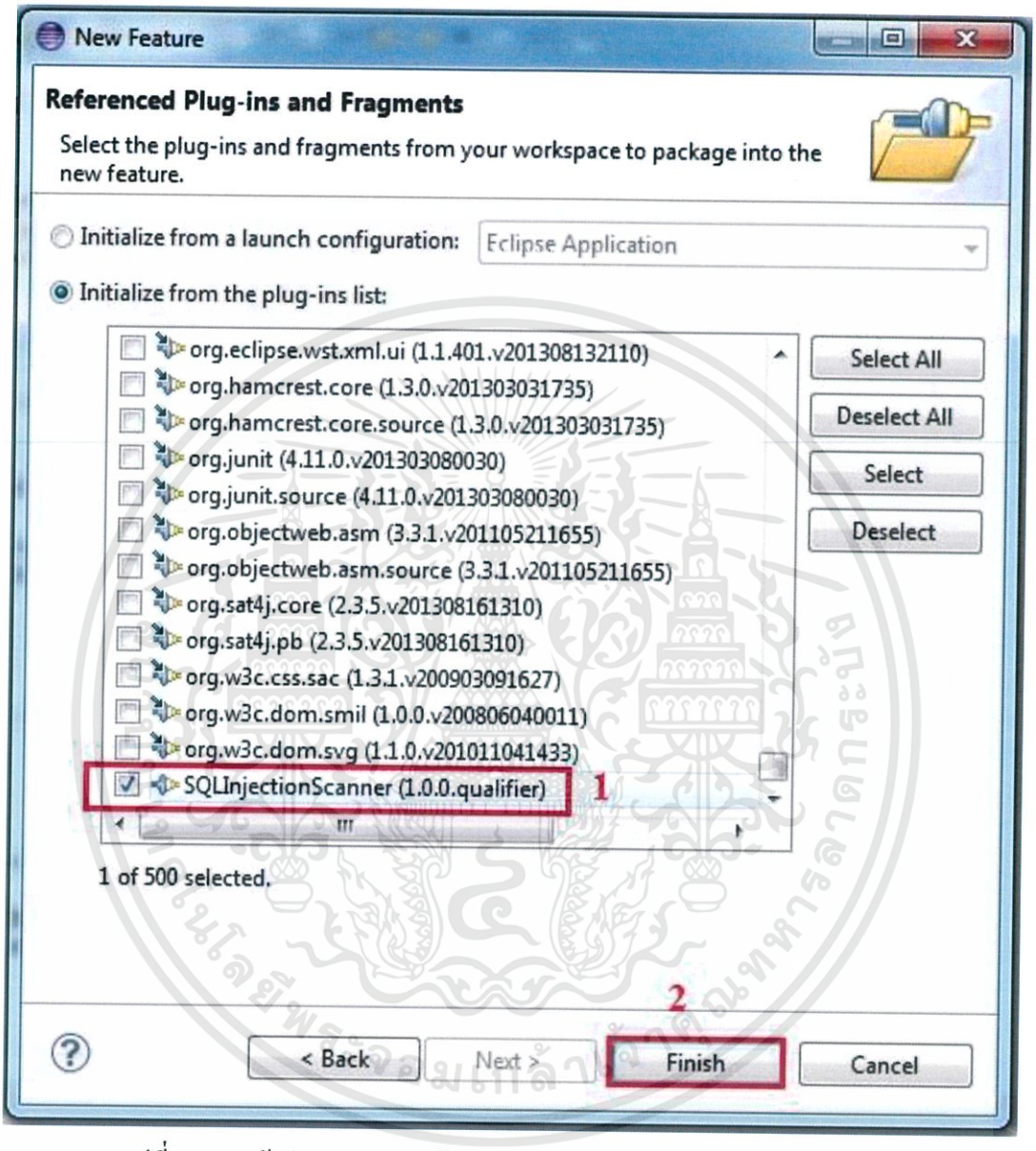
รูปที่ ข.8 หน้าต่างแสดงการเลือก Feature Project

3) ตั้งชื่อ Project name แล้วกด Finish ดังรูปที่ ข.3



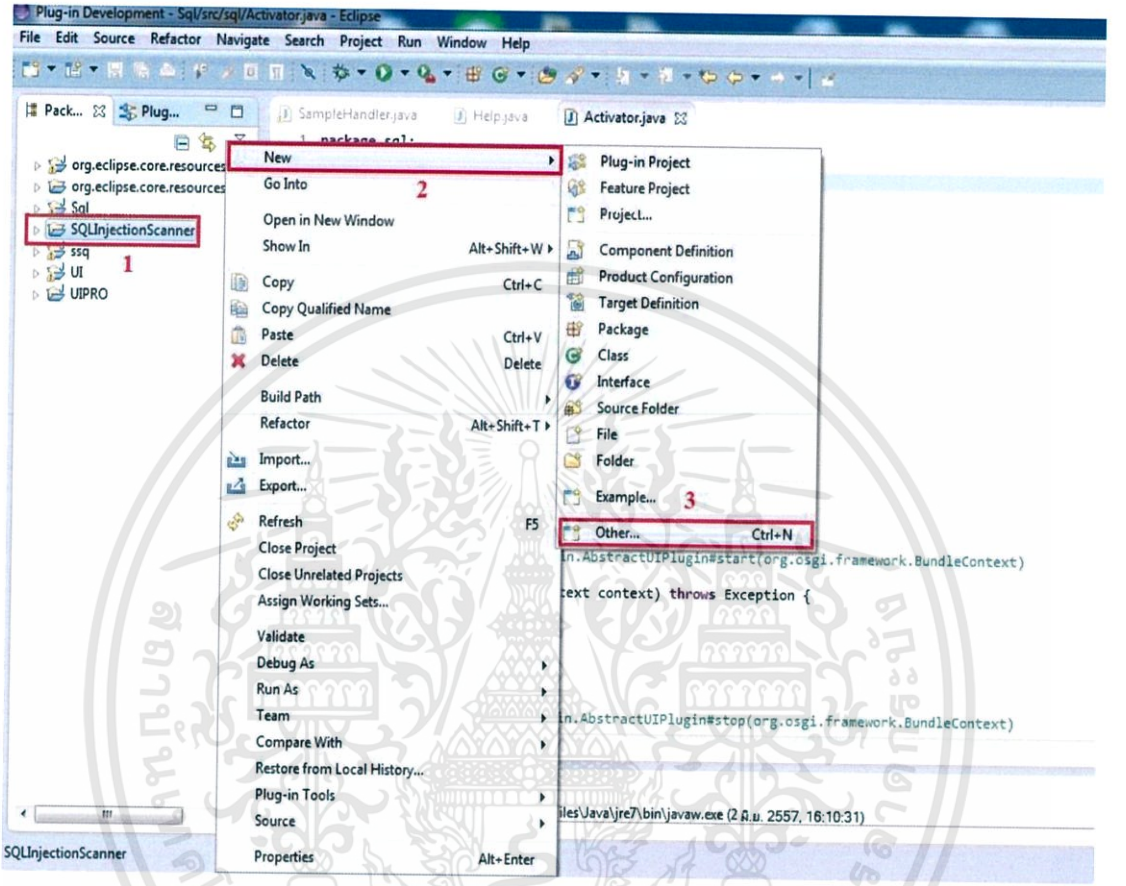
รูปที่ ข.9 หน้าต่างแสดงการตั้งชื่อ Project name

4) คลิกเลือก SQLInjectionScanner (1.0.0.qualifier) แล้วกด Finish



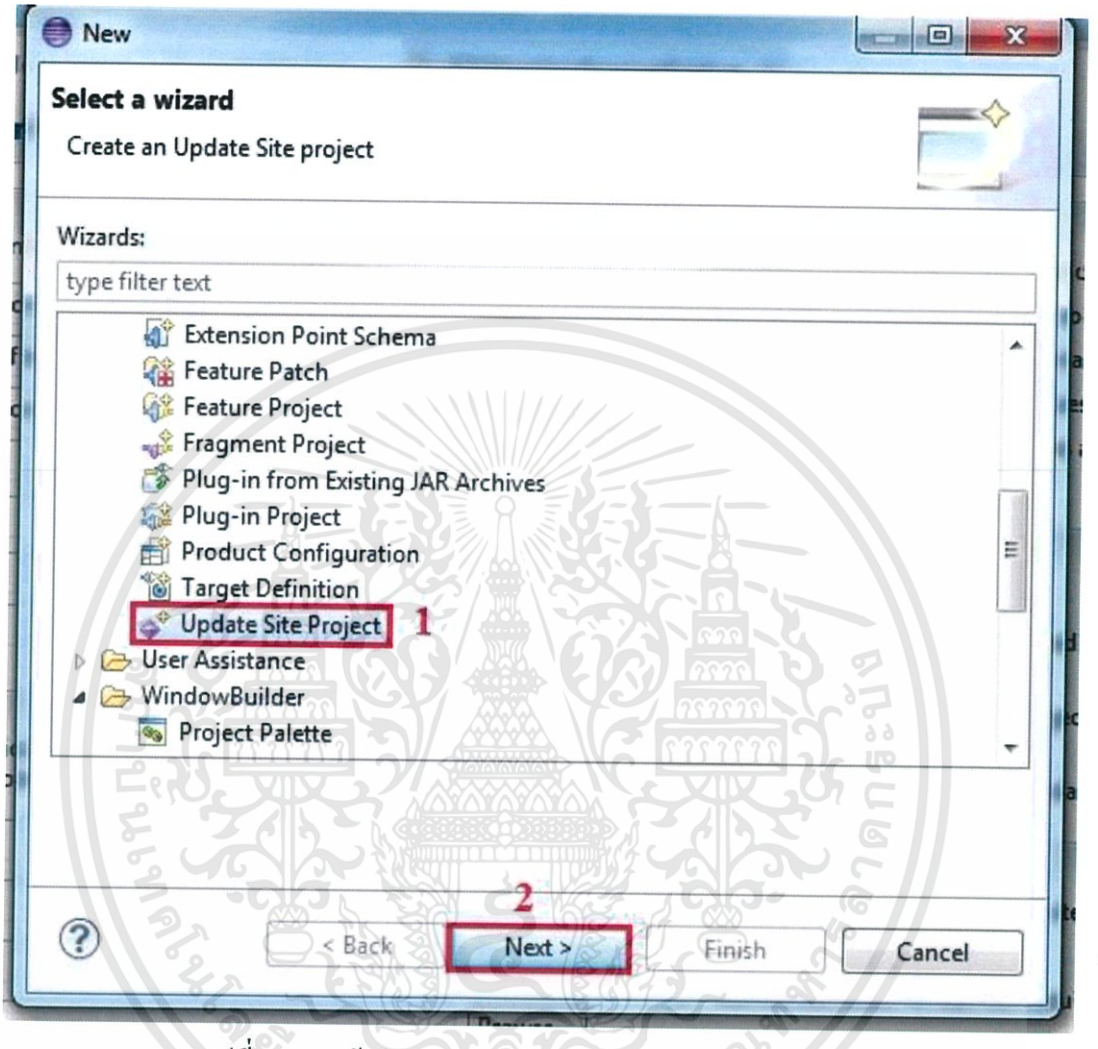
รูปที่ ข.10 หน้าต่างแสดงการเลือก SQLInjectionScanner (1.0.0.qualifier)

5) คลิกขวาที่ SQLInjectionScanner แล้วเลือก New > Other…



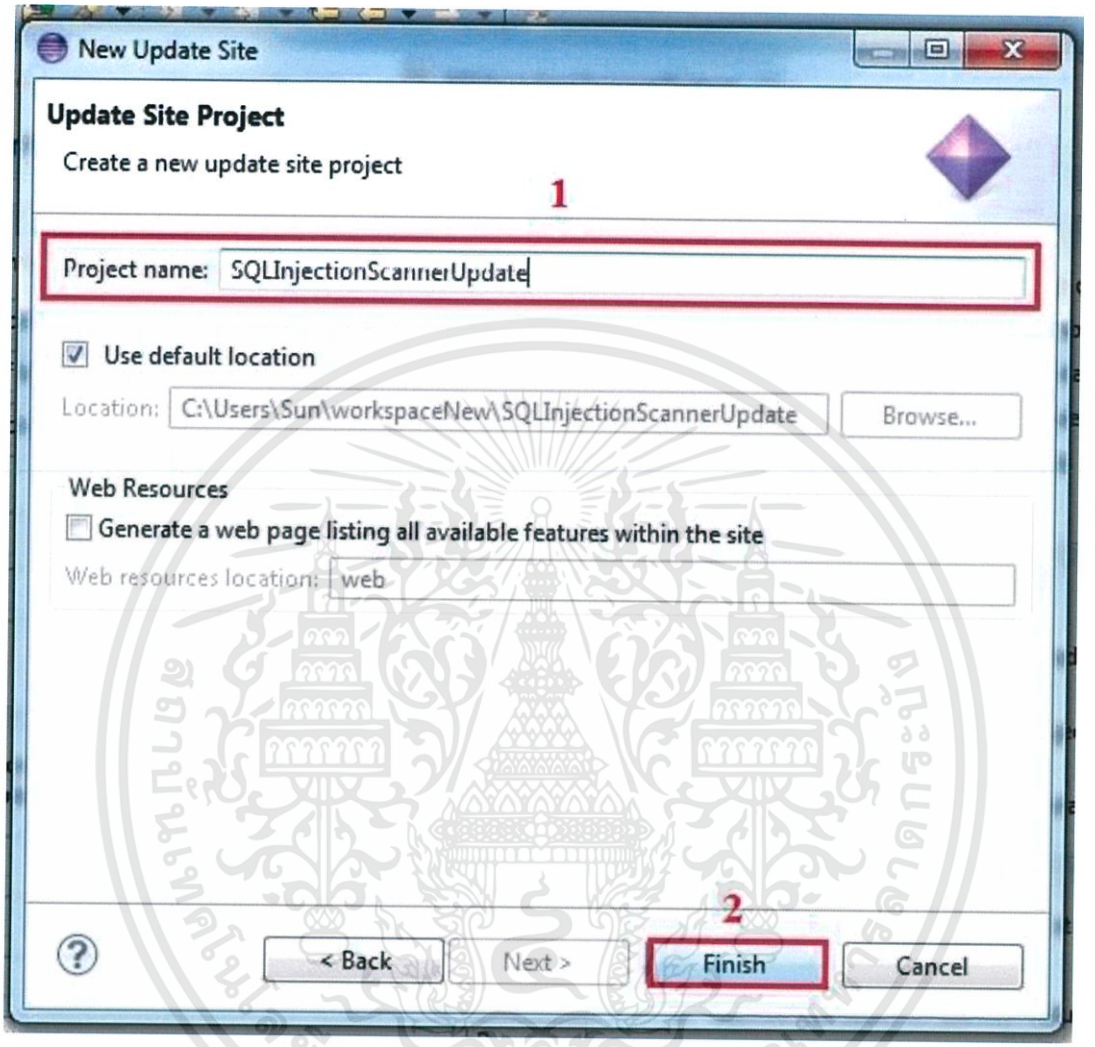
รูปที่ ข.11 หน้าจอการเลือก New > Other…

6) ในหน้าต่าง New เลือก Update Site Project แล้วกด Next >



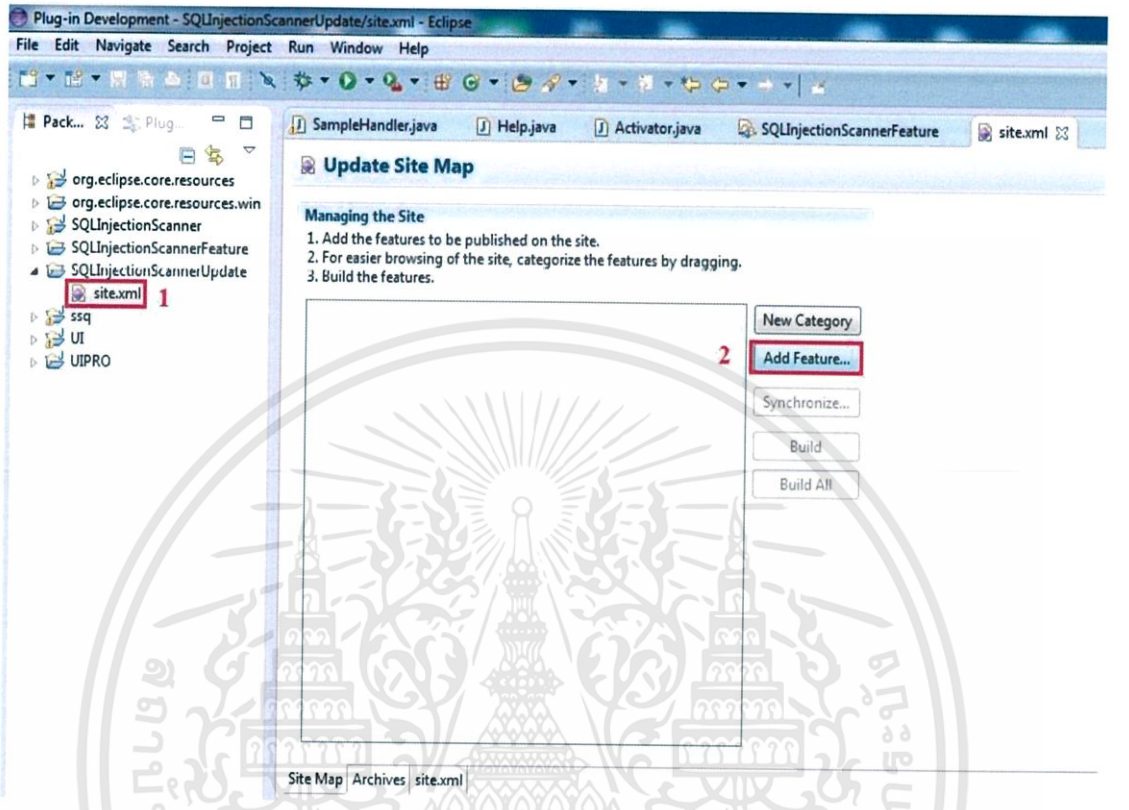
รูปที่ ข.12 หน้าต่าง New แสดงการเลือก Update Site Project

7) ตั้งชื่อ Project name เป็น SQLInjectionScannerUpdate แล้วกดปุ่ม Finish ดังรูปที่ ข.7



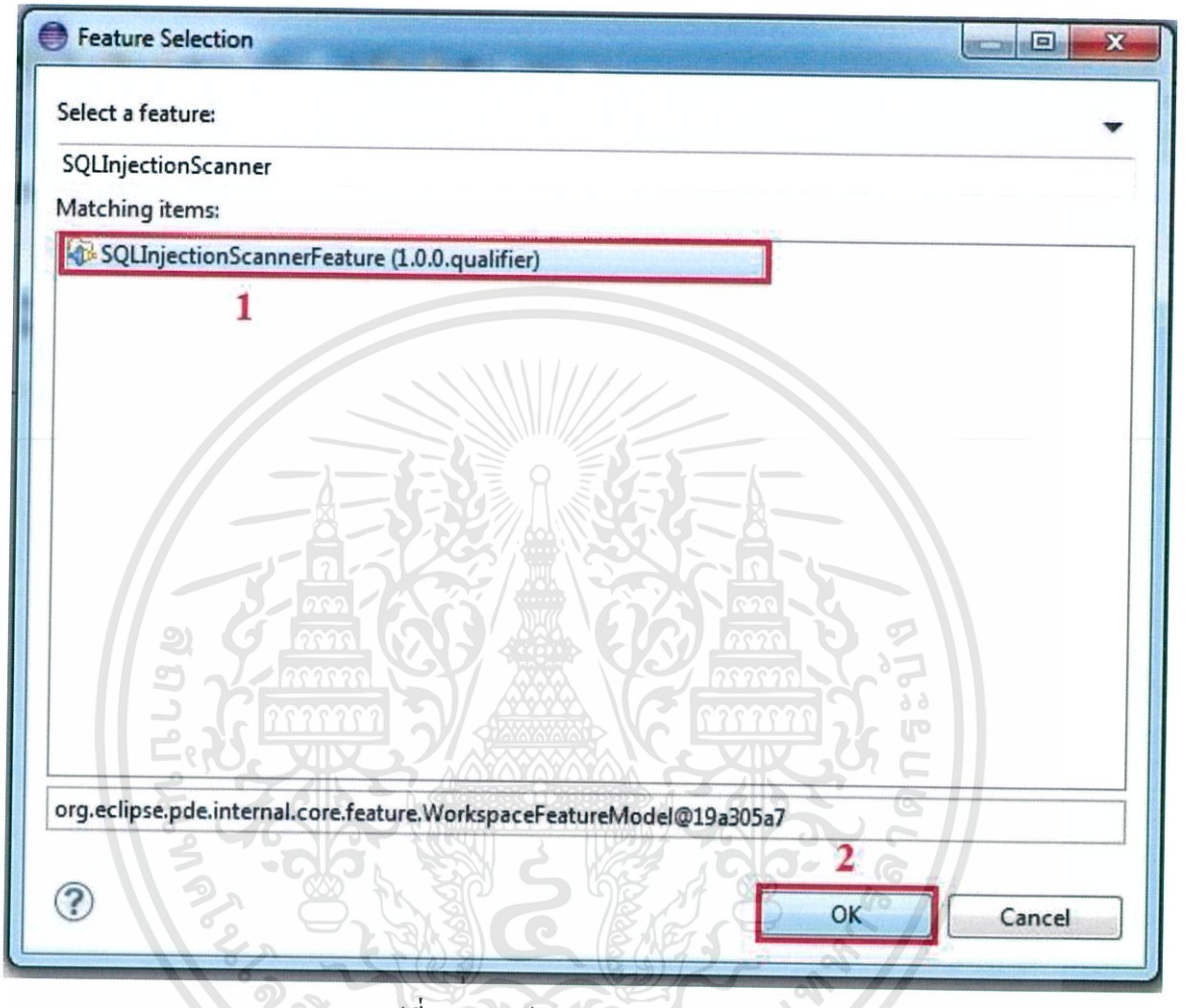
รูปที่ ข.13 หน้าต่างการตั้งชื่อ Project name

8) เลือก site.xml แล้วกดปุ่ม Add Feature ดังรูปที่ ข.8



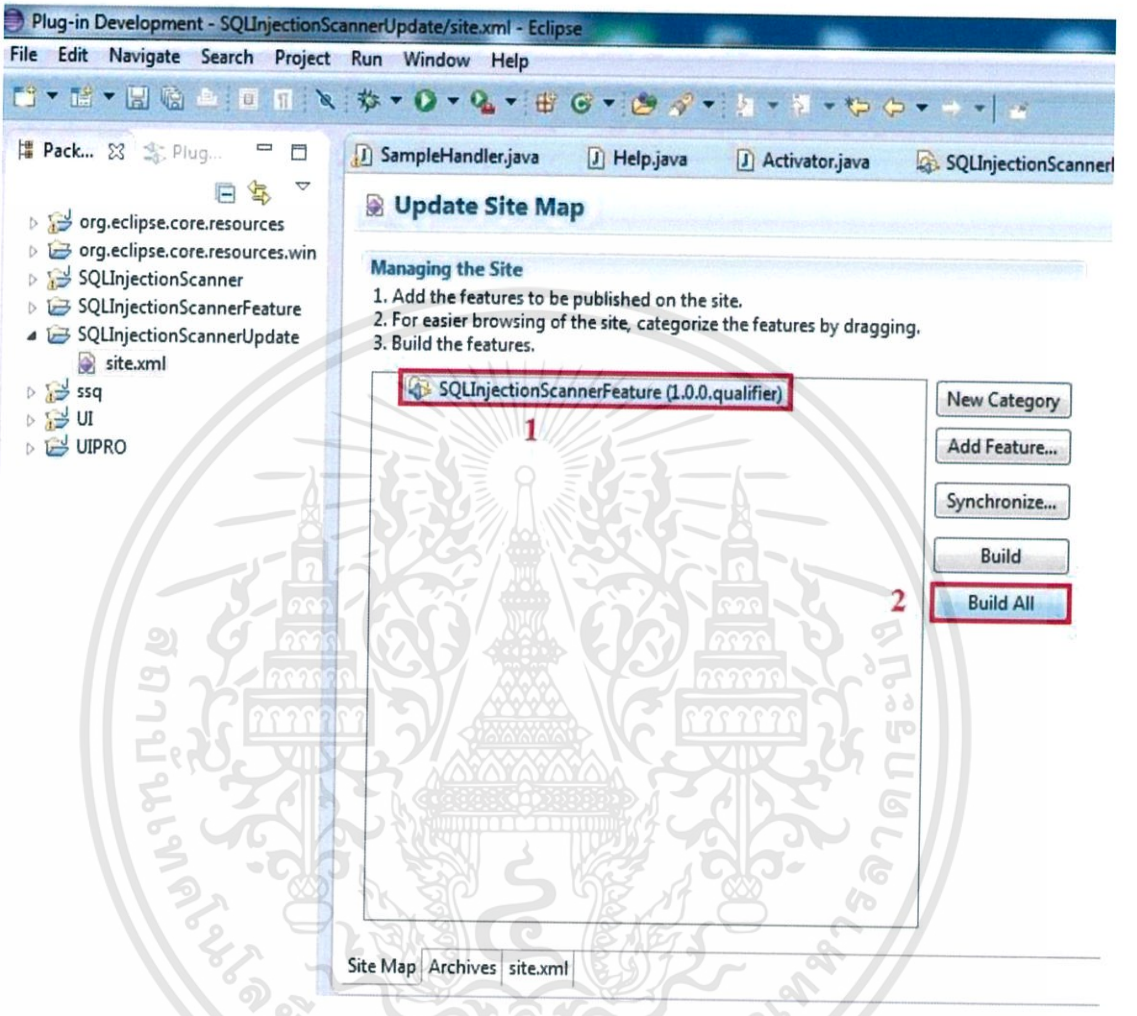
รูปที่ ข.14 หน้าจอแสดงการ Add Feature

9) ในหน้าต่าง Feature Selection เลือก SQLInjectionScannerFeature (1.0.0 qualifier) แล้วกดปุ่ม OK



รูปที่ ข.15 หน้าต่าง Feature Selection

10) ในหน้าต่าง Update Site Map เลือก SQLInjectionScannerFeature (1.0.0 qualifier) แล้วกดปุ่ม Build All



รูปที่ ข.16 หน้าจอแสดงการ Build All

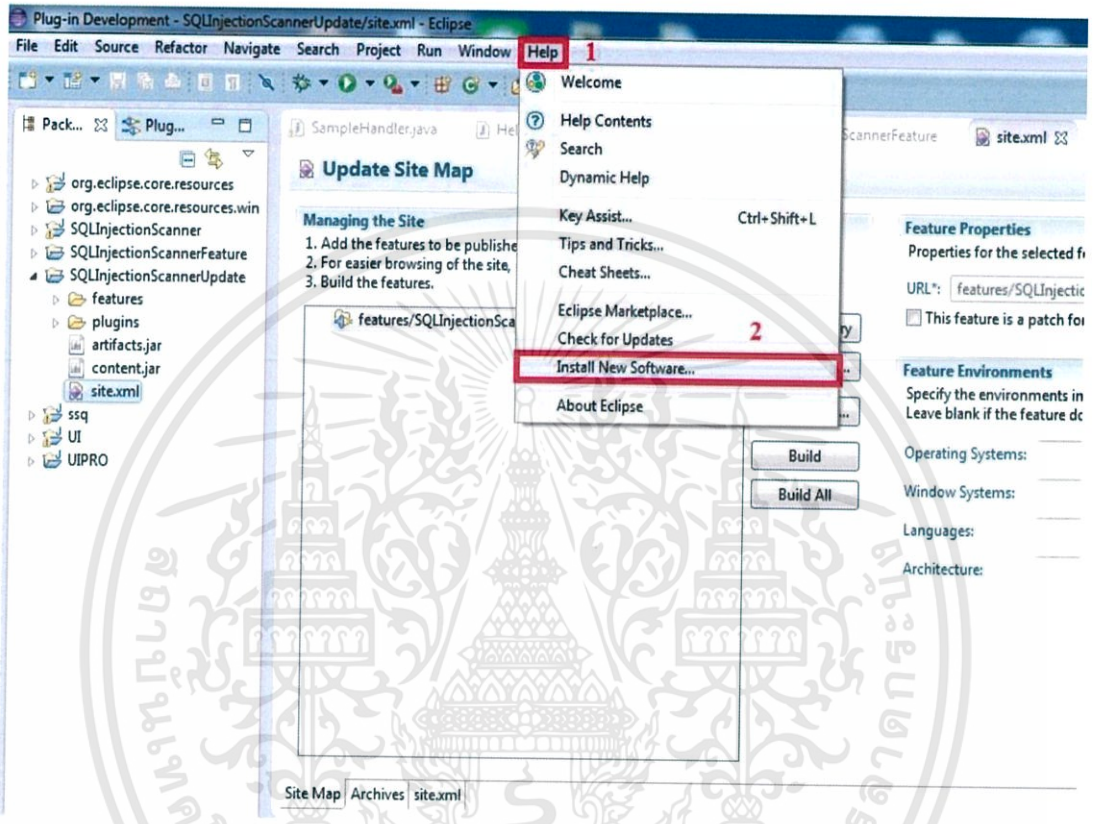
11) เมื่อกดปุ่ม Build All แล้วจะได้ตัวติดตั้งปลั๊กอินเป็นไฟล์ .jar

รูปที่ ข.17 หน้าจอแสดงตัวติดตั้งปลั๊กอินเป็นไฟล์ .jar

ข.3 การติดตั้งโปรแกรมปลั๊กอินบน Eclipse

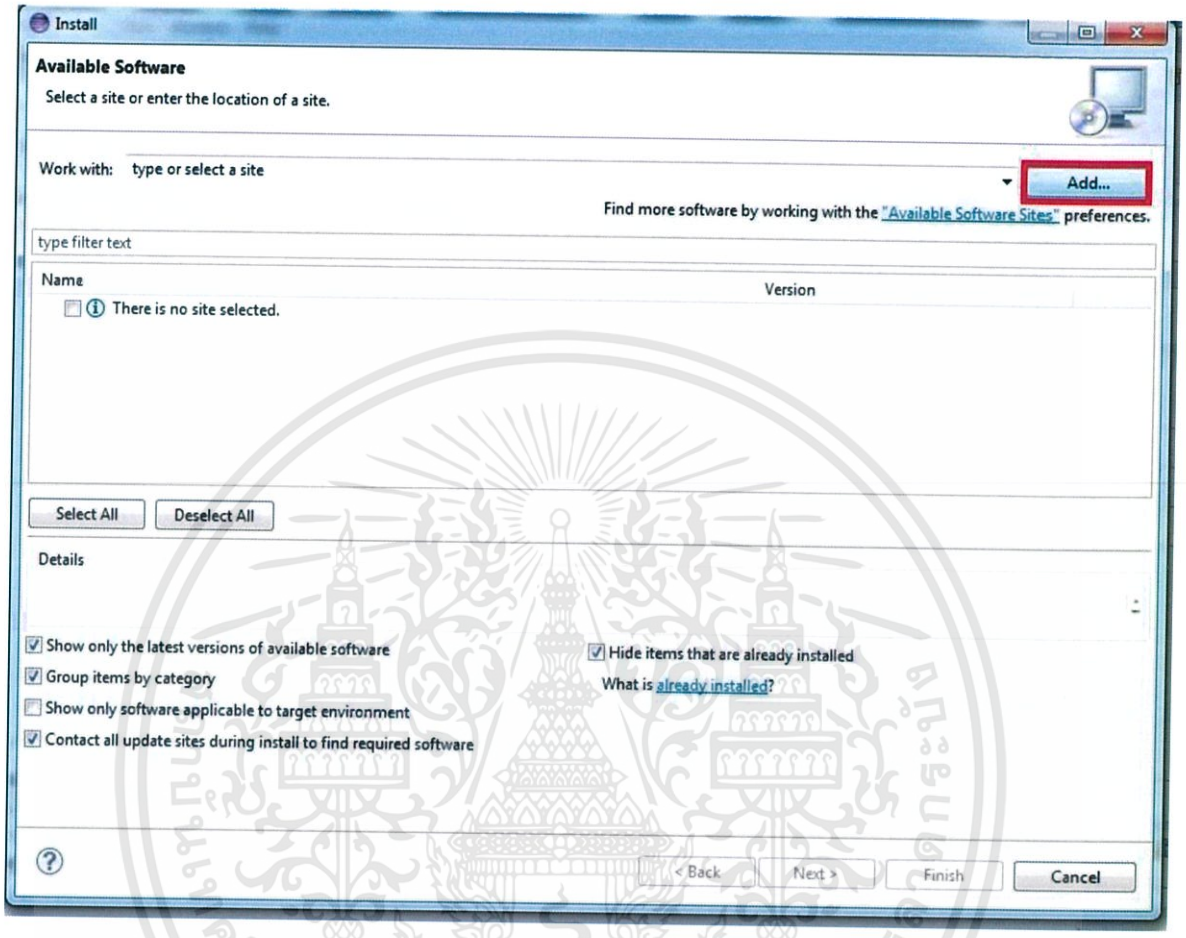
ขั้นตอนการติดตั้งโปรแกรมปลั๊กอินลงบน Eclipse มีวิธีการดังนี้

1) จากหน้าต่างของ Eclipse เลือกเมนู Help > Install New Software… ดังรูปที่ ข.18



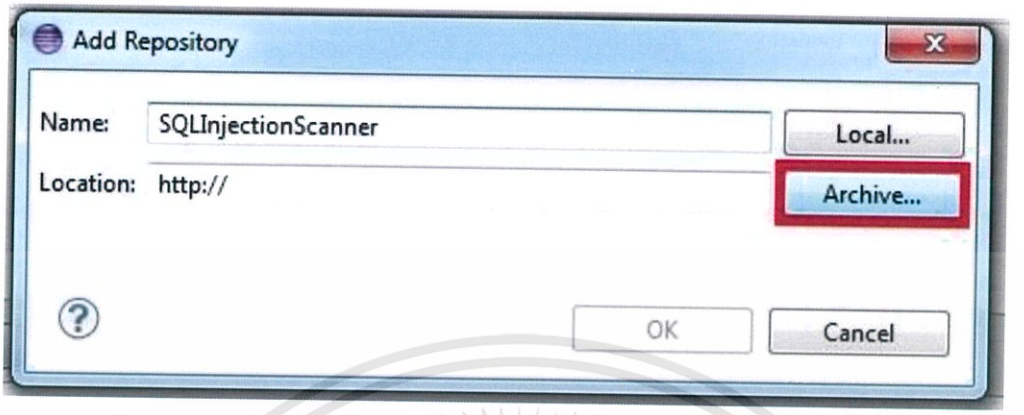
รูปที่ ข.18 หน้าจอแสดงการ Install New Software

2) ในหน้าต่าง Install ของ Eclipse กดปุ่ม Add



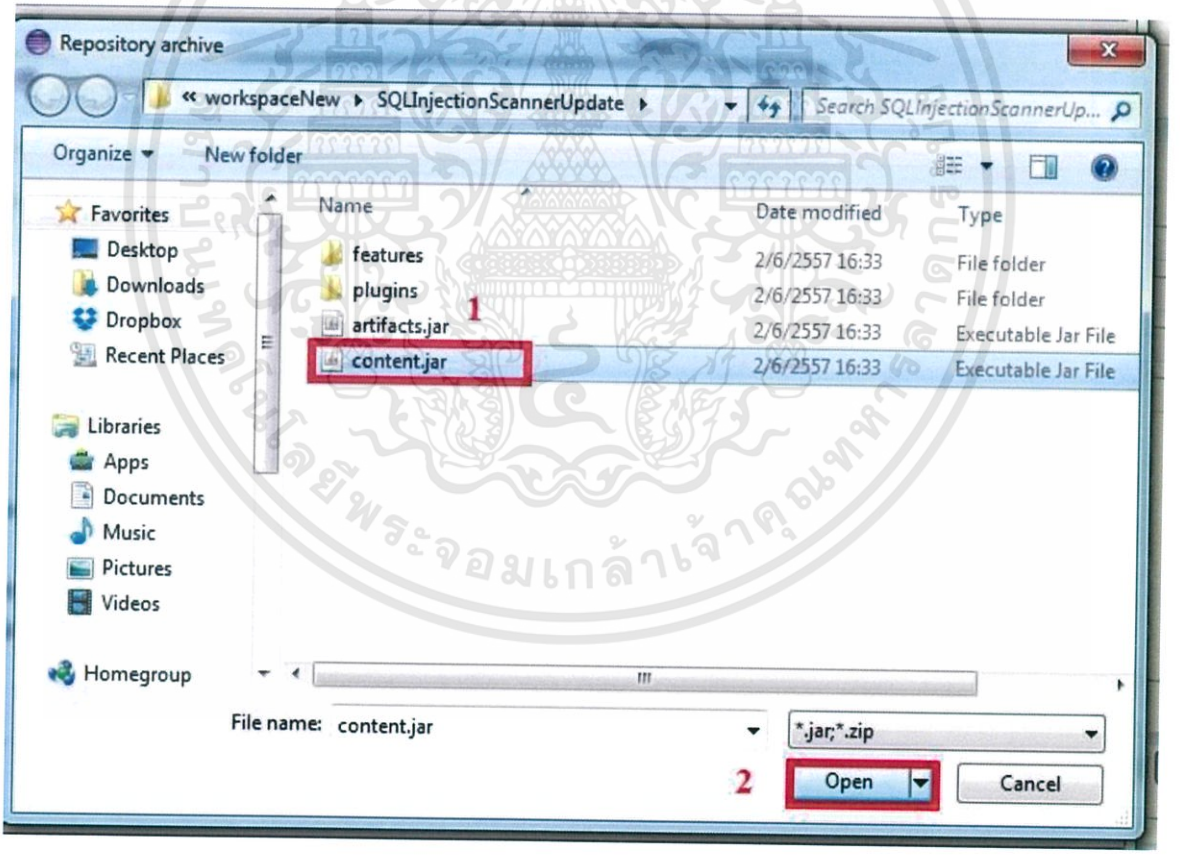
รูปที่ ข.19 หน้าต่างการ Install

3) ในหน้าต่าง Add Repository กดปุ่ม Archive…



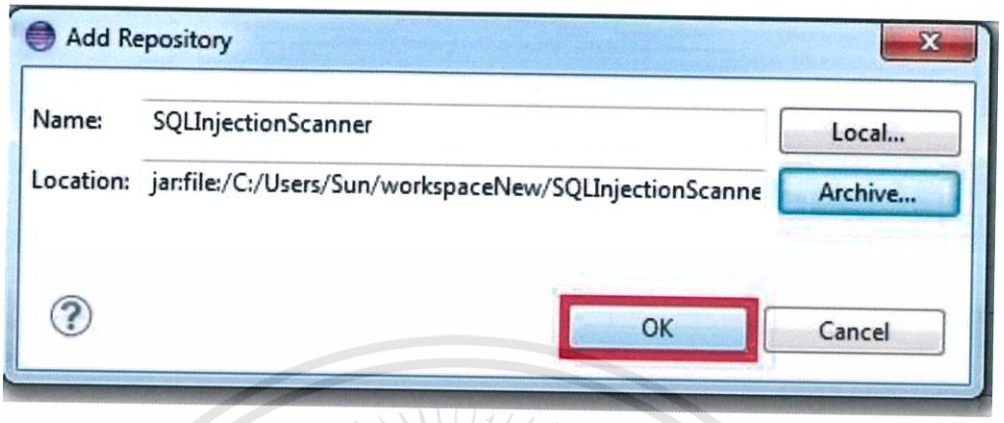
รูปที่ ข.20หน้าต่าง Add Repository

4) จากนั้นเข้าไปที่โฟลเดอร์ของโปรแกรมปลั๊กอิน เลือก content.jar แล้วกดปุ่ม Open



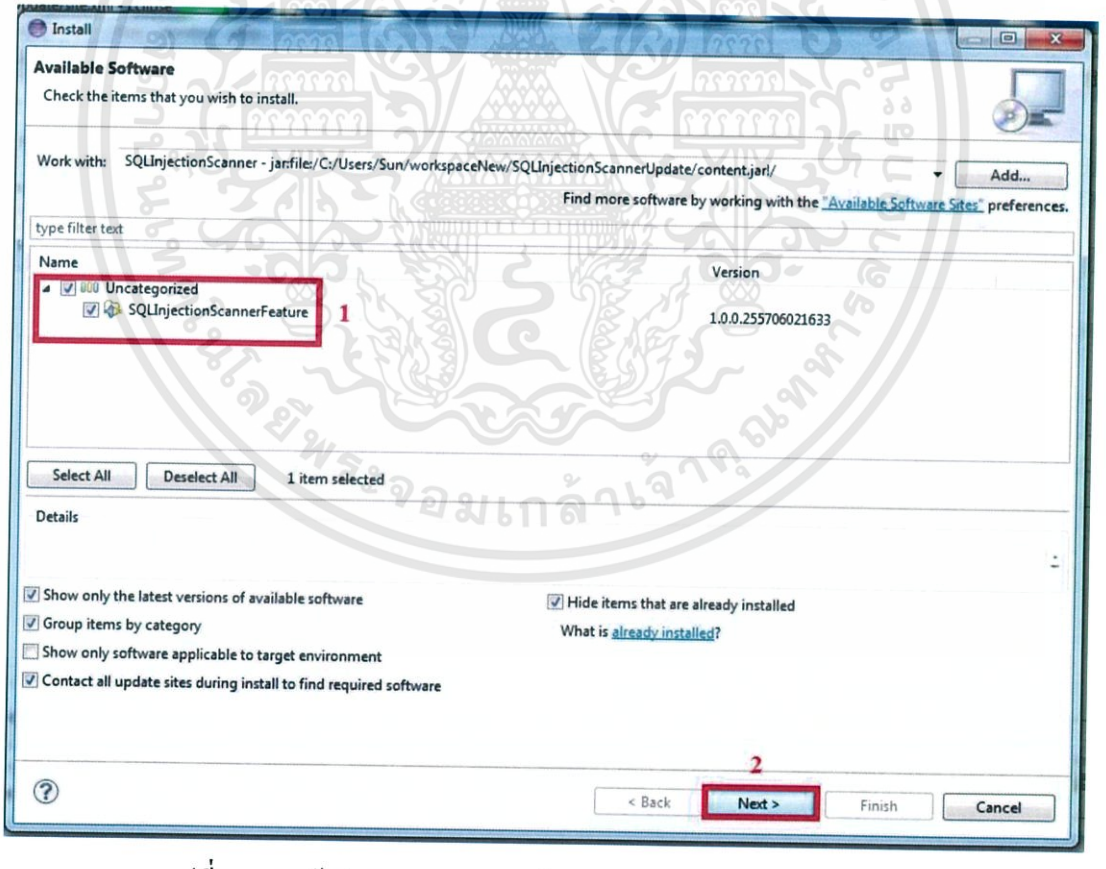
รูปที่ ข.21 หน้าต่างแสดงการเลือก content.jar

5) เมื่อทำข้อ 4) แล้วจะกลับมาที่หน้าต่าง Add Repository อีกครั้ง กดปุ่ม OK



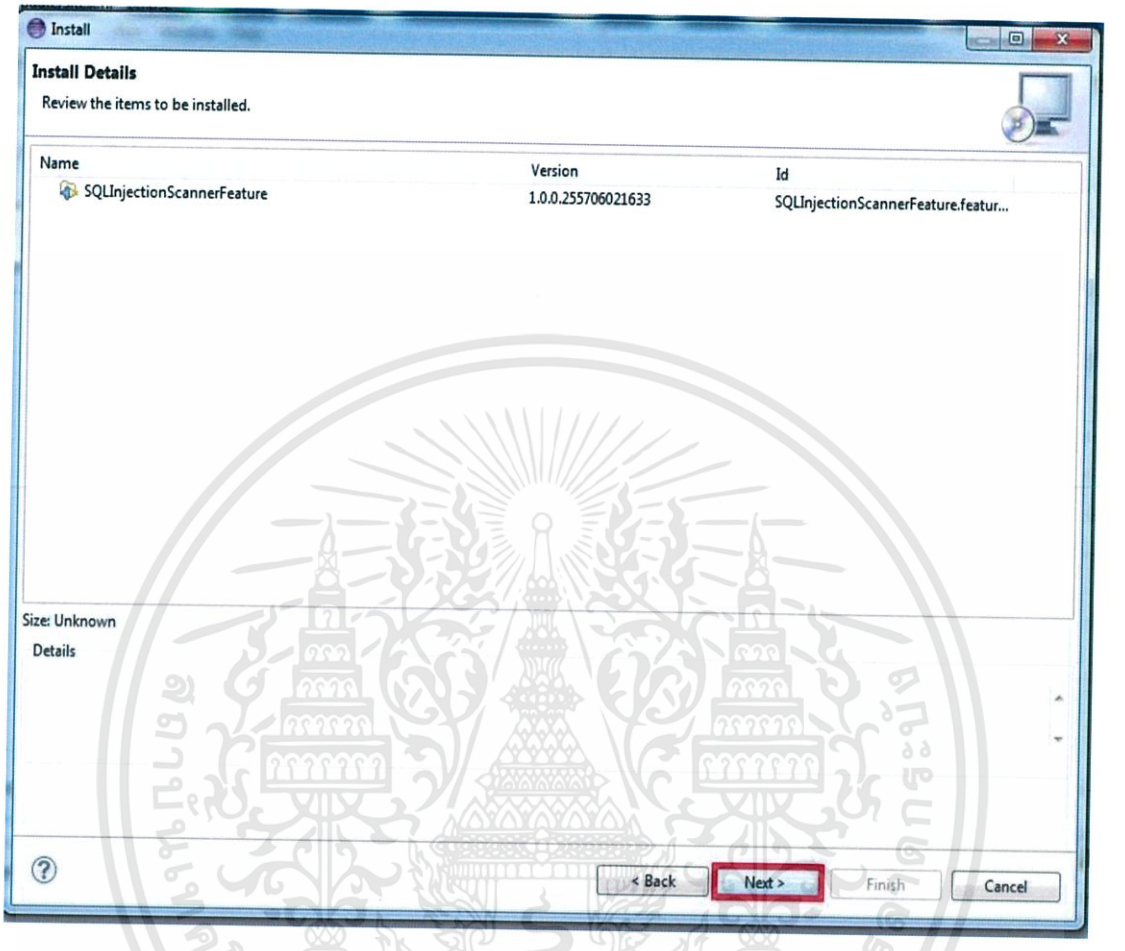
รูปที่ ข.22 หน้าต่าง Add Repository

6) กลับมาที่หน้าต่าง Install คลิกเลือก Uncategorized และ SQLInjectionScannerFeature และกดปุ่ม Next > ดังรูปที่ ข.23



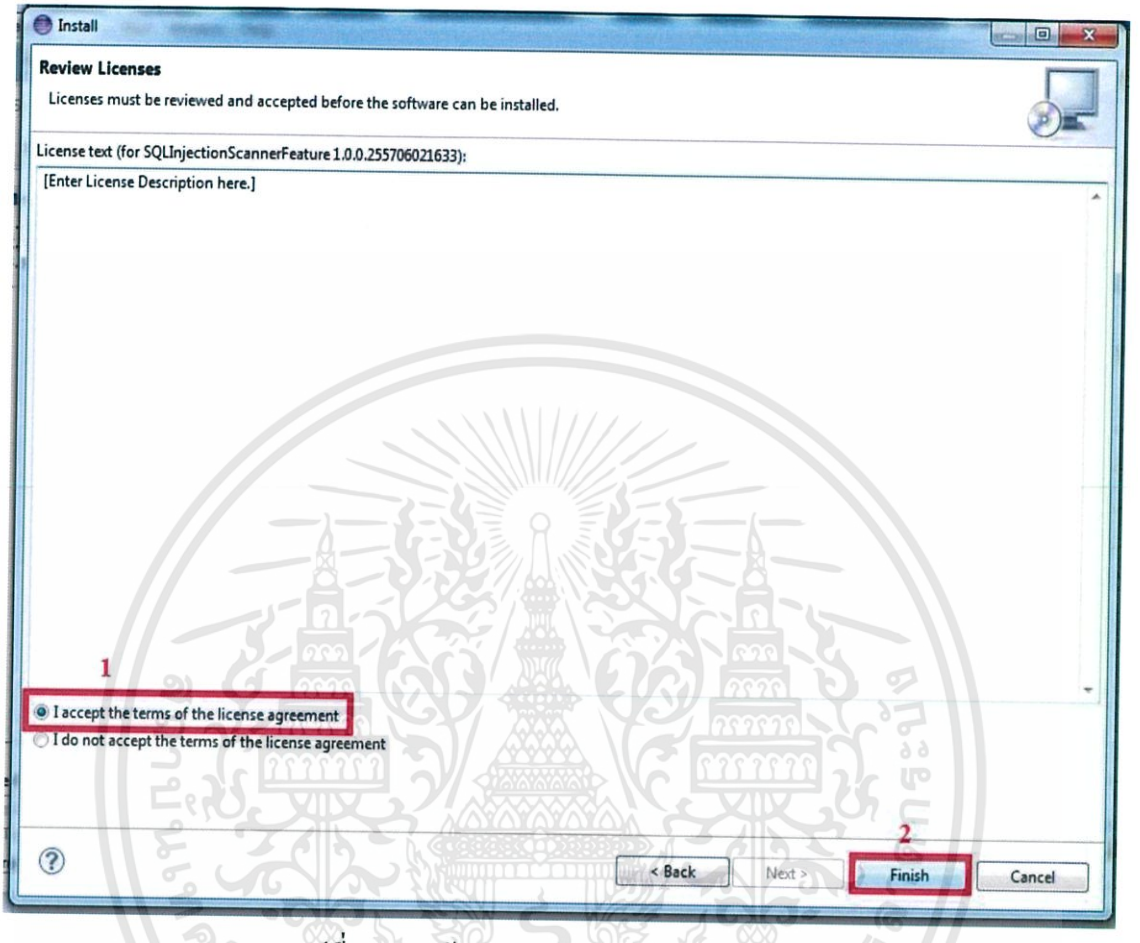
รูปที่ ข.23 หน้าต่าง Install แสดงเลือก SQLInjectionScannerFeature

7) กดปุ่ม Next > อีกครั้งดังรูปที่ ข.24



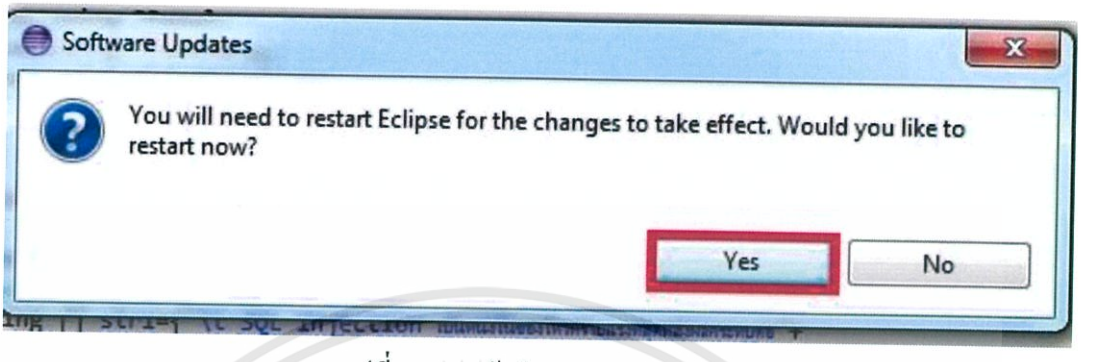
รูปที่ ข.24 หน้าต่างแสดงการ Install

8) จากขั้นตอนที่ 7) คลิกเลือก “I accept the term of the license agreement” แล้วกด Finish



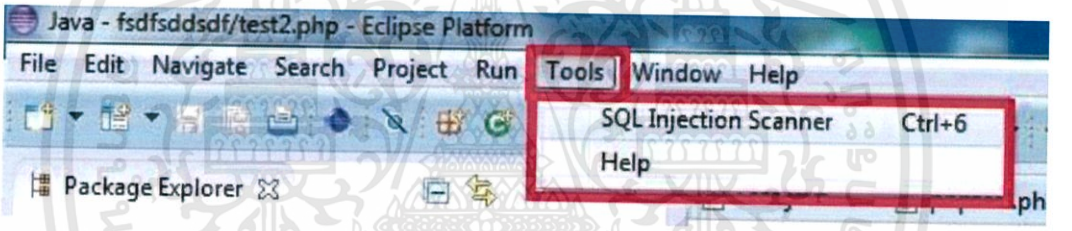
รูปที่ ข.25 หน้าต่างแสดงการยอมรับข้อตกลง

9) หลังจากติดตั้งเรียบร้อยแล้วจะมีหน้าต่าง Software Update ขึ้นมาให้ Restart Eclipse ให้กดปุ่ม Yes



รูปที่ ข.26 หน้าต่าง Restart Eclipse

10) เมื่อ Restart Eclipse หลังจากติดตั้งโปรแกรมปลั๊กอินแล้ว ตัวโปรแกรมจะอยู่บนเมนู Tools ดังรูปที่ ข.21



รูปที่ ข.27 หน้าจอแสดงเมนูของโปรแกรมปลั๊กอินบน Eclipse