**บทที่ 2**

**ทฤษฎีที่เกี่ยวข้อง**

**ทฤษฎีที่เกียวข้องจะประกอบไปด้วย**

**2.1 ทฤษฎีด้านระบบฐานข้อมูล**

2.1.1 ความหมายของระบบฐานข้อมูล

2.1.2 การแบ่งระดับสถาปัตยกรรมของฐานข้อมูล

2.1.3 นิยามพื้นฐานในระบบฐานข้อมูล (Basic Definition)

2.1.4 ข้อดีของการประมวลผลด้วยระบบฐานข้อมูล

**2.2 ทฤษฎีกระบวนการผลิตซอฟต์แวร์**

2.2.1 Prototype Model

**2.3 ทฤษฎีหรือเทคโนโลยีและเทคโนโลยีที่ใช้ในการพัฒนา**

2.3.1 Java

2.3.2 Java Server Page (JSP)

**2.4 งานวิจัยหรือผลงานที่เกี่ยวข้อง**

2.4.1 JSP Search data from Database

2.4.2 JSP Insert data to Database

2.4.3 JSP Edit/Update data in Database

2.4.4 JSP Delete data to Database

* 1. **ทฤษฎีด้านระบบฐานข้อมูล**
     1. **ความหมายของระบบฐานข้อมูล**

ฐานข้อมูล หมายถึง แหล่งที่ใช้สำหรับเก็บรวบรวมข้อมูลซึ่งอยู่ในรูปแฟ้มข้อมูลมารวมไว้ที่เดียวกัน รวมทั้งต้องมีส่วนของพจนานุกรมข้อมูล (data dictionary) เก็บคำอธิบายเกี่ยวกับโครงสร้างของฐานข้อมูล และเนื่องจากข้อมูลที่จัดเก็บนั้นต้องมีความสัมพันธ์ซึ่งกันและกันทำให้สามารถสืบค้น (retrieval) แก้ไข (modified) ปรับปรุงเปลี่ยนแปลงโครงสร้าง ข้อมูล (update) และจัดเรียง (sort) ได้สะดวกขึ้นโดยในการกระทำการดังที่กล่าวมาแล้ว ต้องอาศัยซอฟต์แวร์ประยุกต์สำหรับจัดการฐานข้อมูล ระบบการรวบรวมแฟ้มข้อมูลหลายๆ แฟ้มข้อมูลเข้าด้วยกัน โดยขจัดความซ้ำซ้อนของข้อมูลออก แล้วเก็บข้อมูลไว้ที่ศูนย์กลาง เพื่อการใช้งานร่วมกันในองค์กร ภายในระบบต้องมีส่วนที่เป็นโปรแกรมประยุกต์ที่สร้างขึ้นเพื่อเชื่อมโยงและใช้งานข้อมูลในฐานข้อมูล (database) และจะต้องมีการดูแลรักษาความปลอดภัยของข้อมูลเหล่านั้น มีการกำหนดสิทธิของผู้ใช้งานแต่ละคนให้แตกต่างกัน ตามแต่ความต้องการในการใช้งาน ระบบฐานข้อมูลจะประกอบด้วยแฟ้มข้อมูลจำนวนหลายๆ แฟ้มดังตัวอย่างในรูป แฟ้มข้อมูลเหล่านี้ต้องมีการจัดระบบแฟ้มไว้อย่างดี กล่าวคือ ข้อมูลในแฟ้มข้อมูลเดียวกันต้องไม่มีการซ้ำซ้อนกัน แต่ระหว่างแฟ้มข้อมูลอาจมีการซ้ำซ้อนกันได้บ้าง และต้องเปิดโอกาสให้ผู้ใช้สามารถเข้าถึงข้อมูล และค้นหาได้ง่าย นอกจากนี้ยังสามารถ เพิ่มเติม หรือลบออกได้โดยไม่ทำให้ข้อมูลอื่นเสียหาย

* + 1. **การแบ่งระดับสถาปัตยกรรมของฐานข้อมูล**

การแบ่งระดับดังกล่าวนี้บางครั้งอาจจะเรียกรวมได้ว่า สถาปัตยกรรมของระบบฐานข้อมูลซึ่งจอาศัยลักษณะในการมองภาพรวม (View) ของระบบ เพื่อจำแนกความแตกต่างออกได้เป็น 3 ระดับดังต่อไปนี้

**1. Internal Level** เป็นระดับที่ใช้ในการเก็บข้อมูลจริง ได้แก่ ส่วนที่ทำหน้าที่ในการจัดการเก็บข้อมูลของระบบซึ่งจะครอบคลุมไปถึงการกำหนดชนิดของข้อมูลที่เหมาะสม ตามโครงสร้างที่กำหนดนอกจากนั้งรวมไปถึงการจัดการเกี่ยวกับวิธีการในการเข้าถึงข้อมูลแบบต่างๆ อีกด้วย และในระดับของ Internal Levelนี้จะกล่าวถึงเฉพาะในส่วนของการเข้าถึงข้อมูลของระบบเท่านั้น ทั้งนี้เนื่องจากประสิทธิภาพในการทำงานของระบบไม่ไดขึ้นอยู่กับการออกแบบโครงสร้าง ข้อมูลที่เหมาะสมเพียง

อย่างเดียว แต่ขึ้นอยู่กับวิธีการในการเรียกใช้ข้อมูลเหล่านั้นด้วย ซึ่งวิธีการที่ได้รับความนิยมในระบบฐานข้อมูลทั่วๆไป ได้แก่ Index และ Hashing ซึ่งทั้งสองวิธีนี้ได้นำเอาหลักการทำงานของเซตในรูปแบบของ Search Table มาประยุกต์ในการทำงาน

**2**. **Conceptual Level** เป็นการมองภาพรวมที่เกี่ยวข้องกับข้อมูลทั้งหมด ที่ปรากฏอยู่ในฐานข้อมูลของระบบในเชิงการการออกแบบระบบฐายข้อมูล ตั้งแต่การกำหนดค่า Entity ต่างๆโครงสร้างของข้อมูล ความสัมพันธ์ที่เกิดขึ้นระหว่างข้อมูลนั้นๆ รวมไปถึงกฏเกณฑ์และข้อจำกัดต่างๆของระบบ

**3. External Level** เป็นระดับของข้อมูลที่สนองตอบต่อการใช้งานของผู้ใช้แต่ละคน ซึ่งจะมีการ มองภาพของข้อมูลที่แตกต่างกัน ดังนั้นมุมมองและวิธีการเข้าหาข้อมูลของผู้ใช้แต่ละคนก็แตกต่างกันไปด้วย โดยทั่วไปจะเป็นเพียงการใช้ขอ้มูลกับฐานข้อมูลเป็นบางส่วน แล้วแต่ผู้ออกแบบระบบจะเป็นผู้กำหนด

* + 1. **นิยามพื้นฐานในระบบฐานข้อมูล (Basic Definition)**

**Database** มีคุณลักษณะ 2 ประการ คือ ประการแรกจะต้องเป็นกลุ่มของข้อมูลที่รวมเป็นหนึ่งเดียว (Integrated) และข้อมูลนี้จะถูกผู้ใช้สามารถเรียกใช้ร่วมกันได้ (Share)

**Data** คือความเป็นจริง (Fact) ที่เกี่ยวข้องกับบุคคลสถานที่เหตุการณ์ หรือสิ่งของต่างๆ ซึ่งสามารถนับจำนวนได้

**Information** คือข้อมูลที่ถูกจัดรวบรวมให้อยู่ในรูปที่สามารถจะนำไปใประกอบการตัดสินใจอย่างใดอย่างหนึ่งได้

**Entity** คือสิ่งใดสิ่งหนึ่ง ได้แก่ ชื่อของบุคคล สถานที่ สิ่งของ หรือการกระทำที่ต้องการจัดเก็บนั้นไว้ Entity ที่ใช้สำหรับแสดงความสัมพันธ์กัน ระหว่างข้อมูลในระบบ เช่น Employee, Student, Teacher เป็นต้น สัญลักษณ์ที่ใช้เขียนแทน Entity ได้แก่ รูปสี่เหลี่ยมผืนผ้า

**Attribute** คือรายละเอียดของข้อมูลใน Entity หนึ่งๆ ที่ใช้แสดงลักษณะและคุณสมบัติของ Entity ที่ถูกอ้างถึง เช่น Attribute ของ Student ได้แก่ เลขประจำตัว, ชั้น หรือแผนกที่สังกัด เป็นต้น ค่า Attribute คือค่าที่เก็บอยู่ภายใน Entity นั่นเอง สัญลักษณ์ที่ใช้เขียนแทน ได้แก่ รูปวงรี

**Entity Set** คือ Entity หลายๆ ตัวที่มีค่า Attribute เหมือนกัน และสามารถนำมารวมกันในรูปของ Table เพื่อสะดวกในการเข้าถึงข้อมูลกลุ่มดัลกล่าว เช่น Entity Set ของ Student เป็นต้น

**Field** คือสิ่งที่เกิดจากการรวมตัวของข้อมูลที่เล็กที่สุดภายในคอมพิวเตอร์ที่เรียกว่า บิต (Bit) นำมาประกอบกันจะได้ข้อมูลที่รียกว่า ไบต์ (byte) หรือตัวอักขระ (Character) หากนำอักขระมาประกอบกันเป็นกลุ่ม ก็จะได้ข้อมูลที่ขยายตัวออกเป็นรูปแบบใหม่ๆ ที่เรียกว่า Field และอาจกล่าวได้ว่า ส่วนของ Field ก็จะได้แก่แอททริบิวต์

**File** คือการนำ Record ชนิดเดียวกัน ที่ถูกนำมารวมกันเป็นหมวดหมู่ ข้อมูลที่อยู่ภายในไฟล์จะสามารถมองได้เป็นอาเรย์ 2 มิติ นั่นคือ ในรูปของแถวซึ่งแสดงถึงจำนวน Record และ Column ซึ่งแทนค่าของ Attribute แต่ละตัว

**Association** คือสัญลักษณ์แสดงความสัมพันธ์ (Relationships) กันระหว่าง Entity ซึ่งจะเกิดขึ้นได้กับEntity ตั้งแต่สองตัวขึ้นไป

* + 1. **ข้อดีของการประมวลผลด้วยระบบฐานข้อมูล**
* ลดความซ้ำาซ้อนของข้อมูล(Minimal Data Redundancy)การจัดเก็บข้อมูลในลักษณะเป็นแฟ้มข้อมูล อาจทำให้ข้อมูลประเภทเดียวกันถูกเก็บไว้หลายๆ แห่งทำให้เกิดความซ้ำซ้อนของข้อมูลขึ้นได้ ดังนั้นการนาข้อมูลรวมมาเก็บไว้ในระบบฐานข้อมูลจะช่วยลดปัญหาความซ้ำซ้อนของข้อมูลได้
* หลีกเลี่ยงความขัดแย้งของข้อมูลได้(Consistency of Data)การจัดเก็บข้อมูลในลัษณะเป็นแฟ้มข้อมูลโดยที่ข้อมูลเป็นเรื่องดียวกันอาจมีอยู่ในหลายแฟ้มซึ่งก่อให้เกิดความขัดแย้งของข้อมูลขึ้นได้ ทั้งนี้อาจเนื่องมาจากการแก้ไขข้อมูลที่แฟ้มแห่งหนึ่ง แต่มิได้แก้ไขข้อมูลเรื่องเดียวกันที่อยู่ในไฟล์อื่นๆ ทำให้ข้อมูลนั้นๆ แตกต่างกันได้
* จำกัดความผิดพลาดในการป้อนข้อมูลให้น้อยที่สุด(Data Integrity)บางครั้งความผิดพลาดของข้อมูลอาจเกิดขึ้นจากการป้อนข้อมูลที่ไม่ถูกต้องเข้าสู่ระบบดังนั้นในระบบจัดการฐานข้อมูลจึงจำเป็นที่จะต้องกำหนดกฏเกณฑ์ในการรับข้อมูลจากการปัอนของผู้ใช้ เพื่อรักษาความถูกต้องของข้อมูลให้มากที่สุดเท่าที่จะทำได้
* สามารถใช้ข้อมูลร่วมกันได้(Sharing of Data)เนื่องจากระบบฐานข้อมูลเป็นการจัดเก็บข้อมูลไว้ในที่เดียวกัน เมื่อผู้ใช้ต้องการเรียกใช้ข้อมูลจากแฟ้มข้อมูลที่แตกต่างกัน ก็จะสามารถทำได้โดยง่าย
* สามารถกำหนดความเป็นมาตรฐานเดียวกันได้(Enforcement of Standard)การเก็บข้อมูลไว้ด้วยกันจะสามารถกำหนด และควบคุมความมีมาตรฐานของข้อมูลให้เป็นไปในทิศทางเดียวกันได้ ดังนั้นจึงทำให้ระบบเกิดความเชื่อมั่นมากยิ่งขึ้น
* สามารถกำหนดระบบความปลอดภัยของข้อมูลได้(Security and Privacy Control)เนื่องจากระบบจะทำการกำหนดระดับของผู้ใช้แต่ละคน ตามลำดับความสำคัญของผู้ใช้ดังนั้นจึงสามารถที่จะควบคุมและดูแลความปลอดภัยของข้อมูลภายในระบบได้ดียิ่งขึ้น
* ข้อมูลมีความเป็นอิสระ(Data Independence)ระบบฐานข้อมูลจะทำหน้าที่เป็นตัวเชื่อมโยงกับโปรแกรมประยุกต์ ที่ทำงานกับข้อมูลโดยตรงการแก้ไขข้อมูล เช่น ต้องการเปลี่ยนรหัสไปรษณีย์จากเลข4หลัก เป็นเลข5หลัก ก็จะทำการแก้ไขข้อมูลที่เป็นรหัสไปรษณ๊ย์เฉพาะโปรแกรมที่เรียกใช้รหัสไปรษณีย์เท่านั้น ส่วนโปรแกรมอื่นจะเป็นอิสระต่อการเปลี่ยนแปลงนี้
  1. **ทฤษฎีกระบวนการผลิตซอฟต์แวร์**
     1. **Prototype Model**

**Prototype model** คือกระบวนการพัฒนาระบบซอฟต์แวร์โดยสร้างตัวต้นแบบขึ้นมาเพื่อที่จะให้ผู้ใช้งานระบบได้เห็นถึงรูปร่างลักษณะของระบบที่ตนเองต้องการ การสร้างต้นแบบ เป็นพัฒนาการที่รวดเร็วและเป็นการทดสอบการทำงานของแบบจำลอง (Model) หรือต้นแบบของระบบงานใหม่ ในการโต้ตอบและกระบวนการทำซ้ำประโยคคำสั่งในโปรแกรมที่เรียกว่า การวนรอบ (Interactive หรือ Iterative) โดย นักวิเคราะห์ระบบและผู้ใช้ การทำต้นแบบสามารถทำให้กระบวนการพัฒนาเร็วและง่ายขึ้น โดยเฉพาะอย่างยิ่งโครงการที่ความต้องการของผู้ใช้นั้นยากแก่การเข้าใจอย่าง ชัดเจน การสร้างต้นแบบแบ่งออกเป็น 2 ประเภทใหญ่ๆ คือ

**1. Operational Prototype** เป็นต้นแบบที่สามารถทำงานและเข้าถึงแฟ้มข้อมูลได้จริง สามารถรับข้อมูลเข้าและนำข้อมูลเหล่านั้นมาคำนวณเปรียบเทียบและแสดงผลลัพธ์จริง

**2. Nonoperational Prototype** เป็นต้นแบบจำลองโดยทั่วไปมักรวมถึงข้อมูลต่างๆ ที่นำเข้าและข้อมูลที่แสดงผลลัพธ์ออกมา ข้อมูลที่แสดงผลลัพธ์ออกมาจะอยู่ในรูปแบบของรายงานและทางจอภาพ ข้อมูลที่นำเข้าจะแสดงถึงวิธีที่ข้อมูลถูกจับ ผู้ใช้คำสั่งอะไรต้องเข้าไป และวิธีเข้าถึงระบบแฟ้มข้อมูลอื่นๆ

                การพัฒนาโดยการทำต้นแบบจะใช้เวลาน้อยกว่าวิธีศึกษาแบบดั้งเดิม เป้าหมายของการทำต้นแบบก็เพื่อการสร้างระบบขนาดเล็กที่ไม่แพงแต่รวดเร็ว และเพื่อการเพิ่มหรือทดแทนระบบเชิงปฏิบัติงานแบบเต็มขนาด ขณะที่ผู้ใช้ได้ทำงานร่วมกับระบบก็สามารถให้คำแนะนำที่สามารถเป็นประโยชน์ ให้กับต้นแบบอื่นๆ ได้ แม้ว่าต้นแบบที่จะไปเป็นระบบขั้นสุดท้ายต้องถูกทิ้งไป แต่ความรู้ที่ได้จากการสร้างต้นแบบนั้นก็ยังคงนำไปสร้างให้เป็นระบบจริงขึ้น มาใหม่อีกได้

#### การเลือกใช้การพัฒนาแบบการสร้างต้นแบบควรเลือกใช้ในกรณีต่อไปนี้

                1. ผู้ใช้ยังไม่ทราบความต้องการระบบที่แน่ชัด

                2. ความต้องการของผู้ใช้มีการเปลี่ยนแปลงอย่างต่อเนื่อง

                3. ผู้ใช้มีประสบการณ์เกี่ยวกับระบบที่กำลังพัฒนาอยู่น้อยมาก

                4. มีความจำเป็นจะต้องพัฒนาระบบในระยะเวลาอันสั้นและใช้ค่าใช้จ่ายน้อย

                5. การดำเนินการหรือการประมวลผลไม่มีกฎเกณฑ์แน่ชัด

**กระบวนการสร้างต้นแบบ (Prototyping Process)**

                การ สร้างต้นแบบจะไม่พัฒนาทั้งระบบทีเดียวทั้งหมด แต่จะพัฒนาโดยใช้ต้นแบบ ซึ่งประกอบด้วยส่วนต่างๆ ของระบบใหม่แต่จำลองให้มีขนาดเล็กเพื่อให้ผู้ใช้ได้ทดลองใช้ก่อน และให้ข้อเสนอแนะเพื่อใช้ในการปรับปรุงต้นแบบนี้ให้เหมาะสมต่อไป กระบวนการนี้จะปฏิบัติการซ้ำๆ จนกระทั่งผู้ใช้ยอมรับระบบ จึงจะนำต้นแบบนั้นไปพัฒนาให้เต็มรูปแบบต่อไป

**ขั้นตอนของวิธีการสร้างต้นแบบมี 4 ขั้นตอนดังนี้**

**1. กำหนดความต้องการ** เป็นการหาความต้องการพื้นฐานของผู้ใช้ระบบ นักออกแบบระบบจะต้องมีเวลาเพียงพอในการศึกษาหาความต้องการด้านสารสนเทศพื้นฐานของผู้ใช้

**2. ออกแบบต้นแบบ** นักพัฒนาระบบสามารถใช้เครื่องมือในการพัฒนาต้นแบบออกแบบระบบ เพื่อให้เกิดความรวดเร็ว

**3. นำต้นแบบไปใช้** ผู้ใช้จะนำต้นแบบไปทดลองใช้ว่าสามารถตอบสนองความต้องการของผู้ใช้เพียงใด โดยผู้ใช้สามารถให้ข้อคิดเห็นเกี่ยวกับปัญหาหรือข้อแนะนำในการปรับปรุงต้นแบบได้

**4. การปรับแต่งต้นแบบ** เป็นการนำความเห็นของผู้ใช้มาปรับปรุงต้นแบบให้มีประสิทธิภาพมากขึ้น ขั้นตอนนี้จะเกิดซ้ำไปซ้ำมาจนกระทั่งผู้ใช้เกิดความพอใจ แล้วจึงจะสามารถนำต้นแบบไปใช้งานได้ (Operational Prototype)

* 1. **ทฤษฎีหรือเทคโนโลยีและเทคโนโลยีที่ใช้ในการพัฒนา**
     1. **Java**

Java เป็นภาคภาษาคอมพิวเตอร์ที่ได้รับการออกแบบ สำหรับการใช้บนอินเตอร์เน็ต โดยมีส่วนของการ "look and feel" แบบภาษา C++ แต่ง่ายกว่าการใช้ C++ และสามารถสร้างมุมมองโดยโปรแกรมได้ Java สามารถใช้ในการสร้างการประยุกต์แบบสมบูรณ์ ซึ่งสามารถเรียกใช้ได้เฉพาะเครื่องคอมพิวเตอร์ หรือการกระจายระหว่างเครื่องแม่ข่ายกับลูกข่ายในระบบเครือข่าย และสามารถสร้างโมดูลการประยุกต์ขนาดเล็กหรือ applet สำหรับเป็นส่วนของเว็บเพจ applet ทำให้มีความเป็นได้ในด้านการตอบสนองของผู้ใช้กับเว็บเพ็จ

**คุณลักษณะคือ**

1. โปรแกรมมีขนาดเล็กในระบบเครือข่าย การคอมไพล์จะแปลงโปรแกรมเป็น Java bytecode ซึ่งสามารถเรียกใช้งานได้ทุกที่ภายในเครือข่าย Java virtual machine เป็นตัวแปร bytecode ให้เป็นโปรแกรมเพื่อใช้งานบนเครื่องคอมพิวเตอร์ มีความหมายว่า platform ที่แตกต่างกันของคอมพิวเตอร์ สามารถใช้โปรแกรมนี้ได้
2. คำสั่งเป็นแบบ "Robust" มีความหมายว่า อ็อบเจคของ Java ไม่มีการอ้างอิงข้อมูล หรืออ็อบเจคจากภายนอกซึ่งแตกต่างจาก C++ และภาษาอื่น ๆ เป็นการทำให้มั่นใจไม่มีการเก็บตำแหน่งของข้อมูล ในโปรแกรมประยุกต์อื่น หรือในระบบปฏิบัติการที่ทำให้โปรแกรมไม่ทำงาน Java virtual machine ทำการตรวจสอบแต่ละอ็อบเจคที่ใช้ในโปรแกรม
3. Java เป็นเหมือนกับอ็อบเจคหนึ่ง สามารถได้รับประโยชน์จาก class หรือคำสั่ง เนื่องจากอ็อบเจคมีคุณสมบัติเป็น "นาม" ซึ่งทำให้ติดต่อกับผู้ใช้ได้ ในขณะนี้ภาษาดั้งเดิมมีคุณสมบัติเป็น "กริยา" ดังนั้น method จะได้การรับรู้เป็นความสามารถของอ็อบเจคหรือพฤติกรรม
4. การประมวลผลทำที่เครื่องลูกข่าย ดังนั้น Java applet มีคุณลักษณะในการออกแบบให้ทำงานได้เร็ว
5. Java ง่ายกว่า C++ โดยเปรียบเทียบ

Java ได้รับการแนะนำโดย Sun Microsystems ในปี 1995 และทำให้เกิดทัศนคติการตอบสนองของเว็บ ทำให้ web browser รายหลักได้รวม Java virtual machine เป็นส่วนหนึ่งของ browser ผู้พัฒนาระบบปฏิบัติเกือบทั้งหมดได้ร่วม Java complier เป็นส่วนหนึ่งของผลิตภัณฑ์

Java virtual machine รวมถึงตัวเลือกคือ Just-in-time complier ซึ่งเป็น complier แบบไดนามิคในการคอมไพล์ byte code เป็นคำสั่งที่ประมวลผลได้ เป็นตัวเลือกในการแปร bytecode ในหลาย ๆ กรณี dynamic JIT สามารถคอมไพล์ได้เรียกว่าการแปรของ Java virtual machine

JavaScript เป็นภาษาที่พัฒนาโดย Netscape ซึ่งเป็นตัวแปร (Interpreter) ภาษาระดับสูงและง่ายกว่าการเขียนด้วย Java แต่ขาดความกะทัดรัดเหมือน Java และความเร็วไม่มาก

* + 1. **Java Server Page (JSP)**

Java Server Page (JSP) เป็นเทคโนโลยี่ไว้สำหรับพัฒนา Application บนเว็บไซต์ ในรูปแบบของ Server และ Client แสดงผลและโต้ตอบกับ User Interface ผ่าน Web Browser อย่างพวก IE , Chrome , Firefox และอื่น ๆ โดยอาศัยการทำงานร่วมกับพวก Client Tags เช่นพวก HTML / JavaScript / CSS และพวก jQuery ให้ได้ผลลัพธ์ตามที่ต้องการ ซึ่ง JSP ก็เหมือนกับโปรแกรม ASP , PHP และ .Net รูปแบบการทำงานนั้นไม่แตกต่างกัน แต่แตกต่างกันตรงที่ JSP เป็น Subset ของภาษา Java โดยรูปแบบการเขียนนั้นจะใช้รูปแบบคำสั่งและชุด SDK ของ Java และใน JSP จะมีนามสกุลของไฟล์เป็น .jsp แต่การพัฒนา Website หรือ Web Application ด้วยภาษา JSP จะสามารถแยกย่อยได้อีกประมาณ 2-3 รูปแบบ และแต่ล่ะรูปแบบการวางโครงสร้าง และรูปแบบการเขียนก็ต่างกัน แต่พื้นฐานแล้วยังใช้ Syntax ภาษา Java เหมือนกัน โดยสรุป ๆ แล้วแยกออกเป็น 3 รูปแบบคือ

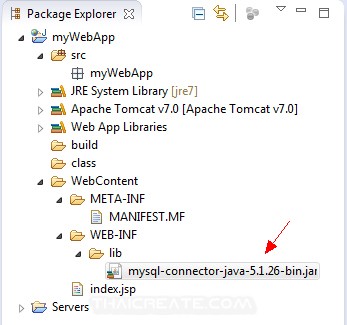
* Scriptlets คือการเขียน JSP ร่วมกับ HTML ให้มองว่าเหมือนพวก ASP และ PHP แต่จะใช้นามสกุลไฟล์ .jsp ส่วน Code นั้นก็สามารถเขียนร่วมกับ HTML หรือสกลับกันไปมาได้
* Servlet เป็นการเขียน JSP ขั้นสูงขั้น ซึ่งจะแยกในส่วนของ Interface (HTML) กับส่วนของ Code ที่เป็นภาษา Java ออกจากกัน ให้มองถึงรูปแบบการเขียน ASP.Net ที่แยก .ASPX และ .VB ไว้คนล่ะไฟล์
* Framework การเขียน Web Application ด้วย JSP จะมี Framework มารองรับการเขียนหลาย ๆ ตัวเช่น Spring , Struts และ Hibernate โดยรูปแบบการเขียนจะเป็น Pattern ในรูปแบบของ MVC

ส่วนจะเลือกใช้ตัวไหนนั้นก็ขึ้นอยู่กับความต้องการ เช่นถ้าเขียนเว็บทั่ว ๆ ไปก็อาจจะใช้แค่ Scriptlets แต่ถ้าเป็น Web ที่ขนาดใหญ่ขึ้นต้องการความเป็นระเบียบเทียบร้อยก็อาจจะเลือกใช้ Servlet และใน Project ขนาดใหญ่เขียนกันหลาย ๆ คนก็ควรจะเลือกใช้ Framework ซึ่งเหมาะอย่างยิ่งสำหรับการเขียน Web Application ทขนาดใหญ่ และ แยกย่อยออกเป็นหลาย ๆ ระบบ

**JSP** เป็นทางเลือกสำหรับการพัฒนา **Application** บนเว็บไซต์ เหตุผลแรกก็คือ JSP สามารถใช้งานได้ฟรี และสามารถรองรับได้ทุก **Platform** ไม่ว่าจะเป็น **Windows , Linux และ iOS** รวมทั้ง Software อื่น ๆ ที่จะเป็นต่อการใช้งานและการพัฒนาก็ฟรีเช่นเดียวกัน อย่างเช่น **Apache Tomcat , Eclipse IDE** รวมทั้งด้านทรัพยากรสำหรับการพัฒนา เนื่องจาก **Java** เป็นภาษาที่มีขนาดใหญ่มาก และ สามารถแบ่งย่อยได้อีกมากมายหลากหลาย และมีตัวเลือก **Library , Class , Framework , Plugin** ให้เลอกใช้มากมาย แทบได้ว่าจะทำอะไรก็มีมาให้หมด โดยบางครั้งก็ไม่ต้องขียนเองให้ยุ่งยาก เพียงแค่หา **Library** มาใช้ก็สามารถนำมาใช้กับ **Web** ของเราได้ และในด้าน **Performance** ต้อง ยอมรับว่า **JSP** ที่พัฒนาด้วยภาษา Java มีประสิทธิภาพการทำงานสูงมาก ฉะนั้นเราจะเห็นว่าเว็บไซต์ใหญ่ ๆ ที่ต้องการความเสถียรสูง เช่นพวก **Bank** จะเลือกใช้ JSP ในการพัฒนา **Application** ซะส่วนใหญ่ และจากที่ผมเองได้ลองเขียนดูแล้ว และเปรียบเทียบระหว่าง **PHP , ASP.Net** และ JSP ต้องยอมรับว่า JSP มีการจัดการจัดสรรพวกค่าตัวแปรใน Web Server ดีมาก เช่น เมื่อสร้างตัวแปรแล้ว สามารถนำไปใช้งานได้ในระดับ **Page , Session และ Application** ได้อย่างง่ายดาย ซึ่งในนส่วนนี้เองขะช่วยลดการทำงานที่ซ้ำซ้อนของ **Server** ได้อย่างดี ซึ่งเหมาะอย่างยิ่งกับ **Application** ที่รองรับการทำงานขนาดใหญ่ได้

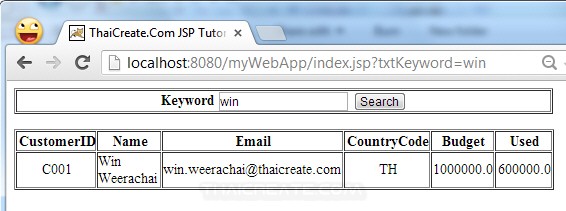
* 1. **งานวิจัยหรือผลงานที่เกี่ยวข้อง**
     1. **JSP Search data from Database**

ตัวอย่างการเขียน JSP ติดต่อกับ Database และทำการค้นหา (Search) ข้อมูลจาก Database จากนั้นหลังจากที่ได้ผลลัพธ์แล้วจะทำการ Loop ข้อมูลใน HTML Table เพื่อแสดงผลลัพธ์ออกทางหน้าจอของไฟล์ JSP

**Example ตัวอย่างการเขียน JSP เพื่อค้นหา (Search) ข้อมูลจาก Database มาแสดง**  


ภาพที่ 2.1 แสดงตัวอย่างการเชื่อมต่อ **MySQL Connector**

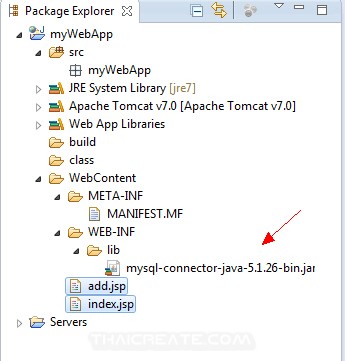
เนื่องจากใช้ **Database** ของ **MySQL** จึงเลือกใช้ **MySQL Connector** ด้วยการเรียกไฟล์ jarโครงสร้างของ Table และ Dataภาพที่ 2.2 แสดงตัวอย่างข้อมูลในฐานข้อมูล MySQL



ภาพที่ 2.3 แสดงตัวอย่างการแสดงผลของการค้นหา

* + 1. **JSP Insert data to Database**

ตัวอย่างการเขียน JSP เพื่อติดต่อกับ Database และตัวอย่างการ Insert ข้อมูลจาก JSP ลงใน Database โดยในขั้นแรกจะใช้การออกแบบ Form เพื่อรับค่า Input ค่าต่าง ๆ จาก User และหลังจากที่ User ทำการ Submit ข้อมูลเหล่านั้นแล้ว เราก็จะใช้ JSP อ่านข้อมูลจาก Input แล้วค่อยนำไป Insert ใน Database

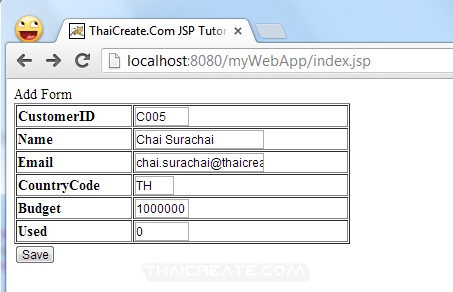
Example ตัวอย่างการเขียน JSP เพื่อ ทำการ Insert เพิ่มข้อมูลลงใน Database  


ภาพที่ 2.4 แสดงตัวอย่างการไฟล์ทั้งหมด

เนื่องจากใช้ Database ของ MySQL จึงเลือกใช้ MySQL Connector ด้วยการเรียกไฟล์ jar

โครงสร้างของ **Table และ Data**  


ภาพที่ 2.5 แสดงตัวอย่างข้อมูลในฐานข้อมูล MySQL

  
ภาพที่ 2.6 แสดงตัวอย่างการเพิ่มข้อมูล

**Form** สำหรับรับข้อมูล  

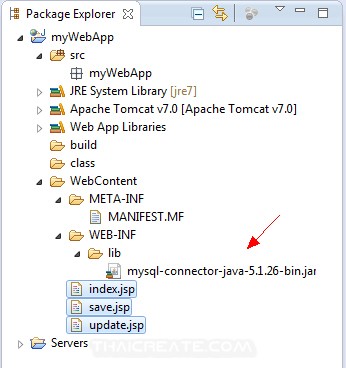

ภาพที่ 2.7 แสดงตัวอย่างเมื่อเพิ่มข้อมูลเรียบร้อย

แสดงสถานะการ **Insert**  
ภาพที่ 2.8 แสดงตัวอย่างฐานข้อมูลฐานเมื่อเพิ่มข้อมูลเรียบร้อย

* + 1. **JSP Edit/Update data in Database**

ตัวอย่างการเขียน JSP เพื่อติดต่อกับ Database และตัวอย่างการ Update แก้ไขด้วย JSP ใน Database โดยในขั้นแรกจะใช้การออกแบบหน้าสำหรับแสดงข้อมูล ในรูปแบบของ Table และสามารถเลือกที่จะแก้ไขรายการใน Record ต่าง ๆ และเมื่อคลิกแก้ไขจะมี Form สำหรับการแก้ไขและบันทึกข้อมูล

Example ตัวอย่างการเขียน JSP เพื่อ ทำการ Update/ Edit แก้ไขข้อมูลลงใน Database



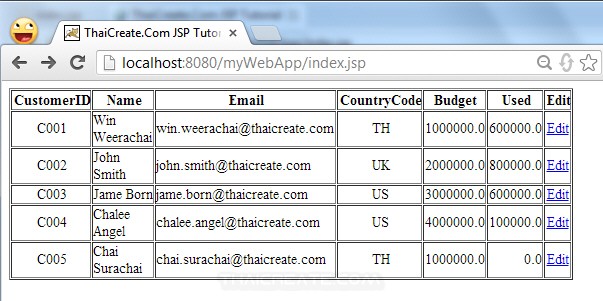
ภาพที่ 2.9 แสดงตัวอย่างการไฟล์ทั้งหมด

เนื่องจากใช้ Database ของ MySQL จึงเลือกใช้ MySQL Connector ด้วยการเรียกไฟล์ jar

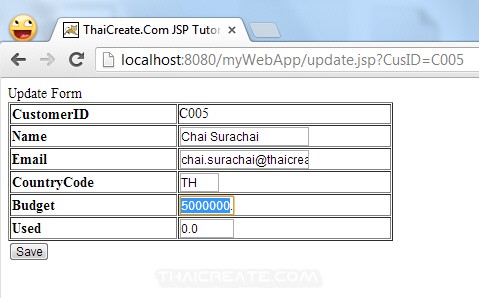
โครงสร้างของ Table และ Data



ภาพที่ 2.10 แสดงตัวอย่างข้อมูลในฐานข้อมูล MySQL



ภาพที่ 2.11 แสดงตัวอย่างแสดงรายการข้อมูลที่สามารถแก้ไขได้



ภาพที่ 2.12 แสดงตัวอย่างการแก้ไขข้อมูล



ภาพที่ 2.13 แสดงตัวอย่างเมื่อแก้ไขข้อมูลเรียบร้อย

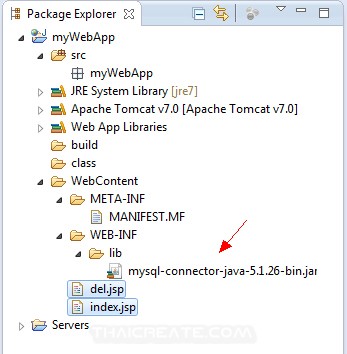


ภาพที่ 2.14 แสดงตัวอย่างฐานข้อมูลฐานเมื่อแก้ไขข้อมูลเรียบร้อย

* + 1. **JSP Delete** **data to Database**

ตัวอย่างการเขียน JSP เพื่อติดต่อกับ Database และตัวอย่างการลบ Delete ข้อมูลใน Database โดยใช้ JSP โดยขั้นแรกจะทำการอ่านข้อมูลจาก Database มาแสดงใน Table จากนั้นสามารถเลือกรายการข้อมูลที่จะทำการลบได้

Example ตัวอย่างการเขียน JSP เพื่อ ทำการ Delete ลบข้อมูลลงใน Database



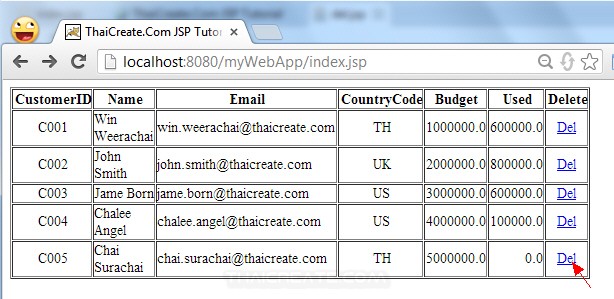
ภาพที่ 2.15 แสดงตัวอย่างการไฟล์ทั้งหมด

เนื่องจากใช้ Database ของ MySQL จึงเลือกใช้ MySQL Connector ด้วยการเรียกไฟล์ jar

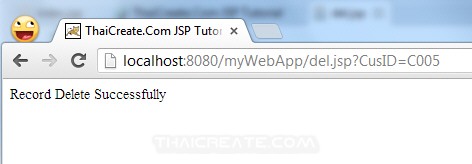
โครงสร้างของ Table และ Data



ภาพที่ 2.16 แสดงตัวอย่างข้อมูลในฐานข้อมูล MySQL



ภาพที่ 2.17 แสดงรายการข้อมูล สามารถเลือกที่จะลบได้



ภาพที่ 2.18 แสดงตัวอย่างเมื่อลบข้อมูลเรียบร้อย



ภาพที่ 2.19 แสดงตัวอย่างฐานข้อมูลฐานเมื่อลบข้อมูลเรียบร้อย