

สาขาวิชาวิทยาศาสตร์และเทคโนโลยี
มหาวิทยาลัยสุโขทัยธรรมมาธิราช

กิจกรรมฝึกปฏิบัติเสริมทักษะภาคสนาม

ชุดวิชาการโปรแกรมเว็บ

(Web Programming)

99420

ภาคเรียนที่ 1/2563

คำนำ

ชุดวิชาการโปรแกรมเว็บ มีลักษณะการเรียนการสอนที่ประกอบด้วยภาคทฤษฎีและภาคปฏิบัติ ซึ่งเป็นการพัฒนาความรู้และความสามารถในการพัฒนาเว็บแอปพลิเคชันด้วยเทคโนโลยีจาวา โดยนำความรู้ทางทฤษฎีมาประยุกต์ใช้ในการปฏิบัติงานเพื่อให้พร้อมที่จะนำไปใช้ในการประกอบอาชีพได้อย่างมีประสิทธิภาพ และเป็นการเพิ่มพูนทักษะด้านการเขียนโปรแกรม และพัฒนาเว็บแอปพลิเคชันอันเป็นพื้นฐานสำคัญในการทำงานได้อย่างเหมาะสมถูกต้องตามหลักทฤษฎี อีกทั้งยังเป็นโอกาสที่จะได้พบปะแลกเปลี่ยนความรู้และประสบการณ์ในการปฏิบัติงานระหว่างคณาจารย์กับนักศึกษา ซึ่งเป็นแนวทางให้เกิดความเข้าใจในวิชาชีพอันเป็นประโยชน์แก่นักศึกษา

คณะทำงานแขนงวิชาเทคโนโลยีสารสนเทศและการสื่อสาร วิชาเอกวิทยาการคอมพิวเตอร์ หวังว่านักศึกษา คงได้รับประโยชน์จากการศึกษา และแนวทางการฝึกกิจกรรมจากแบบฝึกปฏิบัติเสริมทักษะด้วยตนเองเล่มนี้ และหากพบข้อบกพร่องหรือมีข้อเสนอแนะประการใด โปรดแจ้งสาขาวิชาวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยสุโขทัยธรรมาธิราช อีเมล stoffice@stou.ac.th

คณะทำงานแขนงวิชาเทคโนโลยีสารสนเทศและการสื่อสาร
วิชาเอกวิทยาการคอมพิวเตอร์

1. วิธีการศึกษาและการประเมินผล

นักศึกษาที่ลงทะเบียนเรียนในชุดวิชาการโปรแกรมเว็บนี้มีวิธีการศึกษาและการประเมินผลดังนี้

1. วิธีการศึกษา

1.1 ภาควิชา นักศึกษาต้องศึกษาจากเอกสารการสอนชุดวิชาการโปรแกรมเว็บ ซึ่งมีเนื้อหาแบ่งออกเป็น 2 เล่ม คือ เล่มที่ 1 ประกอบด้วยหน่วยการสอนจำนวน 7 หน่วย และเล่มที่ 2 ประกอบด้วยหน่วยการสอนจำนวน 8 หน่วย รวมเป็นหน่วยการสอนทั้งหมด 15 หน่วย ซึ่งนักศึกษาควรใช้เวลาศึกษาเอกสารการสอนวันละ 1-2 ชั่วโมง และควรศึกษา 1 หน่วย ให้จบภายใน 1 สัปดาห์ และก่อนที่นักศึกษาจะทำการศึกษาในเอกสารการสอนขอให้นักศึกษาประเมินผลตนเองก่อนเรียนเพื่อประเมินว่านักศึกษามีความรู้ในเนื้อหาที่จะศึกษามากน้อยเพียงใด หลังจากศึกษาเอกสารการสอนโดยตลอดแล้วขอให้นักศึกษาประเมินผลตนเองหลังเรียนอีกครั้งหนึ่งเพื่อประเมินว่าสามารถอธิบายเนื้อหาที่ศึกษามาแล้วได้หรือไม่ การประเมินผลตนเองก่อนเรียนและหลังเรียนจะช่วยให้นักศึกษาทราบว่าได้รับความรู้เพิ่มขึ้นจากเดิมมากน้อยเพียงใด

1.2 ภาควิชาปฏิบัติ นักศึกษาต้องได้รับการฝึกปฏิบัติเสริมทักษะใน 2 ลักษณะ คือ

1.2.1 ฝึกปฏิบัติเสริมทักษะด้วยตนเอง เป็นการฝึกปฏิบัติที่นักศึกษาฝึกปฏิบัติด้วยตนเองที่บ้าน โดยนักศึกษาจะได้รับกิจกรรมฝึกปฏิบัติเสริมทักษะด้วยตนเอง 1 ชุด ซึ่งใช้ศึกษาควบคู่กับเอกสารการสอน หรืออาจใช้หนังสือด้านหลักการและบริหารเครือข่ายอื่นๆ เพื่ออ่านประกอบในการจัดทำกิจกรรมด้วยตนเอง นักศึกษาต้องฝึกปฏิบัติกิจกรรมต่างๆ ตามที่กำหนดในแบบฝึกปฏิบัติเสริมทักษะด้วยตนเองทุกกิจกรรม ทั้งนี้กิจกรรมฝึกปฏิบัติเสริมทักษะด้วยตนเองนี้ เป็นส่วนหนึ่งของคะแนนสอบในชุดวิชาการโปรแกรมเว็บ และยังเป็นแนวทางในกิจกรรมฝึกปฏิบัติเสริมทักษะภาคสนามด้วย ฉะนั้นในกิจกรรมฝึกปฏิบัติเสริมทักษะด้วยตนเอง นักศึกษาควรกระทำด้วยตนเองให้ครบถ้วนทุกกิจกรรมด้วยความรอบคอบ พร้อมทั้งเขียนงานที่ได้รับมอบหมายจากกิจกรรมในแบบฝึกปฏิบัติเสริมทักษะด้วยตนเองด้วยลายมือหรือพิมพ์ลงกระดาษ A4 และจัดทำเป็นรูปเล่มให้เรียบร้อย และส่งมาที่มหาวิทยาลัยด้วยพร้อมกับกิจกรรมฝึกปฏิบัติเสริมทักษะภาคสนาม

1.2.2 กิจกรรมฝึกปฏิบัติเสริมทักษะภาคสนาม เป็นกิจกรรมที่นำมาใช้ในการฝึกปฏิบัติเสริมทักษะภาคสนาม โดยนักศึกษาจะได้รับกิจกรรมทดแทนการฝึกปฏิบัติเสริมทักษะด้วยภาคสนาม 1 ชุด นักศึกษาจะต้องทำกิจกรรมต่างๆ ตามที่กำหนดในกิจกรรมทดแทนการฝึกปฏิบัติเสริมทักษะภาคสนามทุกกิจกรรม ทั้งนี้กิจกรรมฝึกปฏิบัติเสริมทักษะภาคสนามนี้ เป็นส่วนหนึ่งของคะแนนสอบในชุดวิชาการโปรแกรมเว็บ โดยนักศึกษาทำงานที่ได้รับมอบหมายจากกิจกรรมลงกระดาษ A4 และจัดทำเป็นรูปเล่มให้เรียบร้อย และส่งมาที่มหาวิทยาลัยด้วยพร้อมกับกิจกรรมฝึกปฏิบัติเสริมทักษะด้วยตนเอง

2. การประเมินผล

การประเมินผลในชุดวิชาการโปรแกรมเว็บ มีคะแนนเต็มทั้งหมด 100 คะแนน แบ่งการประเมินผลออกเป็น 2 ส่วน คือ

2.1 การประเมินผลภาควิชา เป็นการประเมินผลจากการสอบภาควิชา ๓ สนามสอบที่จัดไว้ตามวันและเวลาที่กำหนด ซึ่งการประเมินผลภาควิชา คิดเป็น 40 คะแนน

2.2 การประเมินผลการฝึกปฏิบัติเสริมทักษะ เป็นการประเมินผลจากกิจกรรมฝึกปฏิบัติเสริมทักษะ ซึ่งคิดเป็น 60 คะแนน โดยแบ่งเป็นการประเมินผลการกิจกรรมฝึกปฏิบัติเสริมทักษะด้วยตนเอง 10 คะแนน และการประเมินผลจากกิจกรรมทดแทนการฝึกปฏิบัติเสริมทักษะภาคสนาม 50 คะแนน

ทั้งนี้นักศึกษาจะต้องสอบภาคทฤษฎีให้ผ่านร้อยละ 60 และสอบภาคปฏิบัติให้ผ่านร้อยละ 60 จึงจะถือว่านักศึกษาสอบผ่านในชุดวิชาการโปรแกรมเว็บ

2. เนื้อหากิจกรรมฝึกปฏิบัติเสริมทักษะภาคสนาม

วัตถุประสงค์

เมื่อฝึกกิจกรรมฝึกปฏิบัติเสริมทักษะภาคสนามนี้จบแล้ว นักศึกษาสามารถ

1. พัฒนาเว็บแอปพลิเคชันด้วยภาษา JSP และ Servlet ได้
2. พัฒนาเว็บแอปพลิเคชันให้สามารถติดต่อฐานข้อมูลได้
3. สร้างและใช้งาน Restful Web Service และเรียกใช้งาน Web API ได้

กิจกรรมหลัก

ให้นักศึกษาสร้างเว็บแอปพลิเคชันสำหรับการขายสินค้า Smart watch ออนไลน์ โดยที่ลูกค้าสามารถ 1) ลงทะเบียนเป็นสมาชิก 2) การเข้าสู่ระบบและการออกจากระบบ 3) การแสดงรายละเอียดสินค้า 4) การค้นหาสินค้า 5) การเพิ่มสินค้าลงในตระกร้าสินค้า 6) การสั่งซื้อสินค้า และ 7) การจ่ายเงินค่าสินค้า ส่วนผู้ดูแลระบบนั้น นอกจากจะสามารถทำงานเหมือนลูกค้าทั่วไปได้แล้ว ยังสามารถ 1) เพิ่ม ลบ และ เปลี่ยนแปลงสินค้า และ 2) เพิ่ม ลบ และ เปลี่ยนแปลงลูกค้า

กิจกรรมที่ 1 สร้าง และใช้งานคลาสและอ็อบเจกต์

1) วัตถุประสงค์

นำหลักการพื้นฐานที่สำคัญของการโปรแกรมเชิงวัตถุมาประยุกต์ โดยการสร้างคลาสทั้งหมดที่เกี่ยวข้องกับระบบที่จะพัฒนา ความสัมพันธ์ระหว่างคลาส รวมทั้งสร้างอ็อบเจกต์ของคลาสเพื่อจำลองการทำงานของระบบ

2) เครื่องมือหรือไฟล์ที่ใช้

- Java SE 8 ซึ่งเป็น Compiler และ Interpreter ของภาษาจาวา โดยสามารถ download จาก <https://www.oracle.com/java/technologies/javase/javase-jdk8-downloads.html>
- Eclipse Photon for Java EE Developer ซึ่งเป็น Editor สำหรับการเขียนโปรแกรม โดยสามารถ download จาก <https://www.eclipse.org/downloads/packages/release/photon/r>

3) รายละเอียดกิจกรรม

จากการวิเคราะห์ระบบของเว็บแอปพลิเคชันการขายสินค้านี้ดังกล่าวก่อน จะเห็นว่ามีคลาสหลักที่สำคัญ ได้แก่ 1) คลาสลูกค้า (Customer) ทำหน้าที่ในการเก็บรายละเอียดของลูกค้าแต่ละคน 2) คลาสสินค้า (Product) ทำหน้าที่ในการเก็บรายละเอียดของสินค้าแต่ละตัว 3) คลาสตะกร้าสินค้า (Cart) ทำหน้าที่ในการเก็บรายละเอียดของตะกร้าเก็บสินค้าที่ต้องการจะสั่งซื้อ และ 4) คลาสการสั่งซื้อสินค้า (Order) ทำหน้าที่ในการเก็บรายละเอียดของการสั่งซื้อสินค้าแต่ละครั้ง

ให้นักศึกษาสร้าง Java Project ชื่อ WSAct1_รหัสนักศึกษา แล้วสร้างคลาสในข้อ 3.1) - 3.4)

3.1) สร้างและใช้งานคลาส Customer และการบริหารจัดการลูกค้า และจำลองการทำงาน

1. สร้างคลาส Customer ซึ่งทำหน้าที่ในการเก็บรายละเอียดของลูกค้า และคลาส CustomerService ซึ่งทำหน้าที่เหมือนคลังเก็บลูกค้าเพื่อสามารถบริหารจัดการลูกค้าทั้งหมด ดังนี้

คลาส Customer ประกอบด้วย Attribute และ Method ดังนี้

(1) Attribute:

- รหัสลูกค้า (id) : String
- ชื่อลูกค้า (firstname) : String
- นามสกุลลูกค้า (lastname) : String
- ชื่อLogin (loginname) : String
- รหัสผ่าน (password) : String
- วันที่สมัคร (registerdate) : LocalDate

(2) Method:

- Constructor
- Getter
- Setter
- toString

คลาส CustomerService ประกอบด้วย Attribute และ Method ดังนี้

(1) Attribute:

- ลิสต์ลูกค้าทั้งหมด (customers) : HashMap<String, Customer>

(2) Method:

- Constructor
- Getter
- Setter
- addCustomer ทำหน้าที่ในการเพิ่มสมาชิกเข้ามาในคลังลูกค้า
- removeCustomer ทำหน้าที่ในการลบสมาชิกออกจากคลังลูกค้า
- listCustomer ทำหน้าที่ในการแสดงรายละเอียดของลูกค้าทั้งหมดที่อยู่ในคลังลูกค้า
- searchCustomerByName ทำหน้าที่ในการค้นหาลูกค้าตามชื่อลูกค้า

2. จำลองการทำงานโดย 1) ลงทะเบียนสมาชิกหรือเพิ่มลูกค้า 2) ค้นหาสมาชิก 3) แสดงข้อมูลของสมาชิกทั้งหมด 4) เปลี่ยนแปลงข้อมูลของสมาชิก เช่น รหัสผ่าน เป็นต้น และ 5) ลบสมาชิก ดังตัวอย่างในโปรแกรมดังนี้

```
Customer c1 = new Customer("C001", "Urai", "sudjai", "urai", "urai", LocalDate.now());
Customer c2 = new
Customer("C002", "Paitoon", "sukjai", "paitoon", "paitoon", LocalDate.now());
```

```

CustomerService cs = new CustomerService();
cs.addCustomer(c1);
cs.addCustomer(c2);
cs.listCustomer();
System.out.println(cs.searchCustomerByName("Urai"));
c1.setPassword("uraisudjai");
cs.removeCustomer(c2);
cs.removeCustomer("Urai");
cs.listCustomer();

```

3.2) สร้างและใช้งานคลาส Product และการบริหารจัดการสินค้า และจำลองการทำงาน (คล้ายข้อ 1)

1. สร้างคลาส Product ซึ่งทำหน้าที่ในการเก็บรายละเอียดของสินค้า และคลาส ProductService ซึ่งทำหน้าที่เสมือนเป็นคลังสินค้าเก็บสินค้าเพื่อสามารถบริหารจัดการสินค้าทั้งหมด ดังนี้

คลาส Product ประกอบด้วย Attribute และ Method ดังนี้

(1) Attribute:

- รหัสสินค้า (id) : String
- ชื่อสินค้า (name) : String
- คำอธิบายสินค้า (desc) : String
- ราคา (price) : int
- ชื่อไฟล์รูปสินค้า (img) : String

(2) Method:

- Constructor
- Getter
- Setter
- toString

คลาส ProductService ประกอบด้วย Attribute และ Method ดังนี้

(1) Attribute:

- คลังสินค้า (products) : HashMap<String,Product>

(2) Method:

- Constructor
- Getter
- Setter
- addProduct ทำหน้าที่ในการเพิ่มสินค้าใหม่เข้ามาในคลังสินค้า
- removeProduct ทำหน้าที่ในการลบสินค้าออกจากคลังสินค้า

- listProduct ทำหน้าที่ในการแสดงรายละเอียดของสินค้าทั้งหมดที่อยู่ในคลังสินค้า
- searchProductByName ทำหน้าที่ในการค้นหาสินค้าตามชื่อสินค้า

2. จำลองการทำงานโดยการ 1) เพิ่มสินค้าลงในคลังสินค้า 2) ค้นหาสินค้าจากชื่อสินค้า 3) แสดงข้อมูลของสินค้าทั้งหมด 4) เปลี่ยนแปลงข้อมูลของสินค้า เช่น ราคาสินค้า เป็นต้น และ 5) ลบสินค้าออกจากคลังสินค้า ดังตัวอย่างในโปรแกรมดังนี้

```
Product p1 = new Product("P01", "Xiaomi", "Amazfit GTS Gold", 3900, "p01.jpg");
Product p2 = new Product("P02", "Fitbit", "Versa 2 Petal Copper", 6000, "p02.jpg");
Product p3 = new Product("P03", "Suunto", "White Bergundy", 13000, "p03.jpg");
Product p4 = new Product("P04", "Garmin", "Vivifit 4 Activity Tracker Black
L", 3300, "p04.jpg");
ProductService cs = new ProductService();
cs.addProduct(p1);
cs.addProduct(p2);
cs.addProduct(p3);
cs.addProduct(p4);
cs.listProduct();
p1.setPrice(4200);
cs.removeProduct(p2);
cs.listProduct();
System.out.println(cs.searchProductByName("Garmin"));
```

3.3) สร้างและใช้งานคลาส Cart และการบริหารจัดการตะกร้าสินค้า และจำลองการทำงาน (ให้นักศึกษาออกแบบ Attribute และ Method เอง) และจำลองการทำงานโดยการ 1) เพิ่มสินค้าลงในตะกร้าสินค้า 2) ค้นหาสินค้าในตะกร้าสินค้าจากชื่อสินค้า 3) แสดงข้อมูลของสินค้าทั้งหมดในตะกร้าสินค้า 4) เปลี่ยนแปลงจำนวนที่ต้องการจะสั่งซื้อ 5) ลบสินค้าออกจากตะกร้าสินค้า และ 6) เคลียร์ตะกร้าสินค้า ดังตัวอย่างในโปรแกรมดังนี้

```
Product p1 = new Product("P01", "Xiaomi", "Amazfit GTS Gold", 3900, "p01.jpg");
Product p2 = new Product("P02", "Fitbit", "Versa 2 Petal Copper", 6000, "p02.jpg");
Product p3 = new Product("P03", "Suunto", "White Bergundy", 13000, "p03.jpg");
ProductService ps = new ProductService();
ps.addProduct(p1);
ps.addProduct(p2);
ps.addProduct(p3);
ps.listProduct();

Customer c1 = new Customer("C001", "Urai", "sudjai", "urai", "urai", LocalDate.now());
Customer c2 = new Customer("C002", "Paitoon", "sukjai", "paitoon", "paitoon", LocalDate.now());
CustomerService cs = new CustomerService();
cs.addCustomer(c1);
cs.addCustomer(c2);
cs.listCustomer();

Cart cart1 = new Cart(c1.getId());
cart1.addItem(p1, 50);
cart1.addItem(p2, 10);
cart1.addItem(p3, 30);
cart1.removeItem(p1);
System.out.println(cart1.searchItemByProductName("Suunto"));
//cart1.clear();
```



```

Cart cart2 = new Cart(c2.getId());
cart2.addItem(p1, 50);
cart2.addItem(p2, 10);

CartService cts = new CartService();
cts.addCart(cart1);
cts.addCart(cart2);
//cts.removeCart(cart1);
cts.listCart();
System.out.println(cts.searchCartById(c2.getId()));

```

3.4) (ถ้ามีเวลา) สร้างและใช้งานคลาส Order และการบริหารจัดการการสั่งซื้อสินค้า และจำลองการทำงาน (ให้นักศึกษาออกแบบ Attribute และ Method เอง) และจำลองการทำงาน

กิจกรรมที่ 2 สร้างเว็บแอปพลิเคชันด้วยภาษา JSP และ Servlet

1) วัตถุประสงค์

พัฒนาเว็บแอปพลิเคชันโดยใช้เทคโนโลยีจาวาในฝั่ง Web Server โดยใช้ภาษา JSP ซึ่งเป็นภาษา Script สำหรับแสดงผลการทำงาน (ประกอบกับความรู้ภาษา HTML เบื้องต้น) และ Servlet ซึ่งใช้สำหรับควบคุมการทำงานของเว็บแอปพลิเคชัน

*******หมายเหตุ***** กิจกรรมที่ 2 เป็นการใช้งานคลาสและอ็อบเจกต์เพื่อเก็บข้อมูลโดยไม่อนุญาตให้ใช้ Database**

2) เครื่องมือหรือไฟล์ที่ใช้ (เพิ่มเติมจากเครื่องมือที่ใช้ในกิจกรรมที่ 1)

- **Apache Tomcat 9** ซึ่งทำหน้าที่เป็น Web Server โดยสามารถ download จาก <https://tomcat.apache.org/download-90.cgi>

3) รายละเอียดกิจกรรม

ให้นักศึกษาสร้าง Dynamic Web Project ชื่อ WSAct2_รหัสนักศึกษา

ให้นักศึกษาพัฒนา Web Application ด้วย JSP และ Servlet โดยมีหน้าจอและการทำงานตามที่กำหนดให้ สำหรับกิจกรรมนี้เป็นการพัฒนาในส่วนของ Customer Service ซึ่งเป็นส่วนหนึ่งของ Web Application การขายสินค้า โดยประกอบด้วยไฟล์หลักๆ 3 ไฟล์ ได้แก่

- 1) ไฟล์ jsp สำหรับหน้าจอแสดงรายละเอียดของลูกค้านี้ทั้งหมดที่เป็นสมาชิก
- 2) ไฟล์ jsp สำหรับหน้าจอแบบฟอร์มการกรอกข้อมูลเพื่อสมัครเป็นสมาชิกใหม่
- 3) ไฟล์ Servlet สำหรับจัดการคำร้องขอ (request) ที่ส่งมาจากผู้ใช้งาน

รายละเอียดการทำงานของ Customer Service มีดังนี้

หน้าจอแรกของ Customer Service ซึ่งแสดงรายละเอียดของลูกค้านี้หรือสมาชิกทั้งหมด โดยเริ่มต้นจากการที่ไม่มีลูกค้านี้เป็นสมาชิกของร้าน กำหนดให้มีรายละเอียดดังภาพที่ 1

Customer Service

List Customer
New Customer

List Customer

Search Text Search

profileName	firstname	lastname	phone	
-------------	-----------	----------	-------	--

ภาพที่ 1 หน้าจอแรกของ Customer Service

จากภาพที่ 1 เมื่อกดปุ่ม New Customer ก็จะแสดงหน้าจอการเพิ่มข้อมูลลูกค้า ดังภาพที่ 2 (ก)

Customer Service

List Customer
New Customer

New Customer

Profile Name:

Firstname:

Lastname:

Phone:

Save

(ก)

Customer Service

List Customer
New Customer

New Customer

Profile Name:

Firstname:

Lastname:

Phone:

Save

(ข)

ภาพที่ 2 หน้าจอการเพิ่มลูกค้า

จากภาพที่ 2 (ก) ใส่ข้อมูลในกล่องข้อความ ดังภาพที่ 2 (ข) แล้วกดปุ่ม Save แล้วจะได้ดังภาพที่ 3

Customer Service

[List Customer](#)
[New Customer](#)

List Customer

 Search Text
[Search](#)

profileName	firstname	lastname	phone		
urai99	Urai	Sukdee	0981123454	Edit	Delete

ภาพที่ 3 หน้าจอการแสดงผลลูกค้าทั้งหมด

จากภาพที่ 3 สามารถเพิ่มลูกค้าได้โดยการกดปุ่ม

[New Customer](#)

และสมมติว่าใส่ข้อมูลลูกค้าเพิ่ม

เป็นจำนวนทั้งหมด 3 คน ดังภาพที่ 4

Customer Service

[List Customer](#)
[New Customer](#)

List Customer

 Search Text
[Search](#)

profileName	firstname	lastname	phone		
paitoon99	Paitoon	Deemak	0987659876	Edit	Delete
Wanna12	Wanna	Pasuay	0897778987	Edit	Delete
urai99	Urai	Sukdee	0981123454	Edit	Delete

ภาพที่ 4 หน้าจอแสดงลูกค้าทั้งหมด

จากภาพที่ 4 สามารถค้นหาลูกค้าได้ โดยใส่ข้อความที่ต้องการค้นหา

Search Text แล้วกดปุ่ม เพื่อแสดงผลการค้นหา โดย
สมมติให้ใส่ชื่อ Urai ในการค้นหา ก็จะได้ผลลัพธ์ดังภาพที่ 5

The screenshot shows a web application titled "Customer Service". At the top, there are two buttons: "List Customer" (blue) and "New Customer" (red). Below these is a section titled "List Customer". In this section, there is a search bar with the text "Urai" and a green "Search" button. Below the search bar is a table with the following data:

profileName	firstname	lastname	phone	
urai99	Urai	Sukdee	0981123454	<input type="button" value="Edit"/> <input type="button" value="Delete"/>

ภาพที่ 5 หน้าจอผลลัพธ์จากการค้นหา

จากภาพที่ 4 หากต้องการเปลี่ยนแปลงข้อมูลของลูกค้า สมมติว่าต้องการเปลี่ยนชื่อของ Urai ก็

สามารถกดปุ่ม ที่อยู่ในแถวของ Urai จะได้ดังภาพที่ 6

Customer Service

[List Customer](#)
[New Customer](#)

Update Profile

 Profile Name:

 Firstname:

 Lastname:

 Phone:
[Update](#)

ภาพที่ 6 หน้าจอการเปลี่ยนแปลงข้อมูลลูกค้า

จากภาพที่ 6 มีการเปลี่ยนแปลงชื่อและนามสกุลเป็น Uraipan และ Suksaimak ตามลำดับ และ

เมื่อกดปุ่ม [Update](#) ก็จะได้ผลลัพธ์ดังภาพที่ 7 (ก) หากต้องการจะลบข้อมูลลูกค้าชื่อ Wanna ก็สามารถทำได้

โดยการกดปุ่ม [Delete](#) ที่อยู่ในแถวของ Wanna แล้วจะได้ผลลัพธ์ดังภาพที่ 7 (ข)

Customer Service

[List Customer](#)
[New Customer](#)

List Customer

 Search Text [Search](#)

profileName	firstname	lastname	phone		
paicon99	Paicon	Deemak	0987659876	Edit	Delete
Wanna12	Wanna	Pasuy	0897778987	Edit	Delete
urai99	Uraipan	Suksaimak	0981123454	Edit	Delete

Customer Service

[List Customer](#)
[New Customer](#)

List Customer

 Search Text [Search](#)

profileName	firstname	lastname	phone		
paicon99	Paicon	Deemak	0987659876	Edit	Delete
urai99	Uraipan	Suksaimak	0981123454	Edit	Delete

ภาพที่ 7 หน้าจอการแสดงลูกค้าทั้งหมดหลังการเปลี่ยนแปลงและลบข้อมูลลูกค้า

***หมายเหตุ ถ้านักศึกษามีเวลา สามารถปรับปรุงหน้าตาให้สวย (ใช้ความรู้ HTML, CSS และ Java script)

กิจกรรมที่ 3 Jsp, Servlet และการติดต่อฐานข้อมูล MySQL

1) วัตถุประสงค์

เพื่อพัฒนาฐานข้อมูลเชิงสัมพันธ์ของระบบการขายสินค้า Smart watch ออนไลน์ ด้วย MySQL ที่ประกอบด้วยตารางสินค้า ตารางลูกค้า และตารางตระกร้าสินค้า จากนั้นเขียนคำสั่ง JSP, Java และ Servlet เพื่อเชื่อมต่อระบบฐานข้อมูล โดยสามารถเพิ่ม ลบ และแก้ไขข้อมูลได้

2) เครื่องมือหรือไฟล์ที่ใช้

- MySQL Workbench 8.0.22 for Windows ซึ่งเป็นเครื่องมือสำหรับบริหารจัดการระบบฐานข้อมูล โดยสามารถ download จาก <https://dev.mysql.com/downloads/workbench/>

*** หมายเหตุ **** ไฟล์ที่ใช้ในการเรียนการสอน อาจารย์จะ download ให้นักศึกษาที่ MS-Teams ของชุดวิชานี้

3) รายละเอียดกิจกรรม

- 1 การสร้างฐานข้อมูล smartwatch ด้วย MySQL Workbench 8.0.22
 - 1.1 เปิด MySQL Workbench 8.0 CE จากนั้นทำการสร้าง new connection ที่ MySQL Connections ให้คลิกที่เครื่องหมายบวก (+)
 - 1.2 ที่ Setup new connection ให้กำหนดค่าดังนี้

Connection Name : My project1

Username : root

จากนั้นคลิกที่ปุ่ม Configure server management... เพื่อทำการกำหนด password และกำหนดค่าตั้งต้นต่างๆ โดย password จะกำหนดตาม password เมื่อ login เข้าเครื่องคอมพิวเตอร์ ดังภาพที่ 8

Setup New Connection

Connection Name: Type a name for the connection

Connection Method: Method to use to connect to the RDBMS

Parameters SSL Advanced

Hostname: Port: Name or IP address of the server host - and TCP/IP port.

Username: Name of the user to connect with.

Password: The user's password. Will be requested later if it's not set.

Default Schema: The schema to use as default schema. Leave blank to select it later.

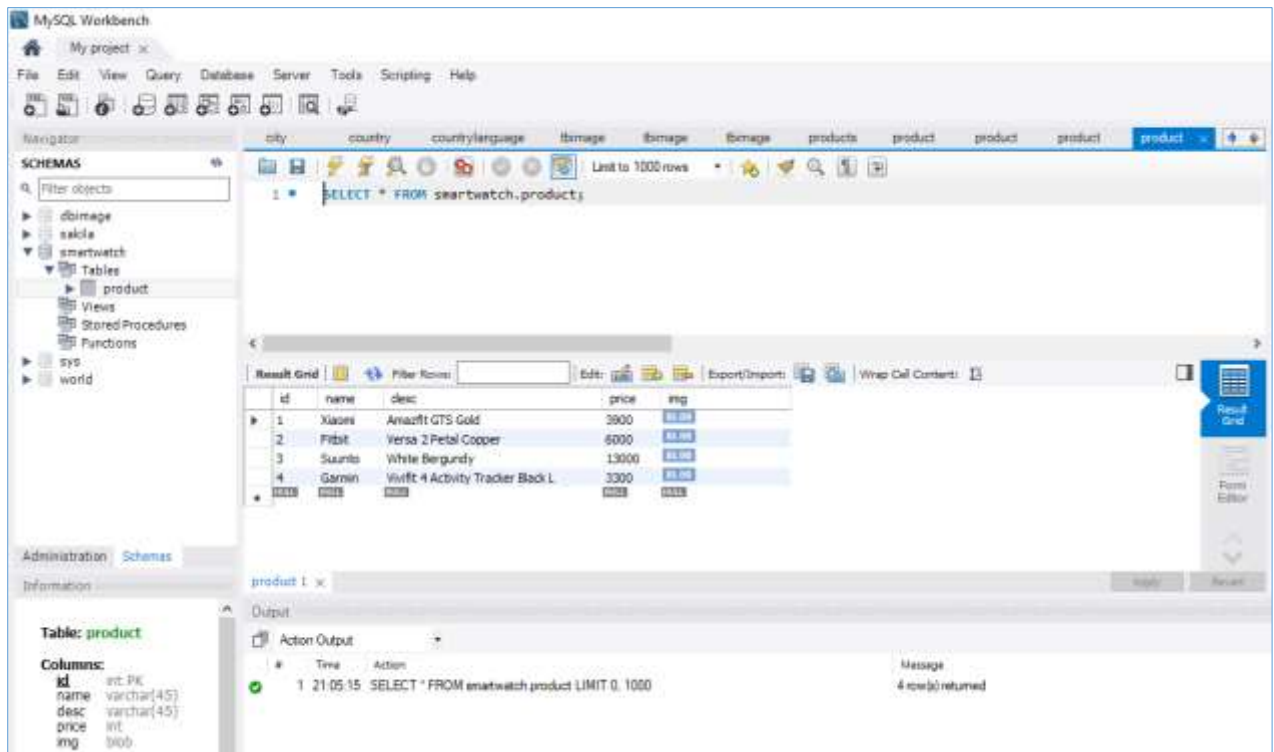
ภาพที่ 8 การกำหนดค่า New Connection ของ My SQL Workbench

- 1.3 สร้าง Schema, Table และกำหนดค่าเริ่มต้นข้อมูลดังนี้
- 1.4 สร้าง Schema : smartwatch
- 1.5 สร้างตารางชื่อ: product

id	int	pk,nn,uq
name	varchar(45)	
desc	varchar(45)	
price	int	
Image	blob	

กรอกข้อมูลในตาราง product (เหมือนกับกิจกรรมที่ 1 ของ WSAct1_รหัสนักศึกษา)

id	name	desc	price	img
1	Xiaomi	Amazfit GTS Gold	3900	BLOB
2	Fitbit	Versa 2 Petal Copper	6000	BLOB
3	Suunto	White Bergundy	13000	BLOB
4	Garmin Vivifit 4 Activity Tracker Black L		3300	BLOB



ภาพที่ 9 แสดงการสร้าง Schema และ Table ใน MySQL Workbench

- 1.6 ที่เว็บไซต์ Maven repository : <https://mvnrepository.com/> พิมพ์คำว่า “mysql jdbc” จากนั้นเลือก MySQL Connector/J เลือก version 8.0.22 และที่บัตริายการ Maven ให้ copy ข้อความทั้งหมด มาใส่ไว้ที่ pom.xml

```
<dependencies>
  <!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java -->
  <dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>8.0.22</version>
  </dependency>
</dependencies>
```

- 2 การสร้าง jsp ทดสอบการเชื่อมต่อฐานข้อมูล

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<%@ page import="java.sql.*" %>
<%@ page import="java.io.*" %>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<body>

<%
```



```

Connection connect = null;
Statement s = null;

try {
    Class.forName("com.mysql.cj.jdbc.Driver");

    connect =
DriverManager.getConnection("jdbc:mysql://localhost:3306/smartwatch" +
        "?user=root&password=123456");

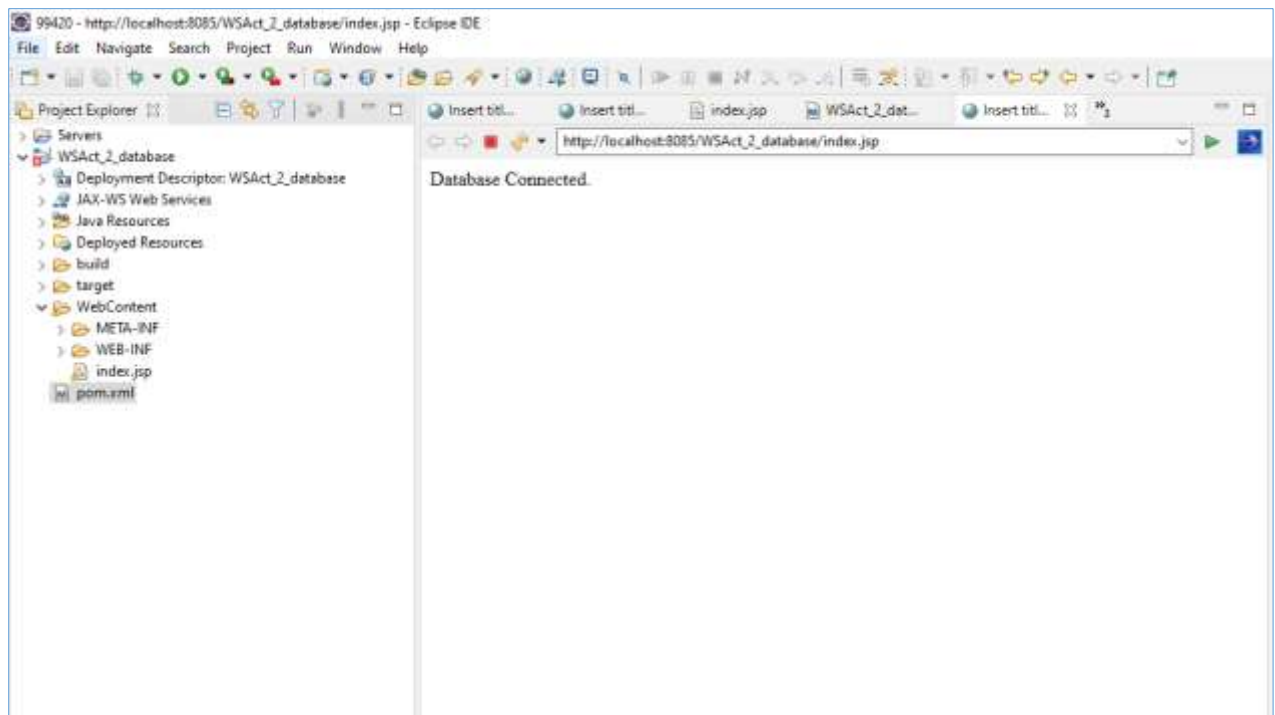
    if(connect != null){
        out.println("Database Connected.");
    } else {
        out.println("Database Connect Failed.");
    }

} catch (Exception e) {
    out.println(e.getMessage());
    e.printStackTrace();
}

try {
    if(s!=null){
        s.close();
        connect.close();
    }
} catch (SQLException e) {
    out.println(e.getMessage());
    e.printStackTrace();
}

%>
</body>
</html>

```



ภาพที่ 10 หน้าจอแสดงการเชื่อมต่อฐานข้อมูลสำเร็จ

จากภาพที่ 10 หน้าจอแสดงการเชื่อมต่อฐานข้อมูล MySQL ด้วย JSP สำเร็จ โดยระบุ username = root และ password = 123456

3 การเขียนคำสั่งแสดงรายการสินค้าทั้งหมดจากตารางสินค้า

The Best Smartwatches for 2021

ID	Name	Description	Price	Action	
1	Xiaomi	Amazfit GTS Gold	3900	Update	Delete
2	Fitbit	Versa 2 Petal Copper	6000	Update	Delete
3	Suunto	White Bergundy	13000	Update	Delete
4	Garmin Vivifit 4	Activity Tracker Black L	3300	Update	Delete

[Add New Smartwatch](#)

ภาพที่ 11 หน้าจอแสดงรายการข้อมูลสินค้าด้วย JSP ที่เชื่อมต่อฐานข้อมูล MySQL

4 โครงสร้างไฟล์ต่างๆ ของ WSAct2_Database ที่ Eclipse > Project Explorer

4.1 /WebContent

4.1.1 Index.jsp : หน้าจอแรก และเรียกใช้ listproduct.jsp ในการแสดงรายการสินค้าทั้งหมด

4.1.2 Listproduct.jsp : แสดงรายการสินค้าทั้งหมด

4.1.3 Product.jsp : แสดงการเพิ่มรายการสินค้าใหม่

4.2 /Java Resources/Src/

4.2.1 Database.java (class) : เขียนคำสั่งเชื่อมต่อฐานข้อมูล

4.2.2 Product.java (class) : เขียนคำสั่งสร้าง class, constructor, getter และ setter

4.2.3 ProductController.java (servlet) : เขียนคำสั่งที่ method : doGet และ doPost ในการส่งผ่านค่า parameter ทั้งหมดกับ class Product

4.2.4 ProductDAO.java (class) : เขียนคำสั่งจัดการเกี่ยวกับฐานข้อมูล ได้แก่ การเพิ่มข้อมูล การลบข้อมูล การแก้ไขข้อมูล และการแสดงรายการข้อมูลทั้งหมด

5 การเขียนคำสั่งเชื่อมต่อฐานข้อมูล ที่ไฟล์ Database.java

```
import java.sql.Connection;
import java.sql.DriverManager;
public class Database {

    public static Connection getConnection() {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con = DriverManager.getConnection(
                "jdbc:mysql://localhost:3306/smartwatch",
                "root", "123456");
            return con;
        }
        catch (Exception ex) {
```

```

        System.out.println("Database.getConnection() Error -->" +
ex.getMessage());
        return null;
    }
}

    public static void close(Connection con) {
        try {
            con.close();
        }
        catch(Exception ex) {
        }
    }
}

```

6 การเขียนคำสั่งสร้าง ProductController.java (servlet)

```

import java.io.IOException;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/ProductController")
public class ProductController extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    /**
     * @see HttpServlet#doGet(HttpServletRequest request,
HttpServletResponse response)
     */

    private static String INSERT_OR_EDIT = "/product.jsp";
    private static String LIST_PRODUCT = "/listproduct.jsp";
    private ProductDAO dao;

    public ProductController() {
        super();
        dao = new ProductDAO();
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        String forward="";
        String action = request.getParameter("action");

        if (action.equalsIgnoreCase("delete")){
            String id = request.getParameter("id");
            dao.deleteProduct(id);
            forward = LIST_PRODUCT;
            request.setAttribute("products", dao.getAllProducts());

```

```

    } else if (action.equalsIgnoreCase("edit")){
        forward = INSERT_OR_EDIT;
        String id = request.getParameter("id");
        Product product = dao.getProductById(id);
        request.setAttribute("product", product);
    } else if (action.equalsIgnoreCase("listProduct")){
        forward = LIST_PRODUCT;
        request.setAttribute("products", dao.getAllProducts());
    } else {
        forward = INSERT_OR_EDIT;
    }

    RequestDispatcher view = request.getRequestDispatcher(forward);
    view.forward(request, response);
}

protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {

    int id = Integer.parseInt(request.getParameter("id"));
    String name=request.getParameter("name");
    String desc =request.getParameter("desc");
    int price =
Integer.parseInt(request.getParameter("price"));

    Product product = new Product(id,name,desc,price);

    product.setId(id);
    dao.checkProduct(product);
//    }
    RequestDispatcher view =
request.getRequestDispatcher(LIST_PRODUCT);
    request.setAttribute("products", dao.getAllProducts());
    view.forward(request, response);
}
}

```

7 การเขียนคำสั่งเพิ่มข้อมูล

```

public void addProduct(Product product) {
    try {
        PreparedStatement preparedStatement =
connection.prepareStatement("insert into smartwatch.products values (?,
?, ?, ?)");
        // Parameters start with 1
        preparedStatement.setInt(1, product.getId());
        preparedStatement.setString(2, product.getName());
        preparedStatement.setString(3, product.getDesc());
        preparedStatement.setInt(4, product.getPrice());
        preparedStatement.executeUpdate();

    } catch (SQLException e) {
        e.printStackTrace();
    }
}

```

8 การเขียนคำสั่งลบข้อมูล

```

public void deleteProduct(String id) {

```

```

        try {
            PreparedStatement preparedStatement =
connection.prepareStatement("delete from smartwatch.products where
id=?");

            preparedStatement.setString(1, id);
            preparedStatement.executeUpdate();

        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

```

- 9 การเขียนคำสั่งแก้ไขข้อมูล

```

public void updateProduct(Product product) {
    try {
        PreparedStatement preparedStatement =
connection.prepareStatement("update smartwatch.products set name=?,
desc=?, price=?"
                            + "where id=?");

        preparedStatement.setInt(1, product.getId());
        preparedStatement.setString(2, product.getName());
        preparedStatement.setString(3, product.getDesc());
        preparedStatement.setInt(4, product.getPrice());
        preparedStatement.executeUpdate();

    } catch (SQLException e) {
        e.printStackTrace();
    }
}

```

- 10 การเขียนคำสั่งแสดงข้อมูลรายการสินค้า listproduct.jsp

```

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
    <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>STOU 99420</title>
</head>
    <h1>The Best Smartwatches for 2021</h1>
<body>
    <table border="1"
        style="width: 100%; text-align: left; background-
color: gold;">
        <thead>
            <tr>

                <th>ID</th>
                <th>Name</th>
                <th>Description</th>
                <th>Price</th>
                <th colspan=2>Action</th>

```

```

        </tr>
    </thead>
    <tbody>
        <c:forEach items="${products}" var="product">
            <tr>
                <td><c:out value="${product.id}" /></td>
                <td><c:out value="${product.name}" /></td>
                <td><c:out value="${product.desc}" /></td>
                <td><c:out value="${product.price}" /></td>

                <td><a href="ProductController?action=edit&id=<c:out
value="${product.id}" />">Update</a></td>
                <td><a
href="ProductController?action=delete&id=<c:out
value="${product.id}" />">Delete</a></td>
            </tr>
        </c:forEach>
    </tbody>
</table>
<p><a href="ProductController?action=insert">Add New
Smartwatch</a></p>
</body>
</html>

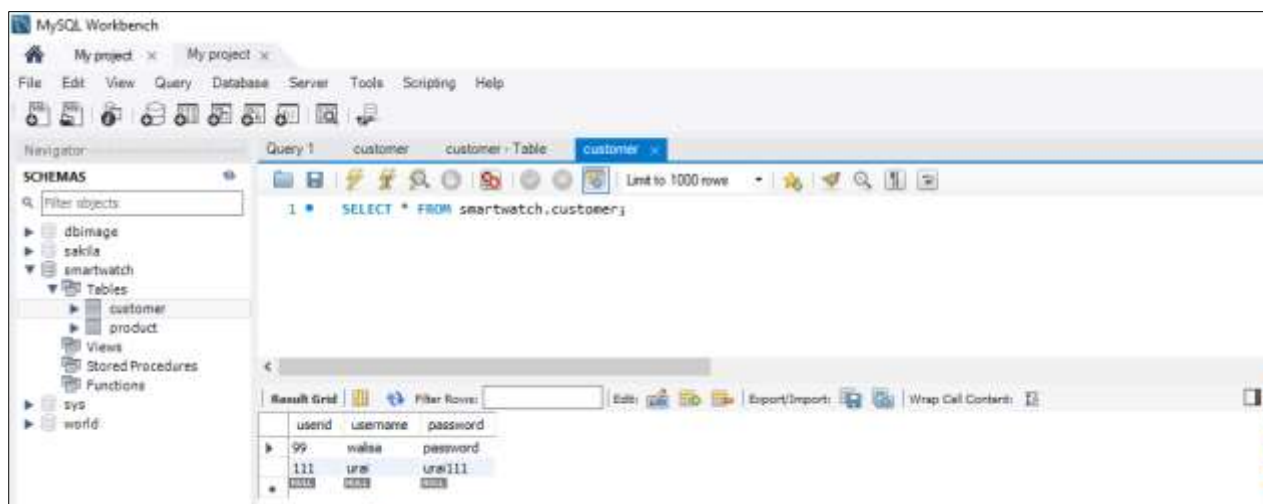
```

11 LAB – 1 : ให้นักศึกษาทำหน้าจอ register ดังภาพที่ 12 พร้อมทั้งสร้างตาราง customer (ข้อมูลจากกิจกรรมที่ 1 ของ WSAct_1_รหัสนักศึกษา) โดยให้กรอกข้อมูลจาก jsp เพิ่มข้อมูลเข้าไปที่ MySQL database

11.1 สร้างหน้าจอ Register : cutomerRegister.jsp

ภาพที่ 12 ไฟล์ customerRegister.jsp

- 11.2 สร้าง Servlet : Register.java ทำหน้าที่เรียกใช้ RegisterDAO ที่เมธอด doPost
- 11.3 สร้าง Class : RegisterDAO.java ทำหน้าที่เชื่อมต่อฐานข้อมูล smartwatch กับตาราง customer และเขียนคำสั่ง insert
- 11.4 สร้าง Class : Customer.java ทำหน้าที่เก็บข้อมูล userid, username และ password (นำข้อมูลมาจากกิจกรรมที่ 1 ของ WSAct_1_รหัสนักศึกษา)



ภาพที่ 13 ตัวอย่างข้อมูลของตาราง Customer

กิจกรรมที่ 4 สร้าง และใช้งาน Web Service

1) วัตถุประสงค์

เรียนรู้หลักการสร้างและใช้งาน Web Service แบบ RESTful พื้นฐาน และการทดสอบการทำงานของ Web Service ที่สร้าง และการทำงานกับโครงสร้างไฟล์ข้อมูลแบบ JSON

2) เครื่องมือหรือไฟล์ที่ใช้สำหรับกิจกรรม (เพิ่มเติมจากเครื่องมือที่ใช้ในกิจกรรมที่ 3)

- โปรแกรม Eclipse แบบ JavaEE
- Tomcat Server 9.0
- การเชื่อมต่อเครือข่ายอินเทอร์เน็ตของเครื่องคอมพิวเตอร์ขณะฝึกปฏิบัติตามกิจกรรม เนื่องจากอาจมีการดาวน์โหลดข้อมูลบางส่วนจาก resource ที่โปรแกรมเรียกใช้งาน เช่น Jersey library

ข้อแนะนำเบื้องต้น

1. ถ้ามีสีติดสีแดงใต้ข้อความหรือคำสั่งใด ๆ ให้ตรวจสอบการพิมพ์ตัวสะกดอักษร ว่าตรงกับตัวอย่างภาพหรือไม่
2. บางครั้งเมื่อแทรกคำสั่งโค้ดในโปรแกรมแล้ว ต้องทำการ save ไฟล์ก่อนเสมอ เพื่อให้โปรแกรม eclipse โหลดข้อมูลบางส่วนทางเน็ตก่อนการรันหรือใช้งานโปรแกรมได้ (อาจใช้เวลาพอสมควร ให้สังเกต status ของโปรแกรมทางมุมล่างขวา)

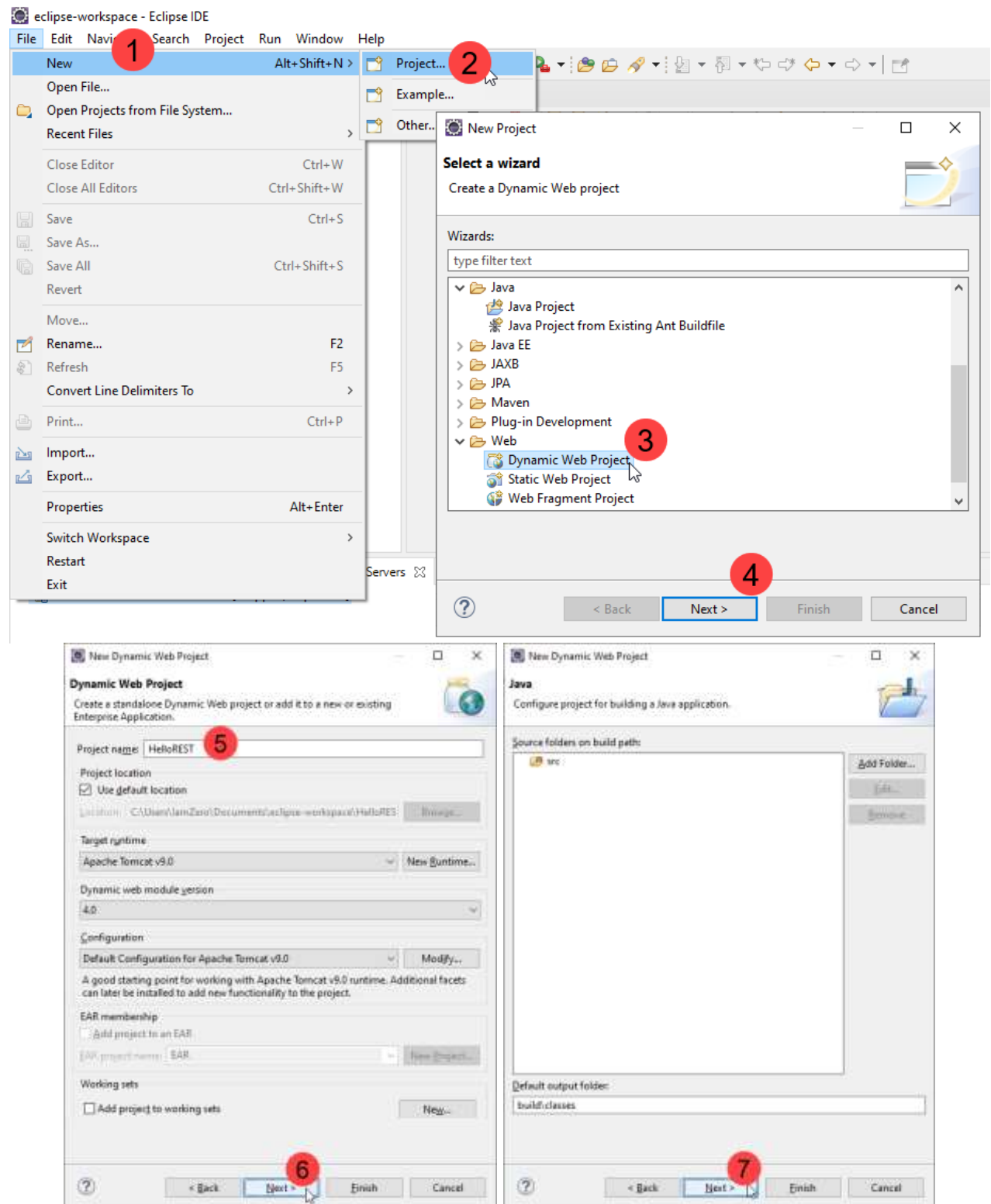
3) รายละเอียดกิจกรรม

กิจกรรมที่ 4 ประกอบด้วยกิจกรรมย่อยที่ต้องฝึกปฏิบัติตามลำดับกิจกรรม ดังนี้

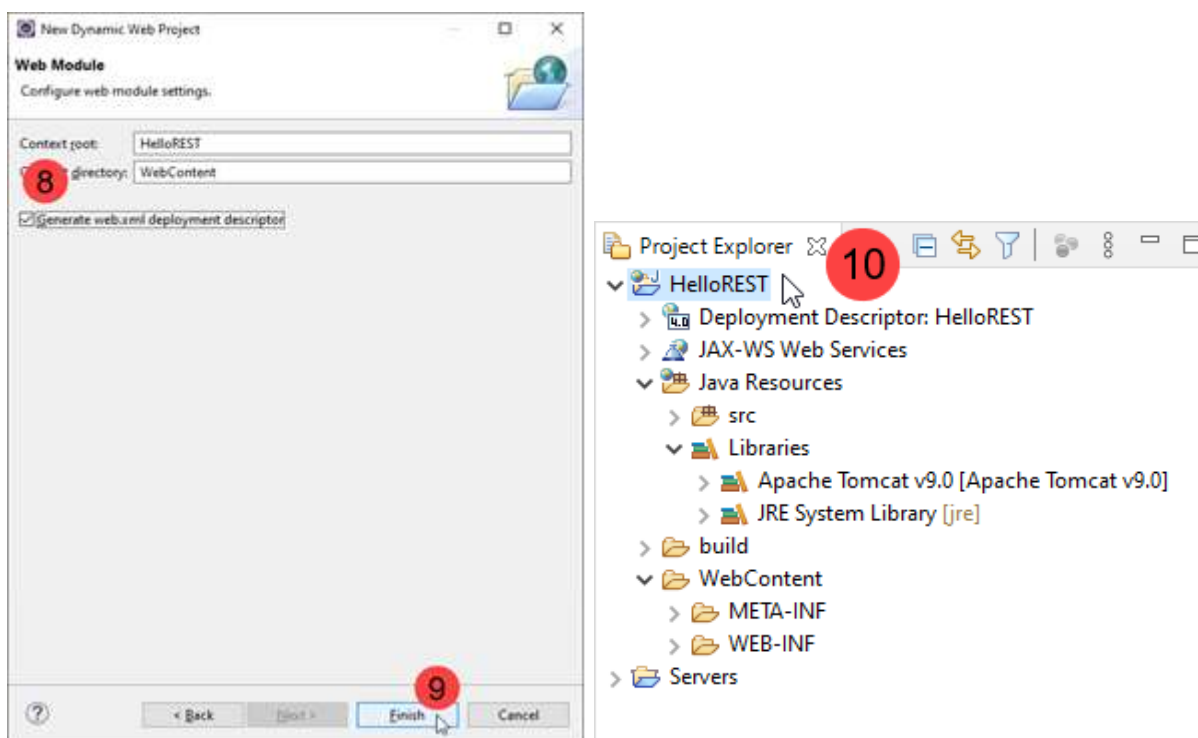
กิจกรรม 4.1 การสร้างไฟล์โปรเจกต์ทำงานทางฝั่งเซิร์ฟเวอร์

เป็นการสร้างไฟล์โปรเจกต์เพื่อให้ทำงานทางฝั่งเซิร์ฟเวอร์ในรูปแบบการใช้ Jersey library ร่วมกับเครื่องมือ build tool แบบ Maven ในโปรแกรม eclipse เพื่อสร้าง RESTful Web Service ดังนี้

4.1.1. สร้างไฟล์โปรเจกแบบ Dynamic Web Project และตั้งชื่อไฟล์โปรเจกเป็น HelloREST พร้อมทั้งกำหนดสร้างไฟล์ web.xml ดังภาพที่ 4.1.1



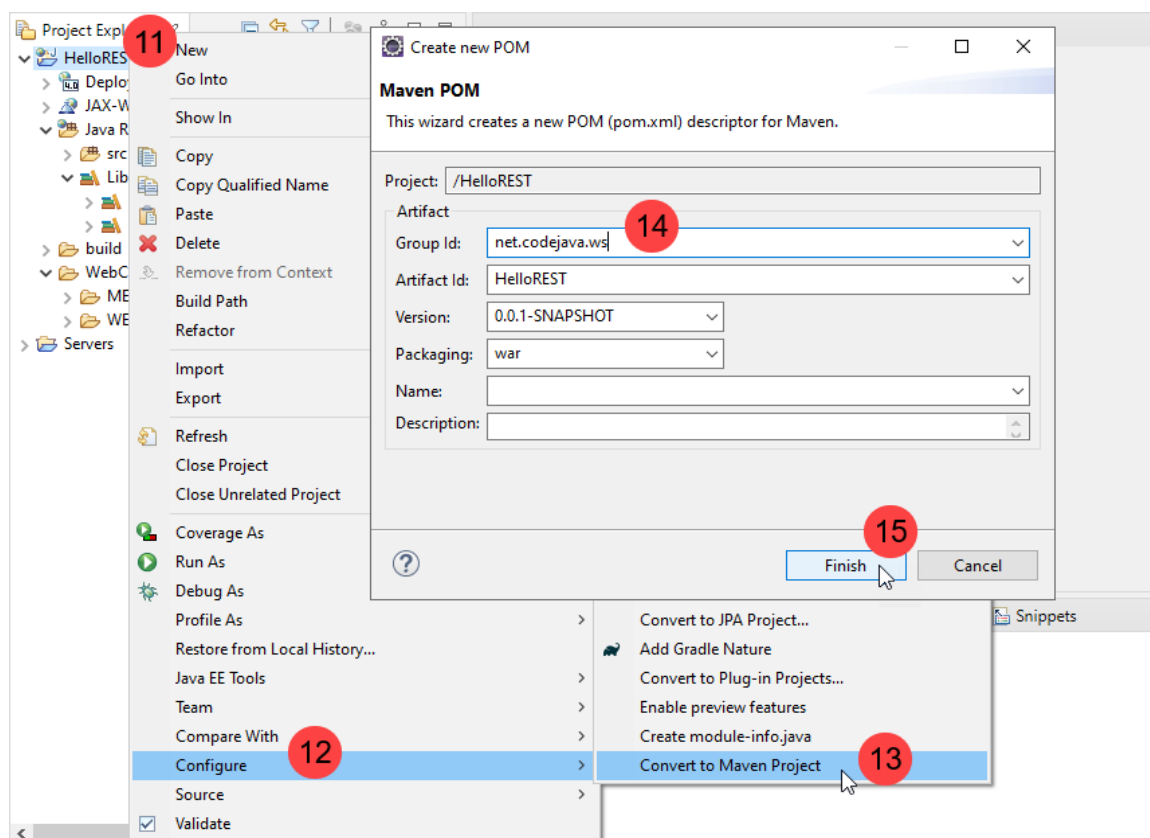
ก. ขั้นตอนการสร้างไฟล์โปรเจกชื่อ HelloREST



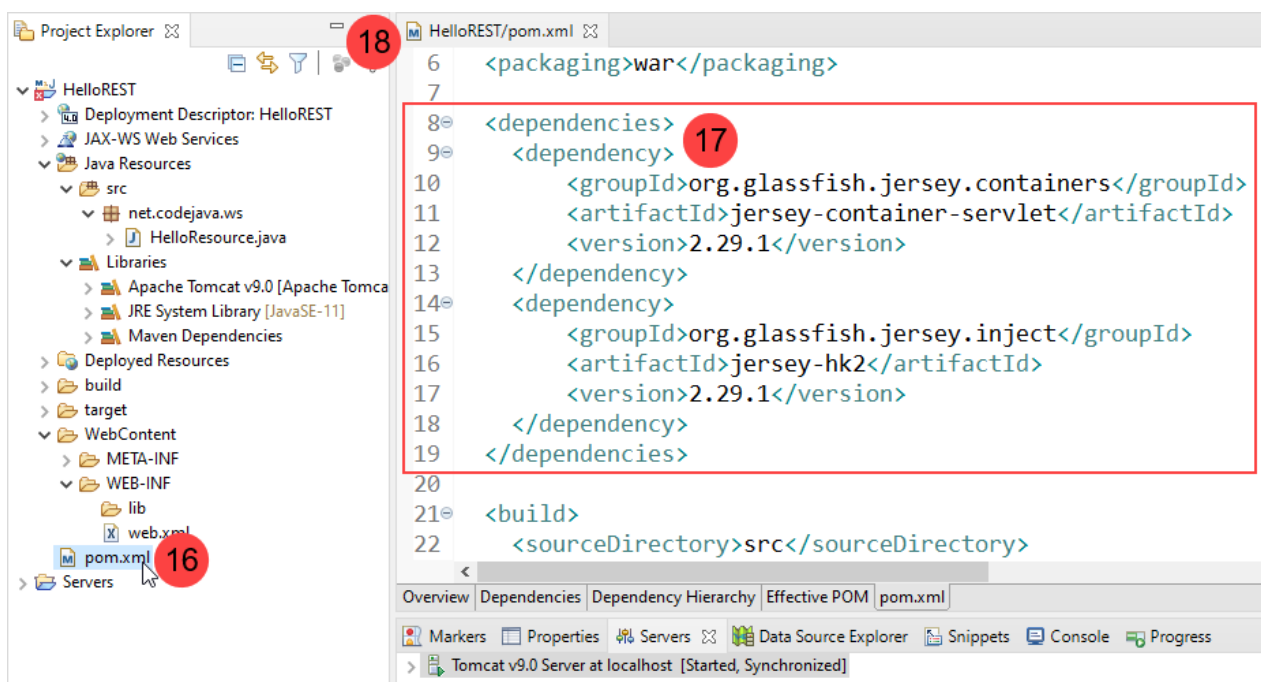
ข. การกำหนดไฟล์ web.xml และหน้าต่างผลลัพธ์ของการสร้างไฟล์โปรเจค HelloREST

ภาพที่ 4.1.1 ลำดับการสร้างและหน้าต่างผลลัพธ์ของไฟล์ HelloREST

4.1.2. ทำการปรับแต่ง (configure) ไฟล์ HelloREST ด้วยการแปลงไฟล์เป็นรูปแบบ Maven เพื่อสะดวกต่อการใช้ Jersey library ดังภาพที่ 4.1.2



ก. การปรับแต่งไฟล์เป็นรูปแบบ Maven

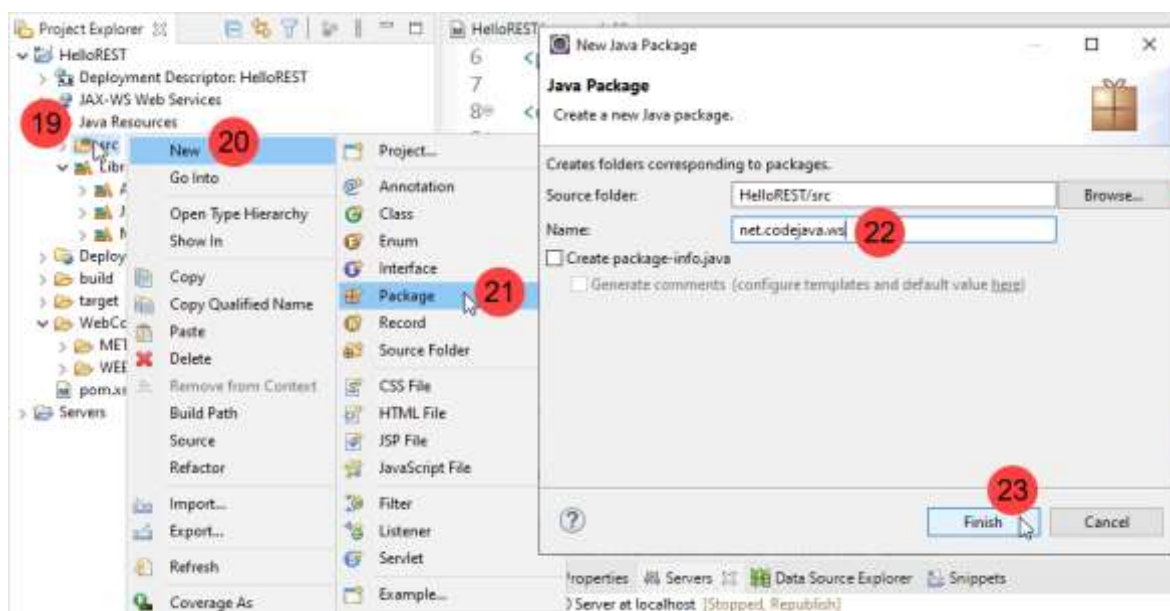


ข. หน้าต่างรูปแบบไฟล์ pom.xml และการเพิ่มแท็กคำสั่ง

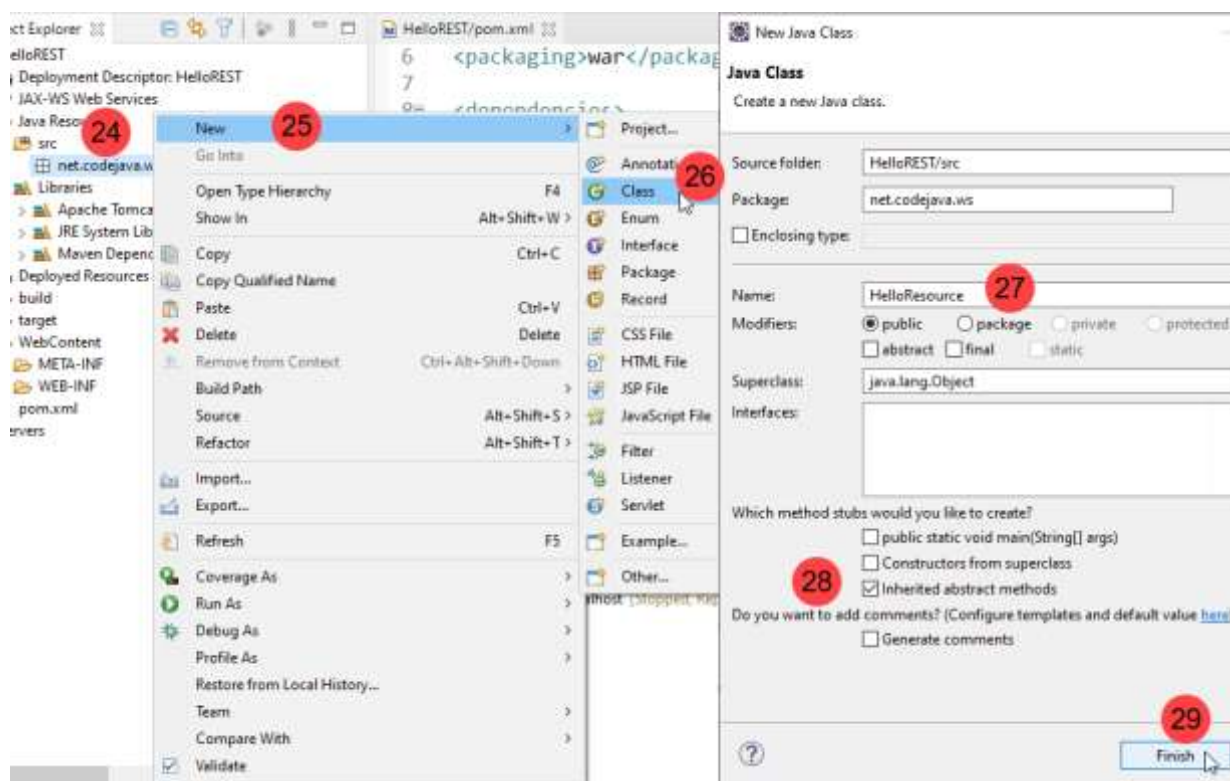
ภาพที่ 4.1.2 ลำดับการแปลงไฟล์เป็นแบบ Maven และเพิ่ม jersey library

เปิดไฟล์ pom.xml ตามหมายเลข 16 และแทรก เพิ่มแท็กคำสั่ง dependency ตามในกรอบหมายเลข 17 ลงในไฟล์ ดังภาพที่ 4.1.2 ข. แล้วทำการ save ไฟล์ (หมายเลข 18) และทำให้โปรแกรม eclipse ทำการ download ไฟล์ที่จำเป็นทางอินเทอร์เน็ต

4.1.3. ทำการสร้างไฟล์แบบ package ชื่อ net.codejava.ws และไฟล์แบบ class ชื่อ HelloResource.java ดังภาพที่ 4.1.3



ก. การสร้างไฟล์แบบ package



ข. การสร้างไฟล์แบบ class ชื่อ HelloResource ภายใต้ไฟล์ package

ภาพที่ 4.1.3 ลำดับการสร้างไฟล์ package และไฟล์ class

4.1.4. เพิ่มคำสั่งตามกรอบหมายเลข 30 ในไฟล์ HelloResource ดังภาพที่ 4.1.4 แล้ว save ไฟล์

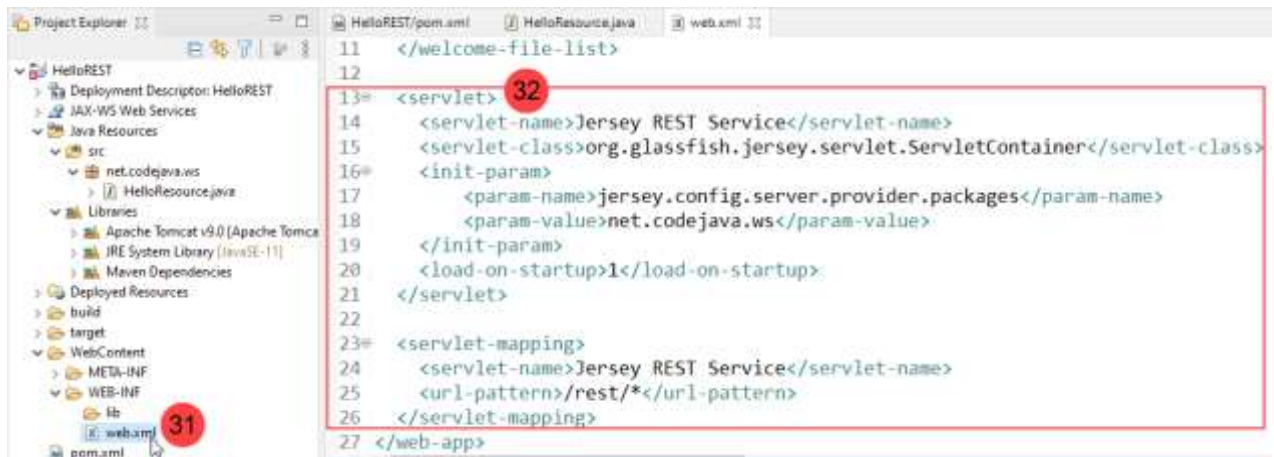
```

1 package net.codejava.ws;
2
3 import javax.ws.rs.GET;
4 import javax.ws.rs.Path;
5 import javax.ws.rs.Produces;
6 import javax.ws.rs.core.MediaType;
7
8 @Path("/stou")
9 public class HelloResource {
10     @GET
11     @Produces(MediaType.TEXT_PLAIN)
12     public String getStou() {
13         return "STOU Welcome";
14     }
15 }

```

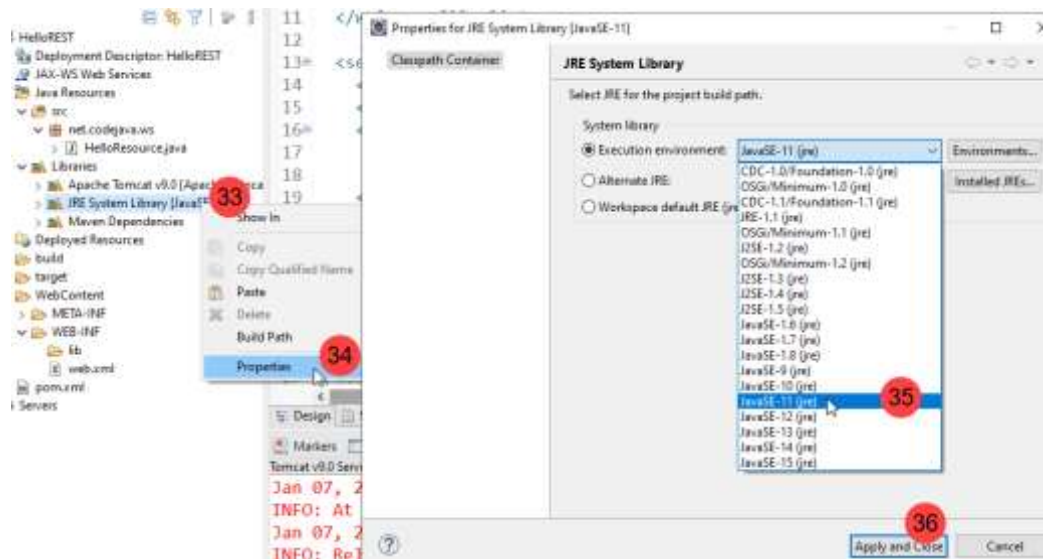
ภาพที่ 4.1.4 คำสั่งในไฟล์ HelloResource.java

4.1.5. เปิดไฟล์ web.xml เพื่อปรับแต่ง (configure) ด้วยการแทรก เพิ่มข้อมูลคำสั่ง Jersey Servlet ตามกรอบหมายเลข 32 ดังภาพที่ 4.1.5 แล้ว save ไฟล์



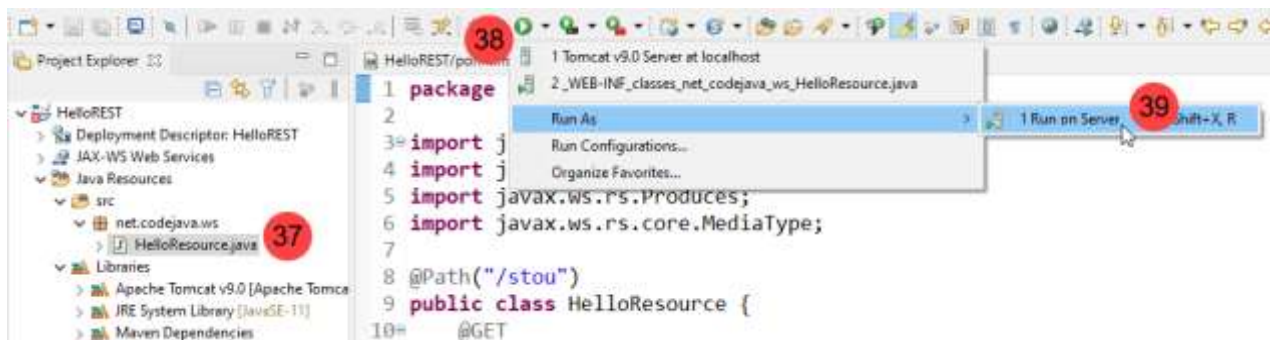
ภาพที่ 4.1.5 ชุดคำสั่ง Jersey Servlet ในไฟล์ web.xml

4.1.6. ทำการเปลี่ยนรุ่นของ JRE System Library เป็น JavaSE-11 ตามลำดับดังภาพที่ 4.1.6

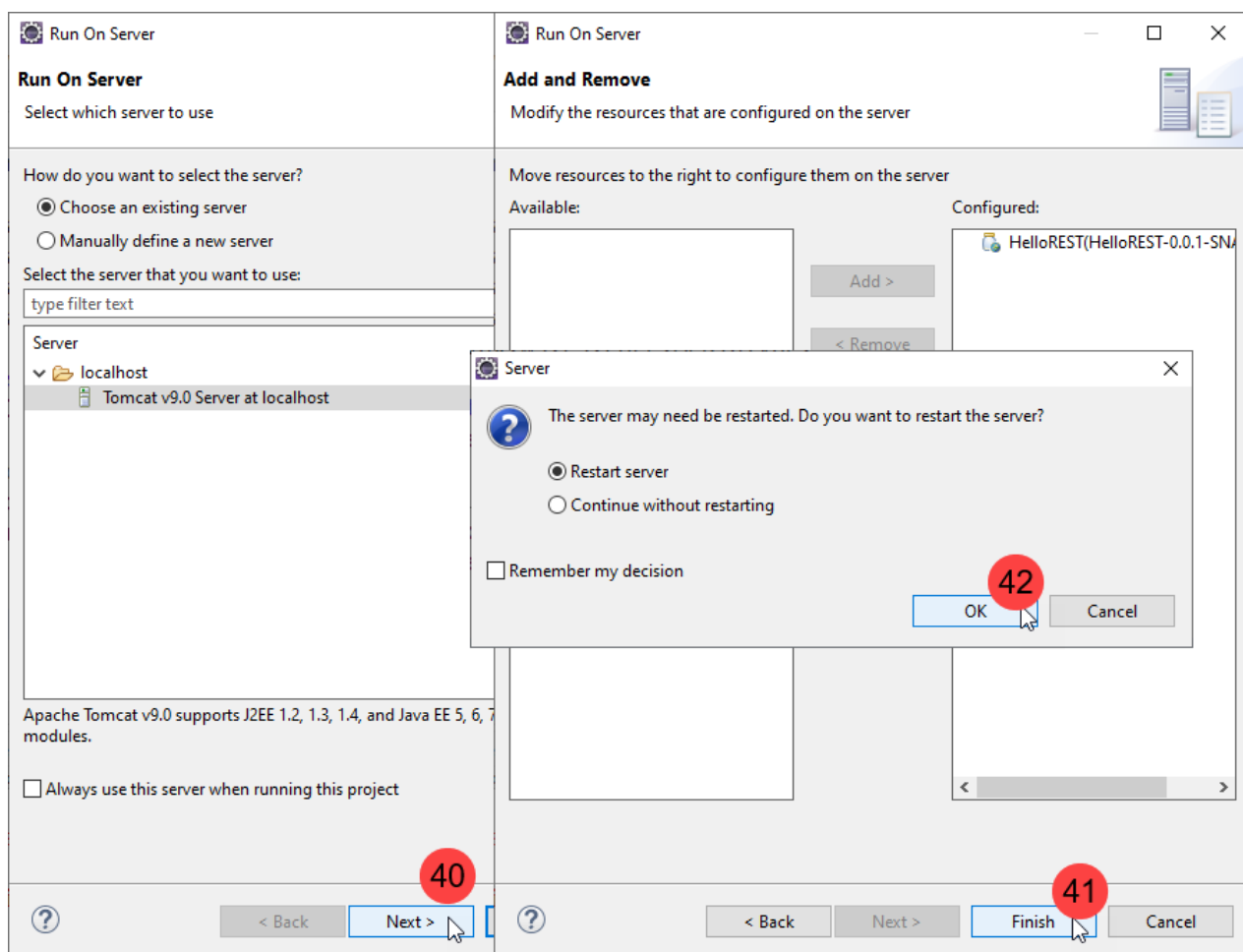


ภาพที่ 4.1.6 การเปลี่ยนรุ่นของ JRE System Library

4.1.7. เลือกไฟล์ HelloResource.java เพื่อทำการ Run as แบบทำงานบน Tomcat Server ดังภาพที่ 4.1.7



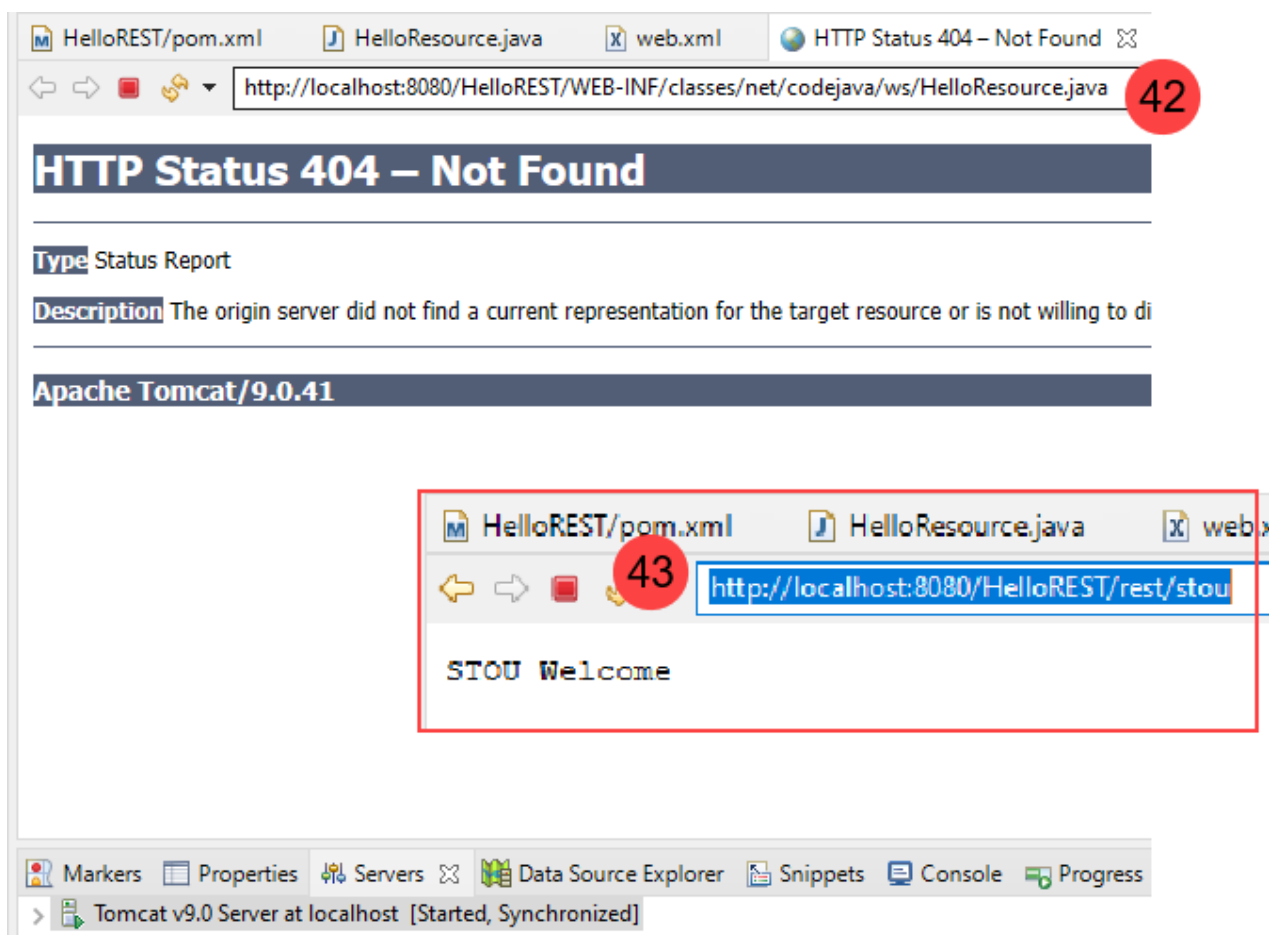
ก. การสั่งรันโปรแกรมของไฟล์ HelloResource.java



ข. การเลือก option ของการรันโปรแกรม Tomcat Server

ภาพที่ 4.1.7 การเริ่มการทำงานของไฟล์บน Tomcat Server

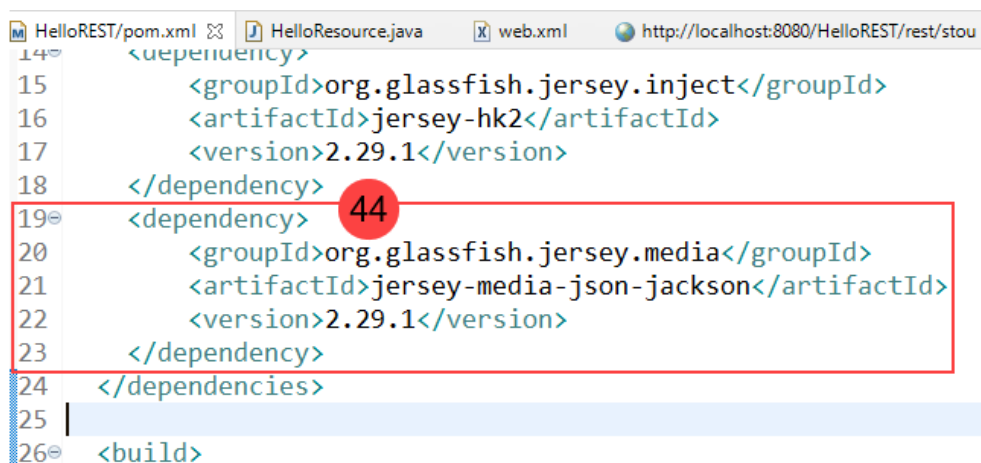
4.1.8. ผลลัพธ์จากการรัน และการปรับแก้ URL ใหม่ คือ “http://localhost:8080/HelloREST/rest/stou” ตามหมายเลข 43 แล้วกด Enter ได้ผลดังภาพที่ 4.1.8 ซึ่งเป็นวิธีการทดสอบการทำงานของ RESTful Web Service รูปแบบหนึ่ง



ภาพที่ 4.1.8 ผลจากการรันไฟล์ HelloResource.java

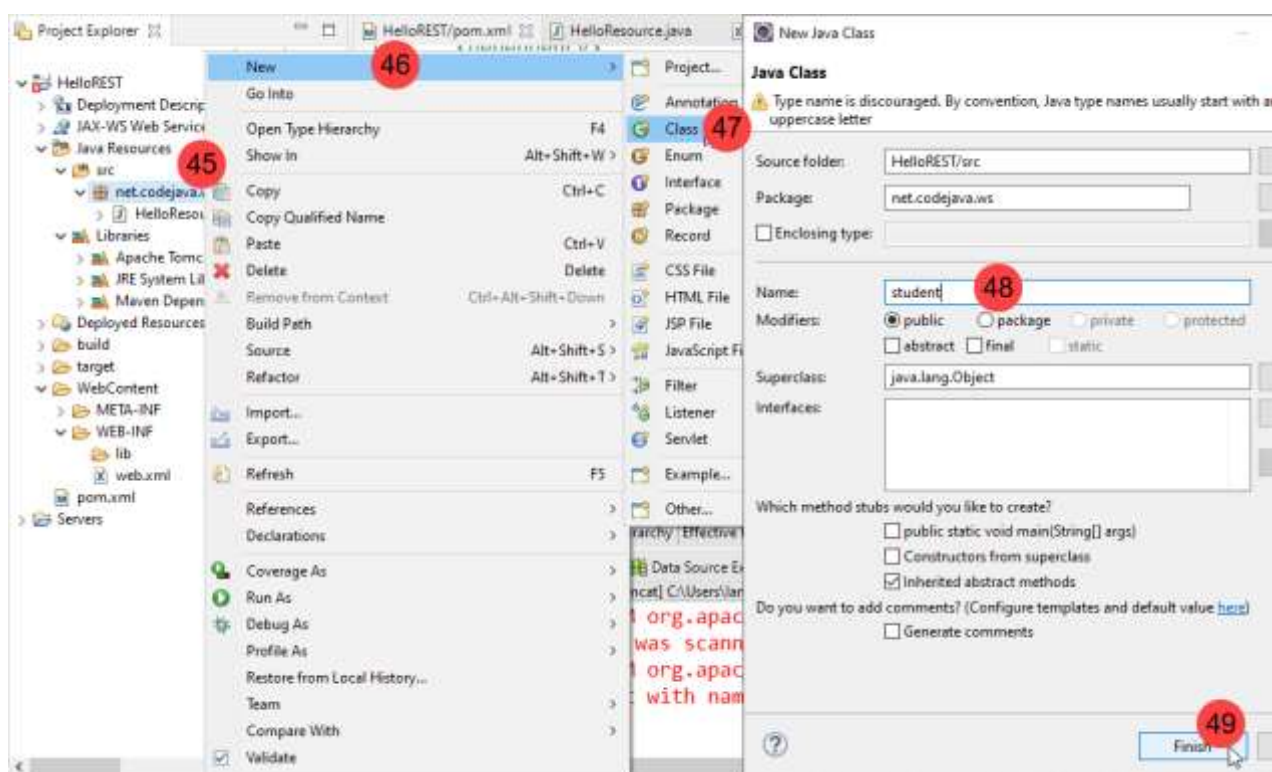
กิจกรรม 4.2 การทำงานของ RESTful Web Service กับข้อมูลแบบ JSON

4.2.1. เปิดไฟล์ pom.xml และเพิ่มคำสั่งตามหมายเลข 44 ดังภาพที่ 4.2.1 แล้วทำการ save ไฟล์



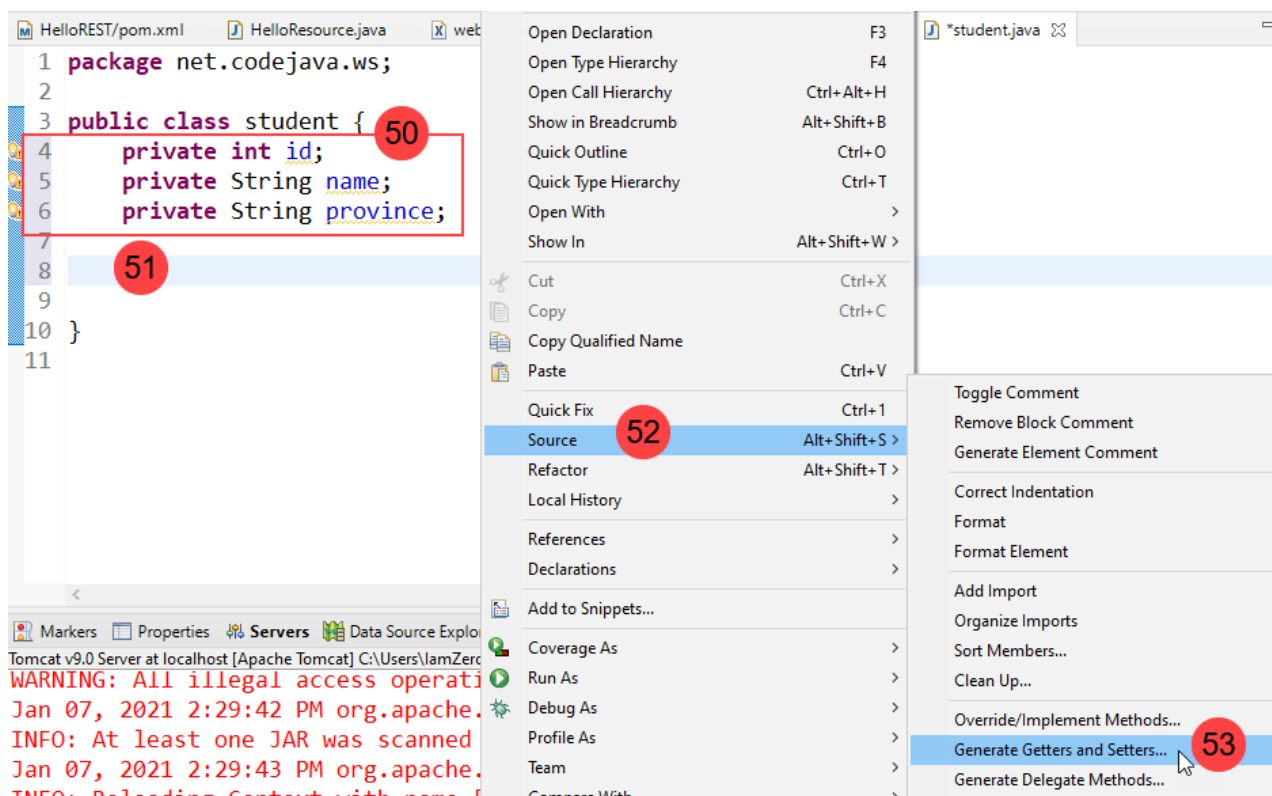
ภาพที่ 4.2.1 การเพิ่มคำสั่งสำหรับการทำงานกับข้อมูลแบบ JSON

4.2.2. สร้าง class ไฟล์ข้อมูลแบบ JSON ชื่อ student.java ดังภาพที่ 4.2.2

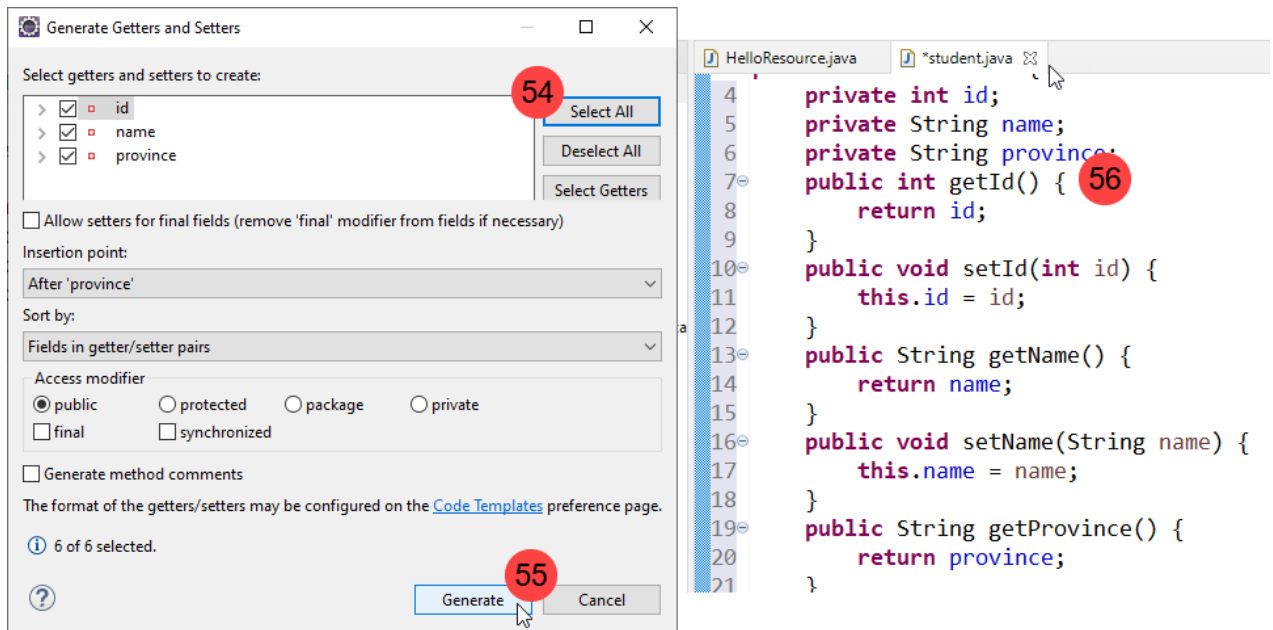


ภาพที่ 4.2.2 การสร้างไฟล์ชื่อ student.java

4.2.3. เปิดไฟล์ student.java แล้วเพิ่มตัวแปรตามในกรอบหมายเลข 50 และในบรรทัดถัดมา คลิกขวามบนเมาส์ (หมายเลข 51) และทำตามลำดับ ดังภาพที่ 4.2.3



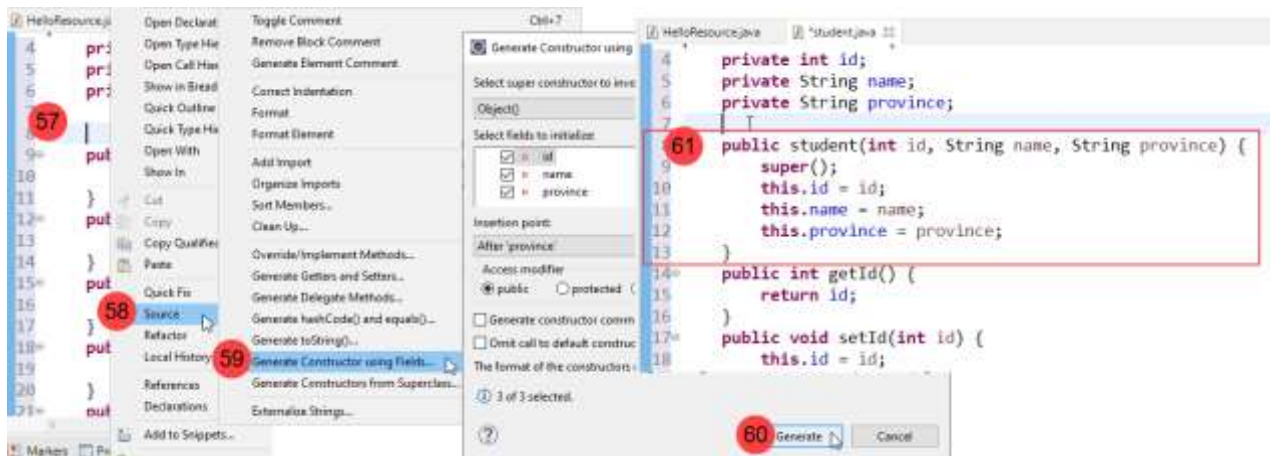
ก. การเพิ่มตัวแปรและคำสั่งในไฟล์ student.java



ข. ผลของการ Generate Getters and Setters ของตัวแปรในหมายเลข 50

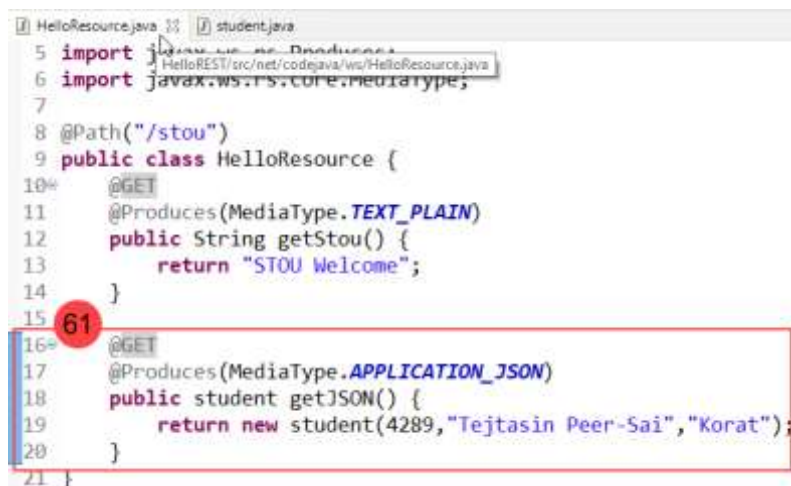
ภาพที่ 4.2.3 การเพิ่มคำสั่งในไฟล์ student.java

4.2.4. แทรกคำสั่งเพิ่มเติมในไฟล์ student.java ด้วยการคลิกขวานเมาส์ (หมายเลข 57) และทำตามลำดับ ดังภาพที่ 4.2.4



ภาพที่ 4.2.4 การแทรกคำสั่งเพิ่มเติมในไฟล์ student

4.2.5 เปิดไฟล์ HelloResource.java เพื่อแทรกคำสั่งสำหรับการจัดการข้อมูลแบบ JSON ตามกรอบหมายเลข 61 ดังภาพที่ 4.2.5



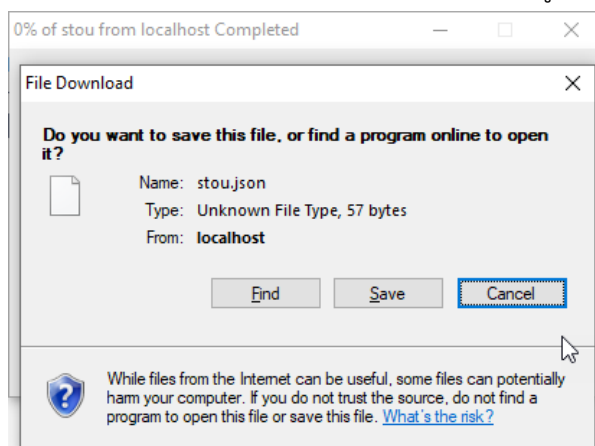
```

5 import javax.ws.rs.Produces;
6 import javax.ws.rs.core.MediaType;
7
8 @Path("/stou")
9 public class HelloResource {
10     @GET
11     @Produces(MediaType.TEXT_PLAIN)
12     public String getStou() {
13         return "STOU Welcome";
14     }
15
16     @GET
17     @Produces(MediaType.APPLICATION_JSON)
18     public student getJSON() {
19         return new student(4289, "Tejtasin Peer-Sai", "Korat");
20     }
21 }

```

ภาพที่ 4.2.5 การแทรกคำสั่งในไฟล์ HelloResource.java

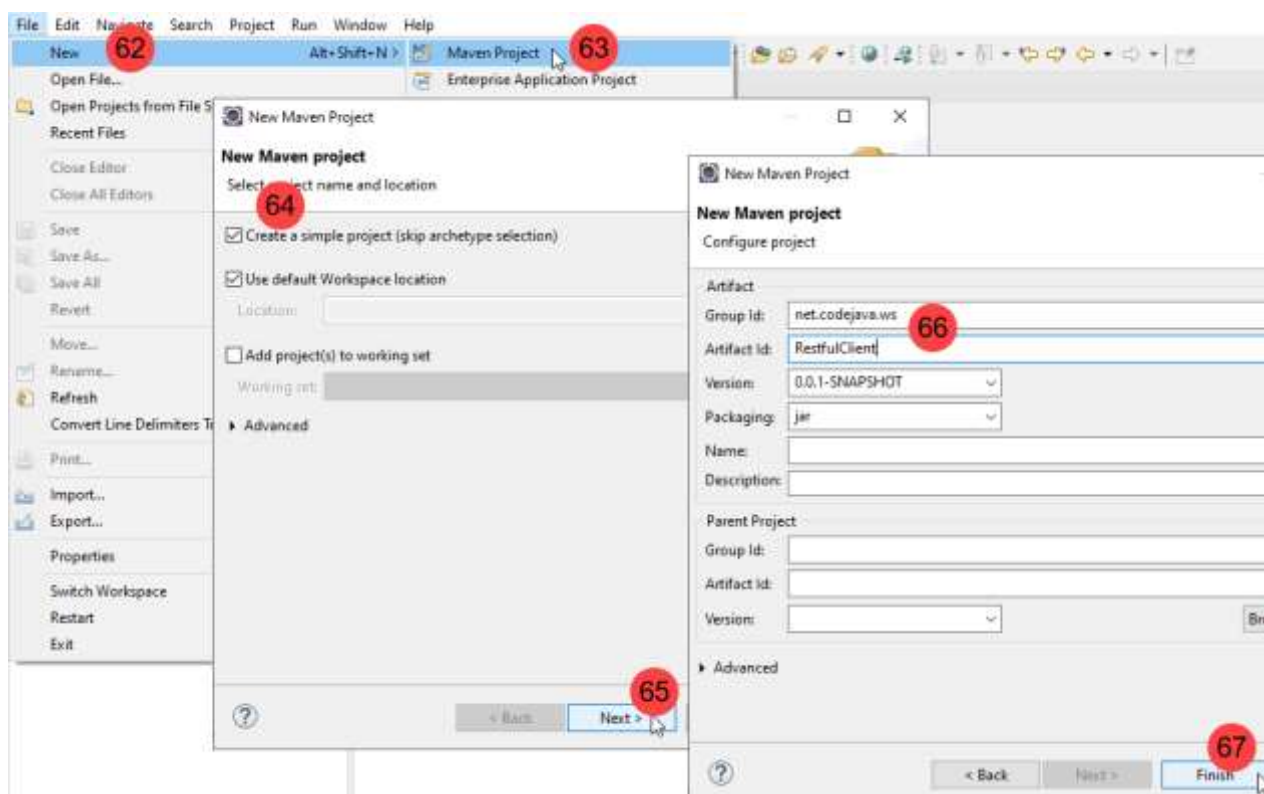
4.2.6. ทดสอบโปรแกรมด้วยการรันไฟล์ HelloResource.java และทำการป้อน address ของ URL ใหม่ด้วย “http://localhost:8080/HelloREST/rest/stou” และได้ผลลัพธ์ของไฟล์ข้อมูลแบบ JSON ดังภาพที่ 4.2.6



ภาพที่ 4.2.6 ผลของการรันกับรูปแบบข้อมูลแบบ JSON

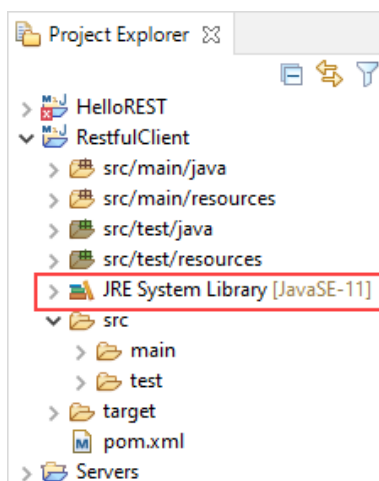
กิจกรรม 4.3 การเรียกใช้ RESTful Web Service ทางฝั่งไคลเอนต์

4.3.1. สร้างไฟล์โปรเจคใหม่ชื่อ RestfulClient สำหรับการทำงานทางฝั่ง Client ด้วยเครื่องมือ Maven project ตามลำดับ ดังภาพที่ 4.3.1



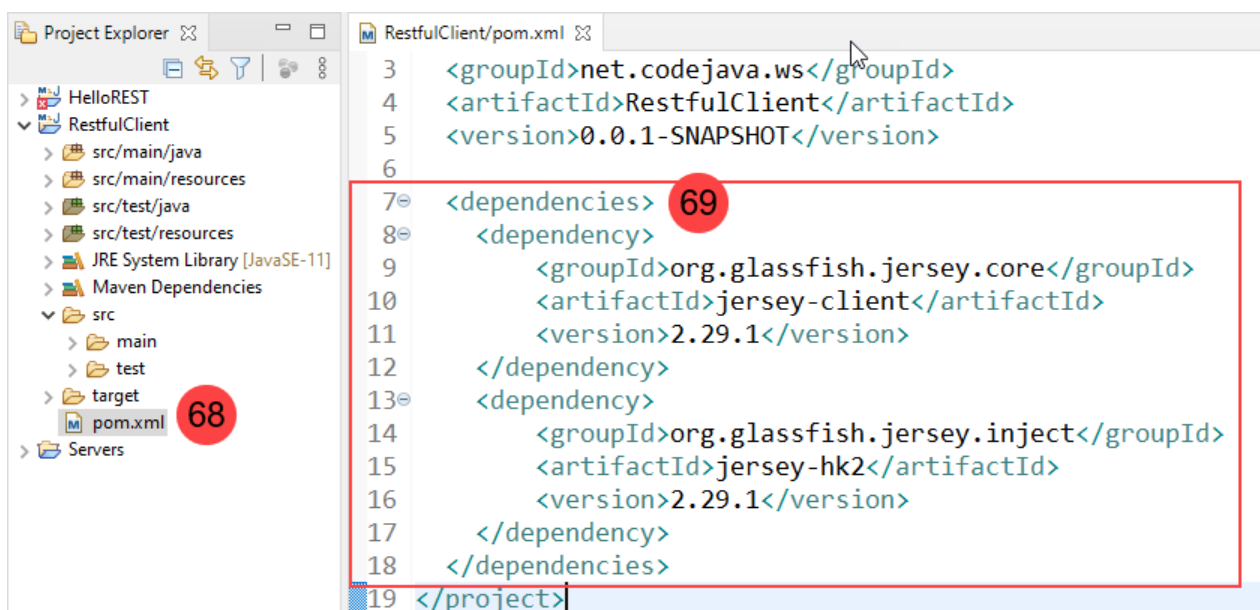
ภาพที่ 4.3.1 การสร้างไฟล์โปรเจก RestfulClient

4.3.2. เปลี่ยนรุ่นของ JRE System Library จาก J2SE-1.5 เป็น JavaSE-11 (ขั้นตอนเหมือน หัวข้อ 4.1.6) ได้ผลดังภาพที่ 4.3.2



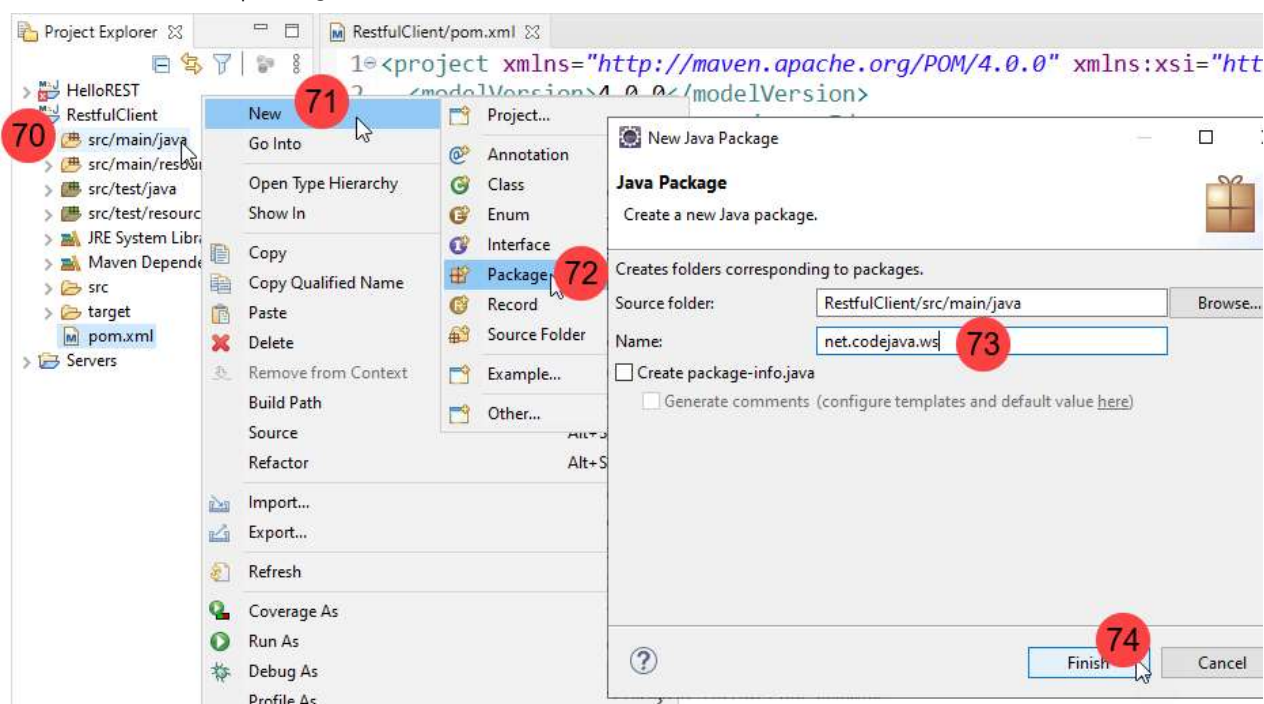
ภาพที่ 4.3.2 การเปลี่ยนรุ่นของ JRE System Library

4.3.3. เปิดไฟล์ pom.xml ของโปรเจก RestfulClient เพื่อกำหนดรูปแบบการทำงานแบบ Jersey RESTful ตามลำดับ ดังภาพที่ 4.3.3

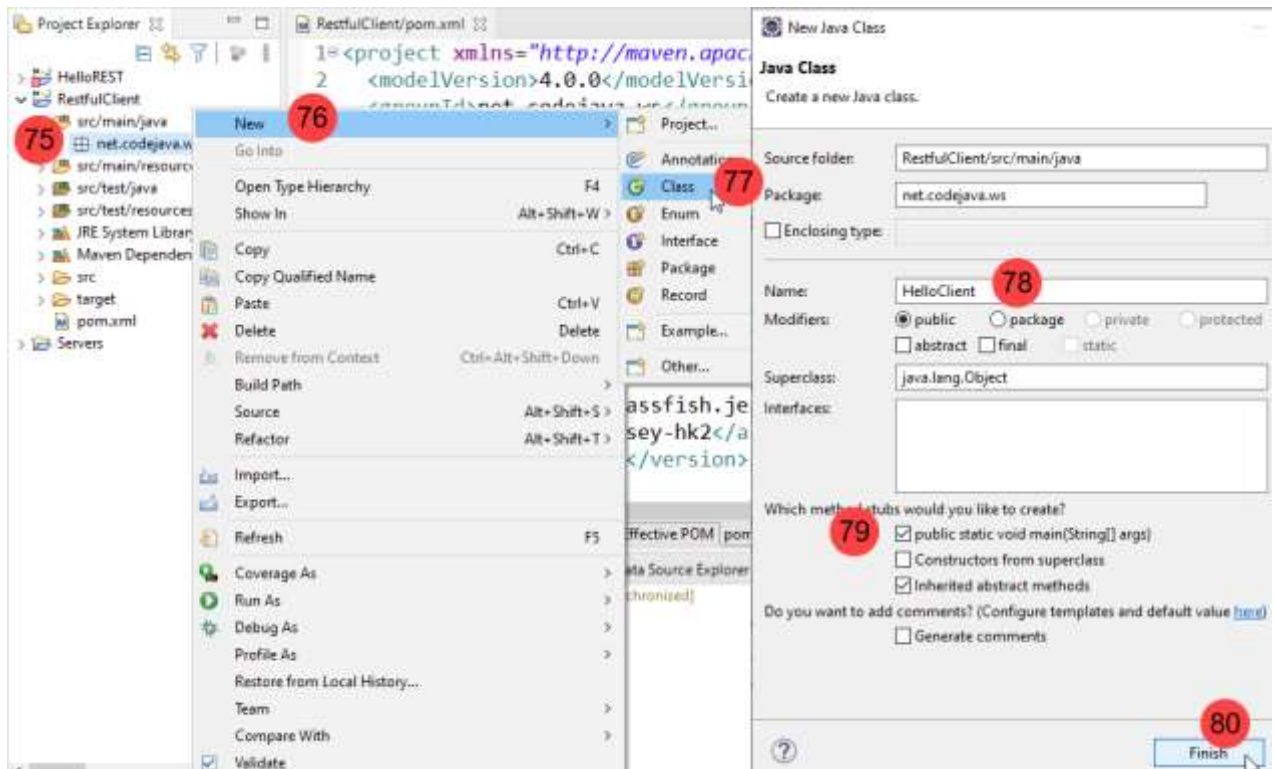


ภาพที่ 4.3.3 การเพิ่มคำสั่งเพื่อกำหนดการทำงานแบบ Jersey ในไฟล์ pom.xml

4.3.4. สร้างไฟล์แบบ package ชื่อ net.codejava.ws และไฟล์แบบคลาส ชื่อ HelloClient.java



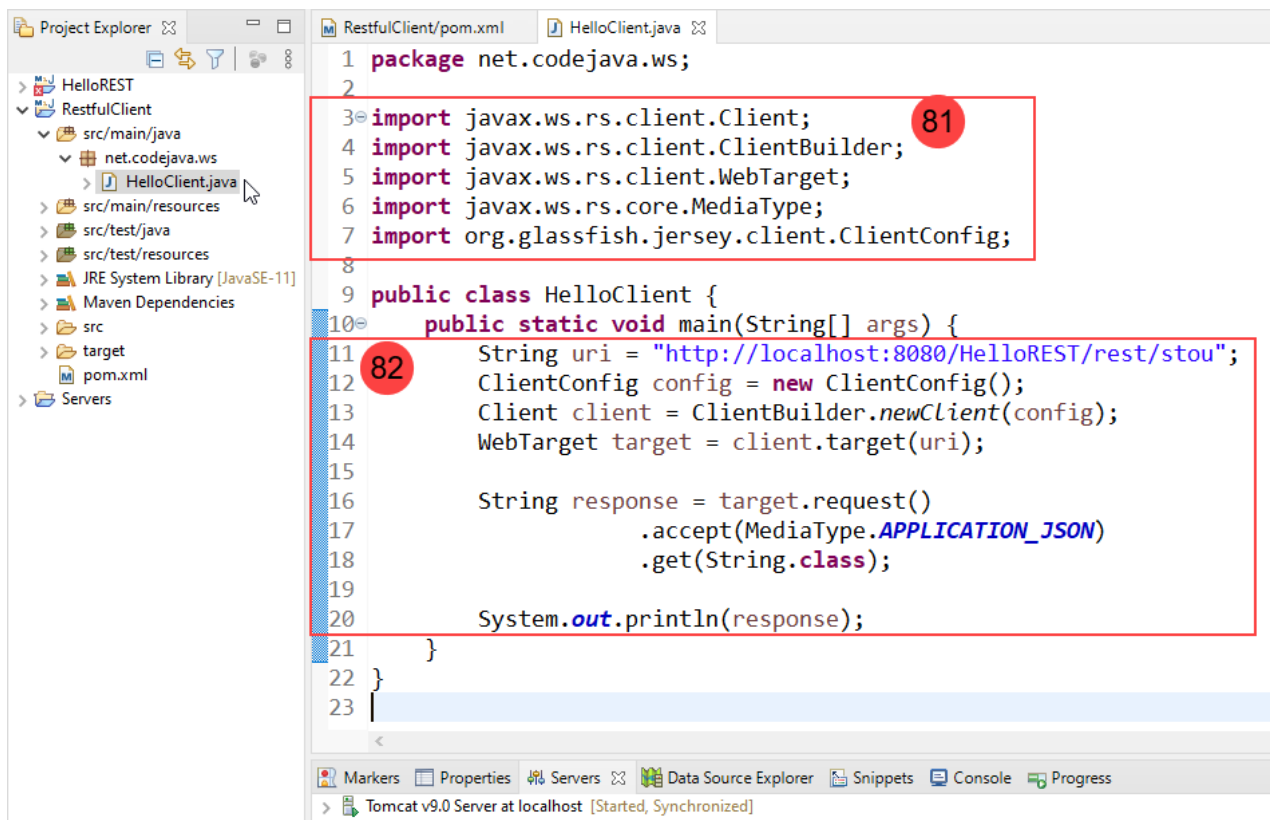
ก. การสร้างไฟล์ package ชื่อ net.codejava.ws



ข. การสร้างไฟล์แบบคลาส ชื่อ HelloClient.java

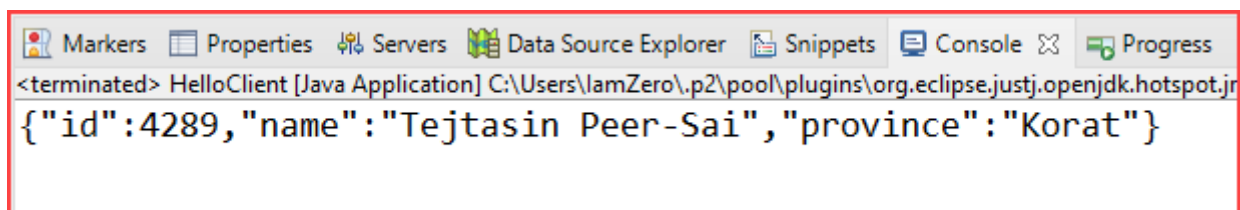
ภาพที่ 4.3.4 ลำดับการสร้างไฟล์แบบ package และแบบคลาส

4.3.5. เปิดไฟล์ HelloClient.java เพื่อแทรกคำสั่ง ดังภาพที่ 4.3.5



ภาพที่ 4.3.5 การแทรกคำสั่งในไฟล์ HelloClient.java

4.3.6. ทำการรันไฟล์ HelloClient.java ทางฝั่ง Client และได้ผลลัพธ์ดังภาพที่ 4.3.6



ภาพที่ 4.3.6 ผลการรันไฟล์โปรเจคทางฝั่ง Client