



## Restaurant Order Management

### จัดทำโดย

- นายเจษฎาพร ไตรวินิจศรีสุข 61070028
- นางสาวฉัตรธิดา แจ้งใจ 61070029
- นายชาญวิทย์ เศรษฐวงศ์สิน 61070040
- นายณภัทร อารยวัฒนาพงษ์ 61070045
- นายระพีพันธ์ มูนไทย 61070181

### เสนอ

ดร. สุพัณณดา โพธิพันธ์  
ดร. ธนาวิเชชฐ์ ชิติจรูญโรจน์

รายงานฉบับนี้ เป็นส่วนหนึ่งของวิชา OBJECT-ORIENTED PROGRAMMING  
ภาคเรียนที่ 1 ปีการศึกษา 2562

หลักสูตรวิทยาศาสตรบัณฑิต คณะเทคโนโลยีสารสนเทศ สาขatechโนโลยีสารสนเทศ  
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

## ចំណាំ ជូន

Restaurant Order Management

1. นายเจษฎาพร ไตรวนิจศรีสุข 61070028
2. นางสาวฉัตรธิดา แจ้งใจ 61070029
3. นายชาญวิทย์ เศรษฐวงศ์สิน 61070040
4. นายณรงค์ อารยะวัฒนาพงษ์ 61070045
5. นายระพีพันธ์ มุนไทย 61070181

อาจารย์ที่ปรึกษา ดร. สุพัณณดา ใจติพันธ์, ดร. ธรรมิเชษฐ์ วิจิจญูโรจน์  
สถานที่ คณะเทคโนโลยีสารสนเทศ (สจล.)  
ปีที่พิมพ์ 2562

บทคัดย่อ

โครงงานฉบับนี้เป็นส่วนหนึ่งของวิชา 06016317 OBJECT-ORIENTED PROGRAMMING โดยมีจุดประสงค์เพื่อการศึกษาความรู้ ความเข้าใจในการเขียนโปรแกรมภาษา Java ซึ่งผู้จัดทำได้เลือกหัวข้อเรื่อง Network Programming ในการทำโปรเจค Restaurant Order Management ซึ่งโปรเจค Restaurant Order Management คือระบบในการสั่งอาหารในร้านอาหาร ที่เลือกทำโปรเจคนี้ เพราะว่า ในปัจจุบันมีร้านอาหารก่อตั้งขึ้นมากหลายแห่ง ทางผู้จัดทำได้เล็งเห็นถึงความสำคัญในการสั่งรายการอาหารที่รวดเร็ว และ ประหยัดเวลา จึงทำให้โปรเจคนี้ได้เกิดขึ้น ซึ่งการสั่งรายการอาหารของโปรแกรม จะมีทั้งระบบของ Client และ Server รวมไปถึงระบบสั่งการพิมพ์ใบเสร็จ โดยใช้การส่งข้อมูลขอเดอร์จาก Client ผ่าน Socket ในรูปแบบของ JSON ไปยัง Server

ทางคณะผู้จัดทำได้ดำเนินการศึกษาโดยสืบหาข้อมูล และนำความรู้ที่ได้เรียนรู้มาเพื่อนำมาพัฒนาโครงการนี้ให้มีคุณภาพ และพร้อมแก่การใช้งานให้มากที่สุด ผู้จัดทำหวังว่าจะเป็นประโยชน์แก่ผู้ที่ต้องการศึกษา และนำไปใช้งาน

# สารบัญ

เรื่อง	หน้า
- วัตถุประสงค์	1
- ประโยชน์ที่ได้รับ	1
- ไดอาแกรม	
▪ ไดอาแกรม (Client)	2
▪ ไดอาแกรม (Server)	3
- โค๊ด	
▪ โค๊ด (Client)	4
○ Class Main4	
○ Class SplashScreen	4
○ Class GUI	5 - 7
○ Class Order	7 - 8
○ Class Total	9
○ Class PrintReceipt	10 - 11
○ Class Json	11
○ Class Clock	12
○ Class Network	13
▪ โค๊ด (Server)	
○ Class Main	14 - 16
○ Class SplashScreen	16 - 17
○ Class GUI	17 - 18
○ Class Countdown	19
○ Class Clock	20
○ Class Network	20 - 21
- หน้าที่สมาชิก	21

## วัตถุประสงค์

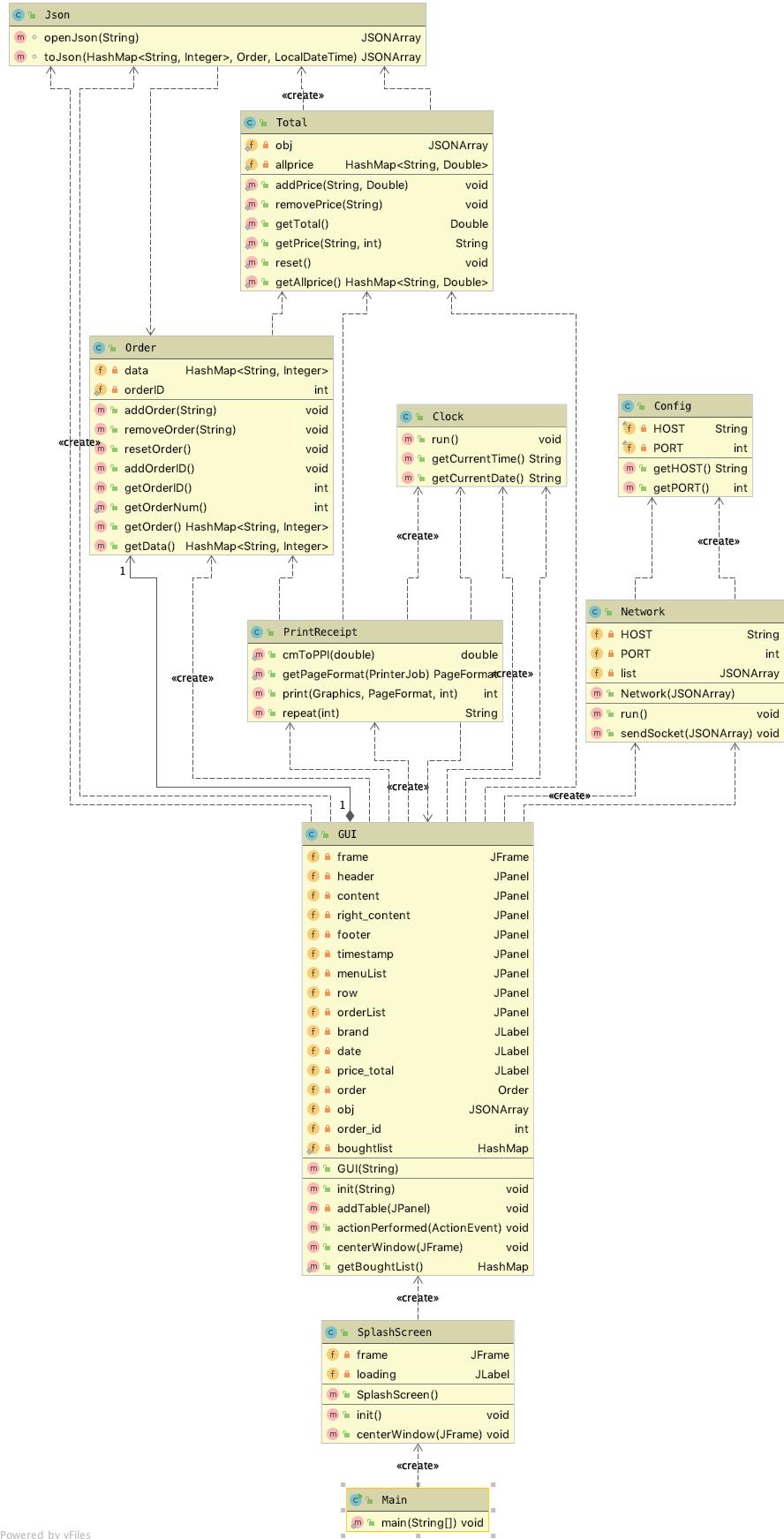
1. เพื่อให้สามารถส่งอเดอร์ไปที่ครัวได้ทันที จากหน้าเคเตอร์ โดยที่ไม่ต้องรอนาน
2. ประหยัดเวลา และ มีความสะดวกรวดเร็ว แก่การสั่งอเดอร์
3. ไม่ต้องใช้กระดาษ ประหยัดงบประมาณในการขาย
4. เพื่อให้เป็นโปรแกรมที่เข้าใจง่าย มีความสวยงาม เหมาะสมแก่การใช้งาน

## ประโยชน์ที่ได้รับ

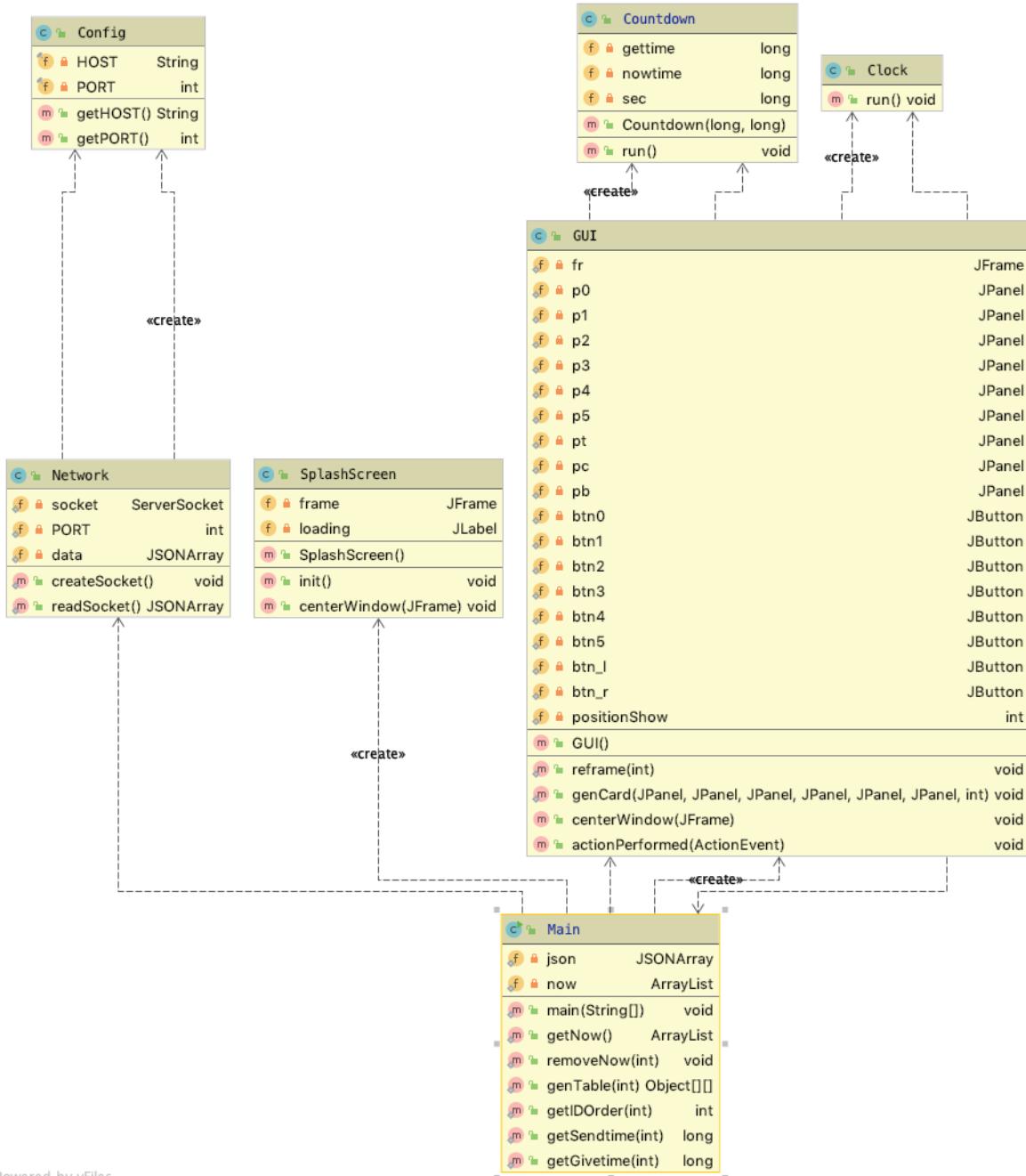
1. ตัวผู้พัฒนาได้มีการทบทวนบทเรียน และ มีความเข้าใจในภาษา Java มากยิ่งขึ้น
2. เพื่อให้สามารถนำภาษา Java มาประยุกต์ใช้กับ Network Programming ได้
3. เพื่อให้ผู้ที่เดินนำโปรเจคนี้ไปใช้มีความพึงพอใจในผลงานของผู้พัฒนา
4. ผู้พัฒนาสามารถต่อยอดไปในด้านต่าง ๆ ได้

# ໄດ້ອາແກຣມ (Diagram)

## ໄດ້ອາແກຣມ (Client)



# ໄດ້ອາແກຣມ (Server)



Powered by yFiles

# โค้ด (Code)

## Code (Client)

1. Class Main -> เป็น Class ที่ใช้ run โปรแกรม

```
public class Main {  
    public static void main(String[] args) { new SplashScreen(); }  
}
```

- Method main - ทำหน้าที่สร้าง Object จาก Class SplashScreen

2. Class SplashScreen -> ทำหน้าต่าง Splash Screen

```
public class SplashScreen {  
    private JFrame frame;  
    private JLabel loading;  
  
    public SplashScreen() { this.init(); }  
  
    public void init() {  
        // frame  
        JFrame frame = new JFrame();  
        // icon for windows  
        frame.setIconImage(new ImageIcon( filename: "img/icon.png").getImage());  
        JLabel loading = new JLabel();  
        loading.setIcon(new ImageIcon( filename: "img/loading.png"));  
        frame.add(loading);  
        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);  
        frame.setUndecorated(true);  
        frame.setVisible(true);  
        frame.setResizable(false);  
        frame.pack();  
        centerWindow(frame);  
        try {  
            Thread.sleep( millis: 2000 );  
        } catch (Exception e) {  
            System.out.println(e);  
        }  
        frame.dispose();  
        new GUI( name: "Restaurant Order Management - Client" );  
    }  
}
```

- Constructor SplashScreen - ทำหน้าที่เรียก Method init();
- Method init - ทำหน้าที่สร้างหน้า Splash Screen และให้หยุดรอ 2 วินาที และให้เรียก Class GUI ขึ้นมา

```

public void centerWindow(JFrame frame) {
    Dimension dimension = Toolkit.getDefaultToolkit().getScreenSize();
    int x = (int) ((dimension.getWidth() - frame.getWidth()) / 2);
    int y = (int) ((dimension.getHeight() - frame.getHeight()) / 2);
    frame.setLocation(x, y);
}

```



- Method centerWindow - ทำให้ JFrame อยู่กลางหน้าจอ ซึ่งจะมี Method นี้อยู่ในทุก ๆ Class ที่เป็น JFrame

### 3. Class GUI -> สร้างรายละเอียด และ โครงสร้างต่าง ๆ ของ GUI

```

public class GUI implements ActionListener {
    private JFrame frame;
    private JPanel header, content, right_content, footer, timestamp, menuList, row, orderList;
    private JLabel brand, date, price_total;
    private Order order;
    private JSONArray obj;
    private int order_id = 1;
    private static HashMap boughtlist;

    public GUI(String name) { init(name); }

    public void init(String name) {
        // frame
        frame = new JFrame( title: "" + name);
        // icon for windows
        frame.setIconImage(new ImageIcon( filename: "img/icon.png").getImage());
        frame.setVisible(true);
        frame.setResizable(false);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        // order
        order = new Order();
        // header
        header = new JPanel();
        header.setLayout(new GridLayout( rows: 1, cols: 2));
        header.setBackground(Color.darkGray);
        // brand
        brand = new JLabel();
    }
}

```

- Constructor GUI - ทำหน้าที่เรียก Method init();

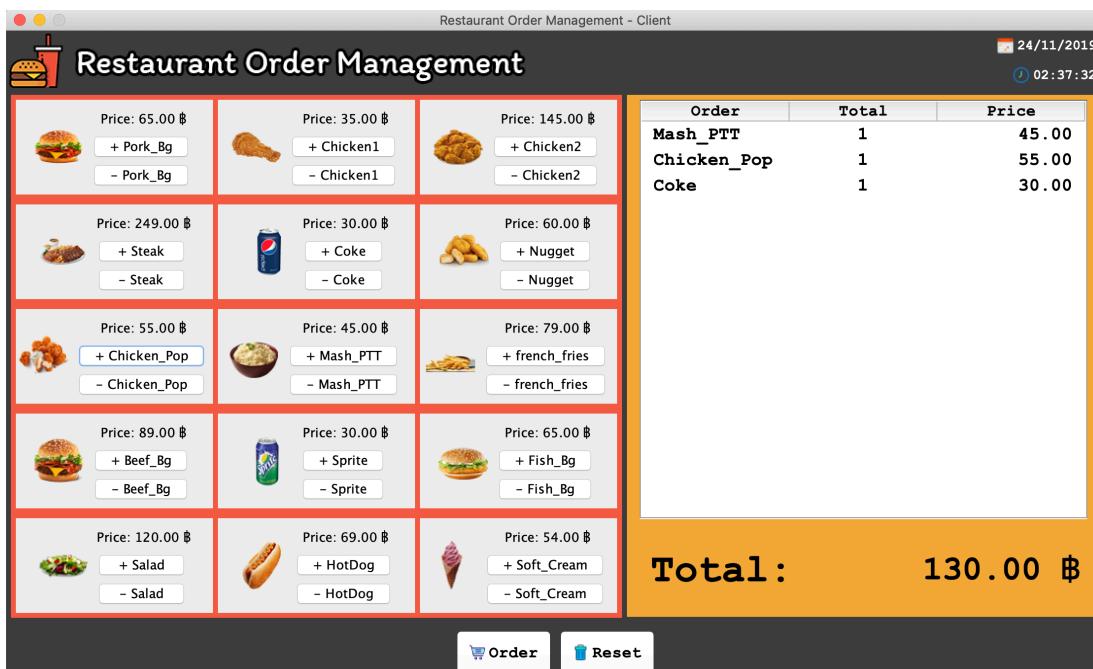
- Method init - สร้างหน้า GUI โดยมีรายละเอียดต่าง ๆ เช่น ตารางเมนู, รายการสั่งซื้อ

```

private void addTable(JPanel orderList) {
    orderList.removeAll();
    orderList.revalidate();
    orderList.repaint();
    HashMap<String, Integer> data = order.getOrder();
    Object[][] tableData = new Object[data.keySet().size()][3];
    String column[] = {"Order", "Total", "Price"};
    int index = 0;
    for (String key : data.keySet()) {
        tableData[index][0] = " " + key;
        tableData[index][1] = data.get(key);
        for (int i = 0; i < obj.size(); i++) {
            JSONObject obj1 = (JSONObject) obj.get(i);
            if (obj1.get("name").equals(key)) {
                double price = Double.parseDouble(" " + obj1.get("price"));
                int total = data.get(key);
                tableData[index][2] = String.format("%.02f ", (price * total));
                Total.addPrice(key, (price * total));
            }
        }
        index++;
    }
}

```

- Method addTable - ทำหน้าที่เพิ่มตาราง เข้าไป panel ซึ่งใช้ทั้งตารางรายการอาหาร และ ตารางรายการสั่งซื้อ



```

public void actionPerformed(ActionEvent ae) {
    System.out.println(ae.getActionCommand());
    if (ae.getActionCommand().equals("Reset")) {
        if (!order.getOrder().isEmpty()) {
            ImageIcon icon = new ImageIcon( filename: "img/alert.png");
            int n = JOptionPane.showConfirmDialog( parentComponent: null,
                message: "Confirm Reset",
                title: "Reset",
                JOptionPane.YES_NO_OPTION, messageType: 3, icon);
            if (n == 0) {
                order.resetOrder();
                Total.reset();
                addTable(orderList);
            }
        }
    }
}

```

- Method actionPerformed (Override มาจาก Class Listener) – ทำหน้าที่ตรวจสอบเหตุการณ์ action ต่าง ๆ จากปุ่ม ซึ่งจะมี Method นี้อยู่ในทุก ๆ Class ที่มีการใช้งาน Button

#### 4. Class Order -> เกี่ยวกับรายละเอียดรายการอาหาร หรือ ออเดอร์

```

public class Order {
    private HashMap<String, Integer> data = new HashMap<>();
    private static int orderID = 0;

    public void addOrder(String name) {
        try {
            int value = (int) data.get(name);
            data.put(name, value + 1);
        } catch (Exception e) {
            data.put(name, 1);
        }
    }
}

```

- Method addOrder - ทำหน้าที่เพิ่มออเดอร์ และ เพิ่มจำนวนออเดอร์นั้นเข้าไปใน Hashmap

```
public void removeOrder(String name) {
    try {
        int value = (int) data.get(name);
        if (value - 1 > 0) {
            data.put(name, value - 1);
        } else if (value - 1 == 0) {
            Total.removePrice(name);
            data.remove(name);
        }
    } catch (Exception e) {
    }
}
```

- Method removeOrder – ทำหน้าที่ลบออเดอร์ และ ลบจำนวนออเดอร์นั้นที่อยู่ใน Hashmap

```
public void resetOrder() { data = new HashMap<String, Integer>(); }
```

- Method resetOrder - ทำหน้าที่ลบออเดอร์ และ จำนวนออเดอร์ทั้งหมดที่อยู่ใน Hashmap

```
public void addOrderID() { orderID++; }
```

- Method addOrderID - ทำหน้าที่เพิ่มเลขลำดับของออเดอร์

```
public int getOrderID() { return orderID; }
```

- Method getOrder - ทำหน้าที่ return เลขลำดับออเดอร์

```
public static int getOrderNum() { return orderID; }
```

- Method getOrderNum - ทำหน้าที่ return เลขลำดับออเดอร์

```
public HashMap<String, Integer> getOrder() { return data; }
```

- Method getOrder – หน้าที่ return Hashmap Data

## 5. Class Total -> เกี่ยวกับค่าราคาอาหารทั้งหมด

```
public class Total {  
    private static JSONArray obj;  
    private static HashMap<String, Double> allprice = new HashMap<>();
```

```
public static void addPrice(String key, Double value) { allprice.put(key, value); }
```

- Method addPrice – ทำหน้าที่เพิ่มค่าเข้าไปใน Hashmap allprice

```
public static void removePrice(String key) { allprice.remove(key); }
```

- Method removePrice – ทำหน้าที่ลบค่าออกจาก Hashmap allprice

```
public static Double getTotal() {  
    Double totalPrice = 0.0;  
    for (String key : allprice.keySet()) {  
        totalPrice += allprice.get(key);  
    }  
    return totalPrice;  
}
```

- Method getTotal - ทำหน้าที่หาผลรวมทั้งหมดของราคา และ return ออกมา

```
public static void reset() { allprice = new HashMap<>(); }
```

- Method reset – ทำหน้าที่ลบค่าทั้งหมดใน Hashmap allprice ออกทั้งหมด

```
public static HashMap<String, Double> getAllprice() { return allprice; }
```

- Method getAllprice – ทำหน้าที่ return ค่าทั้งหมดใน Hashmap allprice

## 6. Class PrintReceipt -> ทำรายละเอียดต่าง ๆ ของใบเสร็จ

```
public class PrintReceipt implements Printable {  
  
    public static double cmToPPI(double cm) { return cm * 0.393600787 * 72d; }
```

- Method cmToPPI - ทำหน้าที่แปลงขนาดจาก cm (centimeters) แล้ว return ค่าขนาดเป็น ppi (Pixel Per inch) เพื่อให้สามารถใช้พิมพ์ร่วมกับเครื่องพิมพ์เอกสารได้อย่างถูกต้อง

```
public static PageFormat getPageFormat(PrinterJob pj) {  
  
    PageFormat pf = pj.defaultPage();  
    Paper paper = pf.getPaper();  
    double middleHeight = 8.0;  
    double headerHeight = 2.0;  
    double footerHeight = 2.0;  
    double width = cmToPPI(8);           //printer know only point p  
    double height = cmToPPI(headerHeight + middleHeight + footerHeight);  
    paper.setSize(width, height);  
    paper.setImageableArea( x: 0, y: 10, width, height - headerHeight - footerHeight );  
    pf.setOrientation(PageFormat.PORTRAIT);           //select orientation  
    pf.setPaper(paper);  
  
    return pf;  
}
```

- Method getPageFormat - ทำหน้าที่ return ขนาดกระดาษเพื่อให้นำไปใช้กับการพิมพ์

```
public int print(Graphics graphics, PageFormat pageFormat, int pageIndex) throws PrinterException {  
  
    Clock clock = new Clock();  
    int result = NO_SUCH_PAGE;  
    if (pageIndex == 0) {  
  
        Graphics2D g2d = (Graphics2D) graphics;  
  
        try {  
            // Draw Header  
            int y = 20;  
            int yShift = 10;  
            int headerRectHeight = 15;  
  
            g2d.setFont(new Font( name: "Courier New", Font.BOLD, size: 10));  
            g2d.drawString( str: " " + clock.getCurrentDate() + " " + clock.getCurrentTime() + " ", x: 12, y: 20 );  
            y += yShift;  
            g2d.drawString( str: "-----", x: 12, y: y );  
            y += yShift;  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
    }  
    return result;  
}
```

- Method print – ทำหน้าที่สร้างใบเสร็จ และ สั่งพิมพ์

```
public String repeat(int count) { return new String(new char[count]).replace( target: "\0", replacement: " "); }
```

- Method repeat - ทำหน้าที่หาซ่องว่างของกระดาษ ระหว่างข้อความ 2 ข้อความ

## 7. Class Json -> เกี่ยวกับข้อมูล JSON

```
public class Json {  
    public JSONArray openJson(String path) {  
        JSONParser jsonParser = new JSONParser();  
        JSONArray obj = null;  
        try {  
            FileReader fopen = new FileReader(path);  
            obj = (JSONArray) jsonParser.parse(fopen);  
        } catch (Exception e) {  
            System.out.println(e);  
        }  
        return obj;  
    }  
}
```

- Method openJson – ทำหน้าที่เปิดไฟล์ JSON ที่อยู่ในเครื่อง

```
public JSONArray toJson(HashMap<String, Integer> data, Order order, LocalDateTime time) {  
    JSONArray list = new JSONArray();  
    for (String key : data.keySet()) {  
        JSONObject temp = new JSONObject();  
        temp.put("name", key);  
        temp.put("total", data.get(key));  
        list.add(temp);  
    }  
    list.add(order.getOrderID());  
    list.add(time.atZone(ZoneId.systemDefault()).toInstant().toEpochMilli());  
    return list;  
}
```

- Method toJson - ทำหน้าที่แปลง Hashmap ให้อยู่ในรูปแบบของ JSON

## 8. Class Clock -> ทำเวลา และ วัน/เดือน/ปี ปัจจุบัน

```
public class Clock extends JLabel implements Runnable {  
    public void run() {  
        while (true) {  
            DateTimeFormatter dtf = DateTimeFormatter.ofPattern("HH:mm:ss ");  
            LocalDateTime now = LocalDateTime.now();  
            this.setText(" " + dtf.format(now));  
            try {  
                Thread.sleep( millis: 1000 );  
            } catch (Exception e) {  
                System.out.println(e);  
            }  
        }  
    }  
}
```

- Method run – ทำหน้าที่ดึงเวลา และ กำหนดเวลาทุก ๆ 1 วิ

```
public String getCurrentTime() {  
    DateTimeFormatter tf = DateTimeFormatter.ofPattern("HH:mm:ss ");  
    LocalDateTime now = LocalDateTime.now();  
    return tf.format(now);  
}
```

- Method getCurrentTime – ทำหน้าที่ดึงเวลาปัจจุบัน

```
public String getCurrentDate() {  
    DateTimeFormatter df = DateTimeFormatter.ofPattern("dd/MM/yyyy");  
    LocalDateTime now = LocalDateTime.now();  
    return df.format(now);  
}
```

- Method getCurrentTime - ทำหน้าที่ดึงวัน, เดือน, ปี ปัจจุบัน

## 9. Class Network -> เกี่ยวกับระบบ Socket และ ส่งค่าไป Server

```
public class Network implements Runnable {  
    private String HOST;  
    private int PORT;  
    private JSONArray list;  
  
    public Network(JSONArray list) { this.list = list; }
```

- Constructor Network – ทำหน้าที่รับค่าจาก parameter มาใส่ใน list

```
public void run() { sendSocket(list); }
```

- Method run - ทำหน้าที่เรียก Method sendSocket พร้อมส่ง argument ของ list ไปด้วย

```
public void sendSocket(JSONArray list) {  
    Config config = new Config();  
    HOST = config.getHOST();  
    PORT = config.getPORT();  
    if (!list.isEmpty()) {  
        try {  
            Socket socket = new Socket(HOST, PORT);  
            ObjectOutputStream object = new ObjectOutputStream(socket.getOutputStream());  
            object.writeObject(list);  
            object.flush();  
            object.close();  
            socket.close();  
        } catch (IOException e) {  
            System.out.println(e);  
        }  
    }  
}
```

- Method sendSocket - ทำหน้าที่ติดต่อกับเครื่องปลายทาง พร้อมส่งข้อมูลไปด้วย

## Code (Server)

1. Class Main -> เป็น Class ที่ใช้ run โปรแกรม

```
public class Main {  
    private static JSONArray json;  
    private static ArrayList now;  
  
    public static void main(String[] args) {  
        Network.createSocket();  
        new SplashScreen();  
        new GUI();  
        now = new ArrayList();  
        while (true) {  
            json = Network.readSocket();  
            if(now.size() > 0) {  
                for(int i=1; i<=now.size(); i++) {  
                    JSONArray last = (JSONArray) now.get(now.size()-i);  
                    last.remove( index: last.size()-1 );  
                    last.add(LocalDateTime.now().atZone(ZoneId.systemDefault()).toInstant().toEpochMilli());  
                }  
            }  
            json.add(LocalDateTime.now().atZone(ZoneId.systemDefault()).toInstant().toEpochMilli());  
            now.add(json);  
            // System.out.println(now);  
            GUI.reframe( p: 0 );  
        }  
    }  
}
```

- Method main – ทำหน้าที่สร้างหน้า Splash Screen จาก SplashScreen และ สร้างหน้า GUI จาก Class GUI ขึ้นมา

```
public static ArrayList getNow() { return now; }
```

- Method getNow - ทำหน้าที่ return ArrayList ของ now ซึ่งคือค่าของเวลาที่เก็บจาก server ก่อนที่จะ reframe เพื่อให้เวลาอัพเดทอยู่ล่าสุด

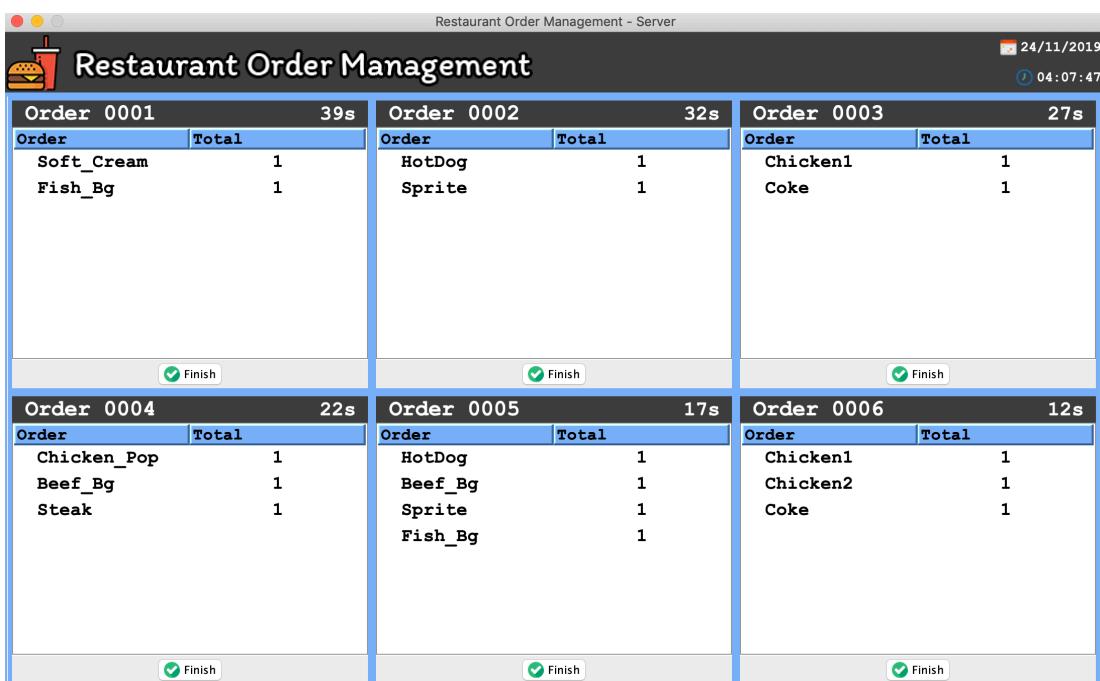
```
public static void removeNow(int fin) {  
    for (int i = 0; i < now.size(); i++) {  
        JSONArray check = (JSONArray) now.get(i);  
        if ((int) check.get(check.size() - 3) == fin) {  
            now.remove(i);  
        }  
    }  
}
```

- Method removeNow – ทำหน้าที่ ส่งค่า Array ที่ต้องการจะลบออก

```

public static Object[][][] genTable(int position) {
    JSONArray data = (JSONArray) now.get(position);
    Object[][] mysave = new Object[data.size()-3][2];
    for(int i=0; i<data.size()-3; i++) {
        JSONObject item = (JSONObject) data.get(i);
        mysave[i][0] = " " + item.get("name");
        mysave[i][1] = item.get("total");
    }
    return mysave;
}

```



- Method genTable - ทำหน้าที่สร้างตารางรายการอาหารขึ้นมา

```

public static int getIDOrder(int position) {
    JSONArray data = (JSONArray) now.get(position);
    return (int) data.get(data.size() - 3);
}

```

- Method getIDOrder - ทำหน้าที่ return ค่าลำดับของออเดอร์

```

public static long getSendtime(int position) {
    JSONArray data = (JSONArray) now.get(position);
    return (long) data.get(data.size() - 2);
}

```

- Method getSendtime – ทำหน้าที่ return ค่าเวลา ที่ได้มาจาก Client

```

public static long getGivetime(int position) {
    JSONArray data = (JSONArray) now.get(position);
    return (long) data.get(data.size() - 1);
}

```

- Method getGivetime – ทำหน้าที่ return ค่าเวลาของอุเดอร์ตัวก่อนล่าสุด

## 2. Class SplashScreen -> หน้าต่าง Splash Screen

```

public class SplashScreen {
    private JFrame frame;
    private JLabel loading;

    public SplashScreen() { this.init(); }

    public void init() {
        JFrame frame = new JFrame();
        JLabel loading = new JLabel();
        loading.setIcon(new ImageIcon( filename: "img/loading2.png"));
        frame.add(loading);
        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
        frame.setUndecorated(true);
        frame.setVisible(true);
        frame.setResizable(false);
        frame.pack();
        centerWindow(frame);
        try {
            Thread.sleep( millis: 2000 );
        } catch (Exception e) {
            System.out.println(e);
        }
        frame.dispose();
    }
}

```

- Constructor SplashScreen - ทำหน้าที่เรียก Method init();

- Method init - ทำหน้าที่สร้างหน้า Splash Screen แล้วให้หยุดรอ 2 วินาที แล้วให้เรียก Class GUI ขึ้นมา

```
public void centerWindow(JFrame frame) {
    Dimension dimension = Toolkit.getDefaultToolkit().getScreenSize();
    int x = (int) ((dimension.getWidth() - frame.getWidth()) / 2);
    int y = (int) ((dimension.getHeight() - frame.getHeight()) / 2);
    frame.setLocation(x, y);
}
```



- Method centerWindow - ทำให้ JFrame อยู่กลางหน้าจอ ซึ่งจะมี Method นี้อยู่ในทุก ๆ Class ที่เป็น JFrame

### 3. Class GUI -> สร้างรายละเอียด และ โครงสร้างต่าง ๆ ของ GUI

```
public class GUI implements ActionListener {
    private static JFrame fr;
    private static JPanel p0, p1, p2, p3, p4, p5, pt, pc, pb;
    private static JButton btn0, btn1, btn2, btn3, btn4, btn5, btn_l, btn_r;
    private static int positionShow = 0;

    public GUI() {
        fr = new JFrame( title: "Restaurant Order Management - Server");
        // icon for windows
        fr.setIconImage(new ImageIcon( filename: "img/icon.png").getImage());

        p0 = new JPanel();
        p1 = new JPanel();
        p2 = new JPanel();
        p3 = new JPanel();
        p4 = new JPanel();
        p5 = new JPanel();
        pt = new JPanel();
        pc = new JPanel();
        pb = new JPanel();
    }
}
```

- Constructor GUI - ทำหน้าที่สร้างหน้า GUI พร้อมรายละเอียดต่าง ๆ

```
public static void reframe(int p) { genCard(p0, p1, p2, p3, p4, p5, p); }
```

- Method reframe - ทำหน้าที่เรียก Method genCard

```
public static void genCard(JPanel p0, JPanel p1, JPanel p2, JPanel p3, JPanel p4, JPanel p5, int p) {
    TableModel model;
    String column[] = {"Order", "Total"};
    try {
        p0 = new JPanel();
```

- Method genCard – ทำหน้าที่สร้างพื้นที่เมนู และตารางของออเดอร์

```
public void centerWindow(JFrame frame) {
    Dimension dimension = Toolkit.getDefaultToolkit().getScreenSize();
    int x = (int) ((dimension.getWidth() - frame.getWidth()) / 2);
    int y = (int) ((dimension.getHeight() - frame.getHeight()) / 2);
    frame.setLocation(x, y);
}
```

- Method centerWindow - ทำหน้าที่ทำให้ JFrame อยู่กลางหน้าจอ ซึ่งจะมี Method นี้อยู่ในทุก ๆ Class ที่เป็น JFrame

```
@Override
public void actionPerformed(ActionEvent ae) {
    if (ae.getSource().equals(btn0)) {
        Main.removeNow(Main.getIDOrder(positionShow));
        positionShow = 0;
        reframe(p: 0);
    } else if (ae.getSource().equals(btn1)) {
        Main.removeNow(Main.getIDOrder( position: positionShow+1));
        positionShow = 0;
```

- Method actionPerformed (Override มาจาก Class Listener) – ทำหน้าที่ตรวจสอบเหตุการณ์ action ต่าง ๆ จากปุ่ม ซึ่งจะมี Method นี้อยู่ในทุก ๆ Class ที่มีการใช้งาน Button

#### 4. Class Countdown -> เกี่ยวกับระบบจับเวลา

```
public class Countdown extends JLabel implements Runnable {  
    private long gettime = 0;  
    private long nowtime = 0;  
    private long sec = 0;  
  
    public Countdown(long x, long y) {  
        this.nowtime = y;  
        this.gettime = x;  
        this.sec = Math.abs((gettime - nowtime)/1000);  
    }  
}
```

- Constructor Countdown – ทำหน้าที่ลบค่าเวลาปัจจุบัน กับค่าเวลาที่ออดิอร์ส่งเข้ามา เป็นการจับเวลา

```
public void run() {  
    while (true) {  
        if (sec < 60){  
            this.setText(sec+"s ");  
        } else if (sec < 3600){  
            this.setText((sec/60)+"m ");  
        } else {  
            this.setText((sec/3600)+"hr ");  
        }  
        try {  
            Thread.sleep( millis: 1000 );  
            sec++;  
        } catch (Exception e) {  
            System.out.println(e);  
        }  
    }  
}
```

- Method run – ทำหน้าที่เปลี่ยนค่า และ ตั้งค่าจับเวลาไปเรื่อย ๆ

## 5. Class Clock -> เกี่ยวกับการแสดงเวลาปัจจุบัน

```
public class Clock extends JLabel implements Runnable {  
    public void run() {  
        while (true) {  
            DateTimeFormatter dtf = DateTimeFormatter.ofPattern("HH:mm:ss ");  
            LocalDateTime now = LocalDateTime.now();  
            this.setText(" " + dtf.format(now));  
            try {  
                Thread.sleep( millis: 1000);  
            } catch (Exception e) {  
                System.out.println(e);  
            }  
        }  
    }  
}
```

- Method run – ทำหน้าที่ดึงเวลา และ กำหนดเวลาทุก ๆ 1 วิ

## 6. Class Network -> เกี่ยวกับระบบ Socket และ รับค่ามาจาก Client

```
public class Network {  
    private static ServerSocket socket;  
    private static int PORT;  
    private static JSONArray data;  
  
    public static void createSocket() {  
        Config config = new Config();  
        PORT = config.getPORT();  
        try {  
            socket = new ServerSocket(PORT);  
        } catch (IOException e) {  
            System.out.println(e);  
        }  
    }  
}
```

- Method createSocket – ทำหน้าที่สร้าง Socket

```

public static JSONArray readSocket() {
    try {
        Socket connection = socket.accept();
        ObjectInputStream input = new ObjectInputStream(connection.getInputStream());
        data = (JSONArray) input.readObject();
    } catch (IOException e) {
        System.out.println(e);
    } catch (ClassNotFoundException e) {
        System.out.println(e);
    }
    return data;
}

```

- Method readSocket – ทำหน้าที่อ่านค่าที่ส่งมาจาก Socket

## หน้าที่ของสมาชิก

1. นายเจษฎาพร ไตรวินิจศรีสุข 61070028
  - ระบบใบเสร็จทั้งหมด
2. นางสาวฉัตรธิดา แจ้งใจ 61070029
 

<ul style="list-style-type: none"> <li>- Design หน้า Client</li> <li>- ตกแต่งหน้า Server</li> </ul>	<ul style="list-style-type: none"> <li>- เล่นรายงาน</li> <li>- PowerPoint</li> </ul>
---	--
3. นายชาญวิทย์ เศรษฐกุวงศ์สิน 61070040
 

<ul style="list-style-type: none"> <li>- ระบบ Client ทั้งหมด</li> <li>- ตกแต่งหน้า Client</li> </ul>	<ul style="list-style-type: none"> <li>- ระบบ Socket</li> <li>- ดึงข้อมูลจาก JSON</li> </ul>
--	--
4. นายณภัทร อารยวัฒนาพงษ์ 61070045
  - ระบบ Server ทั้งหมด
5. นายระพีพันธ์ มุนไวย 61070181
  - Design หน้า Server
  - Tester โปรแกรม