



โครงการบล็อกเชนและสกุลเงินดิจิทัล
เรื่อง Blockchain Coffee delivery

จัดทำโดย

B6400965 กิตติภพ สระแกทอง
B6400989 พงศกร ลั่นใจดี
B6417369 สุภัตสรรา ไวยสุณี
B6425203 ชัยวัฒน์ พุนดี
B6428211 กัญญารัตน์ นิจจอหอ

เสนอ

รศ. ดร.ศิริปฐษ์ บุญครอง

รายงานนี้เป็นส่วนหนึ่งของรายวิชา

1101213 Project in Blockchain and Cryptocurrency

ภาคเรียนที่ 3 ปีการศึกษา 2565

มหาวิทยาลัยเทคโนโลยีสุรนารี

คำนำ

โครงการเล่มนี้จัดทำขึ้นเพื่อเป็นส่วนหนึ่งของ 1101213 โครงการบล็อกเชนและสกุลเงินดิจิทัลเพื่อให้ได้ศึกษาหาความรู้ในเรื่องสกุลเงินออนไลน์ในโลกปัจจุบัน และได้ศึกษาอย่างเข้าใจเพื่อเป็นประโยชน์กับการเรียนรู้ผู้จัดทำหวังว่ารายงานเล่มนี้จะเป็นประโยชน์กับผู้อ่านหรือนักศึกษาที่กำลังหาข้อมูลเรื่องนี้อยู่ หากมีข้อเสนอแนะหรือข้อผิดพลาดประการใด ผู้จัดทำขอน้อมรับไว้และขออภัยมา ณ ที่นี้ด้วย

คณะผู้จัดทำ

สารบัญ

	หน้า
เหตุผลในการทำหัวข้อนี้	1
ความเกี่ยวข้อง หรือ ความหมายสมของ Blockchain/Cryptocurrency	1
ตารางความเหมาะสม	2
เป้าหมาย/วัตถุประสงค์ และขอบเขตของงานที่จะทำ	4
วัตถุประสงค์	4
ขอบเขตของงานที่จะทำ	4
แผนการดำเนินงาน	5
ข้อมูลที่ระบบ Blockchain ต้องจัดเก็บ	6
ออกแบบระบบที่ทำงานบน Blockchain	6
การไหลของข้อมูล	7
ขั้นตอนการทำงานและโค้ดของระบบ	8

Blockchain Coffee delivery

1. เหตุผลในการทำหัวข้อนี้

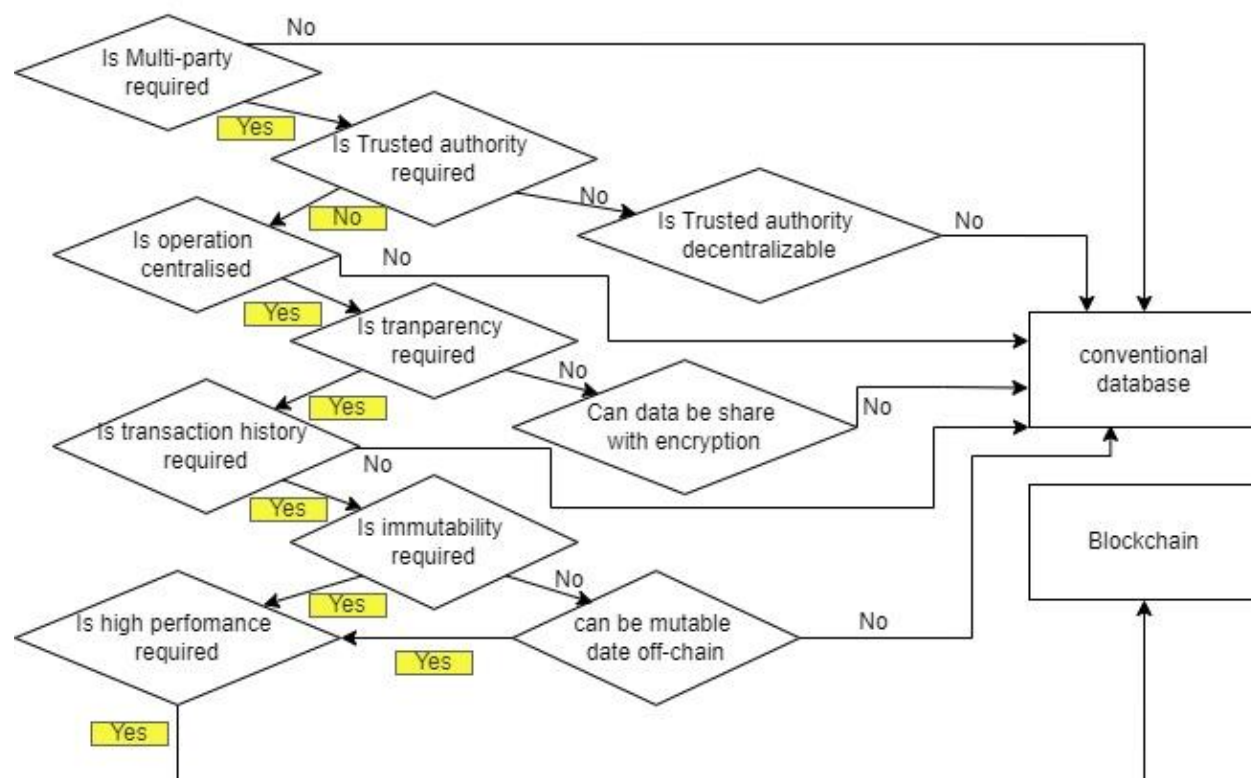
ในปัจจุบันมีร้านกาแฟเปิดใหม่มากมายในประเทศเราจึงอยากเอา เทคโนโลยี Blockchain/Cryptocurrency มาช่วยในการจัดส่งกาแฟเพื่อเพิ่มความปลอดภัยและสินค้าถูกต้องตามที่ผู้ซื้อต้องการและยังสามารถทำให้ผู้ที่ดื่มกาแฟทราบว่ากาแฟที่ตนเองกำลังดื่มนั้นมาจากที่ไหนและเมล็ดพันธุ์อะไรและนำเทคโนโลยี Cryptocurrency มาใช้ในการชำระเงินเพื่อให้ชำระเงินได้สะดวกรวดเร็วยิ่งขึ้นและค่าธรรมเนียมน้อยลงเมื่อต้องจ่ายเงินนอกประเทศ

2. ความเกี่ยวข้อง หรือ ความหมายสมของ Blockchain/Cryptocurrency

การใช้ Blockchain/Cryptocurrency ในการจัดส่งเมล็ดกาแฟ เหมาะสมและมีประโยชน์ เนื่องจากเทคโนโลยีเหล่านี้มีคุณสมบัติที่เหมาะสมกับการจัดการข้อมูลและการทำธุรกรรมออนไลน์ ซึ่งจะช่วยลดเวลาและความซับซ้อนของกระบวนการจัดส่ง และยังช่วยให้กระบวนการดังกล่าวมีความปลอดภัยสูงขึ้นด้วยการใช้ Cryptocurrency ในการชำระเงินและการใช้ Blockchain เพื่อติดตามการจัดส่ง นอกจากนี้การใช้ Blockchain/Cryptocurrency ยังช่วยลดค่าใช้จ่ายของการทำธุรกรรมระหว่างประเทศ และช่วยให้การจัดการข้อมูลและการทำธุรกรรมเป็นไปอย่างเป็นระบบและปลอดภัยมากยิ่งขึ้น ดังนั้น การนำเอา Blockchain/Cryptocurrency มาใช้ในการจัดส่งเมล็ดกาแฟนั้นเหมาะสมและสามารถทำได้อย่างมีประสิทธิภาพและปลอดภัยมากยิ่งขึ้น

Coffee Delivery	
Multi-Party	Yes มีหลายฝ่ายเกี่ยวข้องในระบบการจัดส่งกาแฟ เช่น ร้านกาแฟ, บริการจัดส่ง, และลูกค้า
Trusted Authority	Yes เพื่อสร้างความเชื่อถือและความมั่นใจในกระบวนการส่งมอบกาแฟแก่ผู้รับบริการ รวมถึงเพิ่มระดับความปลอดภัยและความเป็นส่วนตัวของข้อมูลในระบบ
Centralised Operation	No เนื่องจากโครงการนี้มุ่งเน้นในการสร้างระบบให้ผู้เข้าร่วมมีความเป็นเจ้าของข้อมูลและกระบวนการดำเนินการที่ไม่มี Centralised Operation ช่วยเพิ่มความเสถียรและยืดหยุ่นในการดำเนินงาน
Data Transparency or Confidentiality	Yes เพื่อสร้างความเชื่อมั่นให้กับผู้ใช้งาน ผ่านความโปร่งใสของข้อมูลเกี่ยวกับกระบวนการส่งกาแฟ และความลับของข้อมูลสำคัญ เพื่อป้องกันการเข้าถึงและการใช้ข้อมูลที่ไม่เหมาะสม และสร้างความมั่นใจในการให้ข้อมูลส่วนตัวและความเป็นส่วนตัวของลูกค้า
Data Integrity	Yes เพราะมันช่วยให้ข้อมูลที่ถูกบันทึกในระบบ Blockchain ไม่สามารถเปลี่ยนแปลงหรือแก้ไขได้ ซึ่งสร้างความเชื่อมั่นและความถูกต้องในข้อมูลและกระบวนการส่งกาแฟ
Data Immutability	Yes เนื่องจากการจัดส่งกาแฟต้องมีความถูกต้องและน่าเชื่อถือ โดยข้อมูลที่ข้อมูลไม่สามารถเปลี่ยนแปลงได้
High performance	Yes เหมาะสมที่จะมีประสิทธิภาพสูง เนื่องจากการดำเนินการที่รวดเร็วและมีประสิทธิภาพสูงในการจัดส่งกาแฟ ซึ่งอาจเกี่ยวข้องกับเรื่องการผลิตผลข้อมูลรวมถึงประสิทธิภาพของระบบการสื่อสารและการจัดการที่มีความเร็ว
Result	Blockchainระบบการจัดส่งกาแฟใช้เทคโนโลยีบล็อกเชนในการดำเนินงาน ซึ่งมีคุณสมบัติเช่นความปลอดภัย ความโปร่งใส และความสามารถในการเปลี่ยนแปลงข้อมูลได้ จึงเหมาะที่จะนำเทคโนโลยี blockchain เข้ามาใช้พัฒนาระบบนี้

Flowchart ความเกี่ยวข้องของ Blockchain



3. เป้าหมาย/วัตถุประสงค์ และขอบเขตของงานที่จะทำ

เป้าหมายคือนำเทคโนโลยี Blockchain/Cryptocurrency มาปรับใช้กับการจัดส่งเมล็ดกาแฟ

4. วัตถุประสงค์

1. เพื่อออกแบบวิธีการเก็บข้อมูลแหล่งที่มาหรือการจัดส่ง เมล็ดกาแฟแต่ละชนิดไว้ใน Blockchain
2. เพื่อสามารถทำการตรวจสอบแหล่งที่มาของเมล็ดกาแฟได้
3. สามารถนำ Cryptocurrency มาใช้ในการชำระเงินได้อย่างสมบูรณ์

5. ขอบเขตของงานที่จะทำ

1. สามารถตรวจสอบ/ติดตามสินค้าที่จัดส่งได้
2. สามารถตรวจสอบที่อยู่และพันธุ์ของเมล็ดกาแฟที่ส่งไปได้
3. สามารถใช้เงิน Cryptocurrency ในการจ่ายได้

7. ข้อมูลที่ระบบ Blockchain ต้องจัดเก็บ

1. ข้อมูลลูกค้า

- ชื่อลูกค้า (String)
- ที่อยู่ (String)
- จำนวนเงินที่ชำระ (uint256)

2. ข้อมูลการสั่งซื้อ

- ปริมาณเมล็ดกาแฟที่สั่งซื้อ (String)
- สายพันธุ์กาแฟ (String)
- วันที่สั่งซื้อ (String)

3. ข้อมูลการจัดส่ง

- วันที่จัดส่ง (String)
- สถานะการจัดส่ง (String)
- ชื่อผู้รับ (String)

4. ข้อมูลการชำระเงิน

- วิธีการชำระเงินที่ใช้ (String)
- รายละเอียดการทำธุรกรรมเงินสกุลดิจิทัลหรือวิธีการชำระเงินอื่น ๆ (String)

5. ข้อมูลเกี่ยวกับเมล็ดกาแฟ

- สายพันธุ์กาแฟ (String)
- คุณภาพกาแฟ (String)
- จำนวนเมล็ดกาแฟที่ถูกส่ง (uint256)

6. ธุรกรรมทางธุรกิจ (Business Transactions)

- ข้อมูลการสั่งซื้อของธุรกิจที่ซื้อกาแฟจากผู้ผลิตหรือจัดหาสินค้า (String)
- รายละเอียดการชำระเงินระหว่างธุรกิจ (String)
- ข้อมูลการจัดส่งที่เกี่ยวข้องกับการซื้อของธุรกิจ (String)

8. ออกแบบระบบที่ทำงานบน Blockchain

1. ฟาร์ม (Famer)

- รหัสสินค้า (uint256 productCode)
- ที่อยู่ฟาร์ม (string farmAddress)
- รายการการสั่งซื้อ (string orderList)
- เวลาการปลูกของเมล็ดกาแฟ (string coffeePlantingTime)
- รายละเอียดของเมล็ดกาแฟ (string coffeeSeedDetails)
- วันที่จัดส่ง (string orderDate;)
- วันที่ที่สั่งซื้อ (string deliveryDate)

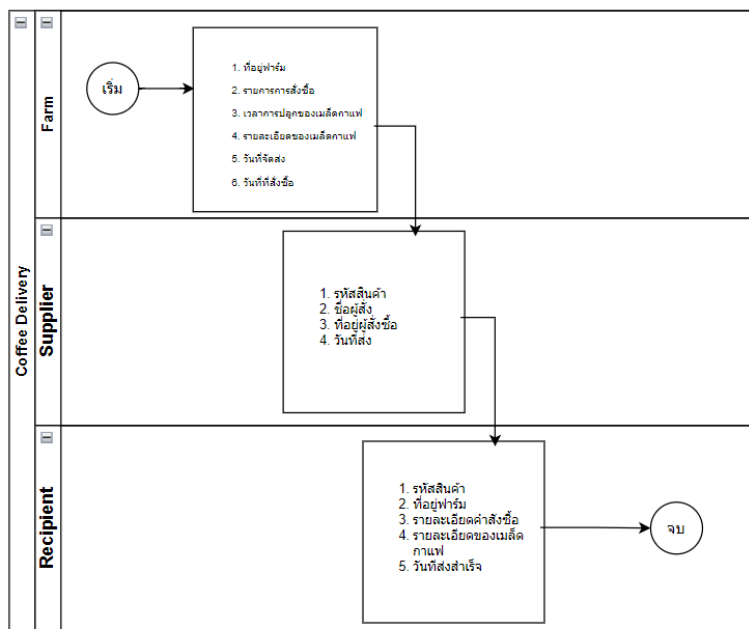
2. ผู้จัดหาสินค้า (Supplier)

- รหัสสินค้า (uint256 productCode)
- ชื่อผู้สั่งซื้อ (string buyerName)
- ที่อยู่ของผู้สั่งซื้อ (string buyerAddress)
- วันที่จัดส่ง (string deliveryDate)

3. ผู้รับสินค้า (Recipient)

- รหัสสินค้า (uint256 productCode)
- ที่อยู่ฟาร์ม (string farmAddress)
- รายการการสั่งซื้อ (string orderList)
- รายละเอียดของเมล็ดกาแฟ (string coffeeSeedDetails)
- วันที่จัดส่ง (string deliveryDate)

9. การไหลของข้อมูล



Start

1. Farm กรอกข้อมูล (รหัสสินค้า, ที่อยู่ฟาร์ม, รายการสินค้า, เวลาการปลูก, รายละเอียดของเมล็ด, วันที่สั่งซื้อ, วันที่จัดส่ง)
 2. Supplier กรอกข้อมูล (รหัสสินค้า, ชื่อผู้จัดส่ง, ที่อยู่ผู้จัดส่ง, วันที่ส่ง, ราคา)
 3. Recipient กรอกข้อมูล (รหัสสินค้า, ที่อยู่ฟาร์ม, รายละเอียดคำสั่งซื้อ, รายละเอียดของเมล็ดกาแฟ, วันที่ส่งสำเร็จ)
- ผู้ที่สามารถถอนเงินออกจากระบบได้คือ ผู้ที่เป็นเจ้าของ Smart contract เท่านั้นและเมื่อเจ้าของ Smart contract ทำการถอนเงินแล้วระบบจะลบข้อมูลของรหัสสินค้านั้นๆทิ้งทั้งหมด ผู้ซื้อสินค้า จะทราบข้อมูลที่อยู่ฟาร์มระยะเวลาในการปลูกสถานที่ปลูกและรายละเอียดของเมล็ดพันธุ์กาแฟนั้นๆที่ผู้ซื้อได้ทำการสั่ง อย่างถูกต้องและเหมาะสมครบถ้วน

10. ขั้นตอนการทำงานและโค้ดของระบบ

1. Farm จะกรอกข้อมูลต่างลงไปในช่วงตรงของ “Function addFarm”

ตัวอย่างข้อมูล “ 22,123 หมู่ 10 อ.เมือง จ.นครราชสีมา, เมล็ดกาแฟกลิ้งทรีฟเฟิล 2 โล, เวลาการปลูก60วัน, เมล็ดกาแฟกลิ้งทรีฟเฟิลข้าวเข้ม, 12/5/65เวลา9:30, 11/5/65เวลา16:22 ”

The screenshot shows a form titled "addFarm" with the following fields and values:

- passcode: "22"
- farmAddress: "123 หมู่ 10 อ.เมือง จ.นครราชสีมา"
- orderList: "เมล็ดกาแฟกลิ้งทรีฟเฟิล 2 โล"
- coffeePlantingTime: "เวลาการปลูก60วัน"
- coffeeSeedDetails: "เมล็ดกาแฟกลิ้งทรีฟเฟิลข้าวเข้ม"
- orderDate: "12/5/65เวลา9:30"
- deliveryDate: "11/5/65เวลา16:22"

At the bottom, there are three buttons: "Calldata", "Parameters", and a prominent orange "transact" button.

โค้ดในส่วนของ Function addFarm ได้เพิ่ม require เอาไว้ตรวจสอบว่า สินค้า เลขนี้ได้ถูกเพิ่มไปแล้ว หรือยังถ้ารหัสสินค้า(passcode)ถูกเพิ่มเข้าไปแล้วระบบไม่ทำการเพิ่มสินค้าลงไปอีก

```
function addFarm(
  uint256 passcode,
  string memory farmAddress,
  string memory orderList,
  string memory coffeePlantingTime,
  string memory coffeeSeedDetails,
  string memory orderDate,
  string memory deliveryDate
) public {
  require(farms[passcode].productCode == 0);
```

2. Supplier จะกรอกข้อมูลต่างลงไปเป็นช่องตรงของ “function addSupplier”

ตัวอย่างข้อมูล “22, ประจวบ, ถนนโชตนา ต.ช้างเผือก อ.เชียงใหม่ จ.เชียงใหม่ 50300, 12/5/65เวลา9:30”

โค้ดในส่วนของ Function addSupplier ได้เพิ่ม require เอาไว้ตรวจสอบว่า สินค้า เลขนี้ได้ถูกเพิ่มเข้าไปในระบบจะตรวจสอบว่ารหัสสินค้า(passcode)ที่เพิ่มไปนั้นตรงกับที่ Farm เพิ่มเข้าไปหรือไม่ถ้าหากตรงจะสามารถเพิ่มข้อมูลเข้าไปได้แต่ถ้าหากไม่ตรง Supplier จะไม่สามารถเพิ่มข้อมูลนั้นเข้าได้

```
function addSupplier(  infinite gas
    uint256 passcode,
    string memory buyerName,
    string memory buyerAddress,
    string memory deliveryDate
) public {
    require(farms[passcode].productCode != 0);
    require(suppliers[passcode].productCode == 0);
```

3. Recipient จะกรอกข้อมูลต่างลงไปในช่องตรงของ “Function addRecipient”

ตัวอย่างข้อมูล “ 22,123 หมู่ 10 อ.เมือง จ.นครราชสีมา, เมล็ดกาแฟลิ้นจี่ฟิเล 2 โล, เวลาการปลูก60วัน เมล็ดกาแฟลิ้นจี่ฟิเลข้าวเข้ม, 12/5/65เวลา9:30, 11/5/65เวลา16:22”

The screenshot shows the 'addRecipient' function interface. The inputs are as follows:

Field	Value
passcode:	"22"
farmAddress:	"123 หมู่ 10 อ.เมือง จ.นครราชสีมา"
orderList:	"เมล็ดกาแฟลิ้นจี่ฟิเล 2 โล"
coffeeSeedDetails:	"เวลาการปลูก60วันเมล็ดกาแฟลิ้นจี่ฟิเล"
deliveryDate:	"12/5/65เวลา9:30"

At the bottom, there are three buttons: 'Calldata', 'Parameters', and 'transact'.

โค้ดในส่วนของ Function addRecipient ได้เพิ่ม require เอาไว้ตรวจสอบว่า สินค้า เลขนี้ได้ถูกเพิ่มเข้าไปในระบบจะตรวจสอบว่ารหัสสินค้า(passcode)ที่เพิ่มไปนั้นตรงกับที่ Farm เพิ่มเข้าไปหรือไม่ถ้าหากตรงจะสามารถเพิ่มข้อมูลเข้าไปได้แต่ถ้าหากไม่ตรง Recipient จะไม่สามารถเพิ่มข้อมูลนั้นเข้าได้

```
function addRecipient(
    uint256 passcode,
    string memory farmAddress,
    string memory orderList,
    string memory coffeeSeedDetails,
    string memory deliveryDate
) public {
    require(farms[passcode].productCode != 0);
    require(recipients[passcode].productCode == 0);
```

4. Function update Status

เป็น Function ที่ให้ Farm,Supplier,Recipient กรอกข้อมูล update status ในช่อง newStatus

ตัวอย่างข้อมูล “22, กำลังดำเนินการจัดส่ง 14/6/65 เวลา 09:55”

The screenshot displays three stacked forms for updating status. Each form has a 'passcode' field with the value '22', a 'newStatus' field, and a 'transact' button. The forms are labeled 'updateFarmStatus', 'updateRecipientStatus', and 'updateSupplierStatus'.

- updateFarmStatus:** newStatus: กำลังดำเนินการจัดส่ง 14/6/65 เวลา 09:55
- updateRecipientStatus:** newStatus: ส่งสำเร็จ 15/6/65 เวลา 10:15
- updateSupplierStatus:** newStatus: กำลังนำส่งไปยังผู้ส่ง 14/6/65 เวลา

เป็น Function ที่ให้ Farm,Supplier,Recipient กรอกข้อมูล update status ในช่อง newStatus เพื่ออัปเดตข้อมูลของ deliveryDate ให้ทราบว่าสถานะของ สินค้าเป็นอย่างไร จัดส่งถึงไหน วันเวลาใด

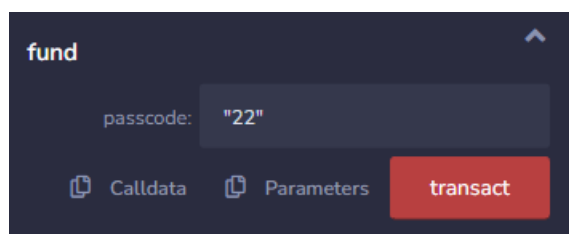
```
function updateFarmStatus(uint256 passcode, string memory newStatus) public { infinite gas
    Farm storage farm = farms[passcode];
    farm.deliveryDate = newStatus;
}

function updateSupplierStatus(uint256 passcode, string memory newStatus) public { infinite gas
    Supplier storage supplier = suppliers[passcode];
    supplier.deliveryDate = newStatus;
}

function updateRecipientStatus(uint256 passcode, string memory newStatus) public { infinite gas
    Recipient storage recipient = recipients[passcode];
    recipient.deliveryDate = newStatus;
}
```

5. จ่ายเงิน Function fund

ตัวอย่างข้อมูล “22” (22 คือรหัสสินค้า /passcode)

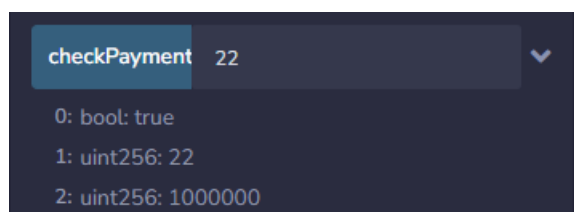


Function fund เป็นฟังก์ชันในการจ่ายเงินโดยที่จะให้กรอกรหัสสินค้า(passcode) เพื่อจ่ายเงินโดยเพิ่ม require เพื่อตรวจสอบว่า ในระบบมีรหัสสินค้าที่กรอกเข้าไปหรือป่าวถ้ามีก็จะสามารถจ่ายเงินได้แต่ถ้าไม่มีระบบจะไม่ทำการหักเงิน

```
function fund(uint256 passcode) public payable {
    passcodeToAmountFunded[passcode] += msg.value;
    require(farms[passcode].productCode != 0);
}
```

6. ตรวจสอบสถานะการจ่ายเงิน Function checkPaymentStatus

ตัวอย่างข้อมูล “22” (22 คือรหัสสินค้า /passcode)



ถ้าหมายเลขรหัสสินค้าของ Farms กับ suppliers และ passcodeToAmountFunded เท่ากับศูนย์ ให้คืนค่า (false, 0, 0) กลับไป ถ้าเงื่อนไขทั้งสามเป็นจริง แสดงว่ามีการชำระเงินในระบบ ซึ่งคืนค่า (true,passcode,passcodeToAmountFunded[passcode])

```
function checkPaymentStatus(uint256 passcode) public view returns (bool, uint256, uint256) {
    if (farms[passcode].productCode == 0 || suppliers[passcode].productCode == 0 || passcodeToAmountFunded[passcode] == 0) {
        return (false, 0, 0);
    }
    return (true, passcode, passcodeToAmountFunded[passcode]);
}
```


7. ค้นหาข้อมูลจากFarm Function Farms

ตัวอย่างข้อมูล “22” (22 คือรหัสสินค้า /passcode)

farms	22	▼
0: uint256:	productCode 22	
1: string:	farmAddress 123 หมู่ 10 อ.เมือง จ.น ครราชสีมา	
2: string:	orderList เมล็ดกาแฟลิ้นจี่ 2 โ ล	
3: string:	coffeePlantingTime เวลาปลูก60วั น	
4: string:	coffeeSeedDetails เมล็ดกาแฟลิ้นจี่ ฟิลเขียวเข้ม	
5: string:	orderDate 12/5/65เวลา9:30	
6: string:	deliveryDate 11/5/65เวลา16:22	

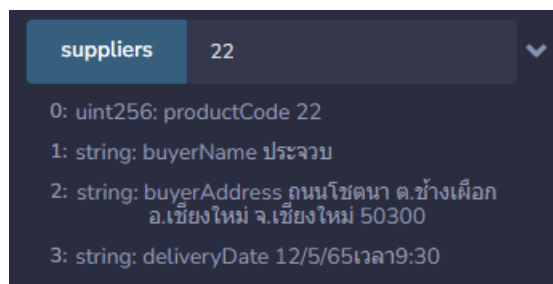
โค้ดของFunction Farms ค้นหาโดยใช้ รหัสสินค้า(passcode)ในการค้นหาจะแสดงข้อมูลที่ Farm กรอกเข้าไปในระบบ

```
mapping(uint256 => Farm) public farms;
mapping(uint256 => Supplier) public suppliers;
mapping(uint256 => Recipient) public recipients;
```

```
    farms[passcode] = Farm(
        passcode,
        farmAddress,
        orderList,
        coffeePlantingTime,
        coffeeSeedDetails,
        orderDate,
        deliveryDate
    );
}
```

8. ค้นหาข้อมูลจากsuppliers Function suppliers

ตัวอย่างข้อมูล “22” (22 คือรหัสสินค้า /passcode)



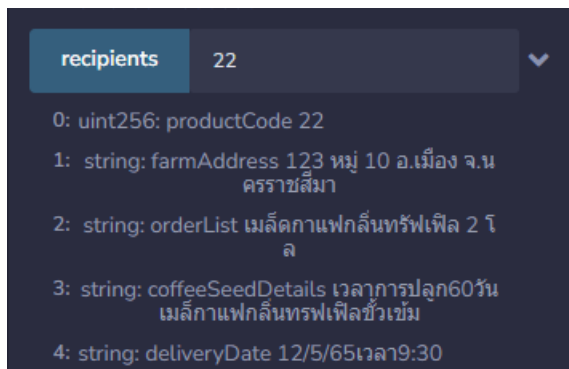
โค้ดของFunction suppliers ค้นหาโดยใช้ รหัสสินค้า(passcode)ในการค้นหาจะแสดงข้อมูลที่ suppliers กรอกเข้าไปในระบบ

```
mapping(uint256 => Farm) public farms;
mapping(uint256 => Supplier) public suppliers;
mapping(uint256 => Recipient) public recipients;
```

```
    suppliers[passcode] = Supplier(
        passcode,
        buyerName,
        buyerAddress,
        deliveryDate
    );
}
```

9. ค้นหาข้อมูลจาก recipients Function recipients

ตัวอย่างข้อมูล “22” (22 คือรหัสสินค้า /passcode)



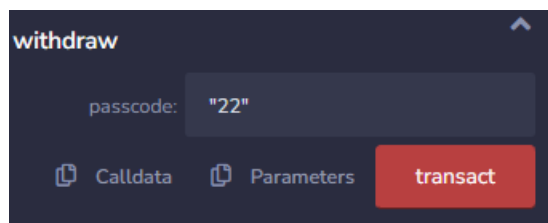
โค้ดของFunction suppliers ค้นหาโดยใช้ รหัสสินค้า(passcode)ในการค้นหาจะแสดงข้อมูลที่ suppliers กรอกเข้าไปในระบบ

```
mapping(uint256 => Farm) public farms;
mapping(uint256 => Supplier) public suppliers;
mapping(uint256 => Recipient) public recipients;
```

```
recipients[passcode] = Recipient(
    passcode,
    farmAddress,
    orderList,
    coffeeSeedDetails,
    deliveryDate
);
```

10. ถอนเงิน Function withdraw

ตัวอย่างข้อมูล “22” (22 คือรหัสสินค้า /passcode)



เมื่อกรอกข้อมูลรหัสสินค้า(passcode)เข้าไประบบจะทำการตรวจสอบว่าข้อมูลที่กรอกเข้านั้นมีอยู่ในระบบหรือไม่หากข้อมูลนั้นมีอยู่ในระบบ ก็จะทำงานถอนเงินออกและระบบจะทำงานลบข้อมูลของรหัสสินค้านั้นทิ้งทั้งหมด

```
function withdraw(uint256 passcode) public onlyOwner payable { infinite gas
    require(farms[passcode].productCode != 0);
    require(suppliers[passcode].productCode != 0);
    require(passcodeToAmountFunded[passcode] != 0);

    uint256 amount = passcodeToAmountFunded[passcode];

    delete farms[passcode];
    delete suppliers[passcode];
    delete recipients[passcode];
    delete passcodeToAmountFunded[passcode];

    payable(msg.sender).transfer(amount);
}
```

11. modifier เพิ่มจำกัดสิทธิ์

ในระบบนี้เพิ่ม modifier ไว้ใน Function withdraw เพื่อให้สิทธิ์แค่ owner ในการถอนเงินออกจากระบบเท่านั้น

```
address public owner;
constructor(){ 1531641 gas 1505800 gas
    owner = msg.sender;
}
```

```
modifier onlyOwner{
    require(msg.sender==owner);

    _;
}
```

```
function withdraw(uint256 passcode) public onlyOwner payable { infinite gas
    require(farms[passcode].productCode != 0);
    require(suppliers[passcode].productCode != 0);
    require(passcodeToAmountFunded[passcode] != 0);
```

12. function calculateETHInUSD (เช็คราคาสินค้า)

ตัวอย่างข้อมูล “22” (22 คือรหัสสินค้า /passcode)

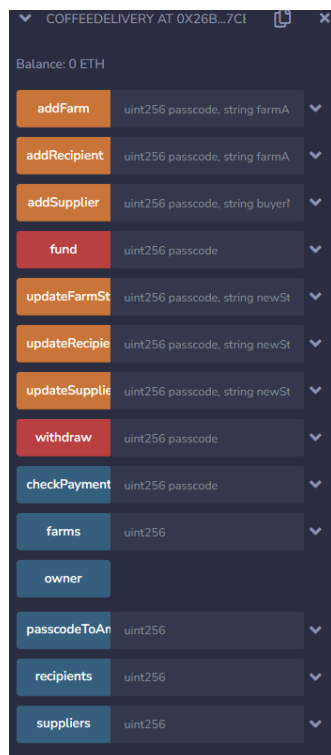
0: uint256: 1727
1: uint256: 1

ระบบจะทำการดึงค่าจาก priceETH ของpasscode นั้นๆ มาทำการคำนวณออกมาเป็นค่าเงิน USDและค่าเงิน ETH

```
function calculateETHInUSD(uint256 passcode) public view returns (uint256,uint256) { infinite gas
    require(suppliers[passcode].productCode != 0, "Invalid passcode");

    uint256 etham = suppliers[passcode].priceETH;
    uint256 ethPrice = getPrice() / 10**8;
    uint256 ethamInUSD = ethPrice * etham;
    return (ethamInUSD, suppliers[passcode].priceETH);
}
```

13. ภาพโดยรวมของระบบ smart contract CoffeeDelivery



ตามขอบเขตที่เราได้กำหนดไว้คือ

1. สามารถตรวจสอบ/ติดตามสินค้าที่จัดส่งได้
2. สามารถตรวจสอบที่อยู่และพันธุ์ของเมล็ดกาแฟที่ส่งไปได้
3. สามารถใช้เงิน Cryptocurrency ในการจ่ายได้

สรุปได้ว่าระบบ ของนั้นสามารถ ทำงานได้ครบตามขอบเขต ระบบของเรานั้นมีหลักๆอยู่ทั้งหมด 6 ฟังก์ชัน

1. เพิ่มข้อมูล (เพิ่มที่อยู่ของเมล็ดกาแฟ,ระยะเวลาการปลูก,รายละเอียดของเมล็ดกาแฟ)
2. อัปเดตข้อมูล (อัปเดตข้อมูลการจัดส่ง สินค้าจัดส่งแล้วหรือยังจัดส่งวันไหน,สถานะของสินค้า)
3. ค้นหาข้อมูล (ค้นหาข้อมูลที่ Farm, Supplier, Recipient) ได้ทำการเพิ่มเข้าไปในข้อที่ 1.
4. จ่ายเงิน (จ่ายเงินด้วยค่าเงินETH)
5. แปลงค่าเงิน (แปลงค่าเงินจาก ETH ให้อยู่ในรูปแบบ USD เพื่อให้ผู้ใช้สามารถทราบได้ว่าจำนวนเงินETHที่ต้องจ่ายมีค่าเป็นกี่ USD)
6. ถอนเงิน (ระบบนี้จะอนุญาตให้แค่ผู้ที่เป็นเจ้าของ Smart contract เป็นคนถอนเงินเท่านั้น)
7. เช็คสถานะการจ่ายเงิน (เช็คสถานะการจ่ายเงิน ว่าจะมาจริงหรือป่าวจำนวนเงินเท่าไร)

หากมีเวลาเพิ่มเติมกลุ่มเราอยากเพิ่มเติม

1. อยากจะปรับปรุงการแปลงค่าเงินให้ผู้ใช้ได้ใช้งานสะดวกสบายมากยิ่งขึ้น
2. อยากปรับปรุงเรื่องการจัดการข้อมูลของวันที่จัดส่ง ให้สมบูรณ์ยิ่งขึ้น

ไฟล์โค้ด Project: https://drive.google.com/file/d/1Mal-r1llwVTCMH_eOdJdNniZ5SpU1uKY/view?usp=sharing

อ้างอิง

ข้อมูลอ้างอิง: Blockchain traceability model in the coffee industry - ScienceDirect

Managing the complexity of coffee through the clarity of blockchain - IBM Blog

ข้อมูลอ้างอิง: Blockchain traceability model in the coffee industry (sciencedirectassets.com)

ข้อมูลอ้างอิง: https://youtu.be/JldLw_h-7Bw?t=2808