ระบบซื้อขายสินค้าเกษตรกรออนไลน์ด้วยสัญญาอัจฉริยะบนเทคโนโลยีบล็อคเชน

Online Agricultural Products Trading System with Smart Contracts based on Blockchain Technology

พรทิพา จันทร์เจริญ

นภัสสร สุขแก่น

สุเมธี คำเอี่ยม

โครงงานวิจัยระดับปริญญาตรีนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตร์บัณฑิต

สาขาวิชาเทคโนโลยีสารสนเทศ คณะวิทยาศาสตร์และเทคโนโลยี

มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี

พ.ศ. 2565

ระบบซื้อขายสินค้าเกษตรกรออนไลน์ด้วยสัญญาอัจฉริยะบนเทคโนโลยีบล็อคเชน

Online Agricultural Products Trading System with Smart Contracts based on Blockchain Technology

พรทิพา จันทร์เจริญ

นภัสสร สุขแก่น

สุเมธี คำเอี่ยม

โครงงานวิจัยระดับปริญญาตรีนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตร์บัณฑิต

สาขาวิชาเทคโนโลยีสารสนเทศ คณะวิทยาศาสตร์และเทคโนโลยี

มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี

พ.ศ. 2565

หัวข้อโครงงานระดับปริญญาตรี	ระบบซื้อขายสินค้าเกษตรกรออนไลน์ด้วยสัญญาอัจฉริยะบนเทคโนโลยีบล็อคเชน Online Agricultural Products Trading System with Smart Contracts based on Blockchain Technology
ชื่อนักศึกษา	พรทิพา จันทร์เจริญ 116210906036-5 นภัสสร สุขแก่น 116210906028-2 สุเมธี คำเอี่ยม 116210906091-0
ปริญญา	วิทยาศาสตรบัณฑิต (เทคโนโลยีสารสนเทศ)
อาจารย์ที่ปรึกษาโครงงาน	ผู้ช่วยศาสตราจารย์ ดร. อนุชา ตุงคัษฐาน

คณะกรรมการสอบโครงงานวิจัย	ลายมือชื่อ
อาจารย์ ดร. สันติ พัฒนะวิชัย ผู้ช่วยศาสตราจารย์ ดร. อนุชา ตุงคัษฐาน อาจารย์ สุเทพ เชาว์สนิท อาจารย์ วุฒิพล วรรณทรัพย์	

วัน/เดือน/ปี ที่สอบ 10 เมษายน 2566

สถานที่สอบ คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี

สาขาวิชาเทคโนโลยีสารสนเทศ คณะวิทยาศาสตร์และเทคโนโลยี

มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี

หัวข้อโครงงานวจัย	ระบบซื้อขายสินค้าเกษตรกรออนไลน์ด้วยสัญญาอัจฉริยะบนเทคโนโลยีบล็อคเชน Online Agricultural Products Trading System with Smart Contracts based on Blockchain Technology
ชื่อนักศึกษา	นางสาว พรทิพา จันทร์เจริญ 116210906036-5 นางสาว นภัสสร สุขแก่น 116210906028-2 นาย สุเมธี คำเอี่ยม 116210906091-0
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร. อนุชา ตุงคัษฐาน
ระดับการศึกษา	ปริญญาตรี
สาขาวิชา	เทคโนโลยีสารสนเทศ
ปีการศึกษา	2565

บทคัดย่อ

โครงงานวิจัยระดับปริญญาตรีเรื่อง ระบบซื้อขายสินค้าเกษตรกรออนไลน์ด้วยสัญญาอัจฉริยะบนเทคโนโลยีบล็อคเชน (Online Agricultural Products Trading System with Smart Contracts based on Blockchain Technology) สาขาวิชาเทคโนโลยีสารสนเทศ คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี จัดทำขึ้นโดยมีวัตถุประสงค์เพื่อพัฒนาระบบซื้อขายสินค้าออนไลน์โดยไม่ผ่านตัวกลาง เป็นการศึกษาเทคโนโลยีใหม่ที่กำลังจะเกิดขึ้นในระยะเวลาอันใกล้ เพื่อรองรับสังคมในยุคดิจิทัลอย่างเต็มรูปแบบ ภายในอนาคตอาจสามารถใช้งานได้ทั่วไป เพียงผู้ซื้อมีกระเป๋าเงินดิจิทัล ก็สามารถใช้งานผ่านเบราว์เซอร์ใดก็ได้ โดยระบบถูกออกแบบให้ทำงานผ่านเว็บเบราว์เซอร์ ที่พัฒนาขึ้นจากภาษา Solidity ทำให้สามารถสร้างสัญญาอัจฉริยะที่ให้ผู้ซื้อและผู้ขายสามารถซื้อขายสินค้าเกษตรได้โดยตรงแบบไม่ผ่านพ่อค้าคนกลาง หากในอนาคตระบบที่นำเสนอนี้ได้รับความนิยมอย่างแพร่หลาย จะทำให้สามารถลดการกดราคาจากพ่อค้าคนกลางได้

กิตติกรรมประกาศ

โครงงานวิจัยระดับปริญญาตรี เรื่อง ระบบขายสินค้าเกษตรกรออนไลน์ด้วยสัญญาอัจฉริยะบนเทคโนโลยีบล็อคเชน (Online farmer sales system with smart contracts on blockchain technology) ณ โอกาสนี้ ผู้วิจัยขอขอบพระคุณ คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี ซึ่งอำนวยความสะดวกในเรื่องของสถานที่ที่ใช้สำหรับดำเนินโครงงาน และขอกราบขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร. อนุชา ตุงคัษฐาน ที่ได้สละเวลาให้ข้อเสนอแนะแนวทางอันเป็นประโยชน์ ตลอดจนช่วยแก้ไขข้อบกพร่องต่างๆ ด้วยความเอาใจใส่และให้ความห่วงใยเสมอมา

ขอกราบขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร.อนุชา ตุงคัษฐาน ดร. สันติ พัฒนะวิชัย นายสุเทพ เชาว์สนิท และนายวุฒิพล วรรณทรัพย์ กรรมการสอบโครงงานระดับปริญญาตรี ที่ได้กรุณาให้คำแนะนำและชี้แนะ จนในที่สุด โครงงานวิจัยระดับปริญญาตรีนี้สำเร็จลุล่วงลงได้

ขอขอบคุณเพื่อนร่วมงานทุกคนที่อำนวยความสะดวกและช่วยเหลือในงานวิจัยครั้งนี้

สุดท้ายนี้ ผู้วิจัยหวังว่างานวิจัยฉบับนี้จะเป็นประโยชน์สำหรับผู้ที่สนใจศึกษาต่อไป

ผู้วิจัย

พรทิพา จันทร์เจริญ

นภัสสร สุขแก่น

สุเมธี คำเอี่ยม

สารบัญ

เรื่อง หน้า

บทคัดย่อ.....ก

กิตติกรรมประกาศ.....ข

สารบัญ.....ค

สารบัญตาราง.....ฉ

สารบัญรูปภาพ.....ช

บทที่ 1.....1

บทนำ.....1

1.1 ความเป็นมาและความสำคัญของปัญหา.....1

1.2 วัตถุประสงค์.....2

1.3 ขอบเขตของโครงงาน.....2

1.4 เครื่องมือและอุปกรณ์ที่ใช้.....3

1.5 วิธีการดำเนินงาน.....3

1.6 ประโยชน์ที่คาดว่าจะได้รับ.....4

บทที่ 2.....5

ทฤษฎีและงานที่วิจัยเกี่ยวข้อง.....5

2.1 ความรู้เกี่ยวกับระบบ Blockchain 5

2.2 ความรู้เกี่ยวกับสกุลเงิน Ethereum.....7

2.3 ความรู้เกี่ยวกับหลักการของ Smart Contract.....7

2.4 ความรู้เกี่ยวกับภาษา Solidity.....9

2.5 โปรแกรม Visual Studio Code....9

2.6 ความรู้เกี่ยวกับ Hardhat....10

2.7 แอพพลิเคชั่น MetaMask....10

2.8 แอพพลิเคชั่น DApps....11

2.9 ความรู้เกี่ยวกับ Remix - Ethereum IDE....11

2.10 ความรู้เกี่ยวกับ Etherscan.io....12

2.11 ความรู้เกี่ยวกับภาษา HTML....13

2.12 ความรู้เกี่ยวกับภาษา CSS....14

2.13 ความรู้เกี่ยวกับภาษา JavaScript....14

2.14 ความรู้เกี่ยวกับหลักการของ React.js....15

2.15 ความรู้เกี่ยวกับหลักการของ Node.js....16

2.16 ความรู้เกี่ยวกับหลักการของ Web3.js....16

2.17 งานวิจัยที่เกี่ยวข้อง....17

บทที่ 3....19

วิธีการดำเนินงาน....19

3.1 แผนงานการดำเนินงาน....19

3.2 เครื่องมือที่ใช้....21

3.3 แผนผังการทำงานของระบบ....22

3.4 ขั้นตอนการพัฒนาระบบ....25

3.5 สรุปขั้นตอนการดำเนินงาน....29

บทที่ 4.....32

ผลการดำเนินงานวิจัย.....32

4.1 ผลการดำเนินงานส่วนผู้ขาย.....32

4.2 ผลการดำเนินงานส่วนผู้ซื้อ.....38

บทที่ 5.....42

สรุปผลการวิจัยและข้อเสนอแนะ.....42

5.1 สรุปผลการดำเนินการ.....42

5.2 ปัญหาที่พบในการศึกษา.....42

5.3 ข้อเสนอแนะ.....43

บรรณานุกรม.....44

ภาคผนวก.....48

ภาคผนวก ก.....49

ขั้นตอนการพัฒนาสัญญาอัจฉริยะ ด้วยภาษา Solidity.....50

ภาคผนวก ข.....56

คู่มือการซื้อสินค้า.....57

ภาคผนวก ค.....62

Source code.....63

ภาคผนวก ง.....141

ประวัติผู้จัดทำ.....142

สารบัญตาราง

ตาราง **หน้า**

ตาราง 1 แผนการดำเนินงานวิจัย.....19

สารบัญรูป

รูป หน้า

รูปที่ 1 รูปแบบการทำงานของ Blockchain.....6

รูปที่ 2 Logo Ethereum.....7

รูปที่ 3 ข้อดีของ Smart contracts.....8

รูปที่ 4 Logo Solidity.....9

รูปที่ 5 Logo Program Visual Studio Code.....9

รูปที่ 6 Logo Hardhat.....10

รูปที่ 7 logo MetaMask.....10

รูปที่ 8 เปรียบเทียบการทำงานของ dApps กับ Apps.....11

รูปที่ 9 ตัวอย่างหน้าเว็บ Remix - Ethereum IDE.....12

รูปที่ 10 ตัวอย่างการทำงานของ Etherscan,io.....13

รูปที่ 11 ตัวอย่างการทำงานของโดยใช้ Tag.....13

รูปที่ 12 Logo CSS.....14

รูปที่ 13 Logo JavaScript.....15

รูปที่ 14 Logo React.....16

รูปที่ 15 Logo Node.js.....16

รูปที่ 16 Logo Wed3.js.....17

รูปที่ 17 คอนเซ็ปต์การทำงานของสัญญาอัจฉริยะบนเทคโนโลยีบล็อคเชน.....21

รูปที่ 18 โครงสร้างระบบขายสินค้าเกษตรกรออนไลน์ด้วยสัญญาอัจฉริยะบนเทคโนโลยีบล็อกเชน.....22

รูปที่ 19 ตัวอย่างฟังก์ชันการเพิ่มข้อมูลสินค้า.....23

รูปที่ 20 ตัวอย่างฟังก์ชันการซื้อสินค้า.....24

รูปที่ 21 การทำงานของระบบ (Flowchart) การเพิ่มสินค้า.....25

รูปที่ 22 การทำงานของระบบ (Flowchart) การซื้อสินค้า.....26

รูปที่ 23 ตัวอย่างสัญญาอัจฉริยะ.....27

รูปที่ 24 ตัวอย่างการใช้ JavaScript ร่วมกับ Solidity 27

รูปที่ 25 ตัวอย่างคำสั่งด้วย Hardhat 28

รูปที่ 26 เขียนสคริปต์เพื่อ deploy.....28

รูปที่ 27 ตัวอย่างบัญชีของ Hardhat ที่ใช้ในการทดสอบ.....29

รูปที่ 28 ตัวอย่างการ deploy สินค้าขึ้น Blockchain.....29

รูปที่ 29 ตัวอย่าง Code ที่ใช้ในการพัฒนาระบบหลังบ้านและหน้าบ้าน.....30

รูปที่ 30 ตัวอย่างหน้าบัญชี MetaMask ที่ Import ด้วย Hardhat.....30

รูปที่ 31 สร้างสัญญาอัจฉริยะด้วยภาษา Solidity 32

รูปที่ 32 เขียนสคริปต์เพื่อให้สัญญาอัจฉริยะทำงานร่วมกันได้.....33

รูปที่ 33 ทดสอบสัญญาอัจฉริยะ.....34

รูปที่ 34 กำหนดรายละเอียดสินค้า.....34

รูปที่ 35 สร้างสคริปต์ในการ deploy สินค้าขึ้นไปยัง blockchain 35

รูปที่ 36 deploy สินค้าขึ้นไปยัง blockchain.....36

รูปที่ 37 การพัฒนาเว็บไซต์.....37

รูปที่ 38 หน้าตาเว็บไซต์ร้านค้าเกษตรกรออนไลน์บนระบบสัญญาอัจฉริยะ.....38

รูปที่ 39 ก่อน-หลังการเชื่อมต่อกระเป๋า MetaMask.....38

รูปที่ 40 แสดงรายการสินค้า.....39

รูปที่ 41 หน้าต่างแสดงข้อมูลสินค้า.....39

รูปที่ 42 หน้าต่างการชำระเงินค่าสินค้า.....40

รูปที่ 43 หน้าต่างประวัติการชำระเงินค่าสินค้า.....41

บทที่1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในปัจจุบันบล็อกเชน (Blockchain) เป็นเทคโนโลยีที่มีคนให้ความสนใจและพูดถึงอย่างมาก โดยเฉพาะการทำธุรกรรมทางการเงินแบบออนไลน์ ทั้งนี้ประโยชน์หลักของมันคือเป็นเทคโนโลยีการเก็บข้อมูลที่มีความปลอดภัย น่าเชื่อถือ โดยไม่ต้องอาศัยคนกลางในการทำธุรกรรม ซึ่งทำให้ธุรกรรมออนไลน์ต่าง ๆ สามารถทำได้อย่างสะดวก รวดเร็ว มีปลอดภัย และที่สำคัญคือเสียค่าธรรมเนียมน้อยมากเมื่อเทียบกับระบบปัจจุบัน แม้ว่าคนสองคนจะไม่เคยรู้จักกันมาก่อนก็สามารถแลกเปลี่ยนข้อมูลกันได้อย่างมั่นใจ ทั้งนี้ความสามารถหนึ่งของเทคโนโลยีบล็อกเชนซึ่งถือว่าเป็นจุดเด่นกว่าเทคโนโลยีอื่น ๆ คือ การสร้าง Smart Contract หรือที่เรียกว่าสัญญาอัจฉริยะ กล่าวคือในเมื่อบล็อกเชนนั้นเป็นระบบฐานข้อมูลรูปแบบหนึ่งที่ไม่สามารถแก้ไขได้ข้อมูลที่บันทึกอยู่ในโครงข่ายบล็อกเชน จะสามารถเชื่อถือได้และเมื่อเราเขียนคำสั่งหรือโปรแกรมใด ๆ ก็ตามลงไปบนบล็อกเชน คำสั่งนั้นจะทำงานอย่างเที่ยงตรงและไม่สามารถแก้ไขได้ เปรียบเสมือนสัญญาที่จะไม่บิดพลิ้วแต่อยู่ในรูปแบบดิจิทัล โดยผู้ที่พัฒนา DApp (Decentralized Application) หรือ สัญญาอัจฉริยะบนโครงข่าย Ethereum นั้นจะต้องใช้ Ether เป็นค่าธรรมเนียม ที่นำ Ethereum มาใช้เพราะเป็นสกุลเงินดิจิตอลที่ได้รับการยอมรับอย่างกว้างขวาง รองรับระบบการเงินเหมือนบิทคอยน์ มีความปลอดภัยและไม่ถูกควบคุมจากกลุ่มบุคคลใดบุคคลหนึ่ง จากประโยชน์ที่กล่าวมาข้างต้น จึงเห็นว่าควรมีการศึกษาการนำเอาเทคโนโลยีบล็อกเชนมาประยุกต์ใช้กับระบบสารสนเทศในด้านต่าง ๆ ได้ เมื่อเรานำเอาบล็อกเชนมาใช้ในเงินดิจิทัล โดยเก็บข้อมูลจำนวนเงินที่ทุกคนมี จำนวนการโอนทุกครั้งของทุกๆ คน เข้าไปในบล็อกแต่ละบล็อกต่อกันไปเรื่อย ๆ และกระจายไปให้ทุกคนรับรู้ ก็ทำให้เงินดิจิทัลนั้นมีความโปร่งใส ปลอดภัยจากการแก้ไขโดยคนใดคนหนึ่ง และที่สำคัญเมื่อเรากระจายไปให้ทุกคนแล้ว จึงลดความเสี่ยงว่าระบบจะล่มพร้อมกันทั้งหมดทำให้ข้อมูลที่ถูกเก็บไว้หายไปได้ ซึ่งบล็อกเชนคือเทคโนโลยีที่ทำให้ความน่าเชื่อถือนี้เกิดขึ้นได้ โดยเฉพาะอย่างยิ่งความปลอดภัยของข้อมูลที่ไม่สามารถแก้ไขได้ ความโปร่งใสของข้อมูลที่ทุกฝ่ายสามารถเห็นข้อมูลเดียวกันได้ การเพิ่มประสิทธิภาพในการลดคนกลาง ปัจจุบันบล็อกเชนจึงได้ถูกนำมาใช้ในหลายๆ รูปแบบ ในหลายๆ อุตสาหกรรม ไม่ได้จำกัดเพียงแค่เงินดิจิทัลเพียงอย่างเดียว และช่วยให้เกษตรกรเข้าถึง smart contract โดยเริ่มจากการใช้แพลตฟอร์มหรือแอปพลิเคชันในการเผยแพร่ความรู้และเทคโนโลยีให้แก่เกษตรกรสามารถทำได้อย่างรวดเร็ว มีต้นทุนต่ำ และสามารถสื่อสารแบบสองทางกับเกษตรกรโดยเปิดโอกาสให้เกษตรกรถามหรือให้ข้อมูลตอบกลับได้ ตลอดถึงสามารถใช้สื่อที่หลากหลายที่ทั้งช่วยเสริมสร้างการเรียนรู้ เช่น ในรูปแบบคลิปวิดีโอที่เข้าใจง่ายและผ่านช่องทาง social media ที่เกษตรกรใช้งานเป็นอยู่แล้ว เพื่อทำความเข้ากับ

ใจกับสกุลเงินดิจิทัล ให้พร้อมรับมือกับการเปลี่ยนแปลงของเหรียญในอนาคตและปรับตัวได้ทัน และยังช่วยดึงดูดและสร้างแรงจูงใจให้เกษตรกรหันมาใช้ เช่น การถ่ายทอดในรูปแบบเกม (gamification) นอกจากนี้ แพลตฟอร์มสามารถใช้ช่วยส่งเสริมการแบ่งปันทางเศรษฐศาสตร์ หรือ sharing economy ในรูปแบบต่าง ๆ

ดังนั้น โครงงานนี้จึงนำเสนอการพัฒนาระบบสัญญาอัจฉริยะด้วยเทคโนโลยีบล็อกเชน (Blockchain) โดยการประยุกต์ใช้การพัฒนาสัญญาอัจฉริยะผ่านเทคโนโลยีบล็อกเชน กับระบบขายสินค้าออนไลน์โดยไม่ผ่านตัวกลาง เป็นการศึกษาเทคโนโลยีใหม่ที่กำลังจะเกิดขึ้นในระยะเวลาอันใกล้ เพื่อรองรับสังคมในยุคดิจิทัลอย่างเต็มรูปแบบในอนาคต

1.2 วัตถุประสงค์ของการวิจัย

1.2.1 เพื่อศึกษาการพัฒนาระบบสัญญาอัจฉริยะบนเทคโนโลยีบล็อกเชน

1.2.2 เพื่อประยุกต์ใช้ระบบสัญญาอัจฉริยะกับระบบขายของออนไลน์โดยไม่ผ่านคนกลาง

1.3 ขอบเขตการวิจัย

1.3.1 ขอบเขตเชิงเนื้อหา

- สามารถเพิ่มสินค้าผ่านสัญญาอัจฉริยะบนเทคโนโลยีบล็อกเชนได้
- สามารถแสดงสินค้าผ่านสัญญาอัจฉริยะบนเทคโนโลยีบล็อกเชนได้
- สามารถซื้อขายสินค้าผ่านสัญญาอัจฉริยะบนเทคโนโลยีบล็อกเชนได้
- สัญญาอัจฉริยะสามารถเชื่อมต่อกับสกุลเงินดิจิทัลในการทำธุรกรรมได้
- ระบบเว็บไซต์ร้านค้าสามารถทำงานร่วมกับสัญญาอัจฉริยะ

1.3.2 ขอบเขตเชิงเทคนิค

- ใช้ Hardhat เป็น private blockchain
- ใช้ภาษา solidity ในการพัฒนาระบบสัญญาอัจฉริยะ
- ใช้ Ethereum จ่ายค่าธรรมเนียมเพื่อพัฒนาระบบสัญญาอัจฉริยะ
- ใช้ Ethereum ในการซื้อขายสินค้า
- ใช้ภาษา html, CSS , JavaScript, react.js , node.js ในการพัฒนาระบบเว็บไซต์ร้านค้า
- พัฒนาระบบสัญญาอัจฉริยะร่วมกับ MetaMask Wallet
- พัฒนาระบบร้านค้าร่วมกับ MetaMask Wallet
- พัฒนาระบบร้านค้าร่วมกับสัญญาอัจฉริยะ

1.4 เครื่องมือที่ใช้ในการดำเนินงาน

1.4.1 ฮาร์ดแวร์ที่ใช้ในโครงงานวิจัย

- Notebook / PC

1.4.2 ซอฟแวร์ที่ใช้ในโครงงานวิจัย

- โปรแกรม Visual Studio Code
- โปรแกรม MetaMask
- Framework Hardhat

1.5 วิธีการดำเนินงาน

1.51 ศึกษาวิเคราะห์ข้อมูลและความเป็นไปได้เกี่ยวกับการพัฒนา ระบบสัญญาอัจฉริยะด้วยเทคโนโลยีบล็อกเชน

1.5.2 ศึกษาอัลกอริทึมในการทำงานของบล็อกเชน

1.5.3 ออกแบบระบบ กำหนดลักษณะ ภาษา และโปรแกรม ที่ต้องใช้

1.5.4 พัฒนาระบบสัญญาอัจฉริยะด้วยเทคโนโลยีบล็อกเชนด้วยโปรแกรม visual studio code

1.5.5 ทดลองใช้ระบบสัญญาอัจฉริยะด้วยเทคโนโลยีบล็อกเชน โดยการเพิ่มสินค้า ซื้อขายสินค้า

1.5.6 ปรับปรุงและแก้ไขข้อผิดพลาดของระบบสัญญาอัจฉริยะด้วยเทคโนโลยีบล็อกเชน

1.5.7 สรุปผลการดำเนินงานและจัดทำเอกสาร

1.6 ประโยชน์ที่คาดว่าจะได้รับ

1.6.1 ระบบสัญญาอัจฉริยะที่สามารถเพิ่มสินค้าได้

1.6.2 ระบบสัญญาอัจฉริยะที่สามารถแสดงรายการสินค้า

1.6.3 ระบบสัญญาอัจฉริยะที่สามารถซื้อขายสินค้าโดยไม่ผ่านคนกลาง

บทที่ 2
ทฤษฎีและงานที่วิจัยเกี่ยวข้อง

โครงงานวิจัย เรื่อง ระบบขายสินค้าเกษตรกรออนไลน์ด้วย smart contract บนเทคโนโลยีบล็อคเชน มีการรวบรวมข้อมูล ศึกษาทฤษฎีและงานวิจัยที่เกี่ยวข้อง ดังนี้

2.1 ความรู้เกี่ยวกับระบบ Blockchain
2.2 ความรู้เกี่ยวกับสกุลเงิน Ethereum
2.3 ความรู้เกี่ยวกับหลักการของ Smart Contract
2.4 ความรู้เกี่ยวกับภาษา Solidity
2.5 โปรแกรม Visual Studio Code
2.6 ความรู้เกี่ยวกับ Hardhat
2.7 แอพพลิเคชั่น MetaMask
2.8 แอพพลิเคชั่น DApps
2.9 ความรู้เกี่ยวกับ Remix - Ethereum IDE
2.10 ความรู้เกี่ยวกับ Etherscan.io
2.11 ความรู้เกี่ยวกับภาษา HTML
2.12 ความรู้เกี่ยวกับภาษา CSS
2.13 ความรู้เกี่ยวกับภาษา JavaScript
2.14 ความรู้เกี่ยวกับหลักการของ React.js
2.15 ความรู้เกี่ยวกับหลักการของ Node.js
2.16 ความรู้เกี่ยวกับหลักการของ Web3.js
2.17 งานวิจัยที่เกี่ยวข้อง

2.1 ความรู้เกี่ยวกับระบบ Blockchain

Blockchain คือ เทคโนโลยีการจัดเก็บข้อมูลแบบ Shared Database หรือที่รู้จักกันในชื่อ “Distributed Ledger Technology (DLT)” โดยเป็นรูปแบบการบันทึกข้อมูลที่รับประกันความปลอดภัยว่าข้อมูลที่บันทึกไปก่อนหน้านั้นไม่สามารถที่จะเปลี่ยนแปลงหรือแก้ไข ซึ่งทุกผู้ใช้งานจะได้เห็นข้อมูลชุดเดียวกันทั้งหมด โดยใช้หลักการ Cryptography และความสามารถขอ Distributed Computing เพื่อสร้างกลไกความน่าเชื่อถือ

หลักการทำงานของเทคโนโลยี Blockchain คือ ฐานข้อมูลจะถูกแชร์ให้กับทุก Node1 ที่อยู่ในเครือข่าย และการทำงานของเทคโนโลยี Blockchain จะไม่มีเครื่องใดเครื่องหนึ่งเป็นศูนย์กลางหรือเครื่องแม่ข่าย ซึ่งการทำงานแบบกระจายศูนย์นี้จะไม่ถูกควบคุมโดยคนเพียงคนเดียว แต่ทุก Node จะได้รับสำเนาฐานข้อมูลเก็บไว้ และจะมีการอัพเดตฐานข้อมูลแบบอัตโนมัติเมื่อมีข้อมูลใหม่เกิดขึ้น ทั้งนี้สำเนาฐานข้อมูลของทุกคนในเครือข่ายจะต้องถูกต้อง และตรงกันกับของสมาชิกคนอื่นในเครือข่าย อีกทั้งการบันทึกข้อมูลเข้าสู่ Block ยังอาศัยหลักการทำ Cryptography และการทำ Consensus จากสมาชิกในเครือข่ายด้วยกันก่อนทำการบรรจุข้อมูลลง

องค์ประกอบของเทคโนโลยี Blockchain ประกอบด้วย 4 องค์ประกอบสำคัญ คือ Block คือ ชุดบรรจุข้อมูลซึ่งมี 2 ส่วนคือส่วนของสิ่งของต่าง ๆ ที่ใส่เข้าไปเรียกว่า Item และส่วนแปะหัวกล่องหรือ Header เพื่อใช้บอกให้คนอื่นทราบว่าบรรจุอะไรมา (แต่เปิดดู Item ภายในนั้นไม่ได้) Chain คือ หลักการจดจำทุก ๆ ธุรกรรมของทุก ๆ คนในระบบและบันทึกข้อมูลพร้อมจัดทำเป็นสำเนาบัญชี Ledger แจกจ่ายให้กับทุกคนในระบบ Consensus คือ ข้อตกลงร่วมกัน Validation คือ การตรวจสอบความถูกต้องแบบทบทวนทั้งระบบและทุก Node

ขั้นตอนการทำงาน

ขั้นตอนที่ 1

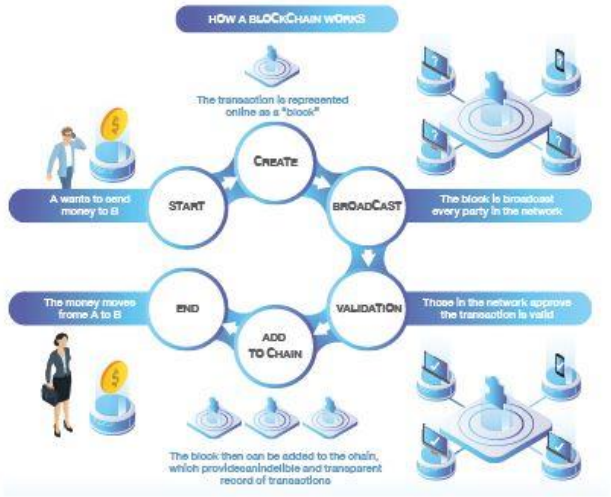
CREATE คือ การสร้าง Block ที่บรรจุคำสั่งขอทำรายการธุรกรรม

ขั้นตอนที่ 2

BROADCAST คือ ทำการกระจาย Block ใหม่นี้ให้กับทุก Node ในระบบ และบันทึกรายการธุรกรรมลง Ledger ให้กับทุก Node เพื่ออัพเดตว่ามี Block ใหม่เกิดขึ้นมา

ขั้นตอนที่ 3

VALIDATION คือ Node อื่น ๆ ในระบบทำการยืนยันและตรวจสอบข้อมูลของ Block นั้นว่าถูกต้องตามเงื่อนไข Validation โดยกระบวนการทำ Consensus ถือว่าเป็นส่วนหนึ่งของกระบวนการทำ Validation [1]



รูปที่ 1 รูปแบบการทำงานของ Blockchain

2.2 ความรู้เกี่ยวกับสกุลเงิน Ethereum

Ethereum เป็นหนึ่งในสกุลเงินดิจิทัลที่มีเทคโนโลยี Blockchain ทำงานรองรับอยู่เบื้องหลัง ได้รับการพัฒนาขึ้นโดย Vitalik Buterin เด็กหนุ่มอัจฉริยะชาวรัสเซีย ผู้เคยอยู่ในทีมงานเบื้องหลังการพัฒนาสกุลเงิน Bitcoin มาก่อน จากนั้นเขาจึงตัดสินใจสร้าง Ethereum หรือ ETH ขึ้นมาในปี 2013 โดยที่เขามีจุดประสงค์ในการสร้าง Ethereum ขึ้นมา เพื่อให้เป็นสกุลเงินดิจิทัลที่สามารถตอบโจทย์ใช้งานได้หลากหลายมากขึ้นกว่า Bitcoin สกุลเงินนี้จึงไม่ได้จำกัดแค่เพียงการใช้งานในเชิงธุรกรรมและการลงทุน แต่ยังสามารถนำไปประยุกต์ใช้ในการทำธุรกิจต่าง ๆ ได้อย่างมีประสิทธิภาพอีกด้วย

ตัวอย่างความสามารถที่นับเป็นข้อดีอันชัดเจนที่สุดอย่างหนึ่งของ Ethereum ได้แก่ ระบบ Smart Contract ซึ่งเป็น Platform ที่ทำให้ผู้ใช้งานสามารถเขียนโปรแกรมลงไปบนข้อมูลของเหรียญ เพื่อออกแบบว่าต้องการนำสกุลเงิน Ethereum ไปใช้ลงทุนหรือจับจ่ายใช้สอยแบบไหนอย่างไรได้ด้วยตนเอง จนส่งผลให้มีการนำ Ethereum ไปประยุกต์ใช้ในธุรกิจหลากหลายรูปแบบทั่วทุกมุมโลก เช่น การชำระค่าบัตรเครดิต การชำระค่าเช่าที่พัก ธุรกิจการเช่ารถ การซื้อขายผ่านช่องทางออนไลน์ การโอนเงินข้ามประเทศ การขอสินเชื่อจากสถาบันการเงิน ฯลฯ [2]

รูปที่ 2 Logo Ethereum

2.3 ความรู้เกี่ยวกับหลักการของ Smart Contract

สัญญาอัจฉริยะ(Smart Contract) คือ โปรแกรมคอมพิวเตอร์ขนาดเล็กซึ่งถูกเก็บไว้ในรูปแบบของบล็อคเชนอีเธอเรียม ตัวสัญญาอัจฉริยะ สามารถใช้งานได้โดยใส่เหรียญอีเธอเรียมเข้าไปในตัวสัญญานั้น ๆ นอกเหนือจากนั้นสัญญาอัจฉริยะจะระบุกฎระเบียบระหว่างผู้ซื้อและผู้ขายไว้ซึ่งมันจะช่วยอำนวยความสะดวกในเรื่องของการแลกเปลี่ยนเอกสาร, หุ้นในบริษัท, เงิน, สินทรัพย์ และอื่น ๆ อีกมากมายที่มีมูลค่าในตัวมันเอง โดยสัญญาอัจฉริยะมีจุดประสงค์หลักคือ ตรวจสอบ, ยืนยัน, บังคับใช้หรือเซ็นสัญญาและข้อตกลงต่าง ๆ ผ่านระบบดิจิตอล

สัญญาอัจฉริยะแตกต่างจากสัญญาทั่ว ๆ ไปตรงที่สัญญาอัจฉริยะสามารถดำเนินการหรือยืนยันข้อตกลงต่าง ๆ ได้อย่างอัตโนมัติโดยไม่ต้องผ่านตัวกลาง

การทำงานของสัญญาอัจฉริยะ

1. การสร้างข้อตกลง

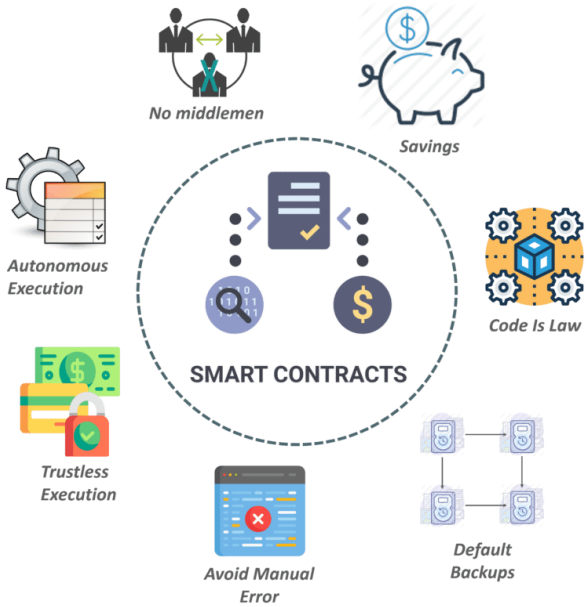
ข้อตกลงระหว่างผู้ซื้อและผู้ขายจะถูกแปลงเป็นรหัสคอมพิวเตอร์ จากนั้นการทำธุรกรรมต่าง ๆ ที่เกิดขึ้นจะถูกบันทึกโดยอัตโนมัติในบล็อคเชนของอีเธอเรียม สัญญาอัจฉริยะจะมีหมายเลขที่อยู่เป็นของตัวเองและเมื่อใดก็ตามที่สัญญาอัจฉริยะถูกบันทึกในบล็อคเชนของอีเธอเรียมใครก็ตามที่มีที่อยู่ของตัวสัญญาอัจฉริยะนั้น ๆ จะสามารถเข้าถึงสัญญาอัจฉริยะได้

2. Triggering Events

สัญญาอัจฉริยะจำเป็นที่จะระบุถึงเหตุการณ์/จุดประสงค์พร้อมด้วยวันหมดอายุของสัญญาเพื่อให้ตัวสัญญาอัจฉริยะทำงานได้ด้วยตัวมันเอง โดยพิจารณาจากข้อตกลงที่ถูกแปลงเป็นรหัส ซึ่งรหัสเหล่านี้จะระบุขั้นตอนต่าง ๆ โดยอาศัยหลักเหตุและผล เราสามารถอธิบายหลักเหตุและผลได้ว่า ถ้าคำสั่งชุดหนึ่งถูกส่งออกมาก็จะให้ผลในรูปแบบรูปแบบหนึ่ง โดยสัญญาอัจฉริยะก็จะทำงานไปเรื่อย ๆ จนกว่าทั้งผู้ซื้อและผู้ขายจะยุติสัญญา

3. การยุติข้อตกลง

เมื่อสัญญาอัจฉริยะถูกสร้างมาแล้ว ผู้ซื้อและผู้ขายจำเป็นที่จะบรรลุจุดประสงค์หรือเหตุการณ์ต่าง ๆ ตามที่ได้ตกลงกันไว้ตอนแรก ถ้าหากฝ่ายใดฝ่ายหนึ่งไม่ได้ทำตามที่ระบุไว้ในสัญญาภายในระยะเวลาที่ตกลงกันไว้บล็อกเชนจะคืนเงินไปให้อีกฝ่ายหนึ่ง [3]



รูปที่ 3 ข้อดีของ Smart contracts

2.4 ความรู้เกี่ยวกับภาษา Solidity

solidity เป็นภาษาทางด้านโปรแกรม ใช้สำหรับการสร้าง Smart Contract บนแพลตฟอร์มบล็อกเชนต่าง ๆ เป็นหลัก โดยเฉพาะ Ethereum ตัว solidity เป็นภาษาที่ได้รับอิทธิพลมาจาก C++, Python และ JavaScript [4]

รูปที่ 4 Logo Solidity

2.5 โปรแกรม Visual Studio Code

Visual Studio Code หรือ VSCode เป็นโปรแกรม Code Editor ที่ใช้ในการแก้ไขและปรับแต่งโค้ด จากค่ายไมโครซอฟท์ มีการพัฒนาออกมาในรูปแบบของ Opensource จึงสามารถนำมาใช้งานได้แบบฟรี ๆ ที่ต้องการความเป็นมืออาชีพ ซึ่ง Visual Studio Code นั้น เหมาะสำหรับนักพัฒนาโปรแกรมที่ต้องการใช้งานข้ามแพลตฟอร์ม รองรับการใช้งานทั้งบน Windows, Mac OS และ Linux สนับสนุนทั้งภาษา JavaScript, Typescript และ Node.js นำมาใช้งานได้ง่ายไม่ซับซ้อน มีเครื่องมือส่วนขยายต่าง ๆ ให้เลือกใช้อย่างมากมาก ไม่ว่าจะเป็นการเปิดใช้งานภาษาอื่น ๆ ทั้ง ภาษา C++, C#, Java, Python, PHP [5]

รูปที่ 5 Logo Program Visual Studio Code

2.6 ความรู้เกี่ยวกับ Hardhat

Hardhat เป็นสภาพแวดล้อมสำหรับการพัฒนาบน Ethereum ที่ช่วยให้นักพัฒนาสามารถจัดการและทำงานที่เกิดซ้ำโดยอัตโนมัติซึ่งมีอยู่ในการสร้างสมาร์ทคอนแทรคและ DApps โดย Hardhat สามารถโต้ตอบโดยตรงกับ Ethereum API [6]

รูปที่ 6 Logo Hardhat

2.7 แอพพลิเคชั่น MetaMask

ระบบนิเวศของ Ethereum คุณจำเป็นต้องมีแอพพลิเคชั่นเช่น Meta Mask ที่เป็นมากกว่ากระเป๋าเงินธรรมดา เพราะมันช่วยให้คุณสามารถใช้งาน DeFi และ dApps ที่มีอยู่มากมายได้อย่างสะดวก ที่ทำให้ผู้ใช้สามารถโต้ตอบกับหน้าเว็บที่รองรับได้โดยตรง

MetaMask เป็นกระเป๋าเงิน Ethereum แบบโอเพนซอร์ส ที่รองรับโทเค็นที่อยู่บน Ethereum ทุกชนิด Meta mask เป็นกระเป๋าเงินดิจิตอลที่ใช้เป็นส่วนเสริมบน Chrome, Firefox และเบราว์เซอร์ หรือให้คุณจัดการกุญแจส่วนตัว Ethereum ผ่านทางเว็บเบราเซอร์ที่กล่าวถึง จะช่วยให้คุณเรียกใช้ Ethereum DApps (แอปพลิเคชันกระจายอำนาจ) จากเบราว์เซอร์โดยไม่ต้องใช้ Ethereum Node เต็มรูปแบบ นั่นหมายความว่าคุณไม่จำเป็นต้องดาวน์โหลดและซิงค์ blockchain ทั้งหมดบนอุปกรณ์หรือคอมพิวเตอร์ของคุณ [7]

รูปที่ 7 logo MetaMask

2.8 แอพพลิเคชั่น DApps

DApps ย่อมาจาก Decentralized แอปพลิเคชัน ซึ่งมีความคล้ายคลึงกับแอพพลิเคชั่นทั่วไปหลายประการ ต่างกันที่แอพพลิเคชั่นปกตินั้นจะอยู่ในรูปแบบ Centralized ซึ่งข้อมูลต่าง ๆ จะถูกเก็บไว้ในเซิฟเวอร์เพียงที่เดียว ทำให้อาจเกิดปัญหา Single Point of Failure ได้ แต่ DApps จะอยู่ในรูปแบบ Decentralized และข้อมูลทั้งหมดจะถูกบันทึกเข้ารหัสลงไปใน Blockchain

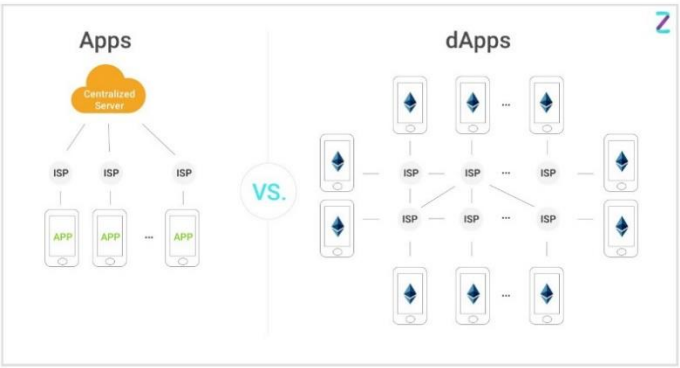
เช่นเดียวกันกับแอพพลิเคชั่นทั่วไป DApps สามารถประยุกต์สร้างอะไรได้หลาย ๆ อย่าง เช่นแอปสำหรับการทำธุรกรรม หรือการชำระเงินแบบ Peer-to-peer ที่ไม่จำเป็นต้องพึ่งตัวกลาง หรือแอปที่ใช้ Smart Contract ในการทำกิจกรรมต่าง ๆ เช่นบริจาคระดมทุน หากระดมทุนไม่ถึงเป้า ก็จะทำการกระขายคืนเงินทั้งหมดทันที เป็นต้น

สรุปลักษณะของ dApps

• ทำงานบน Blockchain

• Code ถูกสร้างขึ้นมาจาก Open-Source และทำงานได้อย่างอิสระโดยไม่มีบุคคลหรือกลุ่มใดควบคุม Token ส่วนใหญ่

• การสร้าง Token ของ dApps เพื่อจัดเตรียมค่าให้กับโหนดที่สนับสนุนและผู้ใช้จะต้องได้รับสิทธิ์การเข้าถึงเพื่อแลกเปลี่ยน Token เช่นเดียวกัน [8]

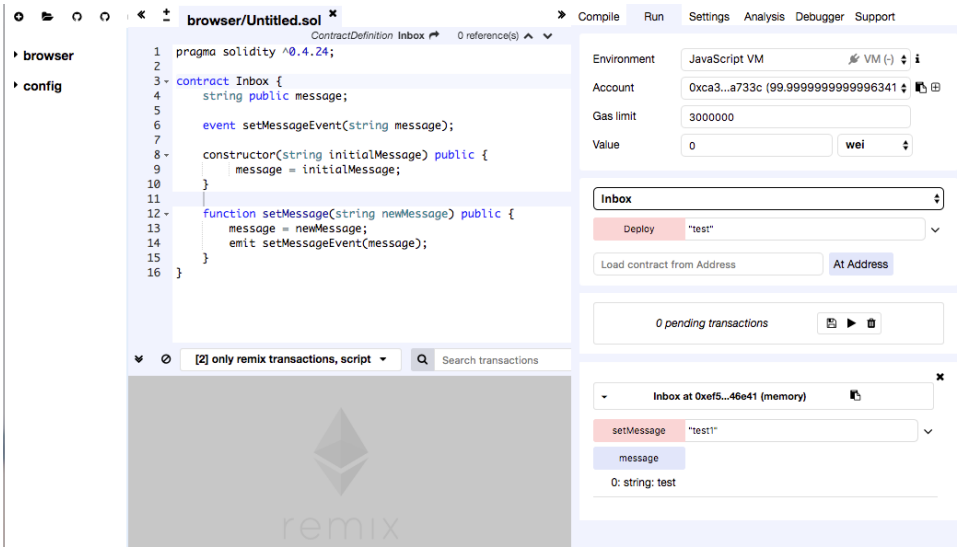


รูปที่ 8 เปรียบเทียบการทำงานของ dApps กับ Apps

2.9 ความรู้เกี่ยวกับ Remix - Ethereum IDE

Remix คือ Solidity IDE ที่รันอยู่บน Browser ซึ่งเราสามารถนำ Solidity code ที่เราเขียนไปรันบนนี้ได้ และสามารถทดสอบ deploy รวมถึงการเรียกใช้ Method setMessage หรือ get ค่าตัวแปร message ที่เก็บอยู่ล่าสุดผ่าน Remix IDE ได้ด้วย

และ Remix IDE ยังเป็น Blockchain tool บน Browser ที่สามารถใช้ในการเขียน smart contract และทดสอบ smart contract ได้แบบง่ายๆ และยังสามารถเชื่อมต่อกับ MetaMask ได้อีกด้วย พัฒนาขึ้นเพื่อสำหรับเขียน code ภาษา Solidity โดยเฉพาะ [9]

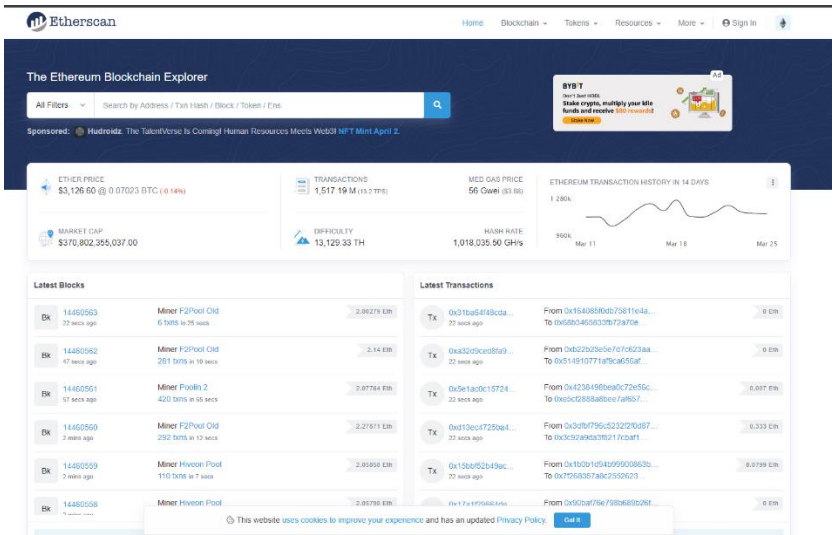


รูปที่ 9 ตัวอย่างหน้าเว็บ Remix - Ethereum IDE

2.10 ความรู้เกี่ยวกับ Etherscan.io

Etherscan ใช้ตรวจสอบว่า contract ที่เราเขียนขึ้นมาถูก deploy ขึ้นบน network เรียบร้อยแล้ว Etherscan คือเครื่องมือในการตรวจสอบ และติดตามธุรกรรม ที่เกิดขึ้นบนสายข้อมูลบล็อกเชนของ Ethereum คอย Scan การบันทึกและตรวจสอบข้อมูลต่างๆ ที่อยู่บนสายข้อมูลของ Ethereum เวลาที่ใครอยากจะเข้าไปตรวจสอบธุรกรรมต่างๆ ก็แค่เอาเลขระบุธุรกรรมใส่เข้าไปในช่องค้นหา ก็จะได้ข้อมูลที่เกี่ยวข้องออกมา เช่น ผู้โอน, ผู้รับโอน, สถานะของธุรกรรม, ค่าธรรมเนียม หรือสำหรับบางคน อาจจะใช้ Etherscan เพื่อตรวจสอบ Smart contract ที่เกี่ยวข้องกับการทำงาน, การลงทุนของตัวเองว่าเป็น Smart contract ที่เชื่อถือได้จริงมั้ย ปลอดภัยรึเปล่า

Etherscan เป็นองค์กรอิสระที่ไม่ได้มีส่วนเกี่ยวข้องหรือรับการสนับสนุนใดๆจากมูลนิธิ Ethereum และไม่ได้ทำหน้าที่เป็นกระเป๋า(Wallet) เก็บสินทรัพย์ดิจิทัล เพียงแต่เป็นกลุ่มคนที่อยากจะเข้ามาอำนวยความสะดวกในการตรวจสอบธุรกรรมต่างๆและดึงเอาข้อดีของเทคโนโลยี Blockchain ซึ่งก็คือความโปร่งใสมาใช้งานให้มีประสิทธิภาพมากขึ้น [10]



รูปที่ 10 ตัวอย่างการทำงานของ Etherscan,io

2.11 ความรู้เกี่ยวกับภาษา HTML

HTML ย่อมาจาก Hypertext Markup Language เป็นภาษาคอมพิวเตอร์รูปแบบหนึ่ง ที่มีโครงสร้างการเขียนโดยอาศัยตัวกำกับ (TAG) ควบคุมการแสดงผลข้อความ รูปภาพ หรือวัตถุ อื่น ๆ ผ่านโปรแกรมเบราเซอร์ แต่ละ Tag อาจจะมีส่วนขยาย เรียกว่า Attribute สำหรับระบุ หรือควบคุมการแสดงผล ของเว็บได้ด้วย HTML เป็นภาษาที่ถูกพัฒนาโดย World Wide Web Consortium (W3C) จากแม่แบบของภาษา SGML (Standard Generalized Markup Language) โดยตัดความสามารถบางส่วนออกไป เพื่อให้สามารถทำความเข้าใจและเรียนรู้ได้ง่าย และด้วยประเด็นดังกล่าว ทำให้บริการ www เติบโตขยายตัวอย่างกว้างขวาง [11]

รูปที่ 11 ตัวอย่างการทำงานของโดยใช้ Tag

2.12 ความรู้เกี่ยวกับภาษา CSS

CSS คือ ภาษาที่ใช้สำหรับตกแต่งเอกสาร HTML/XHTML ให้มีหน้าตา สีสัน ระยะห่าง พื้นหลัง เส้นขอบ และอื่นๆ ตามที่ต้องการ CSS ย่อมาจาก Cascading Style Sheets มีลักษณะเป็นภาษาที่มีรูปแบบในการเขียน Syntax แบบเฉพาะและได้ถูกกำหนดมาตรฐานโดย W3C เป็นภาษาหนึ่งในการตกแต่งเว็บไซต์ ได้รับความนิยมอย่างแพร่หลาย

ประโยชน์ของ CSS

1. ช่วยให้เนื้อหาภายในเอกสาร HTML มีความเข้าใจได้ง่ายขึ้นและในการแก้ไขเอกสารก็สามารถทำได้ง่ายกว่าเดิม เพราะการใช้ CSS จะช่วยลดการใช้ภาษา HTML ลงได้ในระดับหนึ่ง และแยกระหว่างเนื้อหากับรูปแบบในการแสดงผลได้อย่างชัดเจน

2. ทำให้สามารถดาวน์โหลดไฟล์ได้เร็ว เนื่องจาก code ในเอกสาร HTML ลดลง จึงทำให้ไฟล์มีขนาดเล็กลง

3. ช่วยลดเวลาในการปรับปรุงและทำให้การสร้างเอกสารบนเว็บมีความรวดเร็วยิ่งขึ้น นอกจากนี้ยังสามารถควบคุมการแสดงผล ให้คล้ายหรือเหมือนกันได้ในหลาย Web Browser

4. ช่วยในการกำหนดการแสดงผลในรูปแบบที่มีความเหมาะกับสื่อต่างๆ ได้เป็นอย่างดี

5. ทำให้เว็บไซต์มีความเป็นมาตรฐานมากขึ้นและมีความทันสมัย สามารถรองรับการใช้งานในอนาคตได้ [12]

รูปที่ 12 Logo CSS

2.13 ความรู้เกี่ยวกับภาษา JavaScript

JavaScript เป็นภาษาคอมพิวเตอร์สำหรับการเขียนโปรแกรมบนระบบอินเทอร์เน็ต ซึ่งเป็นภาษาสคริปต์เชิงวัตถุ สามารถใช้ร่วมกับภาษา HTML เพื่อการสร้างและพัฒนาเว็บไซต์ ท าให้เว็บไซต์มีการเคลื่อนไหว สามารถตอบสนองผู้ใช้งานได้มากขึ้น ซึ่งมีวิธีการทำงานในลักษณะการแปลความและดำเนินงานไปทีละคำสั่ง เรียกว่า (Object Oriented Programming) ที่มีเป้าหมายในการออกแบบและพัฒนาโปรแกรมในระบบอินเทอร์เน็ต สำหรับผู้เขียนภาษา HTML สามารถทำงานข้ามแพลตฟอร์มได้ โดยทำงานร่วมกับภาษา HTML และภาษาจาวา (Java) ได้ทั้งฝั่งไคลเอนต์ (Client) และฝั่งเซิร์ฟเวอร์ (Server) ซึ่งมีลักษณะการเขียนแบบโปรโตไทพ

(Prototyped-based Programming) ส่วนมากใช้ในหน้าเว็บเพื่อประมวลผลข้อมูลที่ฝั่งของผู้ใช้งาน แต่ก็ยังมีใช้เพื่อเพิ่มเติมความสามารถในการเขียนสคริปต์โดยฝังอยู่ในโปรแกรมอื่นๆ [13]

รูปที่ 13 Logo JavaScript

2.14 ความรู้เกี่ยวกับหลักการของ React.js

React เป็น JavaScript Library ที่ถูกสร้างโดย Facebook

React เป็นแค่ UI โดยสร้างมาจากพื้นฐานแนวความคิดแบบ MVC (Model View Controller) ซึ่งหมายถึงว่า React มีหน้าที่จัดการกับ Model หรือ View แต่ส่วนใหญ่จะเป็น View ใน React มีสิ่งที่เราต้องรู้หลัก ๆ คือ 3 Concept

1. Component – ส่วนต่าง ๆ ในเว็บเราจะมองเป็น Component
2. State – ข้อมูลที่อยู่ใน Component แต่ละชิ้น เราเรียกว่า State
3. Props – ข้อมูลที่ถูกส่งต่อจาก Component ชั้นบนลงไปชั้นล่าง เราเรียกว่าProps (Properties)

Component ก็เหมือนกับการเขียน HTML แต่ใน React ใช้สิ่งที่เรียกว่า JSX ในการแสดงผลเว็บไซต์หน้าตาจะเหมือน HTML มาก แตกต่างตรงที่เราสามารถเข้าไปเขียนในไฟล์ JavaScript ได้เลย แทนการเขียนไฟล์ HTML ทำให้เราสามารถเล่นอะไรกับมันได้มากกว่า

ก่อนอื่นเลยเราต้องเข้าใจกันก่อนว่า React นั้นเป็น JavaScript Framework ที่ทำหน้าที่ Compile Code ของเราให้กลายเป็น Html+Css+Javascript เพื่อให้สามารถใช้งานบน Browser ได้นั่นเอง โดยที่ตัว React นั้นอาศัย NodeJS เพื่อทำการ Compile Code ครับ [14]

รูปที่ 14 Logo React

2.15 ความรู้เกี่ยวกับหลักการของ Node.js

เมื่อพูดถึง JavaScript หลายคนอาจนึกถึงภาษาโปรแกรมมิ่งที่นำไปใช้พัฒนาเว็บไซต์ (Web Development) เพื่อที่จะเพิ่มการตอบโต้กับผู้ใช้ หรือเพิ่มปุ่มหรือสไลด์ต่าง แต่ Node.js สามารถประยุกต์ใช้ความสามารถในการเขียน JavaScript เดิมให้กว้างขึ้น ตั้งแต่การเขียนเครื่องมือให้โปรแกรม Command Line ไปจนถึงการสร้าง เซิฟเวอร์เพื่อนำไปจัดการขอมูลเอกสารในระบบ

Node.js เปิดตัวครั้งแรกเมื่อปี 2009 และถูกนำไปใช้โดยบริษัทใหญ่ๆ มากมาย เช่น Paypal Netflix และ Microsoft เพื่อนำไปสร้าแอพพลิเคชั่นที่สามารถปรับขนาด และทำงานตามสถานะการได้ รวมไปถึงการใช้ทั่วไป

NodeJS หรือเรียกสั้นๆว่า Node คือ JavaScript Runtime Environment พูดง่าย ๆ ก็คือเป็นตัวรันภาษา JavaScript ที่ Server [15]

รูปที่ 15 Logo Node.js

2.16 ความรู้เกี่ยวกับหลักการของ Web3.js

Web3.js เป็น JavaScript Library ที่ใช้ติดต่อกับโหนดของ Ethereum ผ่านทาง Http หรือ IPC

Ethereum Blockchain เกิดจากโหนด หลายๆโหนด ที่มีการแชร์ข้อมูลระหว่างกัน โดยมีข้อมูลเป็นชุดเดียวกัน ทั้งหมด

web3.js ก็เป็นการประกาศว่า เราจะทำการเชื่อม ไปยังโหนดไหน บน Ethereum เพื่ออ่าน หรือ เขียนข้อมูล บนโหนดนั้น ซึ่งเราสามารถ ตั้งโหนด ขึ้นมาเองได้ โดยใช้ geth (Go-Ethereum | Command line Interface) [16]

รูปที่ 16 Logo Wed3.js

2.17 งานวิจัยที่เกี่ยวข้อง

(1) การพัฒนาระบบบริหารสหกรณ์โดยใช้เทคโนโลยีบล็อกเชน สารนิพนธ์นี้เป็นการนำเสนอ การนำเทคโนโลยีบล็อกเชน มาช่วยในการพัฒนาระบบจัดการและบริหารสหกรณ์ ซึ่งจะทำให้การโอนย้ายหุ้น การซื้อขายหุ้นของสมาชิกเป็นไปได้ง่าย เนื่องจากข้อมูลที่เก็บไว้ในบล็อกเชนสามารถตรวจสอบธุรกรรมได้ตลอดเวลา และอยากต่อการเปลี่ยนแปลงหรือแก้ไขข้อมูล สมาชิกยังสามารถโอนหุ้นหรือตรวจเช็คจำนวนหุ้นของตัวเองได้โดยไม่ผ่านคนกลาง ช่วยลดปัญหาการปลอมแปลงเอกสาร เป็นการเพิ่มความปลอดภัยและความน่าเชื่อถือในองค์กรณ์มากยิ่งขึ้น (นันทา จันทร์พิทักษ์ และรัฐพล มณฑา 2562) [17]

ส่วนที่คล้ายในวิจัย คือ การนำเทคโนโลยีบล็อคเชนมาใช่ในการพัฒนาระบบร้านค้า ใก้สามารถทำธุรกรรมได้ง่ายโดยไม่ผ่านคนกลาง

ส่วนที่แตกต่างในวิจัย คือ การพัฒนาตัวระบบจดการและบริหารสหกรณ์ ซึ่งแตกต่างจากระบบร้านค้า

(2) ปัจจัยที่มีผลต่อการตัดสินใจใช้เทคโนโลยีบล็อกเชนบริษัทมหาชนที่จดทะเบียนในตลาดหลักทรัพย์แห่งประเทศไทย สารนิพนธ์นี้เป็นการนำเสนอ ด้านปัจจัยที่มีผลต่อการตัดสันใจใช้เทคโนโลยีบล็อกเชนมาใช้ในธุรกิจทำให้เพิ่มประสิทธิภาพในการทำงาน ทั้งในด้านข้อมูล ด้านระบบงาน และยังสร้างรูปโฉมทางธุรกิจใหม่ได้จริงตามลักษณะและคุณสมบัติของเทคโนโลยีบล็อกเชน (ธัญวรัตน์ ธนัชญ์นันท์ 2562) [18]

ส่วนที่คล้ายในวิจัย คือ ศึกษาอัลกอริทึมของระบบบล็อคเชนและสัญญาอัจฉริยะ

ส่วนที่แตกต่างในวิจัย คือ เป็นกรณีศึกษาปัจจัยที่มีผลต่อการตัดสินใจนำเทคโนโลยีบล็อคเชนมาใช้ในทางธุรกิจ

(3) Smart Contract สัญญาอัจฉริยะ ความฝันและความจริง สารนิพนธ์นี้เป็นการนำเสนอ

การทำสัญญาอัจฉริยะ การทำสัญญามีการปฏิบัติอย่างเข้มงวดเพื่อลดภาระในการบังคับให้เป็นไปตามข้อสัญญาที่กำหนดอันเป็นที่มาของการฟ้องร้อง สัญญาอัจฉริยะเป็นการพัฒนาการเขียนโค้ดในคอมพิวเตอร์ที่ทำให้

ระบบคอมพิวเตอร์สามารถดำเนินการทำสัญญาและบังคับให้เป็นไปตามข้อกำหนดของสัญญา หากปรากฏเหตุการณ์หรือเงื่อนไขที่เขียนไว้ เพื่อทำการปฏิบัติและบังคับให้เป็นไปตามสัญญาไม่ต้องอาศัยมนุษย์มาดำเนิน ทำให้ลดภาระการฝ่าฝืนหรือละเลยไม่ปฏิบัติตามสัญญาที่จะทำให้เกิดภาระตามมาของการฟ้องร้องดำเนินคดี (สรวิศ ลิมปรังษี)

ส่วนที่คล้ายในวิจัย คือ การสร้างเงื่อนไขในสัญญาอัจฉริยะ เพื่อให้เป็นไปตามข้อกำหนดของระบบซื้อขายสินค้า

ส่วนที่แตกต่างในวิจัย คือ การสร้างเงื่อนไขที่แตกต่าง และการ deploy สัญญาอัจฉริยะ

(4) แอปพลิเคชันล็อตเตอรีแบบกระจายศูนย์โดยใช้เทคโนโลยีบล็อกเชน สารนิพนธ์นี้เป็นการนำเสนอ การนำเทคโนโลยีบล็อกเชนมาประยุกต์ใช้กับระบบล็อตเตอรี่เพื่อสร้างแอปพลิเคชันขึ้นมาโดยจะดำเนินการโดยสัญญาอัจฉริยะบนแพลตฟอร์มเพื่อลดต้นทุนและเพิ่มความปลอดภัยในการเก็บข้อมูลของระบบ และความโปร่งใส แก้ปัญหาล็อตเตอรี่เกินราคาที่สำนักงานสลากพบในปัจจุบัน และปัญหาการระบุความเป็นเจ้าของล็อตเตอรี่ (อัษฎางค์ ขำมะโน และ เอกรัฐ รัฐกาญจน์) [19]

ส่วนที่คล้ายในวิจัย คือ การสร้างเงื่อนไขทำให้เกิดความโปร่งใสของข้อมูลในการธุรกรรม

ส่วนที่แตกต่างในวิจัย คือ ระบบล็อตเตอรี่อบบกระจายศูนย์ เป็นการใช้ระบบในการสุ่ม โดยสร้างเงื่อนไขจากจำนวนผู้ซื้อ

(5) ต้นแบบการอนุมัติการจัดการระบบบริการสหกรณ์ร้านค้าดิจิทัลในประเทศไทยที่มีความน่าเชื่อถือด้วยการประยุกต์ใช้บล็อกเชน สารนิพนธ์นี้เป็นการนำเสนอ การพัฒนาระบบการจัดเก็บข้อมูลของสหกรณ์ร้านค้าดิจิทัล โดยใช้เทคโนโลยีบล็อกเชนเข้ามาช่วยในการบันทึกธุรกรรมต่างๆ ที่เกิดขึ้นเพื่อให้เกิดความมั่นใจในข้อมูลและความปลอดภัยของข้อมูล โดยระบบบล็อกเชนนั้นเป็นการเข้ารหัสข้อมูลธุรกรรมต่างๆ ที่เกิดขึ้น และจัดเก็บในรูปแบบที่ทำการแก้ไขหรือเปลี่ยนแปลงข้อมูลได้ยาก เพื่อใช้ในการตรวจสอบความถูกต้องของข้อมูล และสามารถทำงานควบคู่ไปกับฐานข้อมูลการจัดเก็บเดิมของระบบบริการสหกรณ์ร้านค้าได้ (วรวุฒิ บัวทองจันทร์2561) [20]

ส่วนที่คล้ายในวิจัย คือ การพัฒนาระบบการจัดเก็บข้อมูลร้านค้าดิจิทัล โดยใช้เทคโนโลยีบล็อกเชนเข้ามาช่วยในการบันทึกธุรกรรมต่างๆ

ส่วนที่แตกต่างในวิจัย คือ การทำงานร่วมกับระบบฐานข้อมูลเดิมที่มีอยู่แล้ว นำมาพัฒนาร่วมกับทคโนโลยีบล็อคเชนเพื่อให้เกิดความปลอดภัยมากยิ่งขึ้น

บทที่ 3
วิธีการดำเนินงาน

การดำเนินงานและจัดทำการวิจัยนี้ เป็นการพัฒนาระบบสัญญาอิเล็กอัจฉริยะด้วยเทคโนโลยีบล็อกเชน โดยมีการรวบรวมข้อมูลความสนใจของผู้บริโภคแล้วนำไปประยุกต์ใช้เพื่อพัฒนาธุรกิจ เพื่อให้ระบบที่จัดทำขึ้นมีประสิทธิภาพและเลือกแนวทางในการดำเนินงานและวิธีการใช้เครื่องมือ ซึ่งสามารถสรุปขั้นตอนการดำเนินการวิจัยได้ดังนี้

3.1 แผนงานการดำเนินงาน

3.2 เครื่องมือที่ใช้

3.3 แผนผังการทำงานของระบบ

3.4 ขั้นตอนการพัฒนาระบบ

3.5 สรุปขั้นตอนการดำเนินงาน

3.1 แผนงานการดำเนินงาน

ตารางที่ 1 แผนการดำเนินงานวิจัย

ลำดับที่	แผนการดำเนินงาน	ปีการศึกษา											
		2/2563											
		ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.	พ.ค.		ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.
1	คิดหัวข้อโปรเจค	↔											
2	ศึกษาและรวบรวมข้อมูล		↔	↔									
3	ออกแบบระบบเครือข่าย				↔	↔							
4	พัฒนาระบบ						↔	↔	↔	↔	↔		
5	ทำการทดสอบ											↔	
6	สรุปผลกาดำเนินงานและจัดทำเอกสาร												↔

3.1.1 รวบรวมความต้องการของระบบ

การรวบรวมความต้องการได้ทำการขอคำแนะนำจากอาจารย์ที่ปรึกษาเกี่ยวกับระบบการทำงานของระบบสัญญาอัจฉริยะด้วยเทคโนโลยีบล็อกเชน เพื่อรวบรวมข้อมูลความต้องการ ปัญหาที่พบ รวมถึงฟังก์ชันการทำงานที่สามารถช่วยให้การทำงานมีประสิทธิภาพมากขึ้น

3.1.2 ศึกษาความเป็นไปได้

หลังจากที่รวบรวมความต้องการของระบบเรียบร้อยแล้ว จากนั้นนำข้อมูลที่ได้จากการรวบรวมมาทำการศึกษาความเป็นไปได้ในการทำงานของระบบดังนี้จัดเตรียมข้อมูล สร้างแบบจำลองและทดสอบแบบจำลอง

- สามารถเพิ่มสินค้าผ่านสัญญาอัจฉริยะบนเทคโนโลยีบล็อกเชนได้
- สามารถสามารถซื้อขายสินค้าผ่านสัญญาอัจฉริยะบนเทคโนโลยีบล็อกเชนได้

3.1.3 วิเคราะห์ระบบ

จากนั้นนำข้อมูลที่ได้จากการรวบรวมและศึกษาความเป็นไปได้มาทำการวิเคราะห์หาฟังก์ชันการทำงานของระบบ รวมถึงกระบวนการทำงานในแต่ละส่วนของระบบ เพื่อนำไปสู่การออกแบบของระบบ

3.1.4 ออกแบบระบบ

ทำการออกแบบระบบขายสินค้าเกษตรกรออนไลน์ด้วยสัญญาอัจฉริยะ บนเทคโนโลยีบล็อคเชน ตัวระบบควรมีรูปแบบฟังก์ชันการทำงานอย่างไร และระบบสามารถทำอะไรได้บ้าง โดยออกแบบภาพ Flowchart เป็นการอธิบายถึงวิธีการทำงานของระบบ

3.1.5 พัฒนาระบบ

เริ่มลงมือพัฒนาระบบขายสินค้าเกษตรกรออนไลน์ด้วยสัญญาอัจฉริยะ บนเทคโนโลยีบล็อคเชน โดยการเขียนเว็ปแอปพลิเคชันให้ทำงานตามที่ได้ออกแบบไว้

3.1.6 ทำการทดสอบและตรวจสอบปัญหาข้อผิดพลาดแล้วทำการแก้ไข

เมื่อทำการออกแบบระบบขายสินค้าเกษตรกรออนไลน์ด้วยสัญญาอัจฉริยะ บนเทคโนโลยีบล็อคเชน ให้สำเร็จแล้วนำมาทำการตรวจสอบเพื่อหาข้อผิดพลาดที่ทำให้ระบบเกิดปัญหาได้ภายหลังและทำการแก้ไขให้ระบบมีความเสถียรยิ่งขึ้น

3.1.7 สรุปผลและนำเสนอ

เมื่อทำกระบวนการก่อนหน้านี้สำเร็จทั้งหมด ให้นำข้อมูลทั้งหมดจัดทำเอกสารเพื่อใช้ในการ นำเสนอต่อไป ดำเนินการจัดทำเอกสารเกี่ยวกับการดำเนินการในขั้นตอนที่ผ่านมา การจัดทำเอกสารสามารถเริ่มทำเมื่อกระบวนการแรกเกิดขึ้น เพื่อที่จะนำมาเป็นหลักฐานเอกสารอ้างอิงในการจัดทำระบบว่ามีการใช้ทฤษฎีใดที่เกี่ยวข้องและรวมไปถึงจัดทำคู่มือประกอบการใช้งานระบบ

3.2 เครื่องมือที่ใช้

ด้านฮาร์ดแวร์ที่ใช้ในการวิจัยมีดังนี้

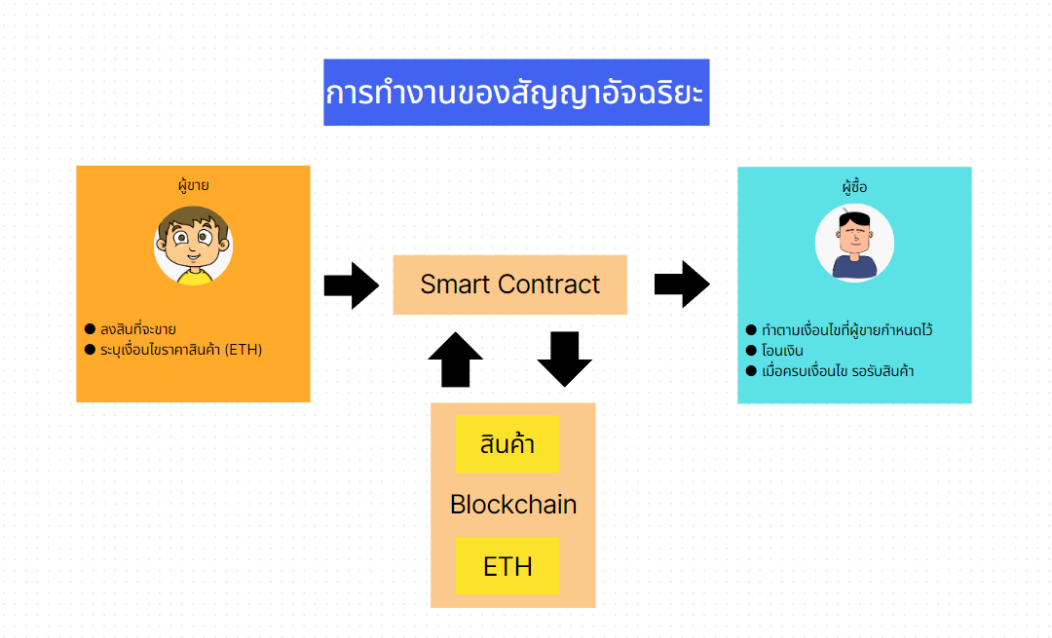
- คอมพิวเตอร์

หน่วยประมวลผลกลาง	Intel(R) Core(TM) i5-8400 CPU@2.80GHz2.81 GHz.
หน่วยความจำหลัก	16.0 GB
หน่วยความจำสำรอง	1 TB HDD
หน่วยประมวลผลกราฟฟิก	Nvidia GeForce GTX 1050ti 4GB
หน้าจอการแสดงผล	23 LED (1920×1080)

- ระบบปฏิบัติการวินโดว์ 10 เป็นระบบปฏิบัติการที่ใช้รองรับการทำงาน
- Hardhat ใช้เป็น Private Blockchain
- Remix - Ethereum IDE,VS code ใช้สำหรับเขียนโค้ดให้กับระบบสัญญาอัจฉริยะบนเทคโนโลยีบล็อกเชน
- MetaMask เป็นกระเป๋าเงิน Ethereum

3.3 แผนผังการทำงานของระบบ

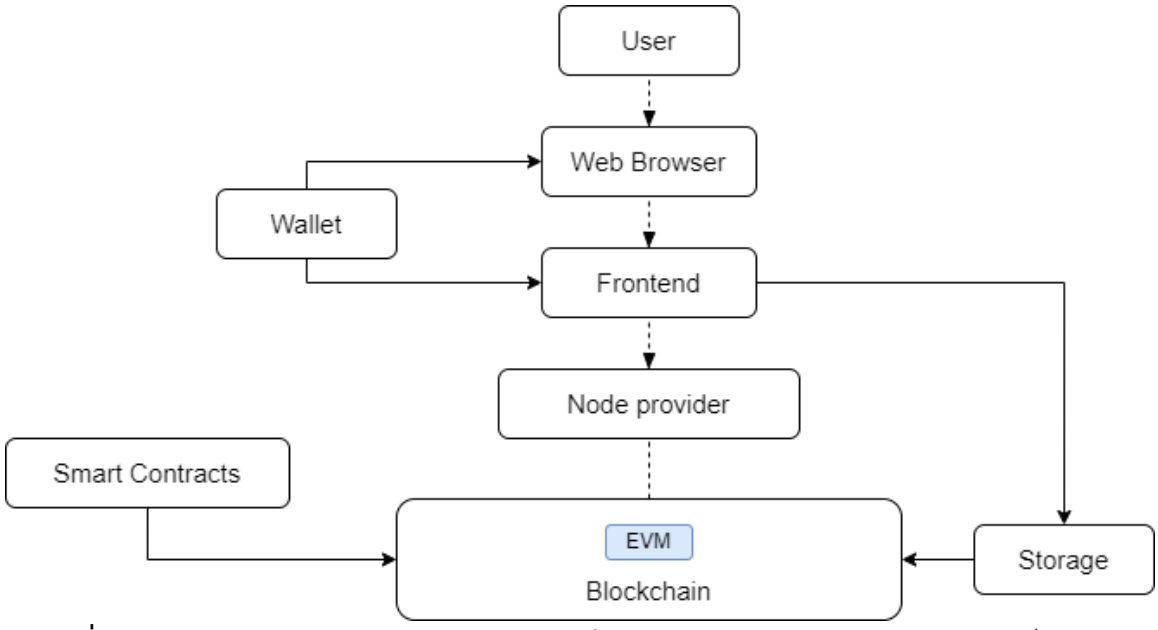
3.3.1 การทำงานของสัญญาอัจฉริยะบนเทคโนโลยีบล็อคเชน



รูปที่ 17 คอนเซ็ปต์การทำงานของสัญญาอัจฉริยะบนเทคโนโลยีบล็อคเชน

จากรูปที่ 17 ผู้ขายทำการสร้างสัญญาอัจฉริยะ โดยการกำหนดสินค้า ราคา แล้วส่งสัญญาอัจฉริยะไปบนบล็อคเชน เมื่อมีผู้ซื้อที่ทำตามเงื่อนไขที่ผู้ขายได้กำหนดไว้ ก็จะทำการโอน ETH ตามเงื่อนไขราคาที่กำหนดไว้ไปยังบล็อคเชน และส่งเข้าสู่กระเป๋าเงิน MetaMask ของผู้ขาย

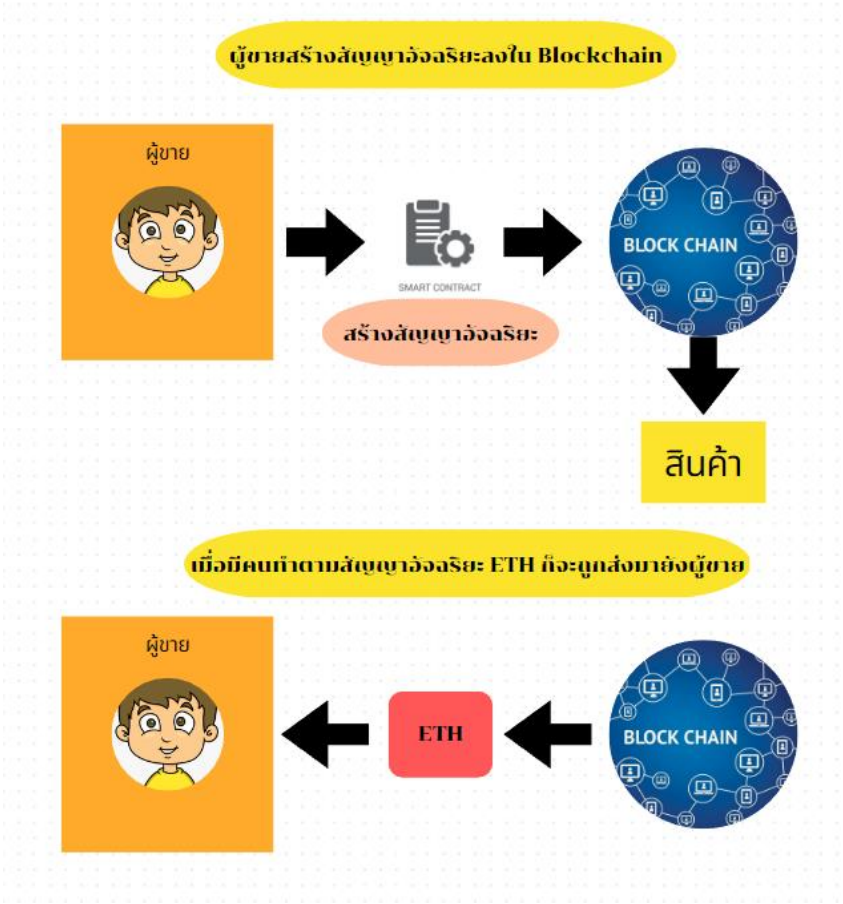
3.3.2 โครงสร้างระบบขายสินค้าเกษตรกรออนไลน์ด้วยสัญญาอัจฉริยะบนเทคโนโลยีบล็อกเชน



รูปที่ 18 โครงสร้างระบบขายสินค้าเกษตรกรออนไลน์ด้วยสัญญาอัจฉริยะบนเทคโนโลยีบล็อกเชน

จากรูปที่ 18 ขั้นตอนการใช้งาน เพียงผู้ซื้อมีกระเป๋า MetaMask ก็สามารถใช้งานผ่าน Browser ใดก็ได้ โดยเข้าใช้งาน Website ที่สร้างขึ้นด้วย HTML, CSS, JavaScript, React.js โดย Website จะทำงานร่วมกับสัญญาอัจฉริยะ ที่เขียนโดยภาษา Solidity ถ้าผู้เข้าใช้งานทำตามเงื่อนไขที่เขียนบนสัญญาอัจฉริยะ หรือ ซื้อสินค้าที่ตรงกับตัวเองต้องการ Wallet ก็จะถูกโอนเข้า MetaMask ของผู้ขายทันที

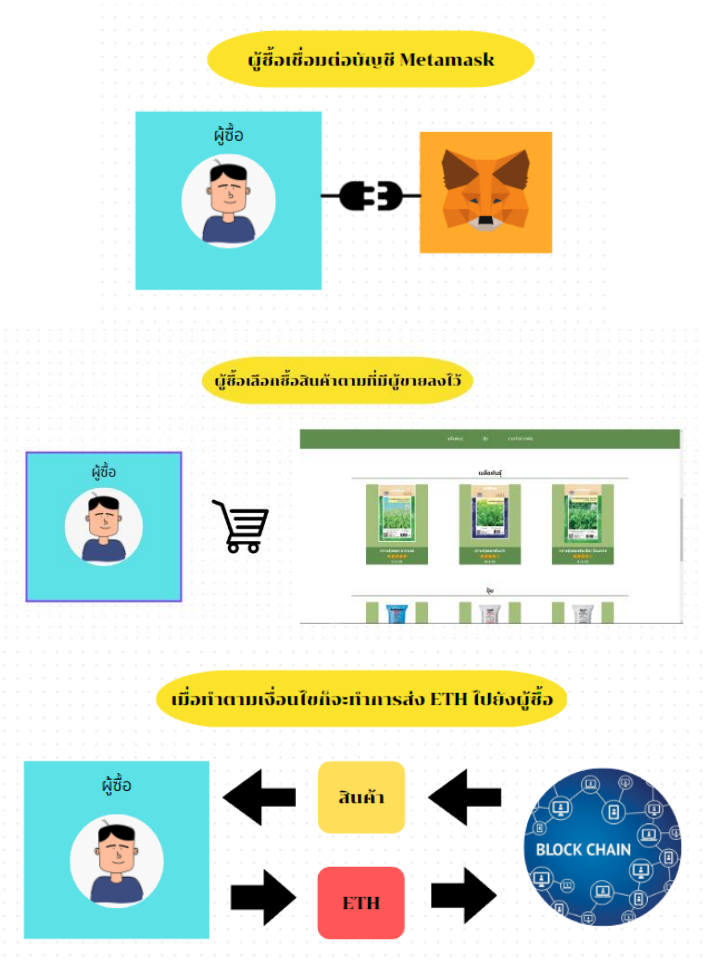
3.3.3 ฟังก์ชันการเพิ่มข้อมูลและขายสินค้า



รูปที่ 19 ตัวอย่างฟังก์ชันการเพิ่มข้อมูลสินค้า

จากรูปที่ 19 ผู้ขายทำการกรอกข้อมูลสินค้า หลังจากนั้น deploy ขึ้นไปเน็ตเวิร์ก เป็นการสร้างเงื่อนไขสัญญาอัจฉริยะขึ้นไปยังบล็อคเชน เพื่อทำให้ข้อมูลเกิดความโปร่งใส บล็อคเชนก็จะเก็บข้อมูลสัญญาอัจฉริยะ และข้อมูลสินค้า เมื่อมีลูกค้าทำตามเงื่อนไขในสัญญาอัจฉริยะ บล็อคเชนก็จะทำการส่งเงิน ETH จากผู้ซื้อมายังผู้ขายอัตโนมัติ

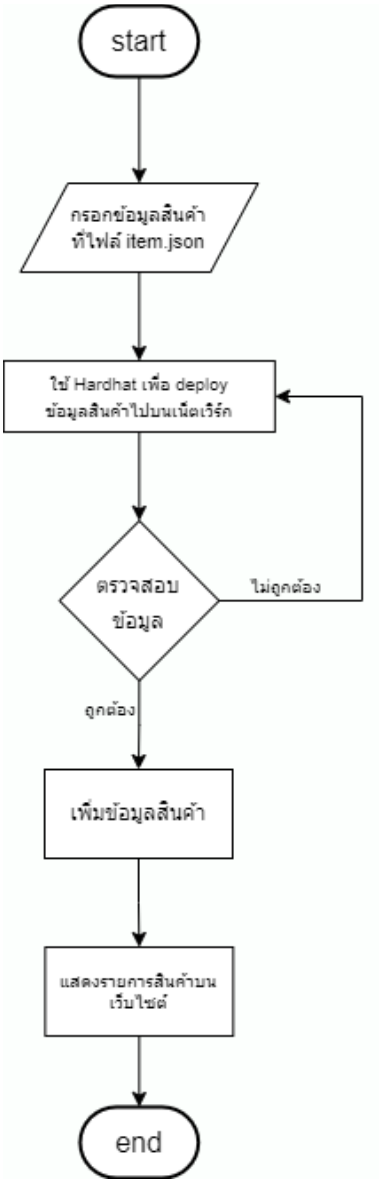
3.3.4 ฟังก์ชันการซื้อสินค้า



รูปที่ 20 ตัวอย่างฟังก์ชันการซื้อสินค้า

จากรูปที่ 20 เมื่อเข้าสู่เว็บไซต์ ผู้ซื้อทำการเชื่อมต่อกับกระเป๋า MetaMask เพื่อซื้อสินค้า หลังจากนั้นเลือกสินค้าที่ต้องการ กดดูข้อมูลสินค้า หลังจากเลือกสินค้าแล้ว กดปุ่ม “ซื้อ” ระบบจะเด้งหน้าต่าง MetaMask ขึ้นมาเพื่อให้ชำระสินค้า กด “Confirm” เพื่อชำระเงินค่าสินค้าเป็นสกุลเงิน ETH โดยจะหักจากกระเป๋า MetaMask ของผู้ซื้อเข้าของผู้ขายโดยอัตโนมัติ สามารถกดดูประวัติการชำระเงินได้ที่แอพพลิเคชั่น MetaMask

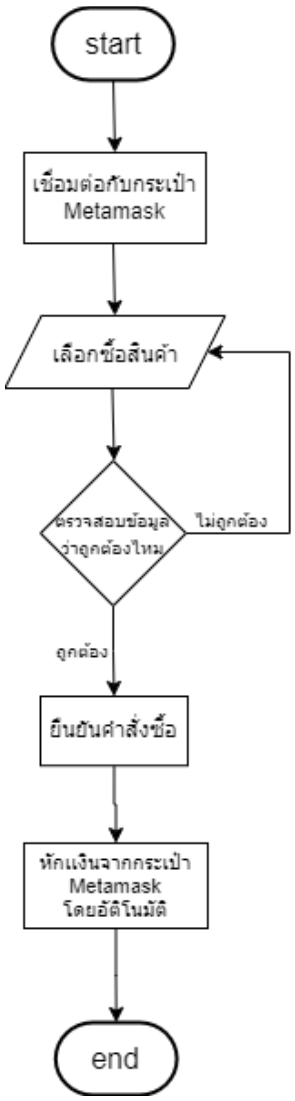
3.3.5 แผนผังการทำงานของระบบในการเพิ่มสินค้า



รูปที่ 21 การทำงานของระบบ (Flowchart) การเพิ่มสินค้า

จากรูปที่ 21 แสดงการทำงานของกระบวนการเพิ่มสินค้า โดยทำการกรอกข้อมูลสินค้า หลังจากนั้นทำการใช้ Hardhat เพื่อ deploy ข้อมูลสินค้าไปบนเน็ตเวิร์กเพื่อไม่ให้เกิดการแก้ไขข้อมูล ทำการตรวจสอบข้อมูลอีกครั้ง ถ้าข้อมูลไม่ถูกต้อง ต้องทำการ deploy ใหม่ ถ้าถูกต้องแล้ว ระบบจะแสดงข้อมูลสินค้าบนหน้าเว็บไซต์ จบการทำงาน

3.3.6 แผนผังการทำงานของระบบในการซื้อสินค้า



รูปที่ 22 การทำงานของระบบ (Flowchart) การซื้อสินค้า

จากรูปที่ 20 แสดงการทำกระบวนการซื้อสินค้าของลูกค้า โดยเข้าหน้าเว็บไซต์ เชื่อมต่อกับกระเป๋า MetaMask ทำการเลือกซื้อสินค้า และตรวจสอบข้อมูล ถ้าไม่ถูกต้องเลือกซื้อสินค้าอีกครั้ง ถ้าถูกต้อง กดยืนยันคำสั่งซื้อ ระบบหักเงินจากกระเป๋า MetaMask โดยอัตโนมัติ จบการทำงาน

3.4 ขั้นตอนการพัฒนาระบบ

3.4.1 เขียนสัญญาอัจฉริยะโดยภาษา solidity

```
// SPDX-License-Identifier: UNLICENSED
pragma solidity ^0.8.9;

contract Farmer {
    address public owner;

    struct Item {
            uint256 id;
            string  name;
            string  description;
            string  category;
            string  image;
            uint256 cost;
            uint256 rating;
            uint256 stock;
    }

    struct Order {
        uint256 time;
        Item item;
    }

    mapping(uint256 => Item) public items;
    mapping(address => mapping(uint256 => Order)) public orders;
    mapping(address => uint256) public orderCount;

    event Buy(address buyer, uint256 orderId, uint256 itemId);
    event List(string name, uint256 cost, uint256 quantity);
```

รูปที่ 23 ตัวอย่างสัญญาอัจฉริยะ

จากรูปที่ 23 สร้างเงื่อนไขสัญญาอัจฉริยะด้วยภาษา Solidity บน Remix IDE เพื่อใช้ในการกำหนดเงื่อนไขในการทำธุรกรรม

3.4.2 เขียน JavaScript ให้ใช้งานร่วมกับ Contract ที่สร้างไว้

```
const { expect } = require("chai")
const { ethers } = require("hardhat")
const tokens = (n) => {
  return ethers.utils.parseUnits(n.toString(), 'ether')
}

// Global constans for listing an item...
  const ID = 1
  const NAME = "กระด้าย"
  const DESCRIPTION = "dsfsdfsdf"
  const CATEGORY = "ปีย"
  const IMAGE = "https://media.discordapp.net/attachments/808186574382563349/1072811561007194122/60251770r.webp"
  const COST = tokens(1)
  const RATING = 4
  const STOCK = 5


describe("Farmer", () => {
  let farmer
  let deployer, buyer

  beforeEach(async () =>{
    // Setup accounts
    [deployer, buyer] = await ethers.getSigners()

    // Deploy contact
    const Farmer = await ethers.getContractFactory("Farmer")
    farmer = await Farmer.deploy()
  })

  describe("Deplyment", () => {
    it("Sets the owner", async () => {
      expect(await farmer.owner()).to.equal(deployer.address)
    })
  })
```

รูปที่ 24 ตัวอย่างการใช้ JavaScript ร่วมกับ Solidity

จากรูปที่ 24 ใช้ภาษา JavaScript ให้ใช้งานร่วมกับตัวสัญญาอัจฉริยะ เพื่อใช้ในการพัฒนาเว็บไซต์ร้านค้า

3.4.3 ทำการทดสอบสัญญาอัจฉริยะโดย Hardhat

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.2728]
(c) Microsoft Corporation. All rights reserved.

C:\Users\axe\farmer>npx hardhat test


  Farmer
    Deployment
      √ Sets the owner (55ms)
    Listing
      √ Returns item attributes
      √ Emits List event
    Buying
      √ Updates buyer's order count
      √ Adds the order
      √ Updates the contract balance
      √ Emits Buy event
    Withdrawing
      √ Updates the owner balance
      √ Updates the contract balance


  9 passing (1s)
```

รูปที่ 25 ตัวอย่างคำสั่งด้วย Hardhat

จากรูปที่ 25 หลังจากเขียนสัญญาอัจฉริยะให้ทำงานร่วมกับ JavaScript สามารถทำการทดสอบฟังก์ชันที่สร้างขึ้น ด้วยเครื่องมือ Hardhat ว่าผ่านไหม

3.4.4 เขียนสคริปต์และใช้คำสั่ง Hardhat เพื่อให้สัญญาเราเข้าไปอยู่ใน private blockchain

```
scripts > JS deploy.js > main
const hre = require("hardhat")
const { items } = require("../src/items.json")

const tokens = (n) => {
  return ethers.utils.parseUnits(n.toString(), 'ether')
}

async function main() {
  // Setup accounts
  const [deployer] = await ethers.getSigners()

  // Deploy Farmer
  const Farmer = await hre.ethers.getContractFactory("Farmer")
  const farmer = await Farmer.deploy()
  await farmer.deployed()

  console.log(`Deployed Farmer Contract at: ${farmer.address}\n`)

  // Listing items...
  for (let i = 0; i < items.length; i++) {
    const transaction = await farmer.connect(deployer).list(
      items[i].id,
      items[i].name,
      items[i].description,
      items[i].category,
      items[i].image,
      tokens(items[i].price),
      items[i].rating,
      items[i].stock,
    )

    await transaction.wait()
```

รูปที่ 26 เขียนสคริปต์เพื่อ deploy

จากรูปที่ 26 ใช้ JavaScript เขียนคำสั่ง เพื่อส่งสัญญาอัจฉริยะ และกำหนดสิ่งที่ต้องการ deploy ไปยังบล็อคเชน

```
C:\Users\axe\farmer>npx hardhat node
Started HTTP and WebSocket JSON-RPC server at http://127.0.0.1:8545/

Accounts
========

WARNING: These accounts, and their private keys, are publicly known.
Any funds sent to them on Mainnet or any other live network WILL BE LOST.

Account #0: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266 (10000 ETH)
Private Key: 0xac0974bec39a17e36ba4a6b4d238ff944bacb478cbed5efcae784d7bf4f2ff80

Account #1: 0x70997970C51812dc3A010C7d01b50e0d17dc79C8 (10000 ETH)
Private Key: 0x59c6995e998f97a5a0044966f0945389dc9e86dae88c7a8412f4603b6b78690d

Account #2: 0x3C44CdDdB6a900fa2b585dd299e03d12FA4293BC (10000 ETH)
Private Key: 0x5de4111afa1a4b94908f83103eb1f1706367c2e68ca870fc3fb9a804cdab365a

Account #3: 0x90F79bf6EB2c4f870365E785982E1f101E93b906 (10000 ETH)
Private Key: 0x7c852118294e51e653712a81e05800f419141751be58f605c371e15141b007a6

Account #4: 0x15d34AAf54267DB7D7c367839AAf71A00a2C6A65 (10000 ETH)
```

รูปที่ 27 ตัวอย่างบัญชีของ Hardhat ที่ใช้ในการทดสอบ

จากรูปที่ 27 เมื่อใช้เน็ตเวิร์กของ Hardhat จะสามารถนำแอคเคาท์ที่มี ETH ไปใช้ในการทดสอบการทำธุรกรรมได้

3.4.5 Deploy สินค้าขึ้น Blockchain โดยใช้คำสั่งของ Hardhat

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

PS C:\Users\axe\farmer> npx hardhat run ./scripts/deploy.js --network localhost
Deployed Farmer Contract at: 0x5FbDB2315678afecb367f032d93F642f64180aa3

Listed item 1: กวางตุ้งดอก ขาวนวล
Listed item 2: กวางตุ้งดอกต้นขาว
Listed item 3: กวางตุ้งดอกต้นเขียว ปิ่นมรกต
Listed item 4: 46-0-0
Listed item 5: 16-20-0
Listed item 6: 15-15-15
Listed item 7: บิวต้า-โปร
Listed item 8: กรมท้า
Listed item 9: ซีซีมิน
```

รูปที่ 28 ตัวอย่างการ deploy สินค้าขึ้น Blockchain

จากรูปที่ 28 เมื่อกรอกรายละเอียดที่ต้องการแล้วทำการ deploy ข้อมูลขึ้นบล็อคเชน โดยใช้คำสั่งของ Hardhat

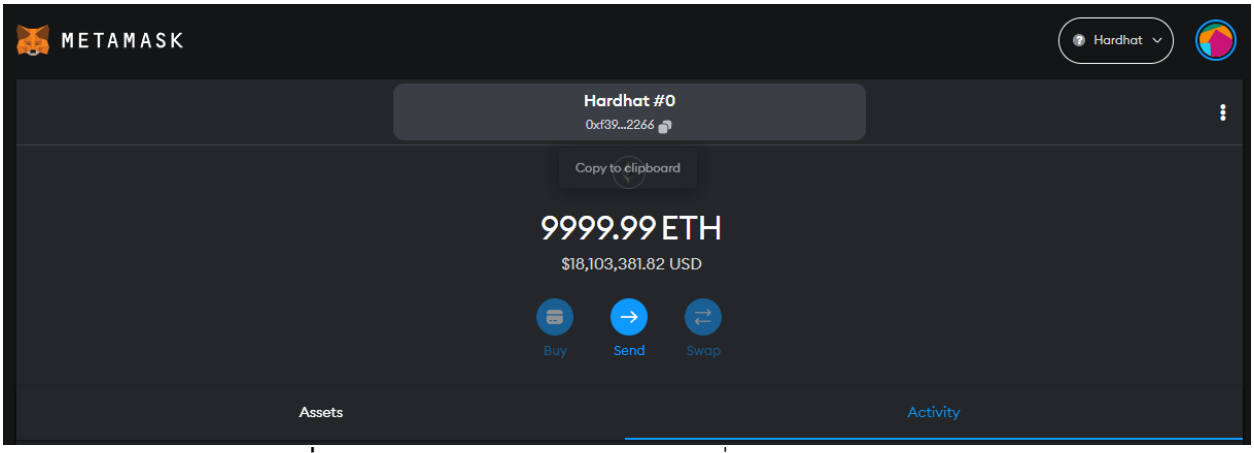
3.4.6 พัฒนาระบบหลังบ้านร่วมกับระบบหน้าบ้าน

```
JS App.js  M ×
src > JS App.js > App
5   import Navigation from './components/Navigation'
6   import Section from './components/Section'
7   import Product from './components/Product'
8   import Titles from './components/Title'
9
10
11
12  // ABIs
13  import Farmer from './abis/Farmer.json'
14
15  // Config
16  import config from './config.json'
17
18  function App() {
19    const [provider, setProvider] = useState(null)
20    const [farmer, setFarmer] = useState(null)
21
22    const [account, setAccount] = useState(null)
23
24    const [seeds, setSeeds] = useState(null)
25    const [fertilizers, setFertilizers] = useState(null)
26    const [herbicides, setHerbicides] = useState(null)
27
28
29    const [item, setItem] = useState({})
30    const [toggle, setToggle] = useState(false)
31
32    const togglePop = (item) => {
33      setItem(item)
34      toggle ? setToggle(false) : setToggle(true)
35    }
36
37    const loadblockchainData = async () => {
38      // Connect to blockchain
39      const provider = new ethers.providers.Web3Provider(window.ethereum)
40      setProvider(provider)
```

รูปที่ 29 ตัวอย่าง Code ที่ใช้ในการพัฒนาระบบหลังบ้านและหน้าบ้าน

จากรูปที่ 29 เป็นการใช้ภาษา React.js เพื่อพัฒนาเว็บไซต์ ในส่วนของหน้าเว็บและในส่วนหลังบ้านร่วมกัน โดยสามารถดึงข้อมูลต่างๆ ที่ทำการ deploy ไปบนบล็อคเชนมาแสดงบนหน้าเว็บไซต์ เพื่อใช้ทำธุรกรรม

3.4.7 หน้าบัญชี MetaMask ที่ Import ด้วย Hardhat



รูปที่ 30 ตัวอย่างหน้าบัญชี MetaMask ที่ Import ด้วย Hardhat

จากรูปที่ 30 ตัวอย่างบัญชี MetaMask ที่ทำการ Import ด้วย Hardhat โดยใช้เน็ตเวิร์กของ Hardhat

3.5 สรุปขั้นตอนการดำเนินงาน

การพัฒนาระบบสัญญาอัจฉริยะด้วยเทคโนโลยีบล็อกเชน เป็นการนำเทคโนโลยีบล็อคเชนมาประยุกต์ใช้เพื่อให้สามารถเพิ่มสินค้า แสดงรายการสินค้า และสามารถซื้อขายสินค้าผ่านสัญญาอัจฉริยะบนเทคโนโลยีบล็อกเชนได้ และนำไปประยุกต์ใช้เพื่อพัฒนาธุรกิจ ซึ่งสามารถสรุปขั้นตอนการพัฒนาได้ดังหัวข้อต่อไปนี้

3.4.1 รวบรวมความต้องการของระบบ จากการศึกษาค้นคว้าข้อมูลเกี่ยวกับระบบสัญญาอัจฉริยะด้วยเทคโนโลยีบล็อกเชน

3.4.2 ศึกษาความเป็นไปได้หลังจากที่รวบรวมความต้องการของระบบเรียบร้อยแล้ว จากนั้นนำข้อมูลที่ได้จากการรวบรวมมาทำการศึกษาความเป็นไปได้ในการทำงานของระบบ

3.4.3 วิเคราะห์ระบบ นำข้อมูลที่ได้จากการรวบรวมและศึกษาความเป็นไปได้มาทำการวิเคราะห์และประยุกต์ใช้กับ ระบบสัญญาอัจฉริยะด้วยเทคโนโลยีบล็อกเชน

3.4.4 ออกแบบและพัฒนาระบบสัญญาอัจฉริยะด้วยเทคโนโลยีบล็อกเชน ให้สามารถใช้งานได้ตรงตามวัตถุประสงค์

บทที่ 4

ผลการดำเนินงานวิจัย

จากการจัดทำโครงงานระบบขายสินค้าเกษตรกรออนไลน์ด้วยสัญญาอัจฉริยะบนเทคโนโลยีบล็อคเชน พบว่าสามารถทำการเพิ่มสินค้า แสดงรายการสินค้าภายในร้านค้า และผู้ซื้อสามารถซื้อขายสินค้าผ่านกระเป๋าเงินดิจิทัลได้ ซึ่งสามารถใช้งานได้ตามวัตถุประสงค์และขอบเขตที่ผู้จัดได้ตั้งเป้าหมายไว้

4.1 ผลการดำเนินงานส่วนผู้ขาย

4.1.1 สร้างสัญญาอัจฉริยะด้วยภาษา Solidity

contracts > Farmer.sol

```
// SPDX-License-Identifier: UNLICENSED
pragma solidity ^0.8.9;

contract Farmer {
    address public owner;

    struct Item {
            uint256 id;
            string  name;
            string  description;
            string  category;
            string  image;
            uint256 cost;
            uint256 rating;
            uint256 stock;
    }

    struct Order {
        uint256 time;
        Item item;
    }

    mapping(uint256 => Item) public items;
    mapping(address => mapping(uint256 => Order)) public orders;
    mapping(address => uint256) public orderCount;

    event Buy(address buyer, uint256 orderId, uint256 itemId);
    event List(string name, uint256 cost, uint256 quantity);

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

    constructor() {
        owner = msg.sender;
    }

```

รูปที่ 31 สร้างสัญญาอัจฉริยะด้วยภาษา Solidity

จากรูปที่ 31 เป็นการสร้างสัญญาอัจฉริยะเพื่อกำหนดเงื่อนไขในการซื้อขายสินค้า โดยกำหนดรูปแบบของสินค้า กำหนดเงื่อนไขในการทำธุรกรรม การเพิ่มข้อมูลสินค้า และการขายสินค้า

4.1.2 เขียนสคริปต์เพื่อให้สัญญาอัจฉริยะทำงานร่วมกันได้

```
17  describe("Farmer", () => {
18    let farmer
19    let deployer, buyer
20
21    beforeEach(async () =>{
22      // Setup accounts
23      [deployer, buyer] = await ethers.getSigners()
24
25      // Deploy contact
26      const Farmer = await ethers.getContractFactory("Farmer")
27      farmer = await Farmer.deploy()
28    })
29
30    describe("Deplyment", () => {
31      it("Sets the owner", async () => {
32        expect(await farmer.owner()).to.equal(deployer.address)
33      })
34    })
35
36    describe("Listing", () => {
37      let transaction
38
39      beforeEach(async () =>{
40        transaction = await farmer.connect(deployer).list(
41          ID,
42          NAME,
43          DESCRIPTION,
44          CATEGORY,
45          IMAGE,
46          COST,
47          RATING,
48          STOCK
49        )
50
51        await transaction.wait()
52      })
53
```

รูปที่ 32 เขียนสคริปต์เพื่อให้สัญญาอัจฉริยะทำงานร่วมกันได้

จากรูปที่ 32 เป็นการเขียนสคิปต์เพื่อดึงสัญญาอัจฉริยะที่เราได้สร้างขึ้นมาทำงานร่วมกับเว็บไซต์ของเรา

4.1.3 ทดสอบสัญญาอัจฉริยะ

```
D:\Program\farmer>npx hardhat test


  Farmer
    Deplyment
      √ Sets the owner (43ms)
    Listing
      √ Returns item attributes
      √ Emits List event
    Buying
      √ Updates buyer's order count
      √ Adds the order
      √ Updates the contract balance
      √ Emits Buy event
    Withdrawing
      √ Updates the owner balance
      √ Updates the contract balance


  9 passing (1s)
```

รูปที่ 33 ทดสอบสัญญาอัจฉริยะ

จากรูปที่ 33 เป็นการทดสอบสัญญาอัจฉริยะที่ผู้ขายได้สร้างขึ้น เพื่อตรวจสอบความผิดพลาดในการเขียนสัญญาอัจฉริยะ

4.1.4 กำหนดรายละเอียดของสินค้าที่ต้องการทำธุรกรรม

items.json M

src > items.json > [ ]items > {} 2


```
{
  "items": [
    {
      "id": 1,
      "name": "กวางตุ้งดอก ขาวนวล",
      "description" : "เมล็ดพันธุ์ กวางตุ้งดอก ขาวนวล น้ำหนัก 10 กรัม เมล็ดพันธุ์เจียไต๋ เมล็ดพันธุ์คุณภาพ ปลูกง่าย โตไว มีให้เลือกกว่า 300 สายพันธุ์ พร้อมจัดส่งถึงมือคุณ",
      "category": "seeds",
      "image": "https://media.discordapp.net/attachments/1090617991143641088/1090618119531278346/01.jpg",
      "price": "0.3",
      "rating": 5,
      "stock": 28
    },

{
      "id": 2,
      "name": "กวางตุ้งดอกต้นขาว",
      "description" : "เมล็ดพันธุ์ กวางตุ้งดอกต้นขาว น้ำหนัก 50 กรัม จำนวน 26000 เมล็ด เมล็ดพันธุ์เจียไต๋ เมล็ดพันธุ์คุณภาพ ปลูกง่าย โตไว มีให้เลือกกว่า 300 สายพันธุ์ พร้อมจัดส่งถึงมือคุ
      "category": "seeds",
      "image": "https://media.discordapp.net/attachments/1090617991143641088/1090618119812288603/02.png",
      "price": "0.6",
      "rating": 4,
      "stock": 46
    },

{
      "id": 3,
      "name": "กวางตุ้งดอกต้นเขียว ปินมรกต",
      "description" : "เมล็ดพันธุ์ กวางตุ้งดอกต้นเขียว ปินมรกต น้ำหนัก 10 กรัม เมล็ดพันธุ์เจียไต๋ เมล็ดพันธุ์คุณภาพ ปลูกง่าย โตไว มีให้เลือกกว่า 300 สายพันธุ์ พร้อมจัดส่งถึงมือคุณ",
      "category": "seeds",
      "image": "https://media.discordapp.net/attachments/1090617991143641088/1090618120214949969/03.jpg",
      "price": "0.1",
      "rating": 4,
      "stock": 32
    },
```


รูปที่ 34 กำหนดรายละเอียดสินค้า

จากรูปที่ 34 กำหนดรายละเอียดสินค้าที่ไฟล์ items.json เช่น ชื่อสินค้า ประเภทสินค้า รายละเอียดสินค้า รวมไปถึงรูปของสินค้า

4.1.5 สร้างสคริปต์ในการ deploy สินค้าขึ้นไปยัง blockchain

```
scripts > JS deploy.js > ...
const hre = require("hardhat")
const { items } = require("../src/items.json")

const tokens = (n) => {
  return ethers.utils.parseUnits(n.toString(), 'ether')
}

async function main() {
  // Setup accounts
  const [deployer] = await ethers.getSigners()

  // Deploy Farmer
  const Farmer = await hre.ethers.getContractFactory("Farmer")
  const farmer = await Farmer.deploy()
  await farmer.deployed()

  console.log(`Deployed Farmer Contract at: ${farmer.address}\n`)

  // Listing items...
  for (let i = 0; i < items.length; i++) {
    const transaction = await farmer.connect(deployer).list(
      items[i].id,
      items[i].name,
      items[i].description,
      items[i].category,
      items[i].image,
      tokens(items[i].price),
      items[i].rating,
      items[i].stock,
    )

    await transaction.wait()

    console.log(`Listed item ${items[i].id}: ${items[i].name}`)
  }
}
```

รูปที่ 35 สร้างสคริปต์ในการ deploy สินค้าขึ้นไปยัง blockchain

จากรูปที่ 35 เป็นการสร้างสคริปต์ในการ deploy สินค้าขึ้นไปยัง blockchain โดยกำหนดให้ไปดึงรายละเอียดสินจากไฟล์ items.json เมื่อทำการ deploy ขึ้น blockchain แล้วจะไม่สามารถแก้ไขได้ แต่ในเน็ตเวิร์คที่ใช้ในการทดสอบสามารถแก้ไขได้ด้วยการ restart คอมพิวเตอร์

4.1.6 deploy สินค้าขึ้นไปยัง blockchain

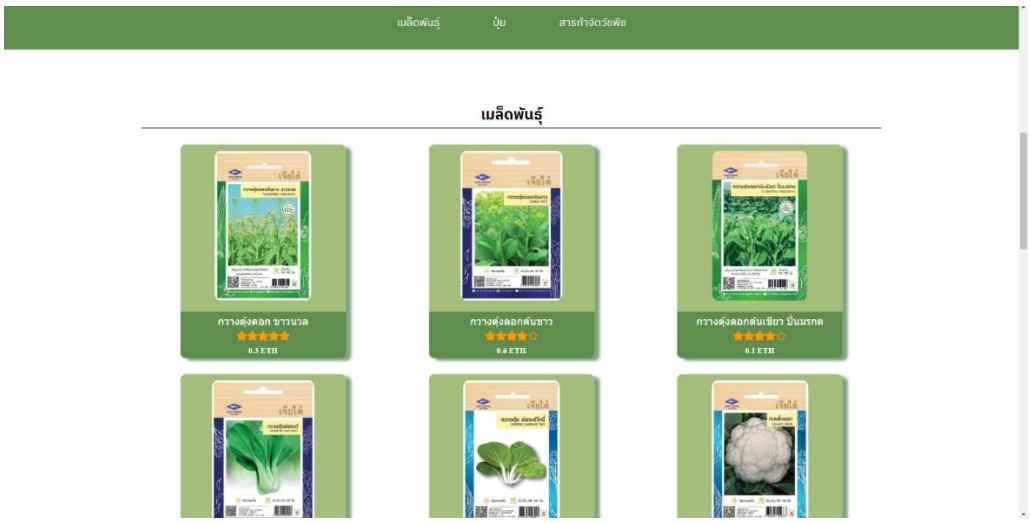
```
PS D:\Program\farmer> npx hardhat run ./scripts/deploy.js --network localhost
Deployed Farmer Contract at: 0x5FbDB2315678afecb367f032d93F642f64180aa3

Listed item 1: กวางตุ้งดอก ขาวนวล
Listed item 2: กวางตุ้งดอกต้นขาว
Listed item 3: กวางตุ้งดอกต้นเขียว บึ้นมรกต
Listed item 4: 46-0-0
Listed item 5: 16-20-0
Listed item 6: 15-15-15
Listed item 7: บิวต้า-โปร
Listed item 8: กรุมก้า
Listed item 9: ซีซี่มีน
Listed item 10: กวางตุ้งฮ่องเต้
Listed item 11: กวางตุ้งฮ่องเต้ไทนี่
Listed item 12: กะหล่ำดอก
Listed item 13: 46-0-0
```

รูปที่ 36 deploy สินค้าขึ้นไปยัง blockchain

จากรูปที่ 36 เป็นการ deploy สินค้าที่ผู้ขายต้องการขาย ขึ้นไปยัง blockchain โดยใช้คำสั่งของ Hardhat ในการ deploy ในการ deploy จะเสียค่าธรรมเนียมทุกครั้งและจะไม่สามารถแก้ไขได้ แต่ในเน็ตเวิร์คที่ใช้ในการทดสอบสามารถแก้ไขได้ด้วยการ restart คอมพิวเตอร์

4.1.7 พัฒนาเว็บไซต์

```
src > JS App.js > ...
import { useEffect, useState } from 'react'
import { ethers } from 'ethers'


import Navigation from './components/Navigation'
import Section from './components/Section'
import Product from './components/Product'
import Titles from './components/Title'

import Farmer from './abis/Farmer.json'

import config from './config.json'

function App() {
  const [provider, setProvider] = useState(null)
  const [farmer, setFarmer] = useState(null)

  const [account, setAccount] = useState(null)

  const [seeds, setSeeds] = useState(null)
  const [fertilizers, setFertilizers] = useState(null)
  const [herbicides, setHerbicides] = useState(null)


  const [item, setItem] = useState({})
  const [toggle, setToggle] = useState(false)

  const togglePop = (item) => {
    setItem(item)
    toggle ? setToggle(false) : setToggle(true)
  }

  const loadblockchainData = async () => {
    // Connect to blockchain
    const provider = new ethers.providers.Web3Provider(window.ethereum)
    setProvider(provider)

    const network = await provider.getNetwork()
    console.log(network)
```



รูปที่ 37 การพัฒนาเว็บไซต์

จากรูปที่ 37 เป็นการเขียนภาษาที่ใช้ในการพัฒนาเว็บไซต์เพื่อแสดงสัญญาอัจฉริยะที่ผู้ขายได้ทำการสร้างขึ้นแล้ว deploy ขึ้นไปยังบล็อคเชน มาแสดงในรูปแบบของเว็บไซต์ โดยกำหนดให้แสดงสิ่งต่างๆ ที่ผู้ขายต้องการทำธุรกรรม

4.2 ผลการดำเนินงานส่วนผู้ซื้อ

4.2.1 เข้าสู่เว็บไซต์ที่สร้างขึ้น

รูปที่ 38 หน้าตาเว็บไซต์ร้านค้าเกษตรกรออนไลน์บนระบบสัญญาอัจฉริยะ

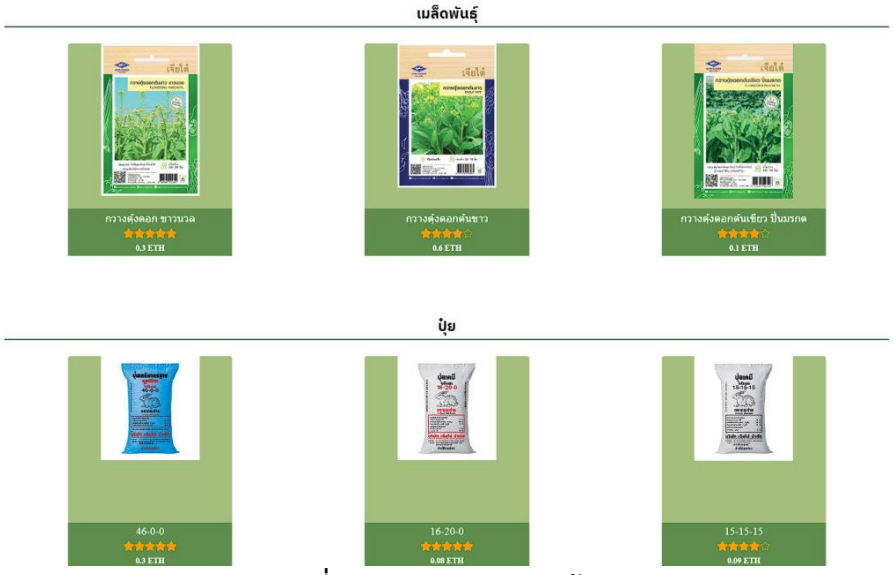
จากรูปที่ 38 เป็นหน้าเว็บไซต์ร้านค้าที่พัฒนาร่วมกับสัญญาอัจฉริยะ เพื่อให้ลูกค้าสามารถเข้ามาทำการซื้อขายสินค้าโดยใช้สกุลเงินดิจิทัลได้

4.2.2 เชื่อมต่อกับกระเป๋า MetaMask

รูปที่ 39 ก่อน-หลังการเชื่อมต่อกระเป๋า MetaMask

จากรูปที่ 39 เป็นการเชื่อมบัญชี MetaMask ของผู้ที่ต้องการทำธุรกรรมเข้ากับเว็บไซต์ หลงเชื่อมกับ MetaMask แล้วจะขึ้นแสดงเป็นหมสยเลข Address ของกระเป๋าตนเอง

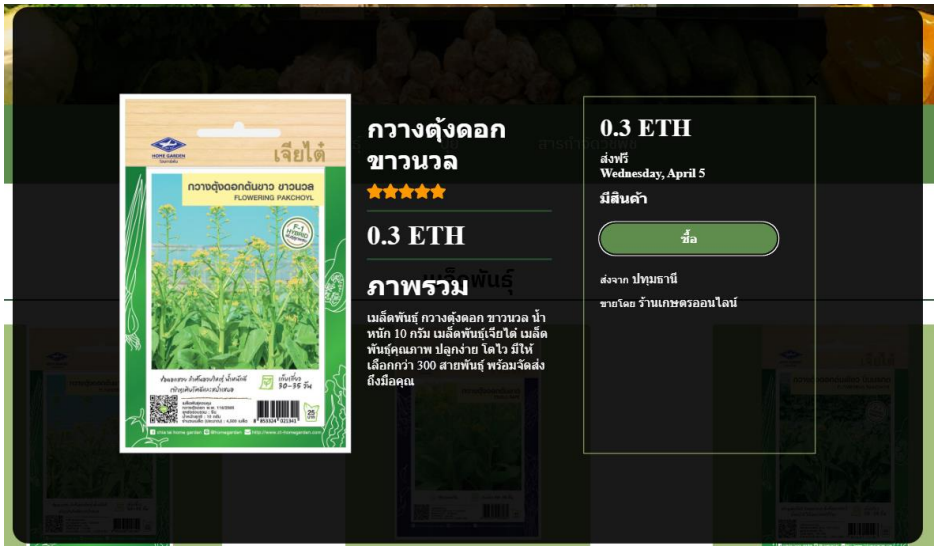
4.2.3 เลือกดูรายการแสดงสินค้า



รูปที่ 40 แสดงรายการสินค้า

จากรูปที่ 40 เป็นสินค้าประเภทเกษตรที่ผู้ขายทำการเขียนลงในสัญญาอัจฉริยะ เพื่อให้ผู้ที่ต้องการมาซื้อสินค้าประเภทเกษตกรได้เลือกซื้อสิ่งที่ต้องการ

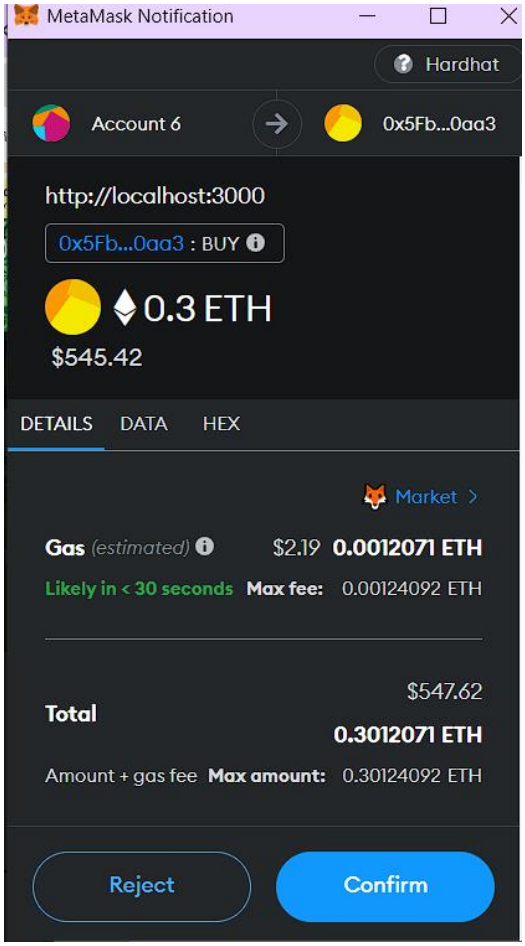
4.2.4 กดเลือกสินค้าที่ต้องการ



รูปที่ 41 หน้าต่างแสดงข้อมูลสินค้า

จากรูปที่ 41 ลูกค้าสามารถกดที่ตัวสินค้า เพื่อดูข้อมูลต่างๆ ของสินค้า เช่น ชื่อ ราคา ชนิดของสินค้า เป็นต้น หลังจากนั้นกดปุ่ม “ซื้อ” เพื่อทำธุรกรรม

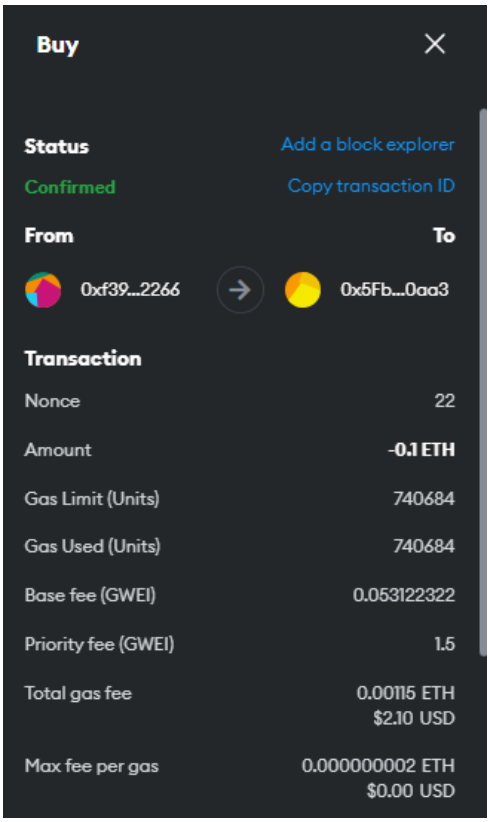
4.2.5 กด Confirm เพื่อยืนยันการซื้อสินค้า



รูปที่ 42 หน้าต่างการชำระเงินค่าสินค้า

จากรูปที่ 42 เมื่อลูกค้ากดที่ปุ่ม “ซื้อ” บัญชี MetaMask ก็จะเด้งขึ้นมาเพื่อให้ลูกค้ายอมรับ เมื่อกด “Confirm” ก็จะเป็นการตกลงในการทำธุรกรรม

4.2.6 สามารถกดดูประวัติการชำระเงินค่าสินค้า



รูปที่ 43 หน้าต่างประวัติการชำระเงินค่าสินค้า

จากรูปที่ 43 เมื่อลูกค้ากดที่ปุ่ม “Confirm” ก็จะเป็นการตกลงในการทำธุรกรรมและจะมีประวัติการโอนเงินไปยังผู้ขายเป็นการเสร็จสิ้นธุรกรรมระหว่างผู้ซื้อและผู้ขาย

บทที่ 5

สรุปผลการวิจัยและข้อเสนอแนะ

ในบทนี้จะกล่าวถึงบทสรุปและข้อเสนอแนะของระบบขายสินค้าเกษตรกรออนไลน์ด้วยสัญญาอัจฉริยะบนเทคโนโลยีบล็อคเชน เพื่อเป็นพื้นฐานที่จะนำไปสู่การพัฒนาต่อไป

5.1 สรุปผลการดำเนินการ

5.2 ปัญหาที่พบในการศึกษา

5.3 ข้อเสนอแนะ

5.1 สรุปผลการดำเนินการ

จากการศึกษาค้นคว้าการเขียนระบบขายสินค้าเกษตรกรออนไลน์ด้วยสัญญาอัจฉริยะบนเทคโนโลยีบล็อคเชน และได้ทำการทดลองตัวอย่าง โดยการซื้อสินค้าผ่านเว็บไซต์ ที่เชื่อมกับสัญญาอัจฉริยะและกระเป๋าเงินดิจิทัล

สรุปผลการทดลองระบบขายสินค้าเกษตรกรออนไลน์ด้วยสัญญาอัจฉริยะบนเทคโนโลยีบล็อคเชน ได้ว่า จากการทดลองตัวระบบร้านขายสินค้าเกษตรออนไลน์ เพียงผู้ซื้อมีกระเป๋า MetaMask ก็สามารถใช้งานผ่าน Browser ใดก็ได้ โดยเข้าใช้งานเว็บไซต์ที่สร้างขึ้น โดยเว็บไซต์จะทำงานร่วมกับสัญญาอัจฉริยะ ที่เขียนโดยภาษา Solidity ถ้าผู้เข้าใช้งานทำตามเงื่อนไขที่เขียนบนสัญญาอัจฉริยะ หรือซื้อสินค้าที่ตรงกับตัวเองต้องการ Wallet ก็จะถูกโอนเข้า MetaMask ของผู้ขายทันที โดยที่ไม่ต้องผ่านคนกลาง ทำให้ง่ายและสะดวกในการซื้อขายสินค้า

5.2 ปัญหาที่พบในการศึกษา

5.2.1 ภาษาที่ใช้งานอาจจะยังไม่มีการเผยแพร่มากนักในประเทศ

5.2.2 การ deploy ขึ้น Network ต้อง Restart เครื่องทุกครั้ง

5.2.3 ไม่มีสกุลเงินดิจิทัลในการทดสอบบนเว็บไซต์จริง

5.2.4 Framework ที่สามารถใช้งานได้บางตัวซับซ้อนเกินไป

5.2.5 ราคาเหรียญ ETH และราคาสินค้ามีการเปลี่ยนแปลงตลอดเวลา

5.3 ข้อเสนอแนะ

5.3.1 ศึกษาจากหลายๆ แหล่ง เช่นจากต่างประเทศ

5.3.2 ตรวจข้อมูลก่อน Deploy ให้แน่ใจทุกครั้ง

5.3.3 เลือกใช้ Network ที่ให้ทดลองใช้สกุลเงินดิจิทัลในการทดลอบ

5.3.4 พยายาทำความเข้าใจกับ Framework ที่ต้องใช้งานให้มากที่สุด

5.3.5 ควรทำราคากลางของสินค้า และกำหนดวันอัพเดตราคาสินค้าให้ชัดเจน

บรรณานุกรม

[1] “ความรู้เกี่ยวกับระบบ Blockchain,” [ออนไลน์] เข้าถึงได้จาก:https://www.depa.or.th/th/article-view/blockchain-work?fbclid=IwAR1NuFEUI_anKlSR_1oNsPmTymrU-3Tbl4M41wl_CgmB37oYGdNpDX3eOAo [สืบค้นเมื่อ 28 มีนาคม 2566]

[2] “ความรู้เกี่ยวกับสกุลเงิน Ethereum,” [ออนไลน์] เข้าถึงได้จาก: https://www.bitkub.com/th/blog/what-is-ethereum-1aac4af8bba?fbclid=IwAR1jHftEJd_q8b74smFrUtlxNdm1irf1GD8C91NJ2iHpvdbyPgsXdBjeMOE [สืบค้นเมื่อ 28 มีนาคม 2566]

[3] “ความรู้เกี่ยวกับหลักการของ Smart Contract,” [ออนไลน์] เข้าถึงได้จาก: https://thematter.co/futureverse/futureword-smart-contract/160767?fbclid=IwAR26_U2HHxqBVKfOLkqslodDrb6tqztKYgiu6cKQtzLFY0JIGKVf5Mans_w [สืบค้นเมื่อ 28 มีนาคม 2566]

[4] “ความรู้เกี่ยวกับภาษา Solidity,” [ออนไลน์] เข้าถึงได้จาก: https://medium.com/20scoops-cnx/%E0%B8%A1%E0%B8%B2%E0%B8%A3%E0%B8%B9%E0%B9%89%E0%B8%88%E0%B8%B1%E0%B8%81%E0%B8%81%E0%B8%B1%E0%B8%9A-solidity-%E0%B8%82%E0%B8%B1%E0%B9%89%E0%B8%99%E0%B8%9E%E0%B8%B7%E0%B9%89%E0%B8%99%E0%B8%90%E0%B8%B2%E0%B8%99%E0%B8%81%E0%B8%B1%E0%B8%99-6f713b3fb64 [สืบค้นเมื่อ 28 มีนาคม 2566]

[5] “โปรแกรม Visual Studio Code,” [ออนไลน์] เข้าถึงได้จาก: https://learn.microsoft.com/th-th/power-apps/maker/portals/vs-code-extension?fbclid=IwAR3kKmNwR3MsCryTvOjOF1mvgAYkm1wFJI4l3v697bbNDWb4KJeN60bId_g [สืบค้นเมื่อ 28 มีนาคม 2566]

[6] “ความรู้เกี่ยวกับ Hardhat,” [ออนไลน์] เข้าถึงได้จาก: https://moonbeam-thailand.medium.com/%E0%B8%81%E0%B8%B2%E0%B8%A3%E0%B8%AA%E0%B8%A3%E0%B

9%89%E0%B8%B2%E0%B8%87-hardhat-%E0%B8%9A%E0%B8%99-moonbeam-4beef2744c3f [สืบค้นเมื่อ 28 มีนาคม 2566]

[7] “แอพพลิเคชั่น MetaMask,” [ออนไลน์] เข้าถึงได้จาก: https://www.ananda.co.th/blog/thegenc/MetaMask/?fbclid=IwAR3hOHSjHFMnV4x-YnLqxPMVK6Cz16reHxeMwWECZ_ND3D6L6OmZLy7wDuE [สืบค้นเมื่อ 30 มีนาคม 2566]

[8] “แอพพลิเคชั่น DApps,” [ออนไลน์] เข้าถึงได้จาก: https://www.mindphp.com/%E0%B8%84%E0%B8%B9%E0%B9%88%E0%B8%A1%E0%B8%B7%E0%B8%AD/73-%E0%B8%84%E0%B8%B7%E0%B8%AD%E0%B8%AD%E0%B8%B0%E0%B9%84%E0%B8%A3/8797-dapps.html?fbclid=IwAR2-IiLcwWQI7HdH34CA4Q-rLqGfoj5rZQntEx_RnIDU3i7u-i7vJfxONgQ [สืบค้นเมื่อ 30 มีนาคม 2566]

[9] “ความรู้เกี่ยวกับ Remix - Ethereum IDE,” [ออนไลน์] เข้าถึงได้จาก: https://www.poolsawat.com/solidity-101-smartcontract-on-remix/ [สืบค้นเมื่อ 30 มีนาคม 2566]

[10] “ความรู้เกี่ยวกับ Etherscan.io,” [ออนไลน์] เข้าถึงได้จาก: https://www.siambitcoin.com/2019/12/04/%e0%b9%84%e0%b8%9f%e0%b8%a3%e0%b9%8c%e0%b8%a7%e0%b8%ad%e0%b8%a5%e0%b8%a5%e0%b9%8c%e0%b8%ad%e0%b8%b4%e0%b8%99%e0%b9%80%e0%b8%97%e0%b8%ad%e0%b8%a3%e0%b9%8c%e0%b9%80%e0%b8%99%e0%b9%87%e0%b8%95%e0%b8%82/ [สืบค้นเมื่อ 30 มีนาคม 2566]

[11] “ความรู้เกี่ยวกับภาษา HTML,” [ออนไลน์] เข้าถึงได้จาก: https://www.mindphp.com/%E0%B8%84%E0%B8%B9%E0%B9%88%E0%B8%A1%E0%B8%B7%E0%B8%AD/73-%E0%B8%84%E0%B8%B7%E0%B8%AD%E0%B8%AD%E0%B8%B0%E0%B9%84%E0%B8%A3/2026-html-%E0%B8%84%E0%B8%B7%E0%B8%AD%E0%B8%AD%E0%B8%B0%E0%B9%84%E0%B8%A3.html?fbclid=IwAR3zqw9TQq7zSt0ToQksoNY9_TVhCY--XuCfIZ5DHs9ho3j0B18Xheaku2U [สืบค้นเมื่อ 30 มีนาคม 2566]

[12] “ความรู้เกี่ยวกับภาษา CSS,” [ออนไลน์] เข้าถึงได้จาก: http://www.kontentblue.com/site/article/article.php?id=css-introduction&fbclid=IwAR0emzw2tr-_0OvaBD9fLFsWRarL0ZMLJB8Uzt35BDn5-IgrcCd5xvX5YK8 [สืบค้นเมื่อ 30 มีนาคม 2566]

[13] “ความรู้เกี่ยวกับภาษา JavaScript,” [ออนไลน์] เข้าถึงได้จาก: https://aws.amazon.com/th/what-is/javascript/?fbclid=IwAR2-IiLcwWQI7HdH34CA4Q-rLqGfoj5rZQntEx_RnIDU3i7u-i7vJfxONgQ [สืบค้นเมื่อ 30 มีนาคม 2566]

[14] “ความรู้เกี่ยวกับหลักการของ React.js,” [ออนไลน์] เข้าถึงได้จาก: https://www.thaiprogrammer.org/2018/01/learn-react-in-5-mins/?fbclid=IwAR2jpH--6gLuM2o3Spl54v5tLQ2vj48QXrDM4EpVqlpujOXbgR-bI2fAmq0 [สืบค้นเมื่อ 30 มีนาคม 2566]

[15] “ความรู้เกี่ยวกับหลักการของ Node.js,” [ออนไลน์] เข้าถึงได้จาก: https://devahoy.com/blog/2015/07/getting-started-with-nodejs?fbclid=IwAR1CMPTWO7Gg5_DOowYECI3Si0Lfs_fpMWBfTaE79sGkk4kcFSbcfjPeN74 [สืบค้นเมื่อ 30 มีนาคม 2566]

[16] “ความรู้เกี่ยวกับหลักการของ Web3.js,” [ออนไลน์] เข้าถึงได้จาก: https://poolsawat-com.medium.com/%E0%B9%80%E0%B8%82%E0%B8%B5%E0%B8%A2%E0%B8%99%E0%B9%82%E0%B8%84%E0%B9%8A%E0%B8%94-javascript-web3-js-c1dc1fb20493 [สืบค้นเมื่อ 30 มีนาคม 2566]

[17] นันทา จันทร์พิทักษ์ และรัฐพล มณฑา. “การพัฒนาระบบบริหารสหกรณ์โดยใช้เทคโนโลยีบล็อกเชน,” [ออนไลน์] เข้าถึงได้จาก: https://ph02.tci-thaijo.org/index.php/JIST/article/download/211214/158110/782559 [สืบค้นเมื่อ 30 มีนาคม 2566]

[18] ธัญวรัตน์ ธนัชญ์นันท์. “ปัจจัยที่มีผลต่อการตัดสินใจใช้เทคโนโลยีบล็อกเชน,” [ออนไลน์] เข้าถึงได้จาก: https://ciba.dpu.ac.th/wp-content/plugins/ciba-article/article/35.pdf?fbclid=IwAR2iC8KfY6tEMnZzQT6RUpc8cXEs7E5g1LrqqI5TdBecEvssBcUrzTHjc34 [สืบค้นเมื่อ 30 มีนาคม 2566]

[19] สรวิศ ลิมปรังษี. “Smart Contract สัญญาอัจฉริยะ ความฝันและความจริง,” [ออนไลน์] เข้าถึงได้จาก:

https://appealsc.coj.go.th/th/content/category/detail/id/8/cid/10851/iid/208350?fbclid=IwAR0EKwITaaz07wGMS-suK3eYR91dI3KSkDwmuIWn06WYB5dlknLZ4EI43RY [สืบค้นเมื่อ 30 มีนาคม 2566]

[20] อัษฎางค์ ขำมะโน และ เอกรัฐ รัฐกาญจน์. “แอปพลิเคชันล็อตเตอรีแบบกระจายศูนย์โดยใช้เทคโนโลยีบล็อกเชน,” [ออนไลน์] เข้าถึงได้จาก: https://as.nida.ac.th/wp-content/uploads/2021/11/6210412028_Adsadang.pdf?fbclid=IwAR0FY5tBn1a3NPRtOdL-Mm5pU2utSFbJDNj8kMeo8YDroUQhOrKUwc51jJ8 [สืบค้นเมื่อ 30 มีนาคม 2566]

[21] วรวุฒิ บัวทองจันทร์. “ต้นแบบการอนุมัติการจัดการระบบบริการสหกรณ์ร้านค้าดิจิทัลในประเทศไทยที่มีความน่าเชื่อถือด้วยการประยุกต์ใช้บล็อกเชน,” [ออนไลน์] เข้าถึงได้จาก: http://dspace.spu.ac.th/handle/123456789/5736?fbclid=IwAR1Rjih3rZ3jAKNkS2RXyIfj4cXZt3RdwXfBEWgLVW1f6OboyWUXhhGdl4E [สืบค้นเมื่อ 30 มีนาคม 2566]

ภาคผนวก

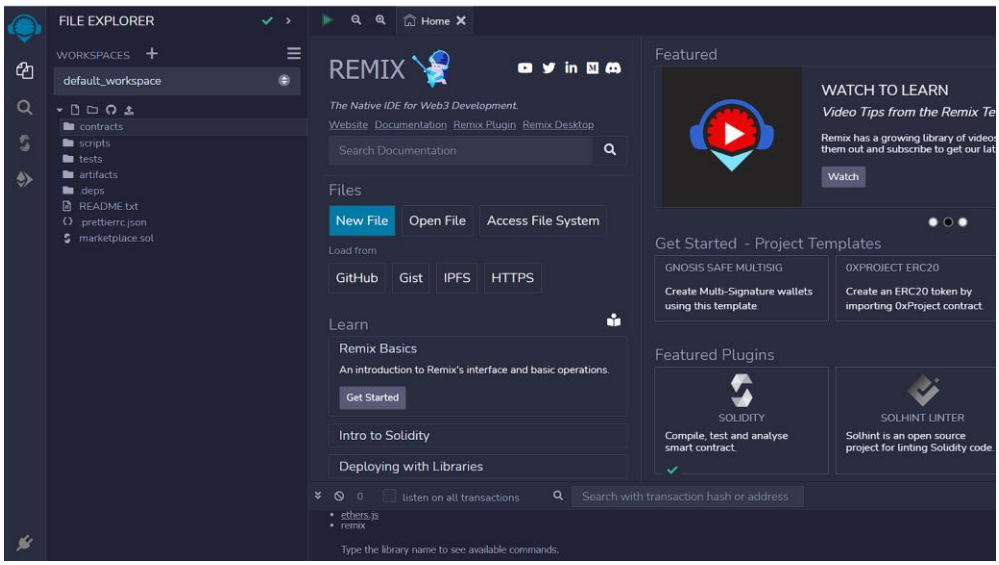
ภาคผนวก ก

ขั้นตอนการพัฒนาสัญญาอัจฉริยะ ด้วยภาษา Solidity

ขั้นตอนการพัฒนาสัญญาอัจฉริยะ ด้วยภาษา Solidity

1. เข้าเว็บ Browser ไปที่ Website Remix IDE

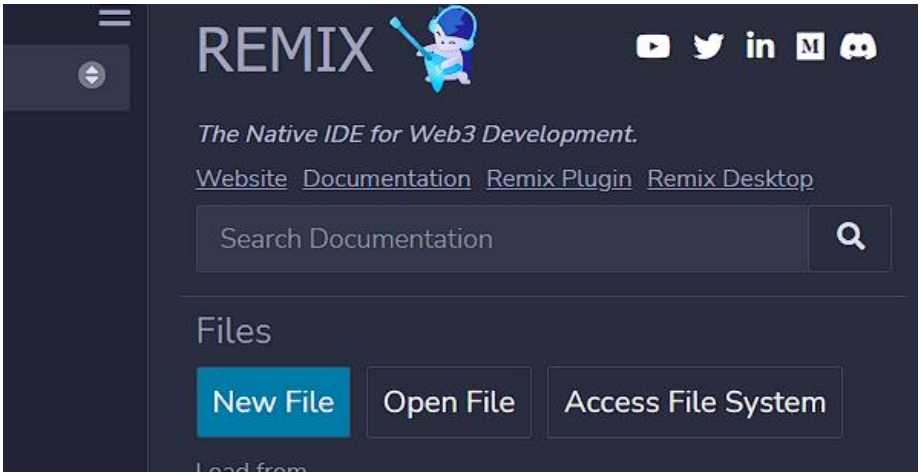
https://remix.ethereum.org/#lang=en&optimize=false&runs=200&evmVersion=null



รูปที่ ก-1 หน้าตาของ Remix IDE

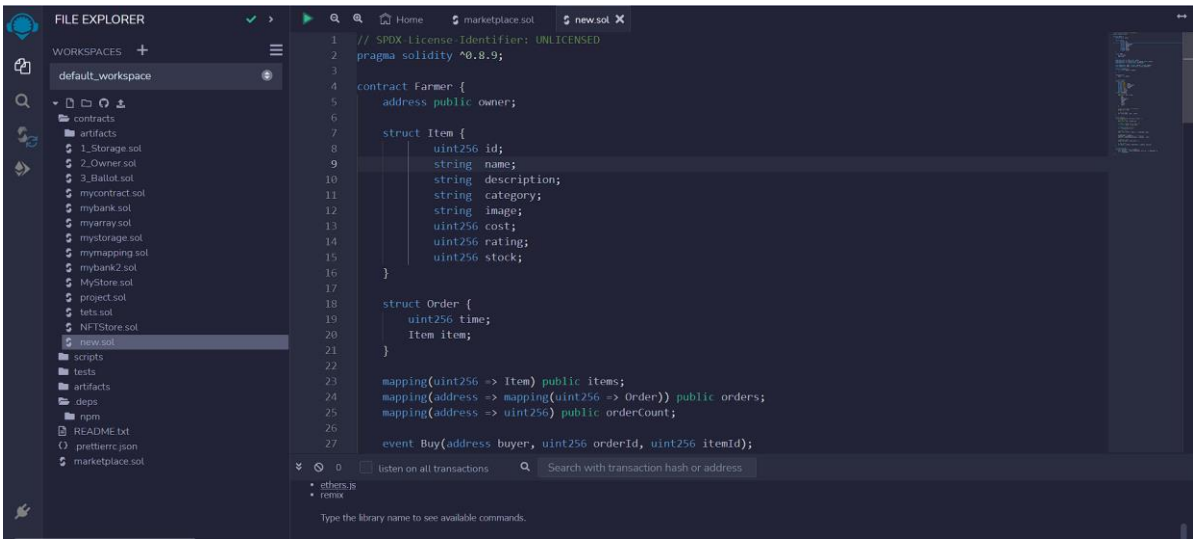
จากรูปที่ ก-1 พัฒนาสัญญาอัจฉริยะด้วยภาษา Solidity ผ่านเว็บไซต์ที่ชื่อว่า Remix IDE

2. คลิกที่ New File เพื่อสร้างไฟล์ใหม่



รูปที่ ก-2 การสร้างไฟล์ใหม่

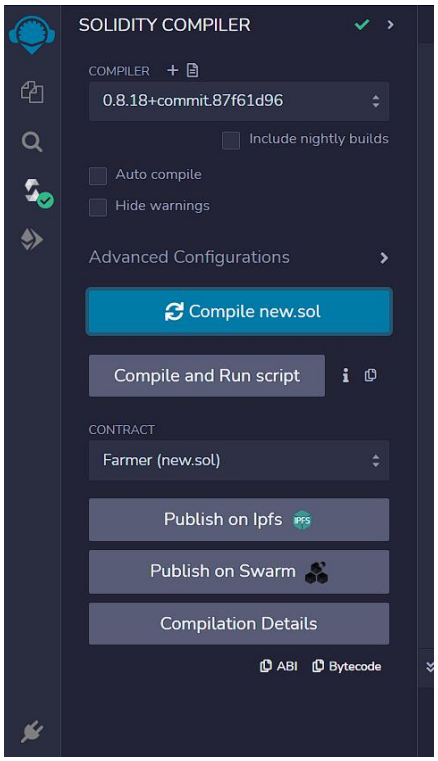
3. พัฒนาระบบร้านค้าด้วยภาษา Solidity



รูปที่ ก-3 ตัวอย่างภาษา Solidity

จากรูปที่ ก-3 ทำการพัฒนาสัญญาอัจฉริยะด้วยภาษา Solidity โดยใช้นามสกุลไฟล์เป็น .sol

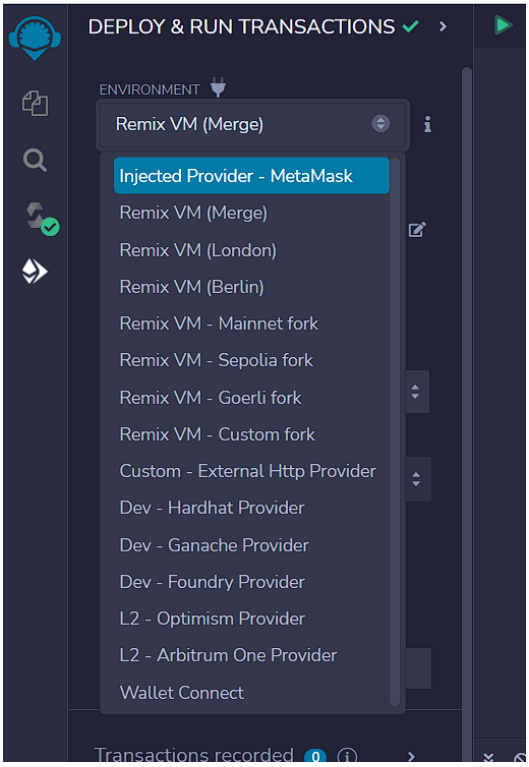
4. กด Compile เพื่อ Run Code (ถ้าขึ้นสีเขียวแสดงว่า Code ผ่าน)



รูปที่ ก-4 การ Run code

จากรูปที่ ก-4 ทำการ Compile ตัว Code เพื่อเช็คว่ารันผ่านหรือไม่ หรือ Error ตรงไหน ถ้าขึ้นสีเขียว แสดงว่า Code รันผ่าน

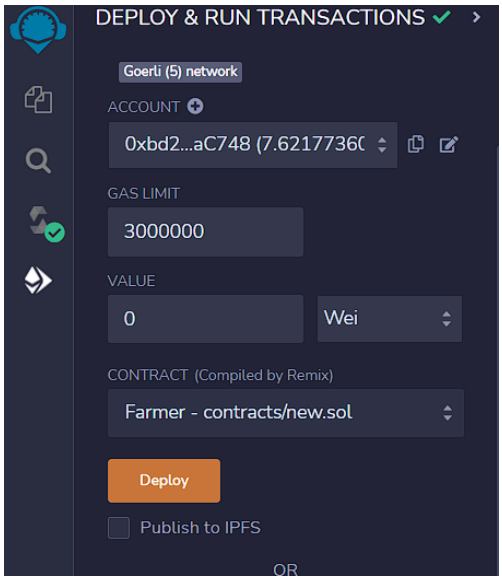
5. เปลี่ยน Environment เป็น Injected Provider – MetaMask เพื่อเชื่อมกับกระเป๋า MetaMask



รูปที่ ก-5 การเปลี่ยน Environment

จากรูปที่ ก-5 หลังจากรันผ่านแล้ว ให้ลงมาที่ฟังก์ชัน Deploy กดเปลี่ยนเป็น Injected Provider - MetaMask เพื่อให้สัญญาอัจฉริยะเชื่อมกับ MetaMask ของเรา

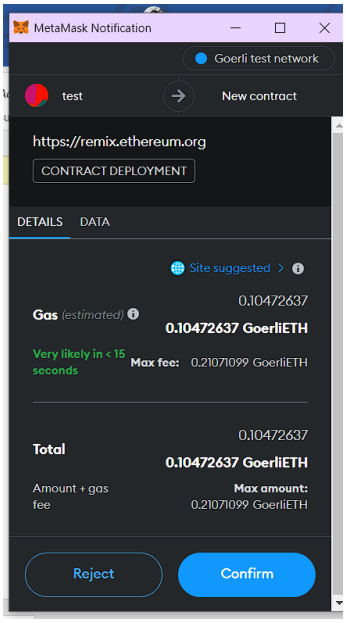
6. กดปุ่ม Deploy เพื่อ deploy code



รูปที่ ก-6 การ deploy code

จากรูปที่ ก-6 เมื่อเชื่อมกับกระเป๋า MetaMask แล้ว ทำการกด Deploy เพื่อ deploy code ให้กลายเป็น Contracts

7. เมื่อกด Deploy จะเด้งหน้า MetaMask เพื่อชำระค่าธรรมเนียม ให้กด Confirm



รูปที่ ก-7 หน้าต่างการชำระค่าธรรมเนียม

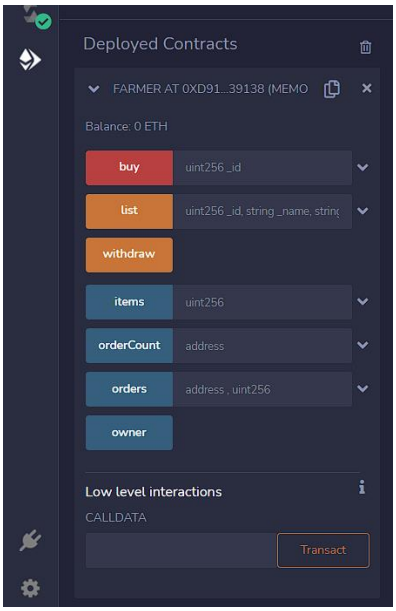
จากรูปที่ ก-7 หลังจากกด Deploy จะขึ้นหน้าต่าง MetaMask ให้ชำระค่าธรรมเนียมในการ Deploy แต่ละครั้ง

8. เมื่อ deploy ผ่าน จะมีสัญญาอัจฉริยะขึ้นมา ซึ่งไม่สามารถแก้ไขข้อมูลภายในได้ หากต้องการแก้ไขข้อมูล ต้องทำการ deploy ใหม่อีกครั้ง

รูปที่ ก-8 ตัวอย่าง Smart contract

จากรูปที่ ก-8 เมื่อ กด Deploy และชำระค่าธรรมเนียมเรียบร้อย จะขึ้น Contracts ของเรา และนำเลข Address ของ Contracts ไปบนเน็ตเวิร์กเพื่อทำให้ข้อมูลโปร่งใส และไม่สามารถแก้ไขข้อมูลภายในได้ ถ้าต้องการแก้ไขข้อมูลต้องทำการ deploy ใหม่เท่านั้น

9. เมื่อ deploy แล้วสามารถกดดูฟังก์ชันต่างๆ ที่เราเขียนขึ้นได้ และลองใช้ได้



รูปที่ ก-9 ฟังก์ชันที่สร้างไว้ใน Smart contract

จากรูปที่ ก-9 สามารถดูฟังก์ชันต่างๆ ที่เราสร้างไว้ในตัว Contract ได้ สามารถใช้งานได้จริง

ภาคผนวก ข

คู่มือการซื้อสินค้า

คู่มือการซื้อสินค้า

1. เข้าสู่เว็บไซต์ที่สร้างขึ้น

รูปที่ ข-1 หน้าตาเว็บไซต์ร้านค้าเกษตรกรออนไลน์บนระบบสัญญาอัจฉริยะ

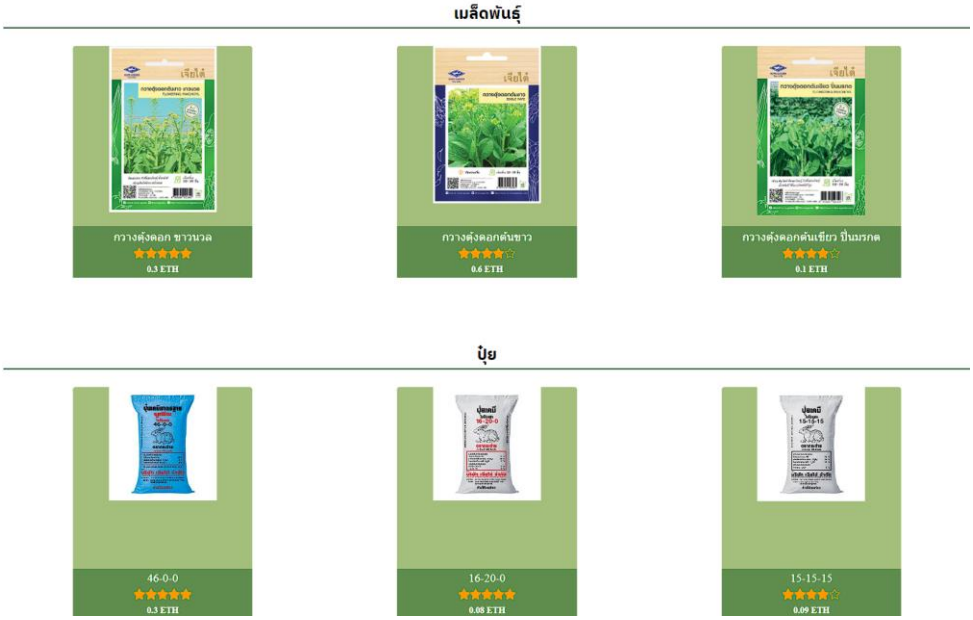
จากรูปที่ ข-1 หน้าเว็บไซต์ร้านค้าที่พัฒนาร่วมกับสัญญาอัจฉริยะ เพื่อให้ลูกค้าสามารถเข้ามาทำการซื้อขายสินค้าโดยใช้สกุลเงินดิจิทัลได้

2. เชื่อมต่อกับกระเป๋า MetaMask

รูปที่ ข-2 ก่อน-หลังการเชื่อมต่อกระเป๋า MetaMask

จากรูปที่ ข-2 เป็นการเชื่อมบัญชี MetaMask ของผู้ที่ต้องการทำธุรกรรมเข้ากับเว็บไซต์

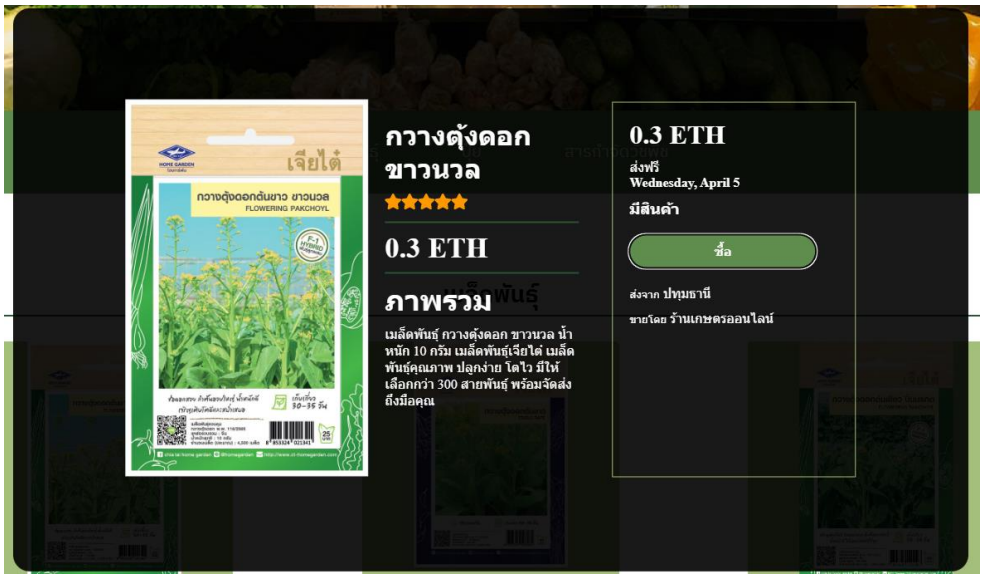
3. เลือกดู รายการแสดงสินค้า



รูปที่ ข-3 แสดงรายการสินค้า

จากรูปที่ ข-3 เป็นสินค้าประเภทเกษตรที่ผู้ขายทำการเขียนลงในสัญญาอัจฉริยะ เพื่อให้ผู้ที่ต้องการมาซื้อสินค้าประเภทเกษตกรได้เลือกซื้อสิ่งที่ต้องการ

4. กดเลือกสินค้าที่ต้องการ



รูปที่ ข-4 หน้าต่างแสดงข้อมูลสินค้า

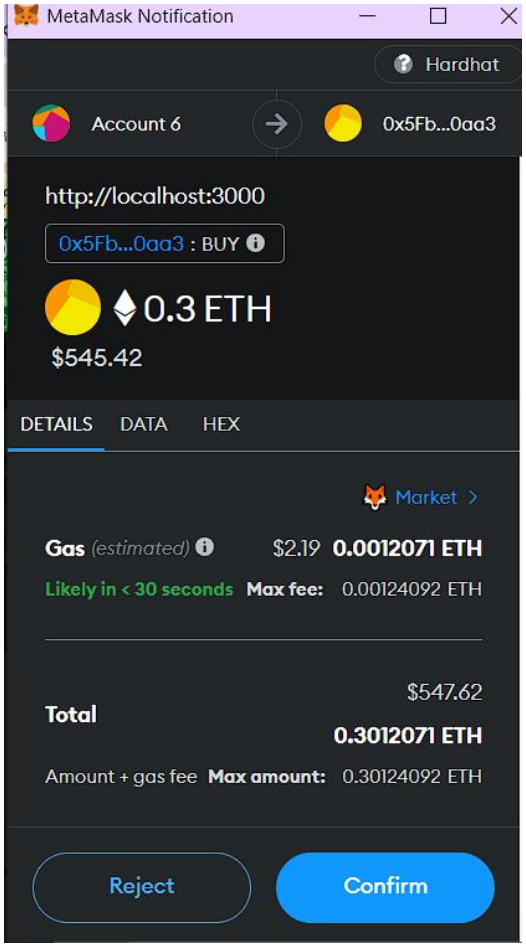
จากรูปที่ ข-4 เมื่อลูกค้าทำการเลือกที่สินค้าก็จะแสดงรายละเอียดสินค้าต่างๆ

5. กดปุ่ม “ซื้อ” เพื่อซื้อสินค้า

รูปที่ ข-5 ปุ่มซื้อสินค้า

จากรูปที่ ข-5 เมื่อลูกค้าทำการเลือกที่สินค้าก็จะแสดงรายละเอียดสินค้าต่างๆ สามารถทำธุรกรรมด้วยการเลือกที่ปุ่ม “ซื้อ” เพื่อซื้อสินค้า

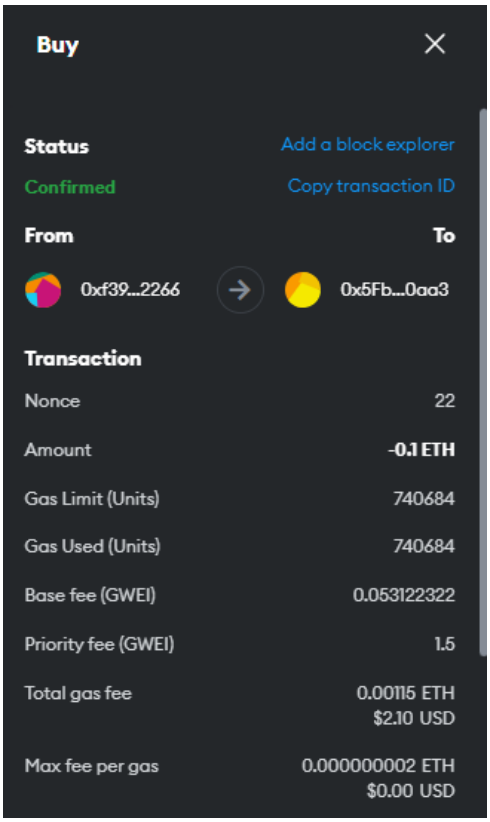
6. กด Confirm เพื่อยืนยันการซื้อสินค้า



รูปที่ ข-6 หน้าต่างการชำระเงินค่าสินค้า

จากรูปที่ ข-6 เมื่อลูกค้ากดที่ปุ่ม “ซื้อ” บัญชี MetaMask ก็จะเด้งขึ้นมาเพื่อให้ลูกค้ายอมรับ เมื่อกด “Confirm” ก็จะเป็นการตกลงในการทำธุรกรรม

7. สามารถกดดูประวัติการชำระเงินค่าสินค้า



รูปที่ ข-7 หน้าต่างประวัติการชำระเงินค่าสินค้า

จากรูปที่ ข-7 เมื่อลูกค้ากดที่ปุ่ม “Confirm” ก็จะเป็นการตกลงในการทำธุรกรรมและจะมีประวัติการโอนเงินไปยังผู้ขายเป็นการเสร็จสิ้นธุรกรรมระหว่างผู้ซื้อและผู้ขาย

ภาคผนวก ค

Source code

Source code

Contracts/farmer.sol

```
// SPDX-License-Identifier: UNLICENSED

pragma solidity ^0.8.9;

contract Farmer {

    address public owner;

    struct Item {

            uint256 id;

            string  name;

            string  description;

            string  category;

            string  image;

            uint256 cost;

            uint256 rating;

            uint256 stock;

    }

    struct Order {

        uint256 time;
```

```
    Item item;
}
mapping(uint256 => Item) public items;
mapping(address => mapping(uint256 => Order)) public orders;
mapping(address => uint256) public orderCount;


event Buy(address buyer, uint256 orderId, uint256 itemId);
event List(string name, uint256 cost, uint256 quantity);


modifier onlyOwner() {
    require(msg.sender == owner);
    _;
}
constructor() {
    owner = msg.sender;
}
// List products
function list(
```

```
    uint256 _id,
    string memory _name,
    string memory _description,
    string memory _category,
    string memory _image,
    uint256 _cost,
    uint256 _rating,
    uint256 _stock
) public onlyOwner {
    // Create Item
    Item memory item = Item(
        _id,
        _name,
        _description,
        _category,
        _image,
        _cost,
        _rating,
```

```
        _stock
    );

    // Save Item struct to blockchain
    items[_id] = item;

    // Emit an event
    emit List(_name, _cost, _stock);
}

// Buy products
function buy(uint256 _id) public payable {
    // Fetch item
    Item memory item = items[_id];

    // Require enough ether to buy item
    require(msg.value >= item.cost);
```

```
    // Require item is in stock
    require(item.stock > 0);

    // Create order
    Order memory order = Order(block.timestamp, item);

    // Add order for user
    orderCount[msg.sender]++; // <-- Order ID
    orders[msg.sender][orderCount[msg.sender]] = order;

    // Subtract stock
    items[_id].stock = item.stock - 1;

    // Emit event
    emit Buy(msg.sender, orderCount[msg.sender], item.id);
}

// Withdraw funds
```

```
    function withdraw() public onlyOwner {
        (bool success, ) = owner.call{value: address(this).balance}("");
        require(success);
    }
}
```

scripts/Deploy.js

```
const hre = require("hardhat")
const { items } = require("../src/items.json")
const tokens = (n) => {
  return ethers.utils.parseUnits(n.toString(), 'ether')
}
async function main() {
  // Setup accounts
  const [deployer] = await ethers.getSigners()
  // Deploy Farmer
  const Farmer = await hre.ethers.getContractFactory("Farmer")
  const farmer = await Farmer.deploy()
```

```
await farmer.deployed()

console.log(`Deployed Farmer Contract at: ${farmer.address}\n`)


// Listing items...

for (let i = 0; i < items.length; i++) {

  const transaction = await farmer.connect(deployer).list(

    items[i].id,

    items[i].name,

    items[i].description,

    items[i].category,

    items[i].image,

    tokens(items[i].price),

    items[i].rating,

    items[i].stock,

  )

  await transaction.wait()

  console.log(`Listed item ${items[i].id}: ${items[i].name}`)

}
```

```
}
main().catch((error) => {
  console.error(error);
  process.exitCode = 1;
});
```

src/abis/Farmer.json

```
[
    {
        "inputs": [],
        "stateMutability": "nonpayable",
        "type": "constructor"
    },
    {
        "anonymous": false,
        "inputs": [
            {
                "indexed": false,
```

```
            "internalType": "address",
            "name": "buyer",
            "type": "address"
        },
        {
            "indexed": false,
            "internalType": "uint256",
            "name": "orderId",
            "type": "uint256"
        },
        {
            "indexed": false,
            "internalType": "uint256",
            "name": "itemId",
            "type": "uint256"
        }
    ],
    "name": "Buy",
```

```
    "type": "event"
},
{
    "anonymous": false,
    "inputs": [
        {
            "indexed": false,
            "internalType": "string",
            "name": "name",
            "type": "string"
        },
        {
            "indexed": false,
            "internalType": "uint256",
            "name": "cost",
            "type": "uint256"
        },
        {
```

```
            "indexed": false,
            "internalType": "uint256",
            "name": "quantity",
            "type": "uint256"
        }
    ],
    "name": "List",
    "type": "event"
},
{
    "inputs": [
        {
            "internalType": "uint256",
            "name": "_id",
            "type": "uint256"
        }
    ],
    "name": "buy",
```

```
    "outputs": [],
    "stateMutability": "payable",
    "type": "function"
},
{
    "inputs": [
        {
            "internalType": "uint256",
            "name": "",
            "type": "uint256"
        }
    ],
    "name": "items",
    "outputs": [
        {
            "internalType": "uint256",
            "name": "id",
            "type": "uint256"
```

```
},
{
    "internalType": "string",
    "name": "name",
    "type": "string"
},
{
    "internalType": "string",
    "name": "description",
    "type": "string"
},
{
    "internalType": "string",
    "name": "category",
    "type": "string"
},
{
    "internalType": "string",
```

```
    "name": "image",
    "type": "string"
},
{
    "internalType": "uint256",
    "name": "cost",
    "type": "uint256"
},
{
    "internalType": "uint256",
    "name": "rating",
    "type": "uint256"
},
{
    "internalType": "uint256",
    "name": "stock",
    "type": "uint256"
}
```

```
    ],
    "stateMutability": "view",
    "type": "function"
},
{
    "inputs": [
        {
            "internalType": "uint256",
            "name": "_id",
            "type": "uint256"
        },
        {
            "internalType": "string",
            "name": "_name",
            "type": "string"
        },
        {
            "internalType": "string",
```

"name": "_description",
"type": "string"
},
{
"internalType": "string",
"name": "_category",
"type": "string"
},
{
"internalType": "string",
"name": "_image",
"type": "string"
},
{
"internalType": "uint256",
"name": "_cost",
"type": "uint256"
},

```
        {
            "internalType": "uint256",
            "name": "_rating",
            "type": "uint256"
        },
        {
            "internalType": "uint256",
            "name": "_stock",
            "type": "uint256"
        }
    ],
    "name": "list",
    "outputs": [],
    "stateMutability": "nonpayable",
    "type": "function"
},
{
    "inputs": [
```

```
        {
            "internalType": "address",
            "name": "",
            "type": "address"
        }
    ],
    "name": "orderCount",
    "outputs": [
        {
            "internalType": "uint256",
            "name": "",
            "type": "uint256"
        }
    ],
    "stateMutability": "view",
    "type": "function"
},
{
```

"inputs": [
    {
        "internalType": "address",
        "name": "",
        "type": "address"
    },
    {
        "internalType": "uint256",
        "name": "",
        "type": "uint256"
    }
],
"name": "orders",
"outputs": [
    {
        "internalType": "uint256",
        "name": "time",
        "type": "uint256"

```
},
{
    "components": [
        {
            "internalType": "uint256",
            "name": "id",
            "type": "uint256"
        },
        {
            "internalType": "string",
            "name": "name",
            "type": "string"
        },
        {
            "internalType": "string",
            "name": "name",
            "type": "description"
        },
```

```
{
    "internalType": "string",
    "name": "category",
    "type": "string"
},
{
    "internalType": "string",
    "name": "image",
    "type": "string"
},
{
    "internalType": "uint256",
    "name": "cost",
    "type": "uint256"
},
{
    "internalType": "uint256",
    "name": "rating",
```

```
                "type": "uint256"
            },
            {
                "internalType": "uint256",
                "name": "stock",
                "type": "uint256"
            }
        ],
        "internalType": "struct Farmer.Item",
        "name": "item",
        "type": "tuple"
    }
  ],
  "stateMutability": "view",
  "type": "function"
},
{
  "inputs": [],
```

```
    "name": "owner",
    "outputs": [
        {
            "internalType": "address",
            "name": "",
            "type": "address"
        }
    ],
    "stateMutability": "view",
    "type": "function"
},
{
    "inputs": [],
    "name": "withdraw",
    "outputs": [],
    "stateMutability": "nonpayable",
    "type": "function"
}
```

]

src/components/Navigations

```
import { ethers } from 'ethers'

import CarImage from '../assets/ETH.png'

const Navigation = ({ account, setAccount }) => {

    const connectHandler = async () => {

        const accounts = await window.ethereum.request({ method:
'eth_requestAccounts' });

        const account = ethers.utils.getAddress(accounts[0])

        setAccount(account);

    }

    return (

        <nav>

            <div className='nav__brand'>

            <div class="nav-img">

            <img src={CarImage} alt="car"/>

            </div>
```

```
<h1>ร้านค้าเกษตกร</h1>
</div>
{account ? (
    <button
        type="button"
        className='nav__connect'
    >
        {account.slice(0, 6) + '...' + account.slice(38, 42)}
    </button>
) : (
    <button
        type="button"
        className='nav__connect'
        onClick={connectHandler}
    >
        เชื่อมต่อบัญชี
    </button>
)}
```

```
        </nav>
    );
}
export default Navigation;
```

src/components/Product.js

```
import { useEffect, useState } from 'react'
import { ethers } from 'ethers'

// Components
import Rating from './Rating'
import close from '../assets/close.svg'

const Product = ({ item, provider, account, farmer, togglePop }) => {
  const [order, setOrder] = useState(null)
  const [hasBought, setHasBought] = useState(false)

  const fetchDetails = async () => {
```

```
    const events = await farmer.queryFilter("Buy")
    const orders = events.filter(
      (event) => event.args.buyer === account && event.args.itemId.toString()
=== item.id.toString()
    )
    if (orders.length === 0) return
    const order = await farmer.orders(account, orders[0].args.orderId)
    setOrder(order)
  }
  const buyHandler = async () => {
    const signer = await provider.getSigner()


    // Buy item...
    let transaction = await farmer.connect(signer).buy(item.id, { value:
item.cost })
    await transaction.wait()
    setHasBought(true)
  }
```

```
useEffect(() => {
  fetchDetails()
}, [hasBought])
return (
  <div className="product">
    <div className="product__details">
      <div className="product__image">
        <img src={item.image} alt="Product" />
      </div>
      <div className="product__overview">
        <h1>{item.name}</h1>
        <Rating value={item.rating} />
        <hr />
        <p>{item.address}</p>
        <h2>{ethers.utils.formatUnits(item.cost.toString(), 'ether')} ETH</h2>
        <hr />
```

```
    <h2>ภาพรวม</h2>
    <p>{item.description}</p>
  </div>
  <div className="product__order">
    <h1>{ethers.utils.formatUnits(item.cost.toString(), 'ether')} ETH</h1>
    <p>
      ส่งฟรี <br />
      <strong>
        {new Date(Date.now() + 345600000).toLocaleDateString(undefined,
{ weekday: 'long', month: 'long', day: 'numeric' })}
      </strong>
    </p>
    {item.stock > 0 ? (
      <h3>มีสินค้า</h3>
    ) : (
      <h3>สินค้าหมด</h3>
    )}
```

```
<button className='product__buy' onClick={buyHandler}>
  ซื้อ
</button>
<p><small>ส่งจาก</small> ปทุมธานี</p>
<p><small>ขายโดย</small> ร้านเกษตรออนไลน์</p>
{order && (
  <div className='product__bought'>
    Item bought on <br />
    <strong>
      {new Date(Number(order.time.toString() +
'000')).toLocaleDateString(
        undefined,
        {
          weekday: 'long',
          hour: 'numeric',
          minute: 'numeric',
          second: 'numeric'
        })}
```

```
            </strong>
          </div>
        )}
      </div>
      <button onClick={togglePop} className="product__close">
        <img src={close} alt="Close" />
      </button>
    </div>
  </div >
  );
}
export default Product;
```

src/components/Rating.js

```
import star_regular from '../assets/star-regular.svg';
import star_solid from '../assets/star-solid.svg';
const Rating = ({ value }) => {
    return (
```

```
        <div className='rating'>
            <img src={value >= 1 ? star_solid : star_regular} width="20px"
height="20px" alt="Star" />
            <img src={value >= 2 ? star_solid : star_regular} width="20px"
height="20px" alt="Star" />
            <img src={value >= 3 ? star_solid : star_regular} width="20px"
height="20px" alt="Star" />
            <img src={value >= 4 ? star_solid : star_regular} width="20px"
height="20px" alt="Star" />
            <img src={value >= 5 ? star_solid : star_regular} width="20px"
height="20px" alt="Star" />
        </div>
    );
}
export default Rating;
```

src/components/Section.js

```
import { ethers } from 'ethers'
import Rating from './Rating'
```

```
const Section = ({ title, items, togglePop }) => {
    return (
        <div className='cards__section'>
            <h3 id={title}>{title}</h3>
            <hr />
            <div className='cards'>
                {items.map((item, index) => (
                    <div className='card' key={index} onClick={() =>
togglePop(item)}>
                        <div className='card__image'>
                            <img src={item.image} alt="Item" />
                        </div>
                        <div className='card__info'>
                            <h4>{item.name}</h4>
                            <Rating value={item.rating} />
                            <p>{ethers.utils.formatUnits(item.cost.toString(), 'ether')}
ETH</p>
                        </div>
```

```
                </div>
            ))}
        </div>
    </div>
  );
}
export default Section;
```

src/components/Title.js

```
const Title = () => {
  return (
    <div class="title">
      <div class="title-text">
      <h1>เพื่อเกษตรกรที่ยั่งยืน โดยร้านเกษตรกรที่ทันสมัย</h1>
      <p>ร้านค้าเกษตรที่แรกของไทย ที่คุณสามารถซื้อได้ผ่านระบบสัญญาอัจฉริยะ
</p>
      </div>
    </div>
```

```
    )
}
export default Title
```

src/App.js

```
import { useEffect, useState } from 'react'
import { ethers } from 'ethers'
// Components
import Navigation from './components/Navigation'
import Section from './components/Section'
import Product from './components/Product'
import Titles from './components/Title'
// ABIs
import Farmer from './abis/Farmer.json'
// Config
import config from './config.json'
function App() {
  const [provider, setProvider] = useState(null)
```

```
const [farmer, setFarmer] = useState(null)

const [account, setAccount] = useState(null)


const [seeds, setSeeds] = useState(null)

const [fertilizers, setFertilizers] = useState(null)

const [herbicides, setHerbicides] = useState(null)

const [item, setItem] = useState({})

const [toggle, setToggle] = useState(false)

const togglePop = (item) => {

  setItem(item)

  toggle ? setToggle(false) : setToggle(true)

}

const loadblockchainData = async () => {

  // Connect to blockchain

  const provider = new ethers.providers.Web3Provider(window.ethereum)

  setProvider(provider)

  const network = await provider.getNetwork()

  console.log(network)
```

```
    // Connect to smartcontracts (Create Js versions)
    const farmer = new
ethers.Contract(config[network.chainId].farmer.address, Farmer, provider)
    setFarmer(farmer)
      // Load products
      const items = []
      for (var i = 0; i < 9; i++) {
        const item = await farmer.items(i + 1)
        items.push(item)
      }
      const seeds = items.filter((item) => item.category === 'seeds')
      const fertilizers = items.filter((item) => item.category === 'fertilizers')
      const herbicides = items.filter((item) => item.category === 'herbicides')


      setSeeds(seeds)
      setFertilizers(fertilizers)
      setHerbicides(herbicides)
  }
```

```
useEffect(() => {
  loadblockchainData()
}, [])
return (
  <div>
    <Navigation account={account} setAccount={setAccount} />
    <Titles/>


    {/* <h2>สินค้าขายดี!!</h2> */}
    {/* <p>{ account}</p> */}




        <ul className='nav__links'>
          <li><a href="#เมล็ดพันธุ์">เมล็ดพันธุ์</a></li>
          <li><a href="#ปุ๋ย">ปุ๋ย</a></li>
          <li><a href="#สารกำจัดวัชพืช">สารกำจัดวัชพืช</a></li>
        </ul>
```

```
      {seeds && fertilizers && herbicides && (
        <>
          <Section title={"เมล็ดพันธุ์"} items={seeds} togglePop={togglePop} />
          <Section title={"ปุ๋ย"} items={fertilizers} togglePop={togglePop} />
          <Section title={"สารกำจัดวัชพืช"} items={herbicides}
togglePop={togglePop} />
        </>
      )}
       {toggle && (
        <Product item={item} provider={provider} account={account}
farmer={farmer} togglePop={togglePop} />
      )
    </div>
  );
}
export default App;
```

src/App.test.js

```
import { render, screen } from '@testing-library/react';

import App from './App';

test('renders learn react link', () => {

  render(<App />);

  const linkElement = screen.getByText(/learn react/i);

  expect(linkElement).toBeInTheDocument();

});
```

src/config.json

```
{

    "31337": {

        "farmer": {

            "address": "0x5FbDB2315678afecb367f032d93F642f64180aa3"

        }

    }

}
```

src/ index.css

```
@import
url('https://fonts.googleapis.com/css2?family=Charmonman&family=IBM+Plex
+Sans+Thai:wght@100;400;700&display=swap');
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}
body {
  margin: 0;
}
h2 {
  max-width: 1200px;
  margin: 20px auto 0;
  font-size: 2.15em;
}
h2:not(.product__overview h2) {
```

```
  padding-left: 20px;
}
nav {
  background-color: #fff;
  display: flex;
  justify-content:space-between;
  width: 80%;
  align-items: center;
  margin: 20px auto ;
}
.nav-img img{
  width: 50px;
  margin-right: 5px;
}
.title {
  background: url(https://images.unsplash.com/photo-1542838132-
92c53300491e?ixlib=rb-
```

```
4.0.3&ixid=MnwxMjA3fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8&auto=form
at&fit=crop&w=1074&q=80);
  background-position: center;
  background-repeat: no-repeat;
  background-size: cover;


  height: 900px;
  display: flex;
  align-items: center;
  margin: 0;
}


.title-text {
  background-color: rgba(0,0,0, 0.8); /* Black w/opacity/see-through */
  color: #fff;
  font-weight: bold;
  border: 3px solid #5F8D4E;
  position: absolute;
```

```
  top: 60%;
  left: 50%;
  transform: translate(-50%, -50%);
  z-index: 2;
  width: 70%;
  padding: 20px;
  text-align: center;
}
.title-text h1 {
    font-family: 'IBM Plex Sans Thai', sans-serif;
    font-size: 4rem;
    line-height: 1.5;
    font-weight: 900;
}
.title-img img {
  width: 100%;
}
.nav__brand {
```

```
  display: flex;
  justify-content: center;
  align-items: center;
  margin: 0 5rem;
}
.nav__brand img {
  max-width: 125px;
  height: auto;
}
.nav__brand h1 {
  font-family: 'IBM Plex Sans Thai', sans-serif;
  /* font-family: 'Charmonman'; */
  font-size: 2.5em;
  color: #285430;
}
hr{
  border: 1px solid #285430;
}
```

```
.nav__links {
  background-color: #5F8D4E;
  height: 100px;
  padding: 10px;
  display: flex;
  justify-content: center;
  align-items: center;
  list-style: none;
}
.nav__links li {
  margin: 0 50px;
}
.nav__links li a {
  text-decoration: none;
  color: #ffff;
  font-size: 20px;
  font-family: 'IBM Plex Sans Thai', sans-serif;
}
```

```
.nav__links li a:hover {
  color: #fcec0c;
}
.nav__connect {
  width: 150px;
  height: 40px;
  margin: 0 5rem;
  background-color: #056632;
  color:#fff;
  border: none;
  border-radius: 4px;
  font-family: "Lalezar";
  font-size: 1.25em;
  font-weight: 600;
  cursor: pointer;
  transition: all 250ms ease;
}
.nav__connect:hover {
```

```
  background-color: #285430;
}
.cards__section {
  max-width: 75%;
  margin: 0 auto 0;
  padding: 0 20px;
  text-align: center;
}
.cards__section h3 {
  font-family: 'IBM Plex Sans Thai', sans-serif;
  font-size: 1.75em;
  padding: 100px 0 0;
}
.cards {
  display: grid;
  gap: 10px;
  grid-template-columns: repeat(auto-fit, minmax(min(100%, 350px), 1fr));
  margin-top: 20px;
```

```
}
.card {
  width: 310px;
  height: 400px;
  background-color: #A4BE7B;
  margin: 10px auto;
  border-radius: 5px;
  cursor: pointer;
  position: relative;
  transition: all 250ms ease;
}


.card:hover {
  box-shadow: #000;
}
.card__image {
  display: flex;
  align-items: center;
```

```
  justify-content: center;
  width: 100%;
  z-index: -1;
}
.card__image img {
  max-width: 60%;
  max-height: 60%;
}
.card__info {
  width: 100%;
  background-color: #5F8D4E;
  color: #fff;
  padding: 5px 10px;
  position:absolute;
  bottom: 0;
  left: 0;
}
.card__info h4 {
```

```
  font-family: "Open Sans";
  font-size: 1.25em;
  font-weight: 400;
}
.card__info .rating {
  color: yellow;
  margin: 5px 0;
}
.card__info p {
  font-family: "Open Sans";
  font-size: 0.95em;
  font-weight: 600;
  margin-top: 5px;
}
.product {
  width: 60vw;
  height: 70vh;
  background-color: rgba(0, 0, 0, 0.9);
```

```
  color: #fff;
 border-radius: 20px;
  position: fixed;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
}
.product__details {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(min(100%, 250px), 1fr));
  align-items: center;


  width: 80%;
  height: 80%;
  margin: 0 auto;
  padding: 20px;


  position: fixed;
```

```
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
  z-index: 2;


  background: var(--clr-white);
}
.product__image {
  max-width: 500px;
  margin: 20px auto;
}
.product__image img {
  max-width: 100%;
}
.product__close {
  color: #fff;
  position: absolute;
  top: 10px;
```

```
  right: 10px;


  width: 30px;
  height: 30px;


  background: transparent;
  border: none;


  cursor: pointer;
}
.product__close img {
  width: 25px;
  height: 25px;
}
.product__overview {
  height: 90%;
  padding: 20px 40px 20px 20px;
}
```

```
.product__overview h2,
.product__overview hr,
.product__overview .rating {
  margin: 10px auto;
}
.product__overview ul {
  margin-left: 40px;
}
.product__order {
  max-width: 300px;
  height: 90%;
  border: 1px solid #A4BE7B;
  padding: 20px;
}
.product__order h1,
.product__order p {
  margin-bottom: 10px;
}
```

```
.product__buy {
  width: 225px;
  height: 40px;
  margin: 20px 0;
  background-color: #5F8D4E;
  color: #fff;
  border: none;
  border-radius: 24px;
  font-size: 0.95em;
  font-weight: 600;
  cursor: pointer;
  transition: all 250ms ease;
}
.product__buy:hover {
  background-color: #285430;
}
.product__bought {
  padding: 5px;
```

```
  border: 1px solid #5F8D4E;
}
@media screen and (max-width: 992px) {
  .product__details {
    height: 95%;
    width: 90%;
  }
  .product__order {
    max-width: 100%;
    height: 100%;
    margin-top: 20px;
    grid-column: span 2;
    color: #fff;
  }
}
@media screen and (max-width: 768px) {
  nav {
    grid-template-columns: repeat(2, 1fr);
```

```
  }
  .nav__links {
    display: none;
  }
  .nav__search {
    display: none;
  }
}

@media screen and (max-width: 576px) {
  .product__details {
    height: 95%;
    overflow-y: scroll;
  }
  .product__image {
    grid-column: span 2;
  }
}
```

src/index.js

```
import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import App from './App';

import reportWebVitals from './reportWebVitals';


const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

  <React.StrictMode>

    <App />

  </React.StrictMode>

);


// If you want to start measuring performance in your app, pass a function

// to log results (for example: reportWebVitals(console.log))

// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals();
```

src/items.json


```
{
  "items": [
    {
      "id": 1,
      "name": "กวางตุ้งดอก ขาวนวล",
      "description" : "เมล็ดพันธุ์ กวางตุ้งดอก ขาวนวล น้ำหนัก 10 กรัม เมล็ดพันธุ์เจียไต๋ เมล็ดพันธุ์คุณภาพ ปลูกง่าย โตไว มีให้เลือกกว่า 300 สายพันธุ์ พร้อมจัดส่งถึงมือคุณ",
      "category": "seeds",
      "image": "https://media.discordapp.net/attachments/1090617991143641088/1090618119531278346/01.jpg",
      "price": "0.3",
      "rating": 5,
      "stock": 28
    },
{
      "id": 2,
```

"name": "กวางตุ้งดอกต้นขาว",

"description" : "เมล็ดพันธุ์ กวางตุ้งดอกต้นขาว น้ำหนัก 50 กรัม จำนวน 26000 เมล็ด เมล็ดพันธุ์เจียไต๋ เมล็ดพันธุ์คุณภาพ ปลูกง่าย โตไว มีให้เลือกกว่า 300 สายพันธุ์ พร้อมจัดส่งถึงมือคุณ",

"category": "seeds",

"image": "https://media.discordapp.net/attachments/1090617991143641088/1090618119812288603/02.png",

"price": "0.6",

"rating": 4,

"stock": 46

},

{

"id": 3,

"name": "กวางตุ้งดอกต้นเขียว ปิ่นมรกต",

"description" : "เมล็ดพันธุ์ กวางตุ้งดอกต้นเขียว ปิ่นมรกต น้ำหนัก 10 กรัม เมล็ดพันธุ์เจียไต๋ เมล็ดพันธุ์คุณภาพ ปลูกง่าย โตไว มีให้เลือกกว่า 300 สายพันธุ์ พร้อมจัดส่งถึงมือคุณ",

"category": "seeds",

"image":
"https://media.discordapp.net/attachments/1090617991143641088/1090618120214949969/03.jpg",
"price": "0.1",
"rating": 4,
"stock": 32
},
{
"id": 4,
"name": "46-0-0",
"description" : "ใช้ได้กับพืชทุกชนิดทั้งนาข้าว พืชไร่ เพื่อสร้างการเจริญเติบโตทางด้านลำต้นและใบ",
"category": "fertilizers",
"image":
"https://media.discordapp.net/attachments/1090617991143641088/1090618165442125885/14.jpg",
"price": "0.3",
"rating": 5,
"stock": 27

},
{
"id": 5,
"name": "16-20-0",
"description" : "นิยมใช้ในนาข้าว และพืชไร่ทุกชนิด บำรุงราก และช่วยการแตกกอ",
"category": "fertilizers",
"image":
"https://media.discordapp.net/attachments/1090617991143641088/1090618165702180864/15.jpg",
"price": "0.08",
"rating": 5,
"stock": 34
},
{
"id": 6,
"name": "15-15-15",
"description" : "ใช้ได้กับพืชทุกชนิด โดยเฉพาะนาข้าว พืชไร่ พืชผัก และไม้ผล เพื่อเร่งการเจริญเติบโตทุกส่วนของพืช",

"category": "fertilizers",

"image": "https://media.discordapp.net/attachments/1090617991143641088/1090618165916094474/16.jpg",

"price": "0.09",

"rating": 4,

"stock": 41

},

{

"id": 7,

"name": "บิวต้า-โปร",

"description" : "เป็นสารกำจัดวัชพืชชนิดดูดซึมเข้าทางยอดอ่อนและราก ใช้พ่นก่อนวัชพืชงอกหรือวัชพืชมีใบจริง 2-3 ใบ",

"category": "herbicides",

"image": "https://media.discordapp.net/attachments/1090617991143641088/1090618203950026762/27.jpg",

"price": "0.05",

"rating": 5,

"stock": 23

},

{

"id": 8,

"name": "กรุมก้า",

"description" : "เป็นสารกำจัดวัชพืชชนิดดูดซึมเข้าทางยอดอ่อนและราก ใช้พ่นก่อนวัชพืชงอกหรือวัชพืชมีใบจริง 3-4 ใบ ใช้ได้กับข้าว พ่นที่ 8-12 วันหลังหว่านข้าว",

"category": "herbicides",

"image": "https://media.discordapp.net/attachments/1090617991143641088/1090618204252028948/28.jpg",

"price": "0.05",

"rating": 4,

"stock": 35

},

{

"id": 9,

"name": "ซีทีมีน",

"description" : "เป็นสารกำจัดวัชพืชชนิดดูดซึม โดยใช้กำจัดวัชพืชชนิดใบกว้างและกก ใช้ได้ในข้าวและอ้อย พ่นหลังวัชพืชงอก",

"category": "herbicides",

"image": "https://media.discordapp.net/attachments/1090617991143641088/1090618204608548875/29.jpg",

"price": "0.15",

"rating": 5,

"stock": 23

}

]

}

src/ reportWebVitals.js

```
const reportWebVitals = onPerfEntry => {
  if (onPerfEntry && onPerfEntry instanceof Function) {
    import('web-vitals').then(({ getCLS, getFID, getFCP, getLCP, getTTFB }) => {
      getCLS(onPerfEntry);
```

```
      getFID(onPerfEntry);
      getFCP(onPerfEntry);
      getLCP(onPerfEntry);
      getTTFB(onPerfEntry);
    });
  }
};
export default reportWebVitals;
```

src/ setupTests.js

```
// jest-dom adds custom jest matchers for asserting on DOM nodes.
// allows you to do things like:
// expect(element).toHaveTextContent(/react/i)
// learn more: https://github.com/testing-library/jest-dom
import '@testing-library/jest-dom';
```

test/Farmer.js

```
const { expect } = require("chai")
```

```
const { ethers } = require("hardhat")

const tokens = (n) => {

  return ethers.utils.parseUnits(n.toString(), 'ether')

}



// Global constans for listing an item...

  const ID = 1

  const NAME = "กระต่าย"

  const DESCRIPTION = "dsfsdfsdf"

  const CATEGORY = "ปุ๋ย"

  const IMAGE =
"https://media.discordapp.net/attachments/808186574382563349/107281156
1007194122/60251770r.webp"

  const COST = tokens(1)

  const RATING = 4

  const STOCK = 5
```

```
describe("Farmer", () => {
  let farmer
  let deployer, buyer
  beforeEach(async () =>{
    // Setup accounts
    [deployer, buyer] = await ethers.getSigners()


    // Deploy contact
    const Farmer = await ethers.getContractFactory("Farmer")
    farmer = await Farmer.deploy()
  })
  describe("Deplyment", () => {
    it("Sets the owner", async () => {
      expect(await farmer.owner()).to.equal(deployer.address)
    })
  })
  describe("Listing", () => {
    let transaction
```

```
beforeEach(async () =>{
  transaction = await farmer.connect(deployer).list(
    ID,
    NAME,
    DESCRIPTION,
    CATEGORY,
    IMAGE,
    COST,
    RATING,
    STOCK
  )
  await transaction.wait()
})
it("Returns item attributes", async () => {
  const item = await farmer.items(ID)
  expect(item.id).to.equal(ID)
  expect(item.name).to.equal(NAME)
  expect(item.description).to.equal(DESCRIPTION)
```

```
      expect(item.category).to.equal(CATEGORY)
      expect(item.image).to.equal(IMAGE)
      expect(item.cost).to.equal(COST)
      expect(item.rating).to.equal(RATING)
      expect(item.stock).to.equal(STOCK)
    })
    it("Emits List event", () => {
      expect(transaction).to.emit(farmer, "List")
    })
  })
  describe("Buying", () => {
    let transaction
    beforeEach(async () => {
      // List a item
      transaction = await farmer.connect(deployer).list(ID, NAME, DESCRIPTION,
CATEGORY, IMAGE, COST, RATING, STOCK)
      await transaction.wait()
```

```
  // Buy a item
  transaction = await farmer.connect(buyer).buy(ID, { value: COST })
  await transaction.wait()
})

it("Updates buyer's order count", async () => {
  const result = await farmer.orderCount(buyer.address)
  expect(result).to.equal(1)
})

it("Adds the order", async () => {
  const order = await farmer.orders(buyer.address, 1)

  expect(order.time).to.be.greaterThan(0)
  expect(order.item.name).to.equal(NAME)
})

it("Updates the contract balance", async () => {
```

```
      const result = await ethers.provider.getBalance(farmer.address)
      expect(result).to.equal(COST)
    })

    it("Emits Buy event", () => {
      expect(transaction).to.emit(farmer, "Buy")
    })
  })
  describe("Withdrawing", () => {
    let balanceBefore

    beforeEach(async () => {
      // List a item
      let transaction = await farmer.connect(deployer).list(ID, NAME,
DESCRIPTION, CATEGORY, IMAGE, COST, RATING, STOCK)
      await transaction.wait()

      // Buy a item
```

```
  transaction = await farmer.connect(buyer).buy(ID, { value: COST })
  await transaction.wait()

  // Get Deployer balance before
  balanceBefore = await ethers.provider.getBalance(deployer.address)

  // Withdraw
  transaction = await farmer.connect(deployer).withdraw()
  await transaction.wait()
})

it('Updates the owner balance', async () => {
  const balanceAfter = await ethers.provider.getBalance(deployer.address)
  expect(balanceAfter).to.be.greaterThan(balanceBefore)
})

it('Updates the contract balance', async () => {
  const result = await ethers.provider.getBalance(farmer.address)
```

```
      expect(result).to.equal(0)
    })
  })
})
```

.gitgnore

```
# See https://help.github.com/articles/ignoring-files/ for more about ignoring files.

# dependencies
/node_modules
/.pnp
.pnp.js

# testing
/coverage

# production
/build
```

```
# misc
.DS_Store
.env.local
.env.development.local
.env.test.local
.env.production.local

npm-debug.log*
yarn-debug.log*
yarn-error.log*

node_modules
.env
coverage
coverage.json
typechain
typechain-types
```

```
#Hardhat files
cache
artifacts

node_modules
.env
coverage
coverage.json
typechain
typechain-types

#Hardhat files
cache
artifacts
```

shardhat.config.js

```
require("@nomicfoundation/hardhat-toolbox");
```

```
/** @type import('hardhat/config').HardhatUserConfig */

module.exports = {

  solidity: "0.8.17",

};
```

ภาคผนวก ง

ประวัติผู้จัดทำ

ประวัติผู้จัดทำโครงงานวิจัย

ชื่อ-นามสกุล : นางสาวพรทิพา จันทร์เจริญ

รหัสนักศึกษา : 116210906036-5

สาขาวิชา/ภาควิชา : สาขาวิชาเทคโนโลยีสารสนเทศ

ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์

วัน/เดือน/ปีเกิด : 02 กุมภาพันธ์ 2543

ประวัติการศึกษา : มัธยมศึกษาตอนต้น (ม.3) โรงเรียนวัดอู่ข้าว

ปีที่สำเร็จการศึกษา พ.ศ.2559

มัธยมศึกษาตอนปลาย (ม.6) โรงเรียนหนองเสือวิทยาคม

ปีที่สำเร็จการศึกษา พ.ศ.2562

ที่อยู่ : บ้านเลขที่ 9/1 หมู่ 9 ตำบลบึงชำอ้อ อำเภอหนองเสือ

จังหวัดปทุมธานี 12170

อีเมล์ : Porntipamauy2543@gmail.com

ประวัติผู้จัดทำโครงงานวิจัย

ชื่อ-นามสกุล :	นางสาวนภัสสร สุขแก่น
รหัสนักศึกษา :	116210906028-2
สาขาวิชา/ภาควิชา :	สาขาวิชาเทคโนโลยีสารสนเทศ
	ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์
วัน/เดือน/ปีเกิด :	27 มิถุนายน 2543
ประวัติการศึกษา :	มัธยมศึกษาตอนต้น (ม.3) โรงเรียนห้วยน้ำหอมวิทยาคาร
	ปีที่สำเร็จการศึกษา พ.ศ.2559
	มัธยมศึกษาตอนปลาย (ม.6) โรงเรียนห้วยน้ำหอมวิทยาคาร
	ปีที่สำเร็จการศึกษา พ.ศ.2562
ที่อยู่ :	บ้านเลขที่ 7 หมู่ 14 ตำบลแม่เล่ย์ อำเภอแม่วงก์
	จังหวัดนครสวรรค์ 60150
อีเมล์ :	junenaphatsorn@gmail.com

ประวัติผู้จัดทำโครงงานวิจัย

ชื่อ-นามสกุล :	นายสุเมธี คำเอี่ยม
รหัสนักศึกษา :	116210906091-0
สาขาวิชา/ภาควิชา :	สาขาวิชาเทคโนโลยีสารสนเทศ ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์
วัน/เดือน/ปีเกิด :	4 มีนาคม 2542
ประวัติการศึกษา :	มัธยมศึกษาตอนต้น (ม.3) โรงเรียนวัดหลักสี่พิพัฒน์ราษฎร์อุปถัมภ์ ปีที่สำเร็จการศึกษา พ.ศ.2558 มัธยมศึกษาตอนปลาย (ม.6) โรงเรียนวัดหลักสี่พิพัฒน์ราษฎร์อุปถัมภ์ ปีที่สำเร็จการศึกษา พ.ศ.2561
ที่อยู่ :	บ้านเลขที่ 39 หมู่ที่ 4 ตำบลหนองสองห้อง อำเภอบ้านแพ้ว จังหวัดสมุทรสาคร 74120
อีเมล์ :	xsumethix123@gmail.com