

KGI ALGO TRADING BOOTCAMP & COMPETITION 2026

เรียนรู้การสร้าง **Algo-Trading** พร้อม Workshop
และการแข่งขันสร้าง**กลยุทธ์การลงทุนที่มีประสิทธิภาพด้วยอัลกอลิทึม**

ชิงทุนการศึกษา
KGI Scholarship
มูลค่ารวมสูงสุด **150,000*** บาท

* สำหรับมหาวิทยาลัยที่เข้าร่วมโครงการ และทีมเข้าร่วมแข่งขัน 40 ทีม ขึ้นไป

TradeSim — What You Need Before You Start

ก่อนที่จะเข้าร่วมการแข่งขัน ให้แน่ใจว่าคุณมีความรู้พื้นฐานตามหัวข้อด้านล่าง และติดตั้งเครื่องมือที่จำเป็นเรียบร้อยแล้ว

- Python 3.13 ONLY
- VS Code with the Jupyter Notebook extension
- git + GitHub account

Download tradeSim via this link: TradeSim

Agenda

1. ข้อควรรู้ก่อนเริ่มซื้อ–ขายหุ้น
2. Algorithmic trading คืออะไร?
3. ขั้นตอนการสร้าง Algorithmic trading
4. TradeSim Overview
5. Competition Details
6. Q & A

ข้อควรรู้ก่อนเริ่มซื้อ–ขายหุ้น

ข้อควรรู้ก่อนเริ่มซื้อ–ขายหุ้น

SET Index คืออะไร

SET Index หรือ ดัชนีราคาหุ้นตลาดหลักทรัพย์ฯ คือ ดัชนีที่สะท้อนความเคลื่อนไหวของราคาหุ้นในตลาดหลักทรัพย์ฯ ทั้งหมด ซึ่งตัวเลขที่ปรับตัวเพิ่มขึ้นหรือลดลงในแต่ละวันเป็นเหมือนตัวชี้วัดให้นักลงทุนเห็นว่าภาพรวมของตลาดหุ้นในวันนั้นๆ เป็นเช่นไร



สีเขียว = มูลค่าตลาดรวมวันนี้ **สูงกว่า** วันก่อนหน้า

สีเหลือง = มูลค่าตลาดรวมวันนี้ **เท่ากับ** วันก่อนหน้า

สีแดง = มูลค่าตลาดรวมวันนี้ **ต่ำกว่า** วันก่อนหน้า

นอกจากดัชนี SET Index แล้ว ในตลาดหลักทรัพย์ฯ มีดัชนีอีกมากมาย เช่น

SET50

ดัชนีสะท้อนความเคลื่อนไหวของราคาหุ้น 50 ตัวแรก

ที่มีมูลค่าหลักทรัพย์ตามราคาตลาดสูงที่สุด เป็นกลุ่มที่มีสภาพคล่องในการซื้อขายสูงที่สุด นักลงทุนนิยม เพราะเป็นกลุ่มธุรกิจแถวหน้าในไทย

SET100

ดัชนีสะท้อนความเคลื่อนไหวของราคาหุ้น 100 ตัวแรก

ที่มีมูลค่าหลักทรัพย์ตามราคาตลาดสูงที่สุด มีสภาพคล่องสูง หุ้นกลุ่มนี้จะอยู่ในบริษัทขนาดใหญ่และขนาดกลางที่เติบโตสูง มีสัดส่วนผู้ถือหุ้นตามที่ตลาดหลักทรัพย์ฯ กำหนดไว้

SETHD

ดัชนีสะท้อนความเคลื่อนไหวของราคาหุ้น 30 ตัว

ที่มีมูลค่าหลักทรัพย์ตามราคาตลาดสูง สภาพคล่องการซื้อขายอยู่ในเกณฑ์ดี มีอัตราผลตอบแทนจากเงินปันผลต่อเนื่อง ซึ่งบริษัทเหล่านี้จะต้องมีนโยบายจ่ายเงินปันผลมากกว่า 85% ของกำไรในช่วง 3 ปีที่ผ่านมา เป็นที่นิยมของนักลงทุนที่ชื่นชอบหุ้นปันผล

sSET

ดัชนีสะท้อนความเคลื่อนไหวของราคาหุ้นที่อยู่นอกเหนือดัชนี SET50 และ SET100

ซึ่งเป็นหุ้นที่มีมูลค่าหลักทรัพย์ตามราคาตลาดอยู่ในลำดับที่ 90% ขึ้นไป แต่จะไม่เกินลำดับที่ 98% สภาพคล่องในการซื้อขายสูงสม่ำเสมอ และมีสัดส่วนผู้ถือหุ้นอยู่ในเกณฑ์ที่กำหนด

SETTHSI

ดัชนีสะท้อนความเคลื่อนไหวราคาหุ้นยั่งยืน (Thailand Sustainability Investment : THSI)

ซึ่งเป็นบริษัทที่มีการดำเนินธุรกิจอย่างยั่งยืน มีความโดดเด่นด้าน ESG โดยคำนึงถึงผู้มีส่วนได้เสียอย่างครอบคลุม ทั้งด้านสิ่งแวดล้อม สังคม และบรรษัทภิบาล

SETWB

ดัชนีสะท้อนความเคลื่อนไหวของราคาหุ้น 30 ตัว ใน 7 หมวดธุรกิจที่โดดเด่น

นักลงทุนต่างชาติให้ความสนใจ ซึ่งการเติบโตของธุรกิจเหล่านี้มีผลต่อการขยายตัว GDP นำมาสู่การสร้างรายได้แก่คนในประเทศ

SETTRI

ดัชนีผลตอบแทนรวมที่คิดจากผลตอบแทนทุกประเภทของการลงทุน

ทั้งเงินปันผล (Dividend) ส่วนต่างราคา (Capital Gain / Loss) และสิทธิในการจองซื้อหุ้น (Right) ซึ่งแตกต่างจากดัชนีทั่วไปที่สะท้อนเฉพาะผลตอบแทนจากส่วนต่างราคา

mai

ดัชนีสะท้อนภาวะการซื้อขายหุ้นในตลาดหลักทรัพย์เอ็ม เอ ไอ

โดยเป็นหุ้นของบริษัทขนาดเล็ก หรือ SME รวมถึงธุรกิจสตาร์ทอัพที่มีการเติบโตสูง แต่มีมูลค่าหลักทรัพย์ตามราคาตลาดอยู่ในระดับที่ต่ำ และมีสภาพคล่องการซื้อขายที่น้อยเช่นกัน

https://www.setinvestnow.com/th/stock/what-is-set-index

ข้อควรรู้ก่อนเริ่มซื้อ–ขายหุ้น

ทำไมต้อง SET50

- กลุ่มหุ้นที่คนส่วนใหญ่ รู้จักและคุ้นเคยเป็นอย่างดี
- ลดปัญหาสภาพคล่อง และการส่งคำสั่งไม่สำเร็จ
- ลดความแตกต่างจาก หุ้นรายตัว และปัจจัยอื่นๆ
- เหมาะกับการใช้ SET50 Index เป็น Benchmark
- เหมาะกับนักศึกษาและผู้เริ่มต้น ในการศึกษาธุรกิจ

ใช้ในการแข่งขัน





ข้อควรรู้ก่อนเริ่มซื้อ–ขายหุ้น

บัญชีที่ใช้ซื้อ–ขายหุ้นมีกี่ประเภท

Cash Balance
บัญชีวางเงินล่วงหน้า

- ก่อนทำการซื้อหุ้น นักลงทุนจะต้องฝากเงินสดเข้าบัญชีซื้อขายหุ้น
- นักลงทุนสามารถซื้อหุ้นได้เท่ากับจำนวนเงินที่มีในบัญชีโดยสังเกตุได้จาก อำนาจซื้อหลักทรัพย์ (Line Available)
- เงินที่ฝากไว้แต่ยังไม่ได้นำไปซื้อหุ้น ก็จะได้รับดอกเบี้ย
- เหมาะสำหรับนักลงทุนมือใหม่ พึ่งเริ่มบริหารเงินลงทุนด้วยตนเอง

ใช้ในการแข่งขัน

Cash Account
บัญชีเงินสด

- ก่อนทำการซื้อหุ้น นักลงทุนต้องวางหลักประกัน เช่น เงินสด หรือ หลักทรัพย์ เป็นจำนวน 20% ของมูลค่าหุ้นที่ต้องการซื้อ
- โบรกเกอร์จะทำกันตัดเงินในบัญชีธนาคารที่ผูกไว้ตามมูลค่าที่ส่งคำสั่งซื้อหุ้นใน อีก 2 วันทำการ
- เงินที่ฝากไว้แต่ยังไม่ได้นำไปซื้อหุ้น ก็จะได้รับดอกเบี้ย
- เหมาะสำหรับนักลงทุนที่สามารถบริหารเงินลงทุนได้อย่างดี

Credit Balance
บัญชีมาร์จิ้น

- บัญชีกู้เงิน นักลงทุนสามารถกู้ยืมเงินเพื่อซื้อหุ้นได้บางส่วน
- ต้องวางหลักประกันการชำระหนี้ก่อนการซื้อหลักทรัพย์ครั้งแรกในจำนวนไม่ต่ำกว่าอัตรามาร์จิ้นเริ่มต้น (Initial Margin) เท่ากับร้อยละ 50 / ตามสัดส่วนที่บริษัทฯ กำหนด
- ลูกค้าต้องจ่ายดอกเบี้ยสำหรับเงินกู้ตามอัตราที่บริษัทฯ
- เหมาะสำหรับนักลงทุนที่มีประสบการณ์สามารถบริหารเงินทุนและหนี้ได้อย่างดี

ข้อควรรู้ก่อนเริ่มซื้อ–ขายหุ้น

ต้นทุนที่ควรคำนึง มีอะไรบ้าง

ใช้ในการแข่งขัน

Commission Fee

- ค่าธรรมเนียมนายหน้า (เรียกเก็บจากโบรกเกอร์)
- ค่าธรรมเนียมตลาด (เรียกเก็บจากผู้ดูแล เช่น SET, TSD, SEC)

Cost

ใช้ในการแข่งขัน

VAT

- 7% ของ Commission Fee

ใช้ในการแข่งขัน

Slippage

- ค่าความคลาดเคลื่อนของราคาซื้อขาย ที่อาจเกิดขึ้น
- เช่น นักลงทุนต้องการซื้อหุ้นที่ราคา 10 หากส่งคำสั่งซื้อบางประเภท เช่น MKT MTL มีโอกาสที่จะไม่ได้ราคาตามที่ต้องการ

ข้อควรรู้ก่อนเริ่มซื้อ–ขายหุ้น

ประเภทและคำสั่งซื้อ–ขายหุ้น มีอะไรบ้าง

ประเภท	ทำอะไร	ใช้ได้ช่วงไหน
Limit	ตั้งราคาเอง ได้ที่ราคานั้นหรือดีกว่า ถ้ายังไม่ได้คู่คำสั่งค้างรอในตลาด	ทุกช่วง
Market (MP–MKT)	ซื้อขายที่ราคาดีที่สุดที่มีตอนนั้น ได้ของแน่ แต่ราคาไม่แน่	ช่วงตลาดเปิด
Market–to–Limit (MP–MTL)	จับคู่แบบ market ก่อน ส่วนที่เหลือกลายเป็น limit ที่ราคาล่าสุดที่จับคู่ได้	ช่วงตลาดเปิด
ATO	ซื้อขายที่ราคาเปิด	ช่วง pre–open
ATC	ซื้อขายที่ราคาปิด	ช่วง pre–close

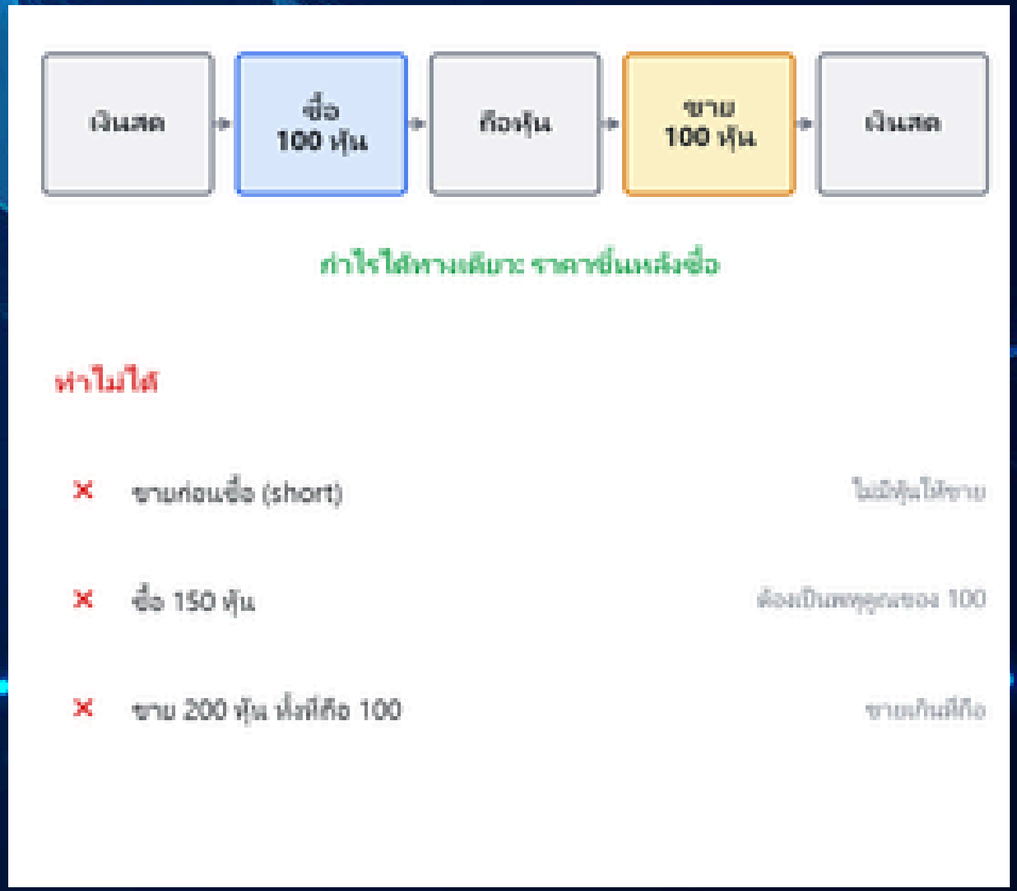
คำสั่งประเภท Limit และ Market

ใช้ในการแข่งขัน

1. **หุ้น SET50 อย่างเดียว**
   - **ซื้อก่อน แล้วขายทีหลัง ขายได้เฉพาะหุ้นที่ถืออยู่**
2. **ซื้อขายทีละ 100 หุ้น (พหุคูณของ 100)**
3. **ประเภทคำสั่ง**
   - **Market order = ไม่ต้องใส่ราคา ใช้ราคาล่าสุดของ tick + slippage (1 ช่วงราคา) นั้นแล้วจับคู่ทันที**
   - **Limit order = ตั้งราคาเองได้ และจ่ายราคานั้นก็ต่อเมื่อมีคนขายราคาที่ตั้งไว้ · ค้างได้ถึงจบวัน (ปรับกฎให้ส่งคำสั่งได้แค่ช่วงตลาดเปิดเท่านั้น)**
     - **Order pending หรือคำสั่งที่ส่งจะอยู่ถึงสิ้นวัน ณ วันที่ส่งคำสั่ง**

การแข่ง ใช้ได้แค่ Limit กับ Market

1. หุ้น SET50 อย่างเดียว
   - ซื้อก่อน แล้วขายทีหลัง ขายได้เฉพาะหุ้นที่ถืออยู่
2. ซื้อขายทีละ 100 หุ้น (พหุคูณของ 100)
3. ประเภทคำสั่ง
   - Market order = ไม่ต้องใส่ราคา ใช้ราคาล่าสุดของ tick + slippage (1 ช่วงราคา) นั้นแล้วจับคู่ทันที
   - Limit order = ตั้งราคาเองได้ และจ่ายราคานั้นก็ต่อเมื่อมีคนขายราคาที่ตั้งไว้ · ค้างได้ถึงจบวัน (ปรับกฎให้ส่งคำสั่งได้แค่ช่วงตลาดเปิดเท่านั้น)
     - Order pending หรือคำสั่งที่ส่งจะอยู่ถึงสิ้นวัน ณ วันที่ส่งคำสั่ง



Example สั่งซื้อตอนราคา 62.75



	Market	Limit 62.75
ราคาที่ได้	63.00	62.75
มูลค่า × 100 หุ้น	6,300.00	6,275.00
+ ค่าคอม 0.157%	+9.89	+9.85
+ VAT 7% ของค่าคอม	+0.69	+0.69
จำนวนจริง	6,310.58	6,285.54
ต่อหุ้น	63.1058	62.8554

Limit ถูกกว่า 25.04 บาท (0.40%) ต่อ 100 หุ้น
แลกกับความเสี่ยงที่จะไม่ได้หุ้น ถ้าไม่มีใครขายถึงราคา

ข้อจำกัด: คำสั่งซื้อขายจะค้างถึงแค่สิ้นวันเท่านั้น !!

What is algorithmic trading?



Algo Trading = การให้คอมพิวเตอร์ตัดสินใจซื้อขายตามกฎที่เราคิดขึ้นมา

Normal Trading VS ALGO Trading

ใช้ Algo Trading เพื่อตัดอารมณ์ออกจากจังหวะตัดสินใจ

ตัดสินใจด้วยอารมณ์

- ราคาขึ้นแรง → ตื่นเต้น กลัวตกรถ → ซื้อ ซึ่งตอนนั้นราคาแพงไปแล้ว
- ราคาลง → กลัว ทนไม่ไหว → ขาย ซึ่งตอนนั้นราคาถูกไปแล้ว
- ไม่มีจุดเข้าออกที่ตั้งไว้ก่อน ตัดสินใจใหม่ทุกครั้งตามสิ่งที่เห็นตรงหน้า
- ผล: ซื้อแพงขายถูกซ้ำสองรอบ รวม – 45.5%

ตัดสินใจซื้อ

ตัดสินใจขาย



Normal Trading VS ALGO Trading

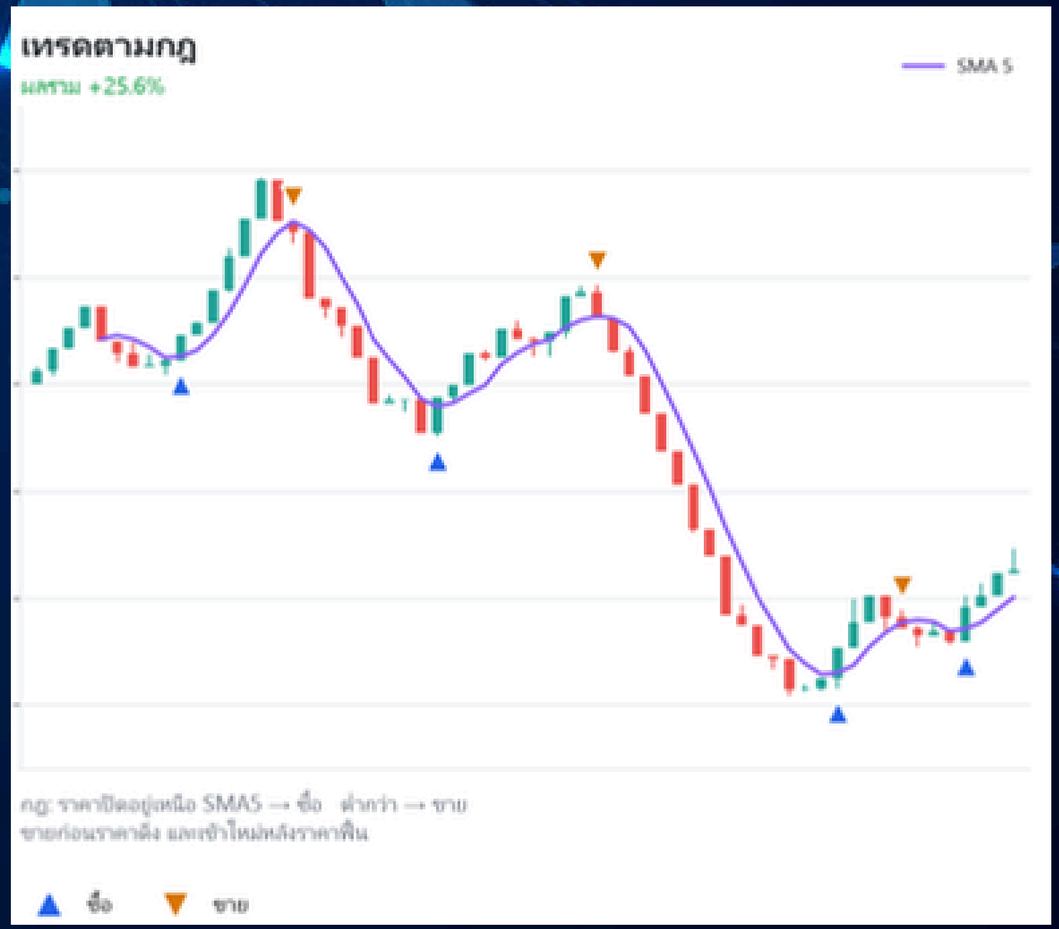
ใช้ Algo Trading เพื่อตัดอารมณ์ออกจากจังหวะตัดสินใจ

ตัดสินใจด้วยกฎ หรือ เงื่อนไข

- คำถามตอนกดซื้อขาย: "เงื่อนไขที่ตั้งไว้เกิดขึ้นหรือยัง"
- กฎตั้งไว้ก่อนเริ่มเทรด: เส้นม่วง (SMA5) ตัดขึ้นเหนือเส้นเทาเข้ม (SMA5) → ซื้อ · ตัดลง → ขาย
- ไม่สนว่าราคาขึ้นแรงแค่ไหน หรือตลาดน่ากลัวแค่ไหน
- ผล: ไม่ได้ซื้อต่ำสุดหรือขายสูงสุด แต่ขายก่อนราคาดิ่งจาก 105 ลงไปแถว 70 รวม + 2.6%

เข้าเงื่อนไขซื้อ

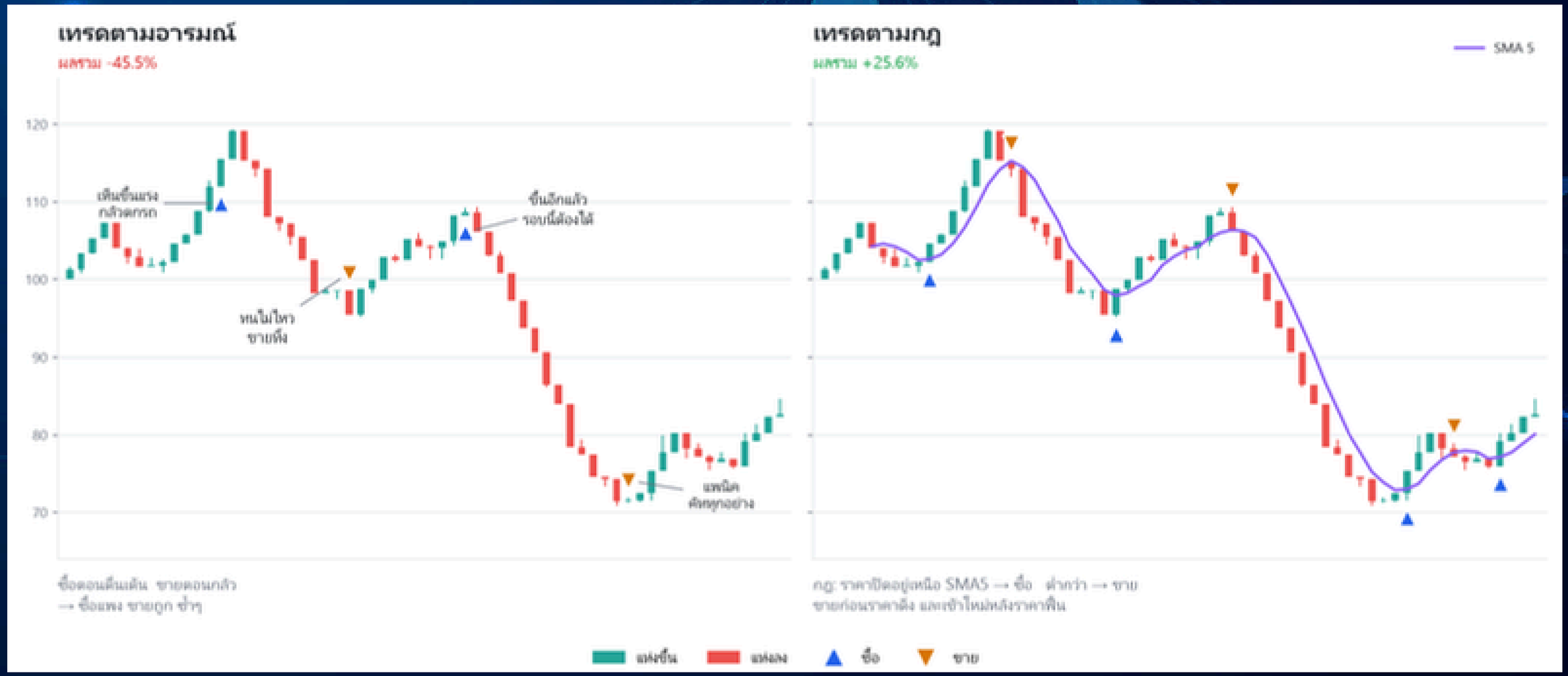
เข้าเงื่อนไขขาย



* SMA หรือ Simple Moving Average = เส้นค่าเฉลี่ยเคลื่อนที่แบบง่าย

Normal Trading VS ALGO Trading

ใช้ Algo Trading เพื่อตัดอารมณ์ออกจากจังหวะตัดสินใจ



Market information : CandleStick

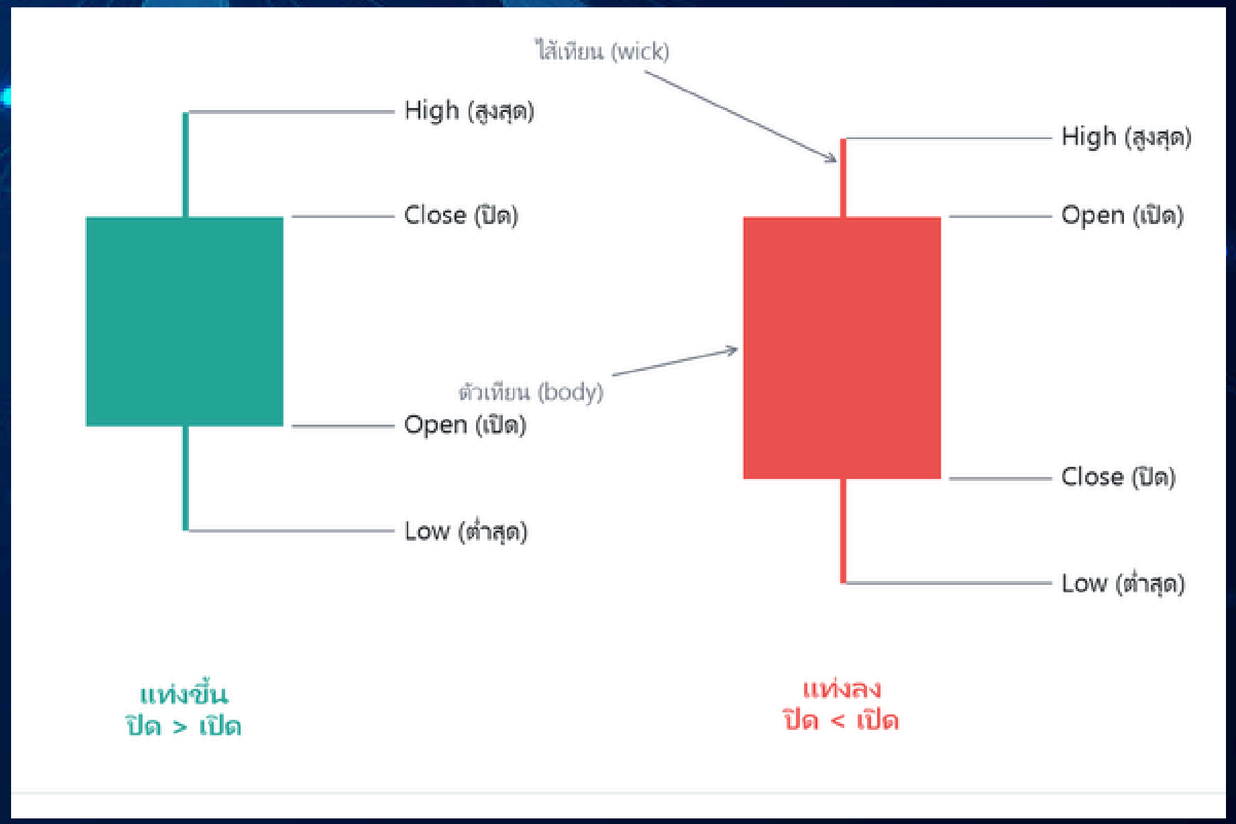
เท่งเทียนหรือ CandleStick

CandleStick

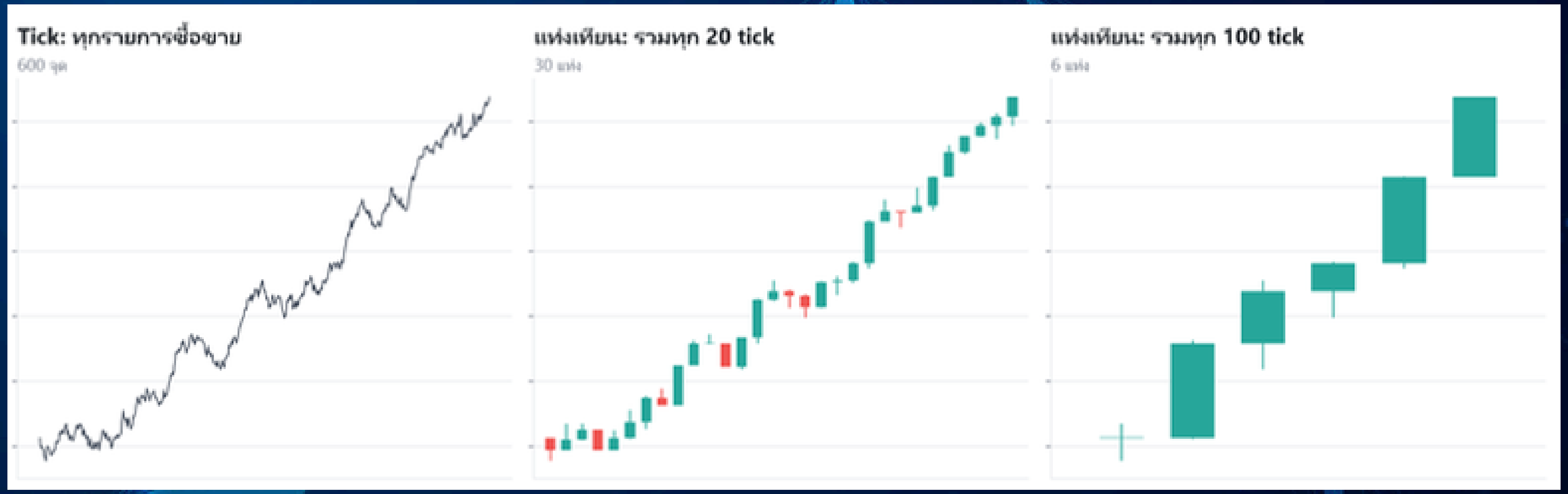
- ตัวเทียน **(ส่วนหนา)** = ช่วงระหว่างราคาเปิดกับราคาปิด
- ไส้เทียน (**เส้นบาง)** = ราคาสูงสุดและต่ำสุดที่เคยไปถึงในช่วงนั้น
- แท่งขึ้น: เปิดอยู่ล่าง ปิดอยู่บน · แท่งลง: เปิดอยู่บน ปิดอยู่ล่าง
- **ไส้ยาว** = ระหว่างทางราคาเหวี่ยงไปไกล แต่ถูกดึงกลับมาก่อนปิด

ข้อมูลจากตลาด

	Date	Open	High	Low	Close	Volume
1	2023-09-21	221.0	224.0	220.0	222.0	3713224
2	2023-09-22	224.0	231.0	223.0	228.0	11478617
3	2023-09-25	229.0	229.0	226.0	226.0	2954810



Market information : Tick, CandleStick และ Timeframe



รายการคำสั่ง

	ShareCode	TradeDateTime	LastPrice	Volume	Value	Flag
1	ADVANC	2026-09-24 09:58:43.000	346.0000000000	252800	87468800.0000000000	OPEN1_E
2	ADVANC	2026-09-24 09:58:44.000	346.0000000000	19000	6574000.0000000000	Buy
3	ADVANC	2026-09-24 09:58:44.000	346.0000000000	400	138400.0000000000	Buy
4	ADVANC	2026-09-24 09:58:44.000	346.0000000000	28100	9722600.0000000000	Buy
5	ADVANC	2026-09-24 09:58:44.000	346.0000000000	21300	7369800.0000000000	Buy
6	ADVANC	2026-09-24 09:58:44.000	346.0000000000	100	34600.0000000000	Buy

ในช่วงเวลา 1 นาที

	Timestamp	Open	High	Low	Close	Volume
1	2026-09-25 09:58:00	344.0	344.0	344.0	344.0	71900
2	2026-09-25 09:59:00	345.0	345.0	344.0	344.0	400
3	2026-09-25 10:00:00	344.0	345.0	344.0	344.0	4100
4	2026-09-25 10:01:00	344.0	345.0	344.0	345.0	3500

ในช่วงเวลา 15 นาที

	Timestamp	Open	High	Low	Close	Volume
1	2026-09-25 09:45:00	344.0	345.0	344.0	344.0	72300
2	2026-09-25 10:00:00	344.0	346.0	343.0	345.0	439000
3	2026-09-25 10:15:00	346.0	346.0	345.0	346.0	93600
4	2026-09-25 10:30:00	346.0	346.0	344.0	345.0	594900

ขั้นตอนการสร้าง ALGO

1. Design Strategy

ออกแบบกลยุทธ์

โดยใช้เครื่องมือทั้ง Fundamental และ Technical

การสร้างกลยุทธ์

1. **เทรดอะไร ตอนไหน (Universe / Time filter)**
   - **หุ้นรายตัวใน SET50 แท่งรายวัน**
2. **เข้าเมื่อไหร่ (Entry)**
   - **ราคาปิดขึ้นเหนือ SMA5**
3. **ออกเมื่อไหร่ (Exit / Stop loss)**
   - **Profit : ราคาปิดต่ำกว่า SMA5**
   - **Stop loss : ราคาปิดต่ำกว่า 3% ของราคาเข้าซื้อ**
4. **เข้าเท่าไหร่ (Position size)**
   - 100 หุ้น



ประเภทกลยุทธ์และ Indicator

Indicator คืออะไร และ กลยุทธ์ มีประเภทอะไรบ้าง

Indicator = ตัวเลขที่คำนวณจากราคาในอดีต

1. **ตัวเลข** : ย่อกราฟหลายแท่งให้เหลือเป็นตัวเลขผลลัพธ์จากการคำนวน
   - เช่น ราคาปิด > SMA5 = ทิศทางที่เห็นว่า ณ ตอนนี้ราคาอยู่สูงกว่าค่าเฉลี่ยเมื่อ 5 แท่งเทียนก่อนหน้า
2. **ตัวเลขท้ายชื่อ (5, 14, 20)** : มองย้อนกี่แท่ง: สั้น = ไวแต่หลอกบ่อย · ยาว = นิ่งแต่ช้า
   - เช่น SMA 5 มองย้อนไป 5 แท่ง, SMA 10 มองย้อนไป 10 แท่ง
3. **Indicator ไม่ใช่กลยุทธ์** : มันแค่ตัวเลขผลลัพธ์ของการคำนวน
4. **กลยุทธ์ :** การนำกฎมาเทียบกับผลลัพธ์ เพื่อบอกว่าผลลัพธ์แบบไหนให้ซื้อ แบบไหนให้ขาย (เช่น ปิดเหนือ SMA5 → ซื้อ)



* SMA หรือ Simple Moving Average = เส้นค่าเฉลี่ยเคลื่อนที่แบบง่าย

ประเภทกลยุทธ์และ Indicator

Indicator คืออะไร และ กลยุทธ์ มีประเภทอะไรบ้าง

SMA5	ตอนนี้ขาขึ้นมั้ย ?
RSI 14	ขึ้นลงแรงไปมั้ย ?
ATR 14	ตอนนี้ราคาแกว่งไปมั้ย ?
Supertrend	ราคาหลุดแถบเดิมมั้ย ?

ประเภทกลยุทธ์และ Indicator



SMA5 ⟶ **ตอนนี้ขาขึ้นมั้ย ?**

ประเภทกลยุทธ์และ Indicator



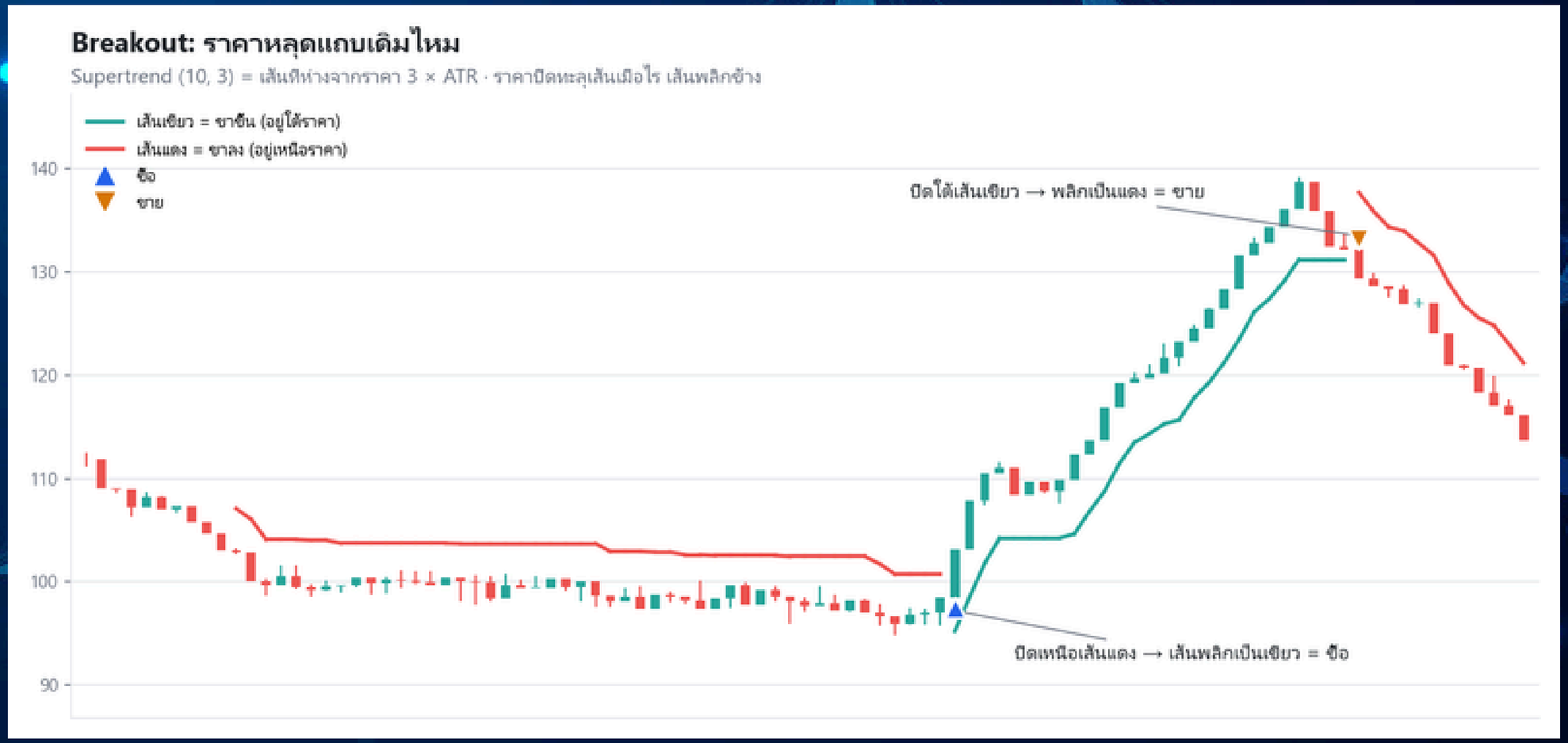
RSI 14 ⟶ **ขึ้นลงแรงไปมั้ย ?**

ประเภทกลยุทธ์และ Indicator



ATR 14 ⟶ ตอนนี้ราคาแกว่งไปมั้ย ?

ประเภทกลยุทธ์และ Indicator



Supertrend ⟶ ราคาหลุดแถบเดิมมั้ย ?

ไม่มี Indicator ไหนชนะทุกสภาพตลาด



2.Backtest

Backtest และการวัดผล



Backtest และการวัดผล

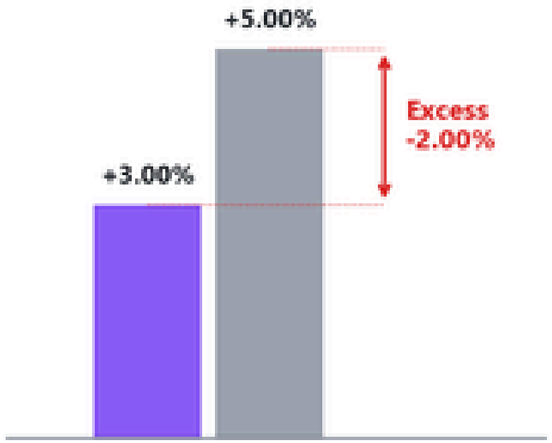


ตัวเลข	ตอบคำถาม	จากภาพ
1. Equity curve	เงินเปลี่ยนไปยังไงตลอดทาง	กราฟเส้นเขียว "PnL สะสม"
2. Total return	สุดท้ายได้หรือเสียเท่าไหร่	PnL ทั้งหมด **+1,635.50 บาท (+16.36%)**
3. Max drawdown	เคยติดลบจากจุดสูงสุดมากสุดกี่ %	ขาดทุนสะสมสูงสุด **416.00 บาท (3.63%)**
4. Win rate	ชนะกี่ไม้จากทั้งหมด	การซื้อขายที่มีกำไร **39.29% (22/56)**
5. Profit factor	กำไรรวม ÷ ขาดทุนรวม	อัตราส่วนกำไรต่อขาดทุน **1.933**

ตัวเลขที่ใช้ตัดสินการแข่งจริง

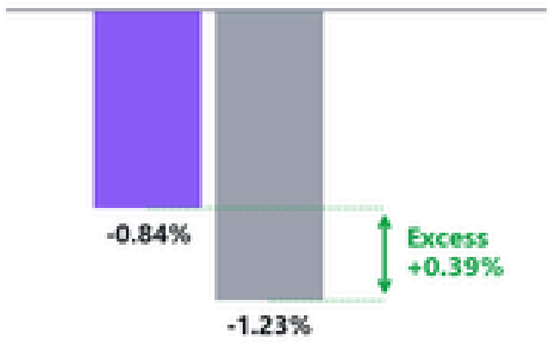
1 · Excess Return = พอร์ต − SET50

การแข่งวัดว่าชนะตลาดแค่ไหน ไม่ได้วัดกำไรสิบ



ตลาดขึ้น 5% · เราได้ 3%

กำไร แต่แพ้ตลาด



ผลจริง 7 วัน (หัวข้อ 10)

ขาดทุน แต่ชนะตลาด

พอร์ตกลยุทธ์ SET50 ช่วงเดียวกัน

2 · ต้องผ่านเกณฑ์ก่อน แล้วค่อยจัดอันดับ

ไม่ผ่านข้อเดียว = ไม่ถูกจัดอันดับ ต่อให้ Excess Return สูงแค่ไหน

เกณฑ์ผ่าน (ต้องครบทุกข้อ)

- ✓ **Max Drawdown < 1.5 × ของ SET50**
  ไม่เสียเกินตลาดมาก
- ✓ **Round-Trip ≥ 5 รอบ**
  ต้องซื้อแล้วขายครบจริง
- ✓ **Average Exposure ≥ 50%**
  ถือหุ้นตลอดๆ ไม่ผ่าน
- ✓ **Expected Payoff > 0**
  เฉลี่ยต่อไม้ต้องได้กำไร

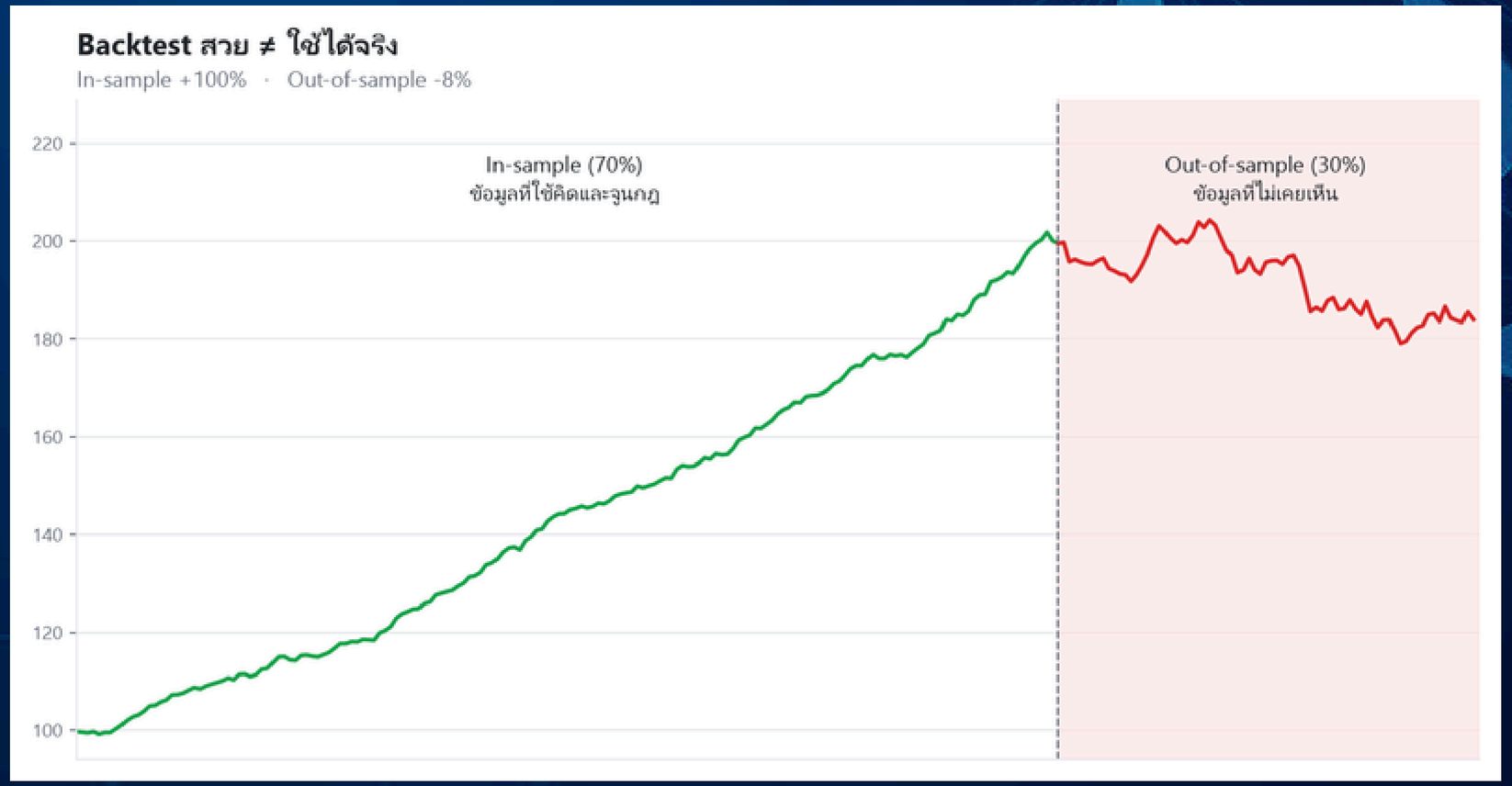
จัดอันดับ Best Alpha

1. **Excess Return**
   ตัวหลัก
2. **Calmar Ratio**
   ถ้าข้อ 1 เสมอ
3. **Total Return**
   ถ้าข้อ 2 เสมอ

Backtest และการวัดผล



กับดักของ Backtest



Overfitting เกินไป

พยายามปรับให้ Indicator เข้ากับข้อมูลตลาดตอนซ้อมเกินไปจนใช้กับตลาดจริงไม่ได้

- SMA3 ตัด SMA8: 0.0%
- SMA5 ตัด SMA10: -7.6%
- SMA5 ตัด SMA20: -19.1%
- SMA10 ตัด SMA20: -26.0%
- ราคาปิด vs SMA5: +25.6%

พยายามปรับให้ Indicator เข้ากับข้อมูลตลาดตอนซ้อมเกินไปจนใช้กับตลาดจริงไม่ได้

3.Forward test by TradeSim

TradeSim — What You Need Before You Start

ก่อนที่จะเข้าร่วมการแข่งขัน ให้แน่ใจว่าคุณมีความรู้พื้นฐานตามหัวข้อด้านล่าง และติดตั้งเครื่องมือที่จำเป็นเรียบร้อยแล้ว

- Python 3.13 ONLY
- VS Code with the Jupyter Notebook extension
- git + GitHub account

Download tradeSim via this link: TradeSim

Tools + รัน Example_strategy

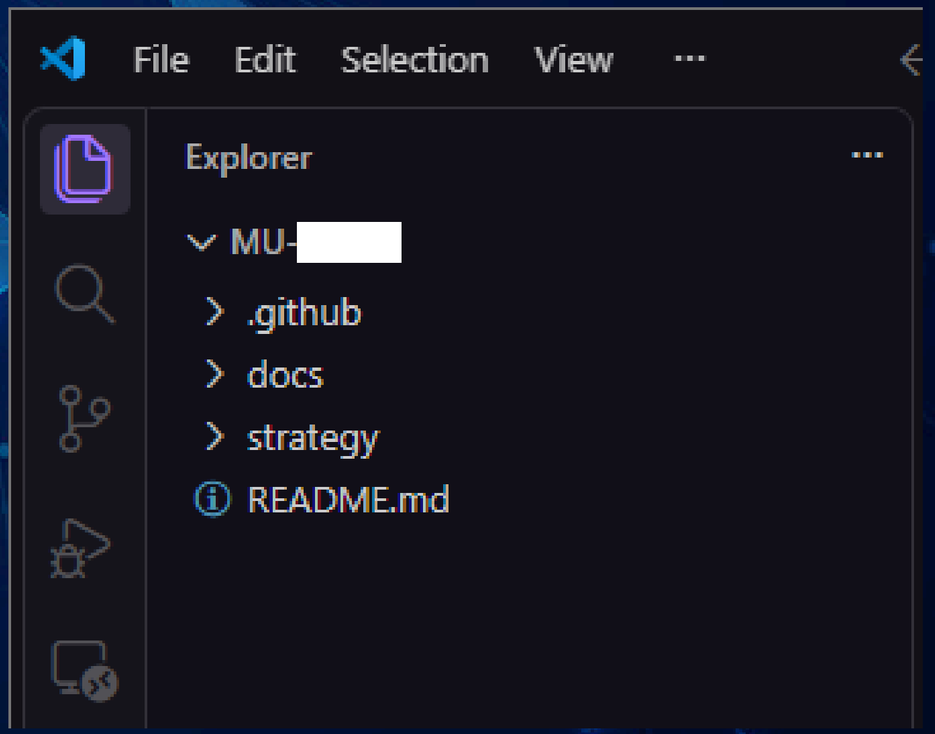
1. Installation

Git clone repository ลงบน PC ที่จะใช้ทดสอบ

```
git clone https://github.com/KGI-ALGO-trading-2026/<UNI>-<TEAM>.git
cd <UNI>-<TEAM>
```

<UNI> = มหาลัยของผู้เข้าแข่งขัน เช่น MU, TU, CU, KMUTT, KMITL, KMUTNB

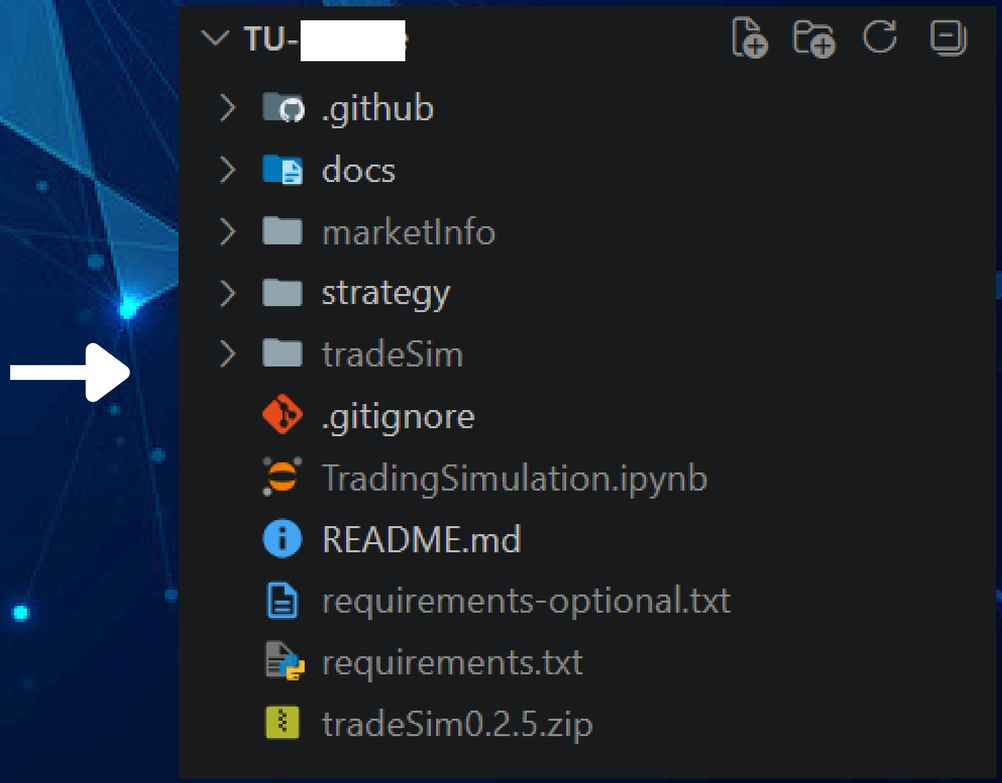
<TEAM> = ชื่อทีมของน้องๆ เช่น Quant01, ALGO02

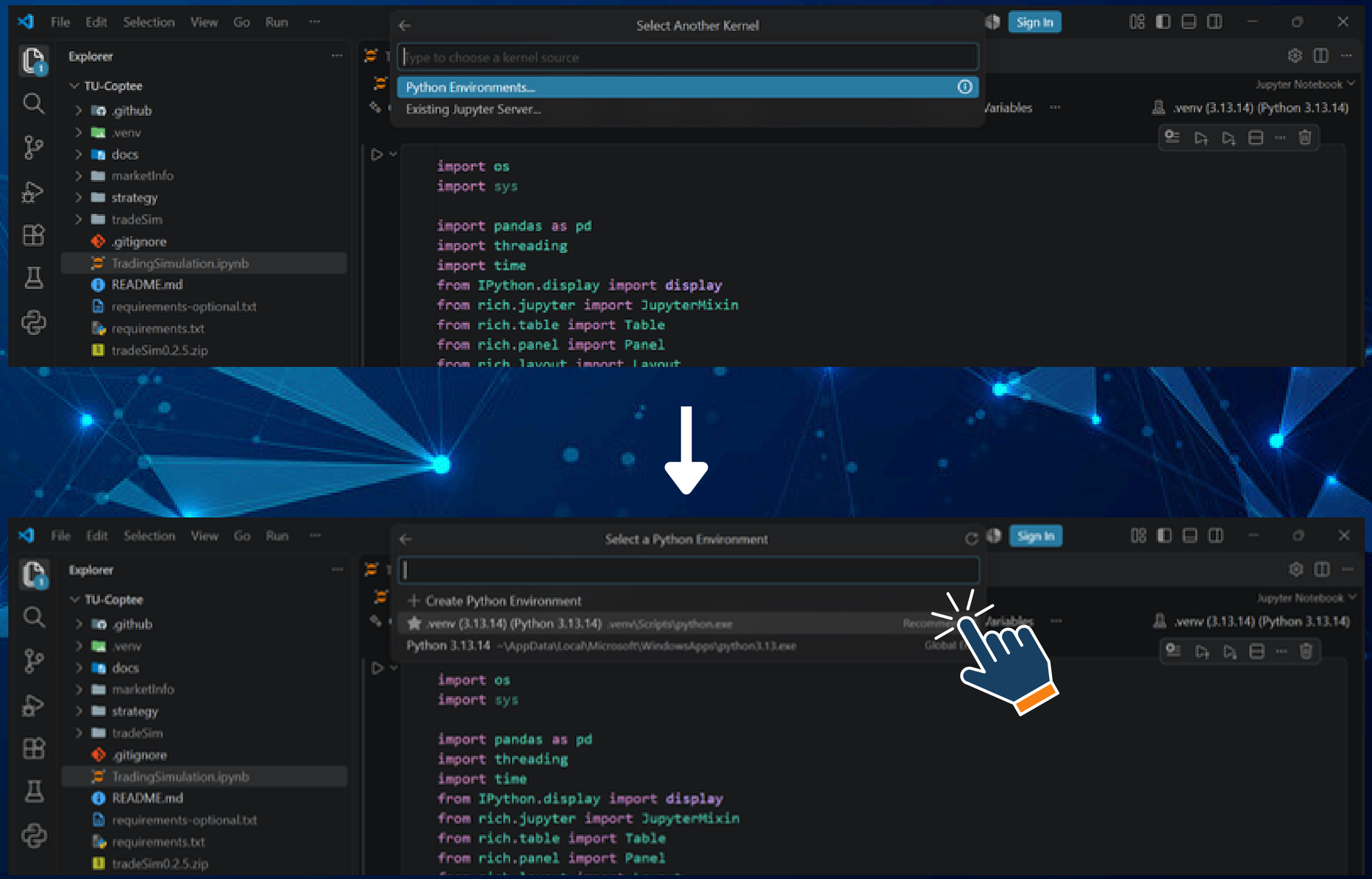


1. Installation

1. โหลด tradeSim0.2.1.zip จากลิงก์ Google Drive
2. แตกทับลงในโฟลเดอร์นี้เลย ไม่ต้องแยกโฟลเดอร์
3. สร้าง venv แล้วทำการ install library ของ TradeSim

```
python -m venv .venv
.venv\Scripts\activate
pip install -r requirements.txt
```




File Edit Selection View Go Run
Select Another Kernel
type to choose a kernel source
Python Environments...
Existing Jupyter Server...
Sign In
Explorer
TU-Coptee
.github
.venv
docs
marketInfo
strategy
tradeSim
.gitignore
TradingSimulation.ipynb
README.md
requirements-optional.txt
requirements.txt
tradeSim0.2.5.zip
Jupyter Notebook
.venv (3.13.14) (Python 3.13.14)
import os
import sys
import pandas as pd
import threading
import time
from IPython.display import display
from rich.jupyter import JupyterMixin
from rich.table import Table
from rich.panel import Panel
Select a Python Environment
Create Python Environment
.venv (3.13.14) (Python 3.13.14) .venv\Scripts\python.exe
Python 3.13.14 ~\AppData\Local\Microsoft\WindowsApps\python3.13.exe

พัก BREAK

How does it work?

1. How does it work?

เปิด TradingSimulation.ipynb แล้วรันจากบนลงล่าง ยังไม่ต้องแก้อะไร

1	import library
2	เลือกไฟล์ tick ของวันที่จะซ้อม
3	ตั้งชื่อทีม + สร้างไฟล์กลยุทธ์
4	import กลยุทธ์เข้ามา
5	with_visual ดูตารางหุ้นวิ่งระหว่างวัน
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่
9	รันหลายวันต่อกัน

1. How does it work?

เปิด TradingSimulation.ipynb แล้วรันจากบนลงล่าง ยัง

ไม่ต้องแก้อะไร

1	import library
2	เลือกไฟล์ tick ของวันที่จะซ้อม
3	ตั้งชื่อทีม + สร้างไฟล์กลยุทธ์
4	import กลยุทธ์เข้ามา
5	with_visual ดูตารางหุ้นวิ่งระหว่างวัน
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่
9	รันหลายวันต่อกัน

```
import os
import sys

import pandas as pd
import threading
from IPython.display import display
from rich.jupyter import JupyterMixin
from rich.table import Table
from rich.panel import Panel
from rich.layout import Layout
import importlib
import shutil
import re

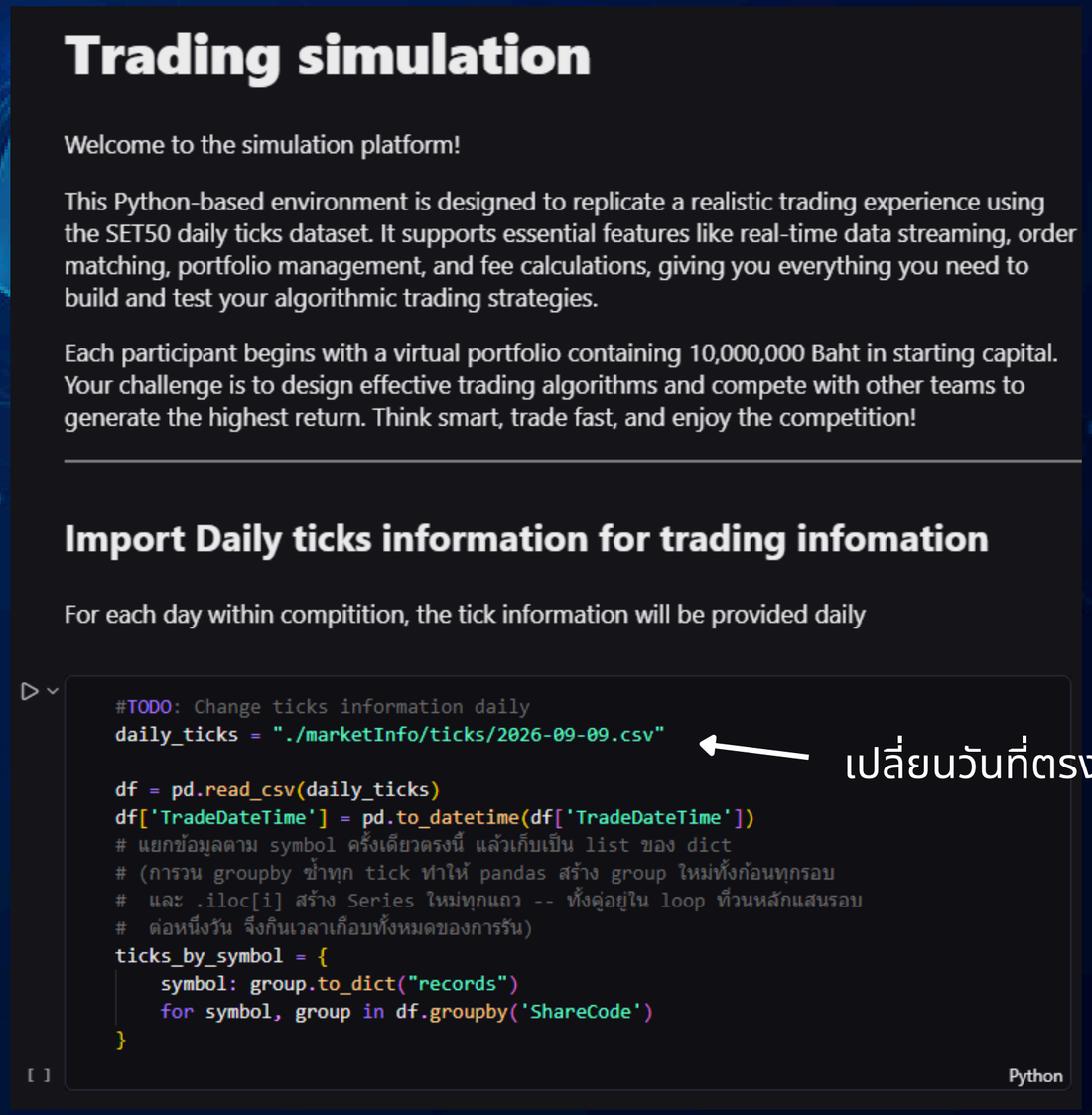
# Import trading library
from tradeSim import TradeSim
from tradeSim import StrategyHandler
```

✓ 0.0s

1. How does it work?

เปิด TradingSimulation.ipynb แล้วรันจากบนลงล่าง ยังไม่ต้องแก้อะไร

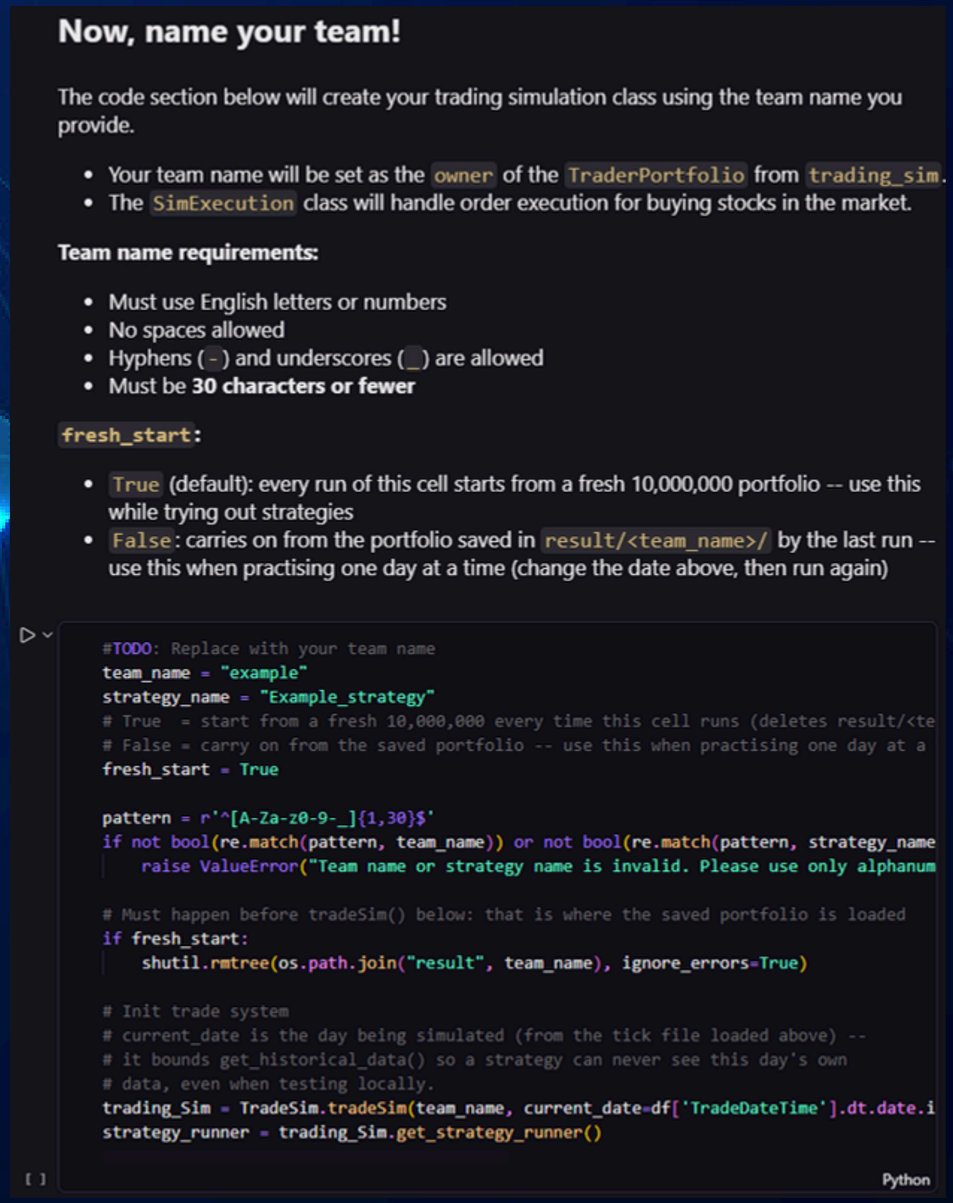
1	import library
2	**เลือกไฟล์ tick ของวันที่จะซ้อม**
3	ตั้งชื่อทีม + สร้างไฟล์กลยุทธ์
4	import กลยุทธ์เข้ามา
5	with_visual ดูตารางหุ้นวิ่งระหว่างวัน
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่
9	รันหลายวันต่อกัน



1. How does it work?

เปิด TradingSimulation.ipynb แล้วรันจากบนลงล่าง ยังไม่ต้องแก้อะไร

1	import library
2	เลือกไฟล์ tick ของวันที่จะซ้อม
3	**ตั้งชื่อทีม + สร้างไฟล์กลยุทธ์**
4	import กลยุทธ์เข้ามา
5	with_visual ดูตารางหุ้นวิ่งระหว่างวัน
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่
9	รันหลายวันต่อกัน



Now, name your team!

The code section below will create your trading simulation class using the team name you provide.

- Your team name will be set as the `owner` of the `TraderPortfolio` from `trading_sim`.
- The `SimExecution` class will handle order execution for buying stocks in the market.

Team name requirements:

- Must use English letters or numbers
- No spaces allowed
- Hyphens (`-`) and underscores (`_`) are allowed
- Must be **30 characters or fewer**

`fresh_start`:

- `True` (default): every run of this cell starts from a fresh 10,000,000 portfolio -- use this while trying out strategies
- `False`: carries on from the portfolio saved in `result/<team_name>/` by the last run -- use this when practising one day at a time (change the date above, then run again)

```
#TODO: Replace with your team name
team_name = "example"
strategy_name = "Example_strategy"
# True  = start from a fresh 10,000,000 every time this cell runs (deletes result/<te
# False = carry on from the saved portfolio -- use this when practising one day at a
fresh_start = True

pattern = r'^[A-Za-z0-9-_]{1,30}$'
if not bool(re.match(pattern, team_name)) or not bool(re.match(pattern, strategy_name
    raise ValueError("Team name or strategy name is invalid. Please use only alphanum

# Must happen before tradeSim() below: that is where the saved portfolio is loaded
if fresh_start:
    shutil.rmtree(os.path.join("result", team_name), ignore_errors=True)

# Init trade system
# current_date is the day being simulated (from the tick file loaded above) --
# it bounds get_historical_data() so a strategy can never see this day's own
# data, even when testing locally.
trading_Sim = TradeSim.tradeSim(team_name, current_date=df['TradeDateTime'].dt.date.i
strategy_runner = trading_Sim.get_strategy_runner()
```

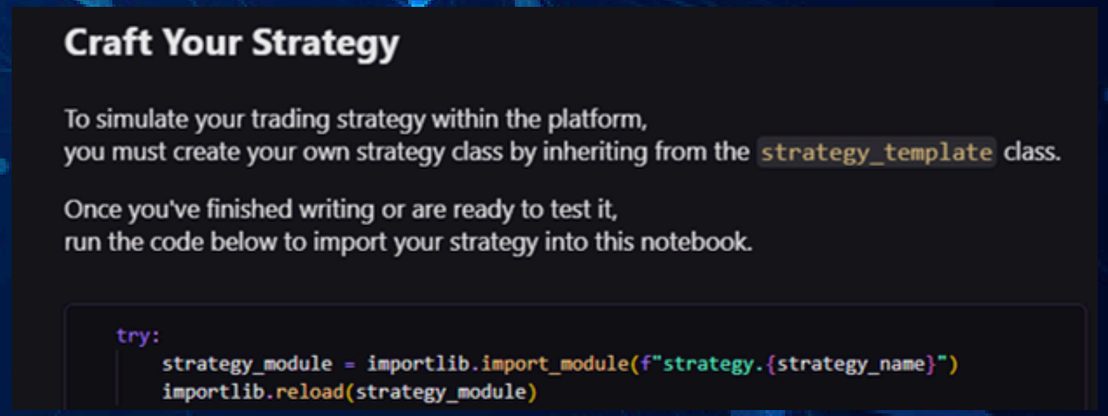
[ ] Python

1. How does it work?

เปิด TradingSimulation.ipynb แล้วรันจากบนลงล่าง ยัง

ไม่ต้องแก้อะไร

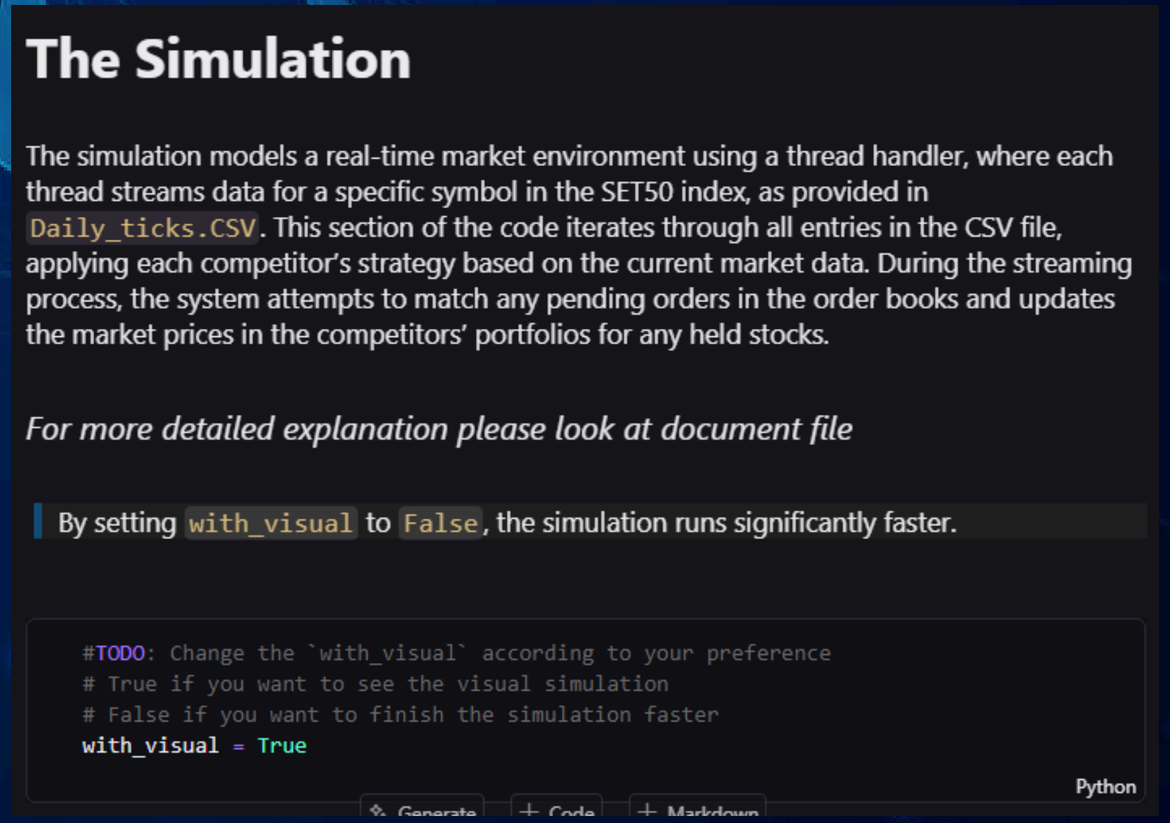
1	import library
2	เลือกไฟล์ tick ของวันที่จะซ้อม
3	ตั้งชื่อทีม + สร้างไฟล์กลยุทธ์
4	import กลยุทธ์เข้ามา
5	with_visual ดูตารางหุ้นวิ่งระหว่างวัน
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่
9	รันหลายวันต่อกัน



1. How does it work?

เปิด TradingSimulation.ipynb แล้วรันจากบนลงล่าง ยังไม่ต้องแก้อะไร

1	import library
2	เลือกไฟล์ tick ของวันที่จะซ้อม
3	ตั้งชื่อทีม + สร้างไฟล์กลยุทธ์
4	import กลยุทธ์เข้ามา
5	with_visual ดูตารางหุ้นวิ่งระหว่างวัน
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่
9	รันหลายวันต่อกัน



1. How does it work?

เปิด TradingSimulation.ipynb แล้วรันจากบนลงล่าง ยังไม่ต้องแก้อะไร

1	import library
2	เลือกไฟล์ tick ของวันที่จะซ้อม
3	ตั้งชื่อทีม + สร้างไฟล์กลยุทธ์
4	import กลยุทธ์เข้ามา
5	with_visual ดูตารางหุ้นวิ่งระหว่างวัน
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่
9	รันหลายวันต่อกัน

Market Stream

Symbol	Last Price	Volume	Flag
ADVANC	354.00	682400	ATC
AOT	62.25	1542100	ATC
AWC	3.10	4731600	ATC
BANPU	15.30	1571200	ATC
BBL	192.00	100	Buy
BCP	58.00	1503100	ATC
BDMS	19.80	100	Buy
BEM	6.75	2521700	ATC
BH	196.50	254900	ATC
BJC	16.50	440100	ATC
CCET	9.35	5767200	ATC
COM7	29.25	1390800	ATC
CPALL	45.75	1900	Buy
CPF	23.70	1193500	ATC
CPN	64.50	759500	ATC
CRC	29.50	1585100	ATC
DELTA	271.00	100	Sell
EGCO	133.00	115100	ATC
GPSC	51.25	2090000	ATC
GULF	62.75	900	Sell
HMPRO	6.25	7278600	ATC
IVL	28.50	10000	Buy
KBANK	244.00	1159900	ATC
KKP	114.50	590900	ATC
KTC	37.00	1030900	ATC
LH	3.62	3526900	ATC
MINT	22.00	15000	Buy
MRDIYT	9.10	502100	ATC
MTC	30.00	1210500	ATC

Portfolio: example

Cash Balance	Symb…	Actu… Vol	Buy Price	Mark… Price	Aver… Cost	Amou… Cost	Mark… Value	Unre…	Unre… %	Real…	Buy Time
8,534,989.86	BBL	100	192.…	192.…	192.…	1928…	1920…	-82.…	-0.4…	0.00	2026-… 14:34…

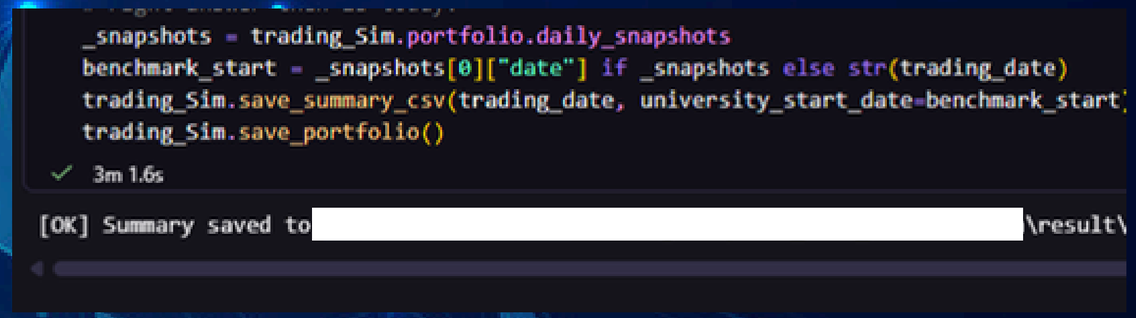
Metrics	
Return (%)	-14.…
Benchmark Return …	0.00%
Excess Return (%)	-14.…
Max Drawdown (%)	0.00%
Buy Orders	8830
Sell Orders	8829
Round-Trips	8829
Profitable Round-…	0
Win Rate	0.00%
Expected Payoff	-163…
Avg Gain	0.00
Avg Loss	163.…
Avg Exposure	0.22%

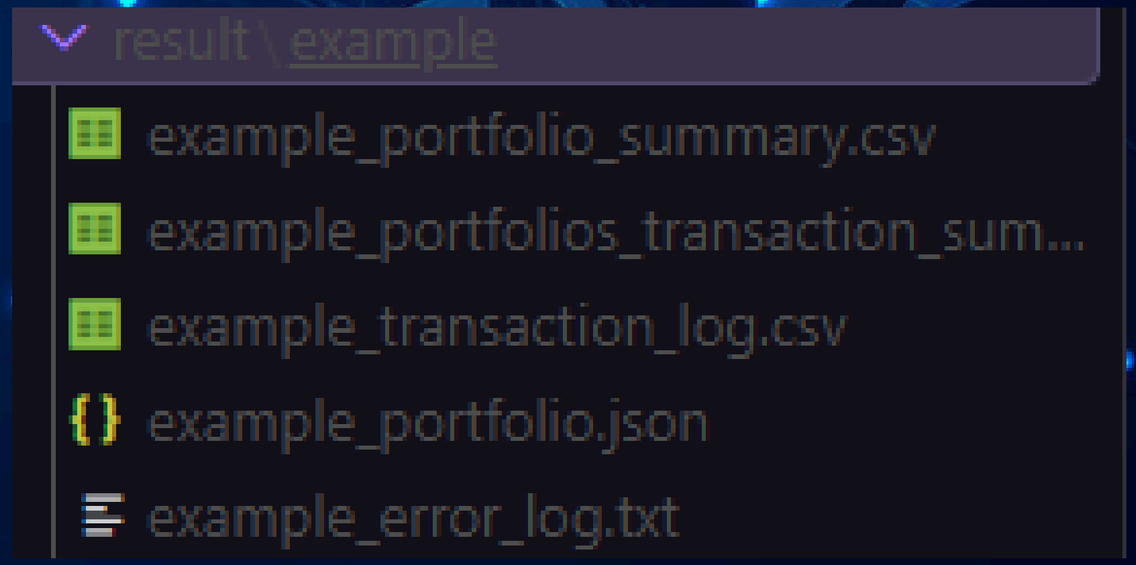
1. How does it work?

เปิด TradingSimulation.ipynb แล้วรันจากบนลงล่าง ยัง

ไม่ต้องแก้อะไร

1	import library
2	เลือกไฟล์ tick ของวันที่จะซ้อม
3	ตั้งชื่อทีม + สร้างไฟล์กลยุทธ์
4	import กลยุทธ์เข้ามา
5	with_visual ดูตารางหุ้นวิ่งระหว่างวัน
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่
9	รันหลายวันต่อกัน





1. How does it work?

เปิด TradingSimulation.ipynb แล้วรันจากบนลงล่าง ยัง

ไม่ต้องแก้อะไร

1	import library
2	เลือกไฟล์ tick ของวันที่จะซ้อม
3	ตั้งชื่อทีม + สร้างไฟล์กลยุทธ์
4	import กลยุทธ์เข้ามา
5	with_visual ดูตารางหุ้นวิ่งระหว่างวัน
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่
9	รันหลายวันต่อกัน

```
portInfo = strategy_runner.get_portfolio_info()
stockInfo = strategy_runner.get_all_stocks_info()
pd.DataFrame([portInfo]).T
```

✓ 0.0s

	0
Owner	example
Number of Stocks	1
Total Cost	19282.34
Unrealized P&L	-82.34
Unrealized %	-0.43
Realized P&L	-1445727.8
Cash Balance	8534989.86
Net Asset Value	8554189.86
Max NAV	8554189.86
Min NAV	8554189.86
Max Drawdown (%)	0
Relative Drawdown	0.0
Calmar Ratio	0
Previous Day Max DD (%)	0
Buy Order Count	8830
Sell Order Count	8829
Round-Trip Count	8829
Profitable Round-Trips	0
Win Rate	0.0
Expected Payoff	-163.75
Avg Gain	0.0
Avg Loss	163.75
Average Exposure	0.0022
Return rate	-14.458101
Benchmark Return	0.0
Benchmark Max DD	0.0
Excess Return	-14.458101

1. How does it work?

เปิด TradingSimulation.ipynb แล้วรันจากบนลงล่าง ยังไม่ต้องแก้อะไร

1	import library
2	เลือกไฟล์ tick ของวันที่จะซ้อม
3	ตั้งชื่อทีม + สร้างไฟล์กลยุทธ์
4	import กลยุทธ์เข้ามา
5	with_visual ดูตารางหุ้นวิ่งระหว่างวัน
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่
9	รันหลายวันต่อกัน

Run Several Days in a Row

Runs the strategy chosen above over several trading days back to back, the way the real daily run does: each day loads the portfolio left by the day before, so a position bought on one day can be sold on a later one.

Run the cells up to **Craft Your Strategy** first (they load the strategy). No visual mode here — it runs headless and prints one row per day.

```
#TODO: The days to run, oldest first
trading_days = ["2026-09-09", "2026-09-10", "2026-09-11", "2026-09-14",
                "2026-09-15", "2026-09-16", "2026-09-17"]
# True  = delete result/<team_name>/ first, so the run starts from a fresh 10,000,000
# False = carry on from whatever portfolio is already saved there
fresh_start = True

if fresh_start:
    shutil.rmtree(os.path.join("result", team_name), ignore_errors=True)

strategy_dir = os.path.dirname(strategy_module.__file__)
day_rows = []

for day in trading_days:
    day_df = pd.read_csv(f"./marketInfo/ticks/{day}.csv")
    day_df['TradeDateTime'] = pd.to_datetime(day_df['TradeDateTime'])
    day_ticks = {symbol: group.to_dict("records") for symbol, group in day_df.groupby('ShareCode')}
    trading_date = day_df['TradeDateTime'].dt.date.iloc[0]

    # A new tradeSim per day loads result/<team_name>/<team_name>_portfolio.json,
    # which is what carries cash and holdings over from the previous day.
    trading_Sim = TradeSim.tradeSim(team_name, current_date=trading_date)
    strategy_runner = trading_Sim.get_strategy_runner()
    handlers = {
        symbol: StrategyHandler.StrategyHandler(strategy_class, strategy_runner, strategy_dir)
        for symbol in day_ticks
    }
```

```
[INFO] Created new portfolio for 'example'
[OK] Summary saved to c:[redacted]\result\example\example_portfolios_transaction_summar
[INFO] Loaded existing portfolio from 'result\example\example_portfolio.json'
[OK] Summary saved to c:[redacted]\result\example\example_portfolios_transaction_summar
[INFO] Loaded existing portfolio from 'result\example\example_portfolio.json'
[OK] Summary saved to c:[redacted]\result\example\example_portfolios_transaction_summar
[INFO] Loaded existing portfolio from 'result\example\example_portfolio.json'
```

1. How does it work?

เปิด TradingSimulation.ipynb แล้วรันจากบนลงล่าง ยังไม่ต้องแก้อะไร

1	import library
2	เลือกไฟล์ tick ของวันที่จะซ้อม
3	ตั้งชื่อทีม + สร้างไฟล์กลยุทธ์
4	import กลยุทธ์เข้ามา
5	with_visual ดูตารางหุ้นวิ่งระหว่างวัน
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่
9	รันหลายวันต่อกัน

	Date	Cash	Return (%)	Benchmark Return (%)	Excess Return (%)
0	2026-09-09	8534989.86	-14.4581	0.0000	-14.4581
1	2026-09-10	7413810.17	-25.2990	-0.1163	-25.1827
2	2026-09-11	6042154.97	-39.3726	-0.7820	-38.5905
3	2026-09-14	4730460.46	-52.5039	-1.1717	-51.3322
4	2026-09-15	3637660.81	-63.6057	-1.2557	-62.3500
5	2026-09-16	34474.72	-99.6375	-2.3729	-97.2646
6	2026-09-17	692.66	-99.9931	-1.2271	-98.7660

Excess Return (%)	Buy Orders	Sell Orders	Round-Trips	Holding
-14.4581	8830	8829	8829	BBL 100
-25.1827	14935	14931	14931	AWC 100, BANPU 100, BBL 100, ADVANC 100
-38.5905	23694	23691	23691	AWC 100, BANPU 100, BBL 100
-51.3322	32199	32198	32198	BBL 100
-62.3500	38378	38376	38376	BANPU 100, AWC 100
-97.2646	51331	51329	51329	AWC 100, BANPU 100
-98.7660	52622	52622	52622	-

กลยุทธ์แรก: โครง → ซื้อ → ขาย

2. You first Strategy

สร้าง Strategy ตัวแรก

1	Strategy เปล่าๆ
2	ส่งคำสั่งซื้อครั้งแรก
3	ซื้อวันนี้ ขายวันหลัง
4	ใช้ข้อมูลย้อนหลังในการทำ Strategy + Indicator SMA
5	แนะนำ Library สำหรับทำ Indicator
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่
9	รันหลายวันต่อกัน

2. You first Strategy

สร้าง Strategy ตัวแรก

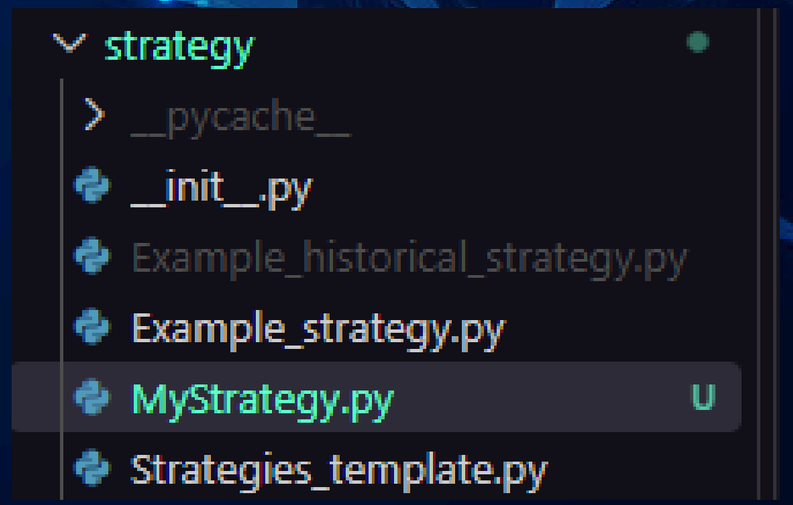
1	Strategy เปล่าๆ
2	ส่งคำสั่งซื้อครั้งแรก
3	ซื้อวันนี้ ขายวันหลัง
4	ใช้ข้อมูลย้อนหลังในการทำ Strategy + Indicator SMA
5	แนะนำ Library สำหรับทำ Indicator
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่
9	รันหลายวันต่อกัน

```python
from strategy.Strategies_template import Strategy_template


class MyStrategy(Strategy_template):
    def __init__(self, handler):
        super().__init__("MyTeam", "MyStrategy", handler, timeframe="1D")

    def on_data(self, bar):
        pass

```



2. You first Strategy

สร้าง Strategy ตัวแรก

1	Strategy เปล่าๆ
2	**ส่งคำสั่งซื้อครั้งแรก**
3	ซื้อวันนี้ ขายวันหลัง
4	ใช้ข้อมูลย้อนหลังในการทำ Strategy + Indicator SMA
5	แนะนำ Library สำหรับทำ Indicator
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่
9	รันหลายวันต่อกัน

strategy > MyStrategy.py > MyStrategy

```python
from strategy.Strategies_template import Strategy_template


class MyStrategy(Strategy_template):
    TAKE_PROFIT = 1.01

    def __init__(self, handler):
        super().__init__("CopterTeam", "MyStrategy", handler, timeframe="1D")
        self.buy_price = None

    def on_data(self, bar):
        if bar["ShareCode"] != "AOT":
            return
        price = bar["Close"]

        if self.buy_price is None:
            result = self.handler.create_order_to_limit(100, price, "Buy", "AOT")
            if "successfully" in str(result):
                self.buy_price = price
            return

        if price >= self.buy_price * self.TAKE_PROFIT:
            if self.handler.check_port_has_stock("AOT", 100):
                self.handler.create_order_to_limit(100, price, "Sell", "AOT")
```

Portfolio: CopterTeam

Cash Balance	Symb…	Actu… Vol	Buy Price	Mark… Price	Aver… Cost	Amou… Cost	Mark… Value	Unre…	Unre… %	Reali…	Buy Time
9,993,664.37	AOT	100	63.36	62.25	63.36	6335…	6225…	-110…	-1.7…	0.00	2026… 09:5…

Metrics	
ROI (%)	-0.0…
Max Drawdown (%)	0.00%
Buy Orders	1
Sell Orders	0
Round-Trips	0
Wins	0
Win Rate	0.00%
Expected Payoff	0.00
Avg Exposure	0.00%

2. You first Strategy

สร้าง Strategy ตัวแรก

1	Strategy เปล่าๆ
2	ส่งคำสั่งซื้อครั้งแรก
3	ซื้อวันนี้ ขายวันหลัง
4	ใช้ข้อมูลย้อนหลังในการทำ Strategy + Indicator SMA
5	แนะนำ Library สำหรับทำ Indicator
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่
9	รันหลายวันต่อกัน

strategy > MyStrategy.py > MyStrategy > __init__

```python
from strategy.Strategies_template import Strategy_template


class MyStrategy(Strategy_template):
    SYMBOL = "KBANK"
    TAKE_PROFIT = 1.02                      # ขายเมื่อราคาสูงกว่าต้นทุน 2%

    def __init__(self, handler):
        super().__init__("CopterTeam", "MyStrategy", handler, timeframe="1D")
        self.ordered_today = False          # วันละ 1 คำสั่ง กันสั่งรัว

    def on_data(self, bar):
        if bar["ShareCode"] != self.SYMBOL or self.ordered_today:
            return
        price = bar["Close"]

        if not self.handler.check_port_has_stock(self.SYMBOL, 100):
            # ยังไม่มีหุ้น → ซื้อ
            result = self.handler.create_order_to_limit(100, price, "Buy", self.SYMBOL)
        elif price >= self.cost() * self.TAKE_PROFIT:
            # มีหุ้นอยู่ (อาจซื้อไว้ตั้งแต่วันก่อน) และถึงเป้า → ขาย
            result = self.handler.create_order_to_limit(100, price, "Sell", self.SYMBOL)
        else:
            return                          # มีหุ้นแต่ยังไม่ถึงเป้า → ถือต่อ

        if "successfully" in str(result):
            self.ordered_today = True

    def cost(self):
        # ต้นทุนเฉลี่ยต่อหุ้น (รวมค่าธรรมเนียมแล้ว) — เก็บในพอร์ต จึงยังอยู่แม้ข้ามวัน
        for stock in self.handler.get_all_stocks_info():
            if stock["Symbol"] == self.SYMBOL:
                return stock["Average Cost"]

```

2. You first Strategy

สร้าง Strategy ตัวแรก

1	Strategy เปล่าๆ
2	ส่งคำสั่งซื้อครั้งแรก
3	ซื้อวันนี้ ขายวันหลัง
4	ใช้ข้อมูลย้อนหลังในการทำ Strategy + Indicator SMA
5	แนะนำ Library สำหรับทำ Indicator
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่
9	รันหลายวันต่อกัน



Portfolio: CopterTeam

Cash Balance	Symb…	Actu… Vol	Buy Price	Mark… Price	Aver… Cost	Amou… Cost	Mark… Value	Unre…	Unre… %	Reali…	Buy Time
9,975,158.34	KBANK	100	248.…	244.…	248.…	2484…	2440…	-441…	-1.7…	0.00	2026… 09:5…

Metrics	
ROI (%)	-0.0…
Max Drawdown (%)	0.00%
Buy Orders	1
Sell Orders	0
Round-Trips	0
Wins	0
Win Rate	0.00%
Expected Payoff	0.00
Avg Exposure	0.00%

2. You first Strategy

สร้าง Strategy ตัวแรก

1	Strategy เปล่าๆ
2	ส่งคำสั่งซื้อครั้งแรก
3	ซื้อวันนี้ ขายวันหลัง
4	ใช้ข้อมูลย้อนหลังในการทำ Strategy + Indicator SMA
5	แนะนำ Library สำหรับทำ Indicator
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่
9	รันหลายวันต่อกัน



Portfolio: CopterTeam

Cash Balance	Symb…	Actu… Vol	Buy Price	Mark… Price	Aver… Cost	Amou… Cost	Mark… Value	Unre…	Unre… %	Reali…	Buy Time
9,975,158.34	KBANK	100	248.…	244.…	248.…	2484…	2440…	-441…	-1.7…	0.00	2026… 09:5…

Metrics	
ROI (%)	-0.0…
Max Drawdown (%)	0.00%
Buy Orders	1
Sell Orders	0
Round-Trips	0
Wins	0
Win Rate	0.00%
Expected Payoff	0.00
Avg Exposure	0.00%

Portfolio: CopterTeam

Cash Balance	Symb…	Actu… Vol	Buy Price	Mark… Price	Aver… Cost	Amou… Cost	Mark… Value	Unre…	Unre… %	Reali…	Buy Time
9,975,158.34	KBANK	100	248.…	245.…	248.…	2484…	2450…	-341…	-1.3…	0.00	2026… 09:5…

Metrics	
ROI (%)	-0.0…
Max Drawdown (%)	0.00%
Buy Orders	1
Sell Orders	0
Round-Trips	0
Wins	0
Win Rate	0.00%
Expected Payoff	0.00
Avg Exposure	0.24%

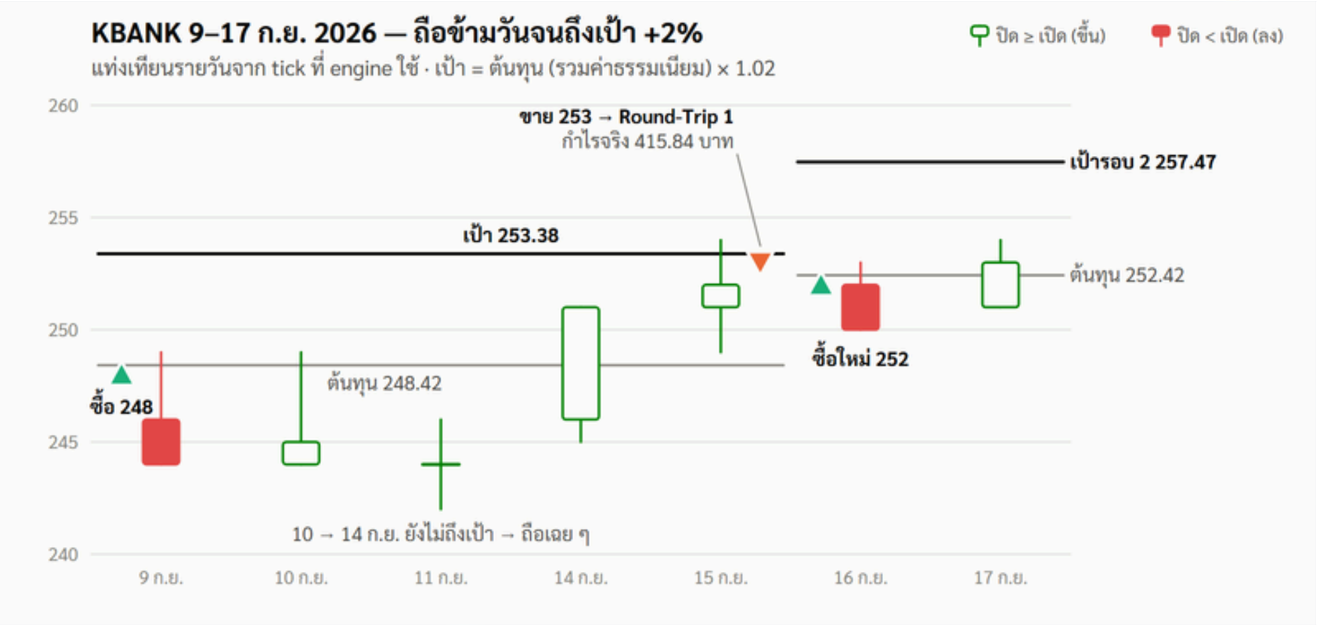
2. You first Strategy

สร้าง Strategy ตัวแรก

1	Strategy เปล่าๆ
2	ส่งคำสั่งซื้อครั้งแรก
3	ซื้อวันนี้ ขายวันหลัง
4	ใช้ข้อมูลย้อนหลังในการทำ Strategy + Indicator SMA
5	แนะนำ Library สำหรับทำ Indicator
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่
9	รันหลายวันต่อกัน

```
[INFO] Created new portfolio for 'CopterTeam'
[OK] Summary saved to c:\Code\TradeSim_template\MU-Pooh\result\CopterTeam\CopterTeam_
[INFO] Loaded existing portfolio from 'result\CopterTeam\CopterTeam_portfolio.json'
[OK] Summary saved to c:\Code\TradeSim_template\MU-Pooh\result\CopterTeam\CopterTeam_
[INFO] Loaded existing portfolio from 'result\CopterTeam\CopterTeam_portfolio.json'
[OK] Summary saved to c:\Code\TradeSim_template\MU-Pooh\result\CopterTeam\CopterTeam_
[INFO] Loaded existing portfolio from 'result\CopterTeam\CopterTeam_portfolio.json'
[OK] Summary saved to c:\Code\TradeSim_template\MU-Pooh\result\CopterTeam\CopterTeam_
[INFO] Loaded existing portfolio from 'result\CopterTeam\CopterTeam_portfolio.json'
[OK] Summary saved to c:\Code\TradeSim_template\MU-Pooh\result\CopterTeam\CopterTeam_
[INFO] Loaded existing portfolio from 'result\CopterTeam\CopterTeam_portfolio.json'
[OK] Summary saved to c:\Code\TradeSim_template\MU-Pooh\result\CopterTeam\CopterTeam_
```

	Date	Cash	ROI (%)	Buy Orders	Sell Orders	Round-Trips	Holding
0	2026-09-09	9975158.34	-0.0044	1	0	0	KBANK 100
1	2026-09-10	9975158.34	-0.0034	1	0	0	KBANK 100
2	2026-09-11	9975158.34	-0.0034	1	0	0	KBANK 100
3	2026-09-14	9975158.34	0.0016	1	0	0	KBANK 100
4	2026-09-15	10000415.84	0.0042	1	1	1	-
5	2026-09-16	9975173.50	0.0017	2	1	1	KBANK 100
6	2026-09-17	9975173.50	0.0047	2	1	1	KBANK 100



	Date	Cash	ROI (%)	Buy Orders	Sell Orders	Round-Trips	Holding
0	2026-09-09	9975158.34	-0.0044	1	0	0	KBANK 100
1	2026-09-10	9975158.34	-0.0034	1	0	0	KBANK 100
2	2026-09-11	9975158.34	-0.0034	1	0	0	KBANK 100
3	2026-09-14	9975158.34	0.0016	1	0	0	KBANK 100
4	2026-09-15	10000415.84	0.0042	1	1	1	-
5	2026-09-16	9975173.50	0.0017	2	1	1	KBANK 100
6	2026-09-17	9975173.50	0.0047	2	1	1	KBANK 100

	ซื้อ (09-09)	ขาย (09-15)
ราคาที่ได้จริง [1]	248	253
มูลค่าหุ้น (× 100 หุ้น)	24,800.00	25,300.00
ค่าคอมมิชชัน 0.157%	38.94	− 39.72
VAT 7% ของค่าคอมฯ	2.73	− 2.78
จ่ายไป / ได้กลับมา	24,841.66	25,257.50
ต่อหุ้น	24.842	25.258

[1] หมายเหตุ: ราคาที่ Order จะมีการคิด slippage เผื่อไว้ที่ 1 ticksize เสมอเพื่อให้สมจริงกับการซื้อในตลาด

	บาท	%
ส่วนต่างราคา (253 − 248) × 100	500.00	2.02%
ค่าธรรมเนียมรวม 2 ขา	− 84.16	
กำไรจริง	415.84	1.67%

Strategy + Indicator
SMA

2. You first Strategy

สร้าง Strategy ตัวแรก

1	Strategy เปล่าๆ
2	ส่งคำสั่งซื้อครั้งแรก
3	ซื้อวันนี้ ขายวันหลัง
4	ใช้ข้อมูลย้อนหลังในการทำ Strategy + Indicator SMA
5	แนะนำ Library สำหรับทำ Indicator
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่

```python
from strategy.Strategies_template import Strategy_template


class MyStrategy(Strategy_template):
    def __init__(self, handler):
        super().__init__("CopterTeam", "MyStrategy", handler, timeframe="1D")

        # ดึงข้อมูลย้อนหลังตรงนี้ – ต่อจาก super().__init__() เสมอ
        history = self.handler.get_historical_data("AOT", 5)
        print("ได้มา", len(history), "แท่ง")
        for day in history:
            print(day["Date"], "ปิดที่", day["Close"])

    def on_data(self, bar):
        pass
```

```
ได้มา 5 แท่ง
2026-09-02 ปิดที่ 61.75
2026-09-03 ปิดที่ 62.75
2026-09-04 ปิดที่ 63.5
2026-09-07 ปิดที่ 62.5
2026-09-08 ปิดที่ 63.0
ได้มา 5 แท่ง
2026-09-02 ปิดที่ 61.75
2026-09-03 ปิดที่ 62.75
2026-09-04 ปิดที่ 63.5
2026-09-07 ปิดที่ 62.5
2026-09-08 ปิดที่ 63.0
ได้มา 5 แท่ง
2026-09-02 ปิดที่ 61.75
2026-09-03 ปิดที่ 62.75
2026-09-04 ปิดที่ 63.5
2026-09-07 ปิดที่ 62.5
2026-09-08 ปิดที่ 63.0
```


ราคาปิด vs SMA5
RSI 14: <40 ซื้อ >60 ขาย
MACD hist + RSI
ราคาปิด − SMA5 (> 0 ซื้อ · < 0 ขาย)
RSI 14 (< 40 ซื้อ · > 60 ขาย)
MACD histogram (> 0 ซื้อ · < 0 ขาย)
สัญญาณ = RSI 14 (แกนขวา)
ซื้อ
ขาย

2. You first Strategy

สร้าง Strategy ตัวแรก

1	Strategy เปล่าๆ
2	ส่งคำสั่งซื้อครั้งแรก
3	ซื้อวันนี้ ขายวันหลัง
4	ใช้ข้อมูลย้อนหลังในการทำ Strategy + Indicator SMA
5	แนะนำ Library สำหรับทำ Indicator
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่

```python
from strategy.Strategies_template import Strategy_template


class MyStrategy(Strategy_template):
    LOOKBACK = 5
    WATCHLIST = ["AOT"]

    def __init__(self, handler):
        super().__init__("CopterTeam", "MyStrategy", handler, timeframe="1D")

        # คำนวณครั้งเดียว เก็บไว้ใน self.sma
        self.sma = {}
        for symbol in self.WATCHLIST:
            history = self.handler.get_historical_data(symbol, self.LOOKBACK)
            if len(history) < self.LOOKBACK:
                continue                        # ข้อมูลไม่พอ → ข้ามตัวนี้

            # ↓ คำนวณ SMA เอง ไม่ใช้ library: บวกราคาปิด 5 วัน แล้วหาร 5
            closes = [day["Close"] for day in history]
            self.sma[symbol] = sum(closes) / len(closes)

    def on_data(self, bar):
        symbol = bar["ShareCode"]
        sma = self.sma.get(symbol)              # แค่เปิดดูค่าที่คำนวณไว้
        if sma is None:                         # ไม่อยู่ใน watchlist หรือข้อมูลไม่พอ
            return

        price = bar["Close"]
        if price > sma and not self.handler.check_port_has_stock(symbol, 100):
            self.handler.create_order_to_limit(100, price, "Buy", symbol)
        elif price < sma and self.handler.check_port_has_stock(symbol, 100):
            self.handler.create_order_to_limit(100, price, "Sell", symbol)

```

2. You first Strategy

สร้าง Strategy ตัวแรก

1	Strategy เปล่าๆ
2	ส่งคำสั่งซื้อครั้งแรก
3	ซื้อวันนี้ ขายวันหลัง
4	ใช้ข้อมูลย้อนหลังในการทำ Strategy + Indicator SMA
5	แนะนำ Library สำหรับทำ Indicator
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่

Portfolio: CopterTeam

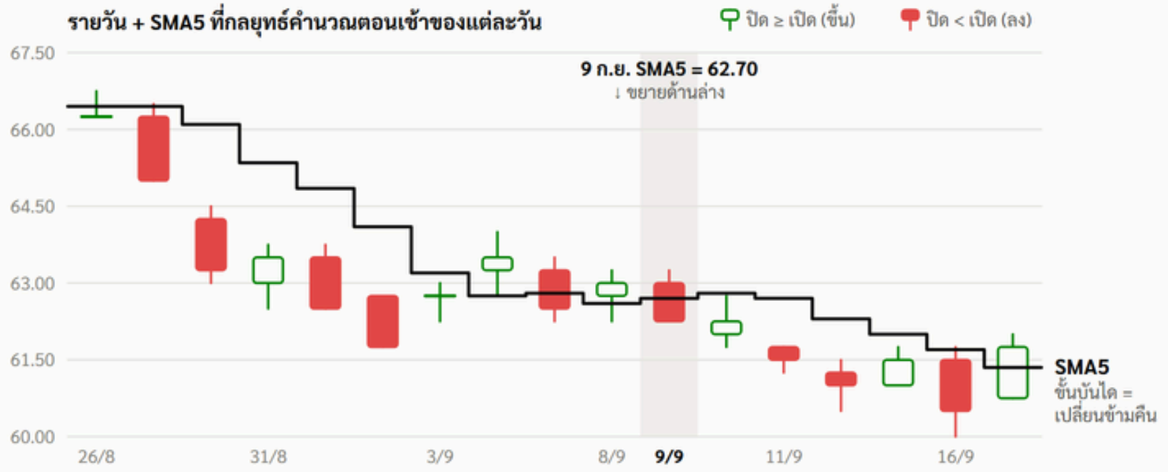
Cash Balance	Symb…	Actu… Vol	Buy Price	Mark… Price	Aver… Cost	Amou… Cost	Mark… Value	Unre…	Unre… %	Reali…	Buy Time
9,989,615.72	-	0	-	-	-	-	-	-	-	-	-

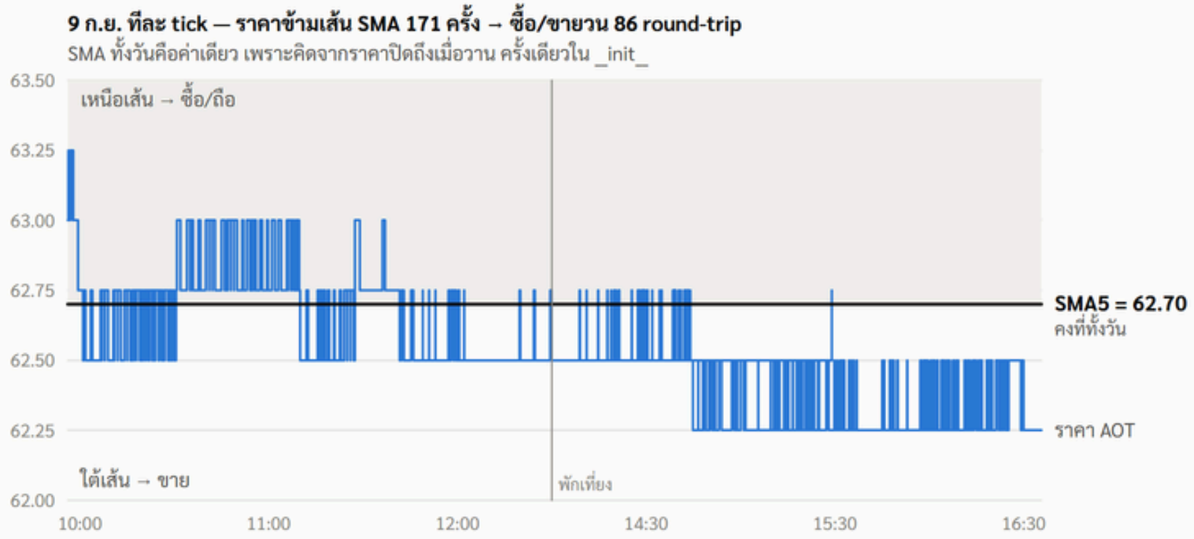
Metrics	
ROI (%)	-0.1…
Max Drawdown (%)	0.00%
Buy Orders	105
Sell Orders	105
Round-Trips	105
Wins	0
Win Rate	0.00%
Expected Payoff	-98.…
Avg Exposure	0.00%

	Order Number	owner	Volume	Price	Side	Symbol	Timestamp
1	20260909-ORD00218	CopterTeam	100	63.25	Buy	AOT	2026-09-09 09:56:24
2	20260909-ORD00219	CopterTeam	100	62.25	Sell	AOT	2026-09-09 10:01:00
3	20260909-ORD00220	CopterTeam	100	63.0	Buy	AOT	2026-09-09 10:01:08
4	20260909-ORD00221	CopterTeam	100	62.25	Sell	AOT	2026-09-09 10:01:50
5	20260909-ORD00222	CopterTeam	100	63.25	Buy	AOT	2026-09-09 10:01:58
6	20260909-ORD00223	CopterTeam	100	62.25	Sell	AOT	2026-09-09 10:01:58
7	20260909-ORD00224	CopterTeam	100	63.0	Buy	AOT	2026-09-09 10:03:28
8	20260909-ORD00225	CopterTeam	100	62.25	Sell	AOT	2026-09-09 10:03:33
9	20260909-ORD00226	CopterTeam	100	63.5	Buy	AOT	2026-09-09 10:06:00
10	20260909-ORD00227	CopterTeam	100	62.25	Sell	AOT	2026-09-09 10:06:00
11	20260909-ORD00228	CopterTeam	100	63.0	Buy	AOT	2026-09-09 10:06:35
12	20260909-ORD00229	CopterTeam	100	62.25	Sell	AOT	2026-09-09 10:07:00
13	20260909-ORD00230	CopterTeam	100	63.0	Buy	AOT	2026-09-09 10:07:07
14	20260909-ORD00231	CopterTeam	100	62.25	Sell	AOT	2026-09-09 10:08:00

AOT — เส้น SMA5 เปลี่ยนวันละครั้ง แต่ระหว่างวันคงที่

บน: รายวัน · ล่าง: ขยายวันที่ 9 ก.ย. ทีละ tick — กลยุทธ์หัวข้อ 7 ใช้ SMA ค่าเดียวทั้งวัน





2. You first Strategy

สร้าง Strategy ตัวแรก

1	Strategy เปล่าๆ
2	ส่งคำสั่งซื้อครั้งแรก
3	ซื้อวันนี้ ขายวันหลัง
4	ใช้ข้อมูลย้อนหลังในการทำ Strategy + Indicator SMA
5	แนะนำ Library สำหรับทำ Indicator
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่

```python
import numpy as np
import talib

from strategy.Strategies_template import Strategy_template


class MyStrategy(Strategy_template):
    LOOKBACK = 60
    RSI_PERIOD = 14
    WATCHLIST = ["AOT", "MTC"]

    def __init__(self, handler):
        super().__init__("CopterTeam", "MyStrategy", handler, timeframe="1D")
        self.rsi = {symbol: self._rsi(symbol) for symbol in self.WATCHLIST}

    def _rsi(self, symbol):
        history = self.handler.get_historical_data(symbol, self.LOOKBACK)
        if len(history) < self.RSI_PERIOD + 1:
            return None
        closes = np.array([day["Close"] for day in history], dtype=np.float64)
        value = talib.RSI(closes, timeperiod=self.RSI_PERIOD)[-1]
        return None if np.isnan(value) else float(value)

    def on_data(self, bar):
        symbol = bar["ShareCode"]
        rsi = self.rsi.get(symbol)
        if rsi is None:
            return

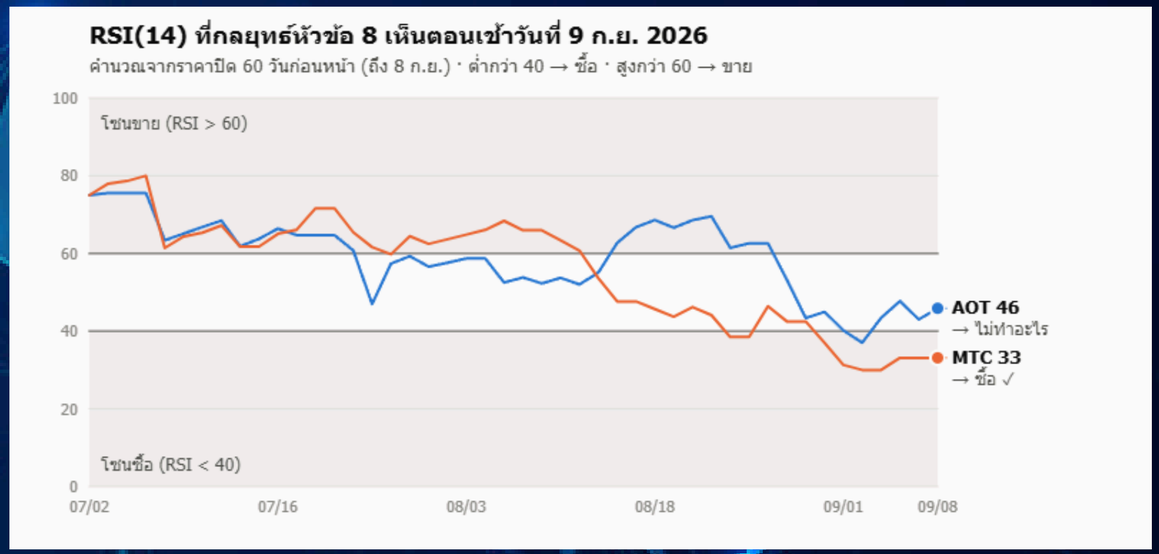
        price = bar["Close"]
        if rsi < 40 and not self.handler.check_port_has_stock(symbol, 100):
            self.handler.create_order_to_limit(100, price, "Buy", symbol)
        elif rsi > 60 and self.handler.check_port_has_stock(symbol, 100):
            self.handler.create_order_to_limit(100, price, "Sell", symbol)

```

2. You first Strategy

สร้าง Strategy ตัวแรก

1	Strategy เปล่าๆ
2	ส่งคำสั่งซื้อครั้งแรก
3	ซื้อวันนี้ ขายวันหลัง
4	ใช้ข้อมูลย้อนหลังในการทำ Strategy + Indicator SMA
5	แนะนำ Library สำหรับทำ Indicator
6	ทดสอบ Strategy กับตลาดจริง
7-8	ดูสรุปพอร์ต และหุ้นที่ถืออยู่



Portfolio: CopterTeam

Cash Balance	Symb…	Actu… Vol	Buy Price	Mark… Price	Aver… Cost	Amou… Cost	Mark… Value	Unre…	Unre… %	Reali…	Buy Time
9,996,969.92	MTC	100	30.30	30.00	30.30	3030…	3000…	-30.…	-0.9…	0.00	2026… 10:0…

Metrics	
ROI (%)	-0.0…
Max Drawdown (%)	0.00%
Buy Orders	1
Sell Orders	0
Round-Trips	0
Wins	0
Win Rate	0.00%
Expected Payoff	0.00
Avg Exposure	0.00%

ส่ง Strategy เพื่อเข้าแข่งขัน

รูปแบบการแข่งขัน

การแข่งขันในครั้งนี้จะเป็นการนำกลยุทธ์ของผู้เข้าแข่งขันมาทำการจำลองการซื้อขายหุ้นกับข้อมูลตลาดจริงโดยนำข้อมูลการซื้อขายของตลาดในวันการแข่งขันมาใช้ในการจำลอง

1. ในช่วงเดือน ตุลาคม ผู้เข้าแข่งขันออกแบบและสร้างกลยุทธ์การลงทุน (Strategy) ที่สมบูรณ์และพร้อมใช้งานบน PC ของท่าน
2. นำส่ง Strategy ให้กับทีมงานเพื่อเตรียมตัวเข้าแข่งขันกับข้อมูลการซื้อขายของตลาด ณ วันการแข่งขัน

ช่วงก่อนเริ่มการแข่งขัน

- Download Trading_Simulator.zip เพื่อนำไปทดลองใช้งาน และพัฒนา Strategy ก่อนเริ่มการแข่งขัน
- ในเดือนตุลาคม ทีมงานจะส่ง daily tick ชุด 20 วันให้เพิ่ม
- **หนึ่งอาทิตย์**ก่อนเริ่มการแข่งขันจริง ผู้เข้าแข่งขันจะต้องส่ง Strategy มาเพื่อทดสอบระบบ [1]

[1] หากมีการใช้ Library อื่นๆในการทำกลยุทย์โปรดแจ้งทีมงานตอนนำส่ง code ด้วย

ในช่วงการแข่งขัน

- ทีมงานจะส่ง Daily tick ของตลาดล่าสุดให้ทุกเช้า
- ผู้เข้าแข่งขันสามารถปรับปรุงแก้ไข Strategy ได้ตลอดช่วงการแข่งขัน แต่จะต้องส่งให้ทีมงานก่อนเช้า 9 โมงในวันที่ตลาดเปิด
- ในกรณีที่เป็นวันตลาดเปิด และผู้เข้าแข่งขันไม่ได้ส่ง Strategy เข้ามาก่อน 9 โมง ทีมงานจะใช้ Strategy ล่าสุดของผู้เข้าแข่งขันในการประมวลผล

1.ส่ง Strategy

git add MyStrategy.py เปลี่ยนชื่อ team ให้เรียบร้อย

```python
    def __init__(self, handler):
        super().__init__("CopterTeam", "MyStrategy", handler, timeframe="1D")

```

เข้า CMD ไปที Directory ของ Repository แล้วรัน

```
git add strategy\MyStrategy.py
```

Commit to the repository then push

```
git commit -m "Add MyStrategy"

git push
```

```
C:\Users\03915\Code Testing\template_ALGO_2026\MU-Copter>git add strategy\MyStrategy.py

C:\Users\03915\Code Testing\template_ALGO_2026\MU-Copter>git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   strategy/MyStrategy.py


C:\Users\03915\Code Testing\template_ALGO_2026\MU-Copter>git commit -m "Add MyStrategy"
[main 98737e5] Add MyStrategy
 1 file changed, 72 insertions(+)
 create mode 100644 strategy/MyStrategy.py

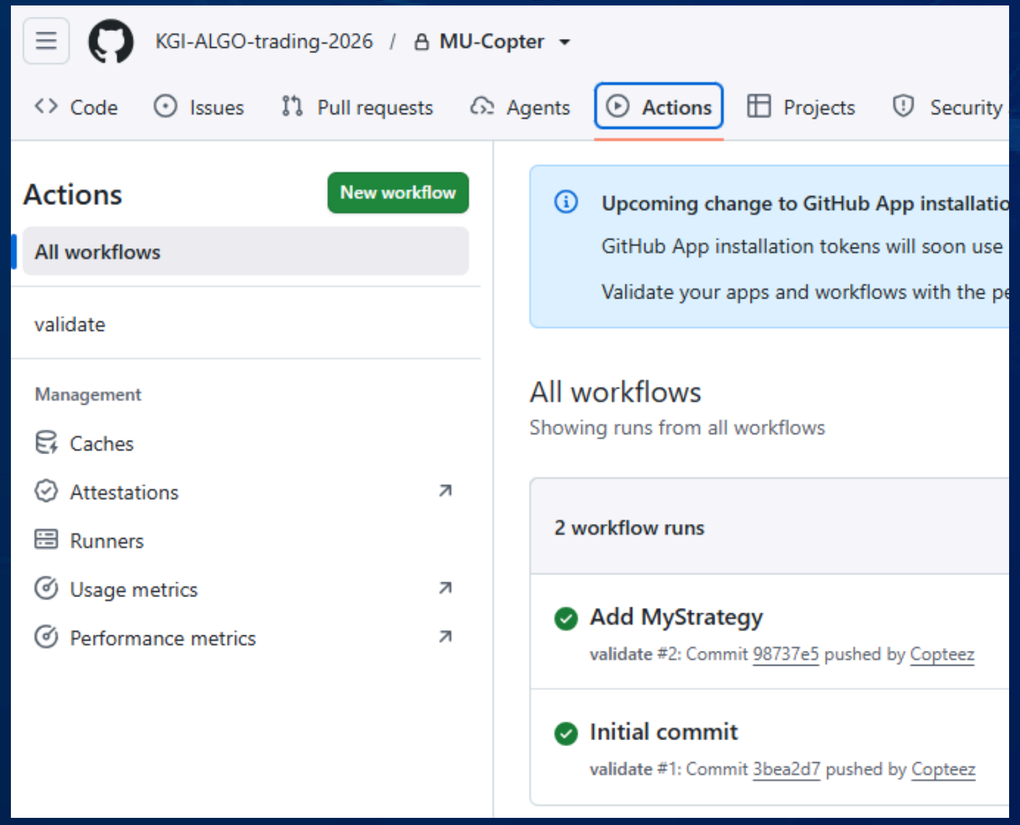
C:\Users\03915\Code Testing\template_ALGO_2026\MU-Copter>git push
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 6 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 1.67 KiB | 856.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/KGI-ALGO-trading-2026/MU-Copter.git
   3bea2d7..98737e5  main -> main

C:\Users\03915\Code Testing\template_ALGO_2026\MU-Copter>D
```

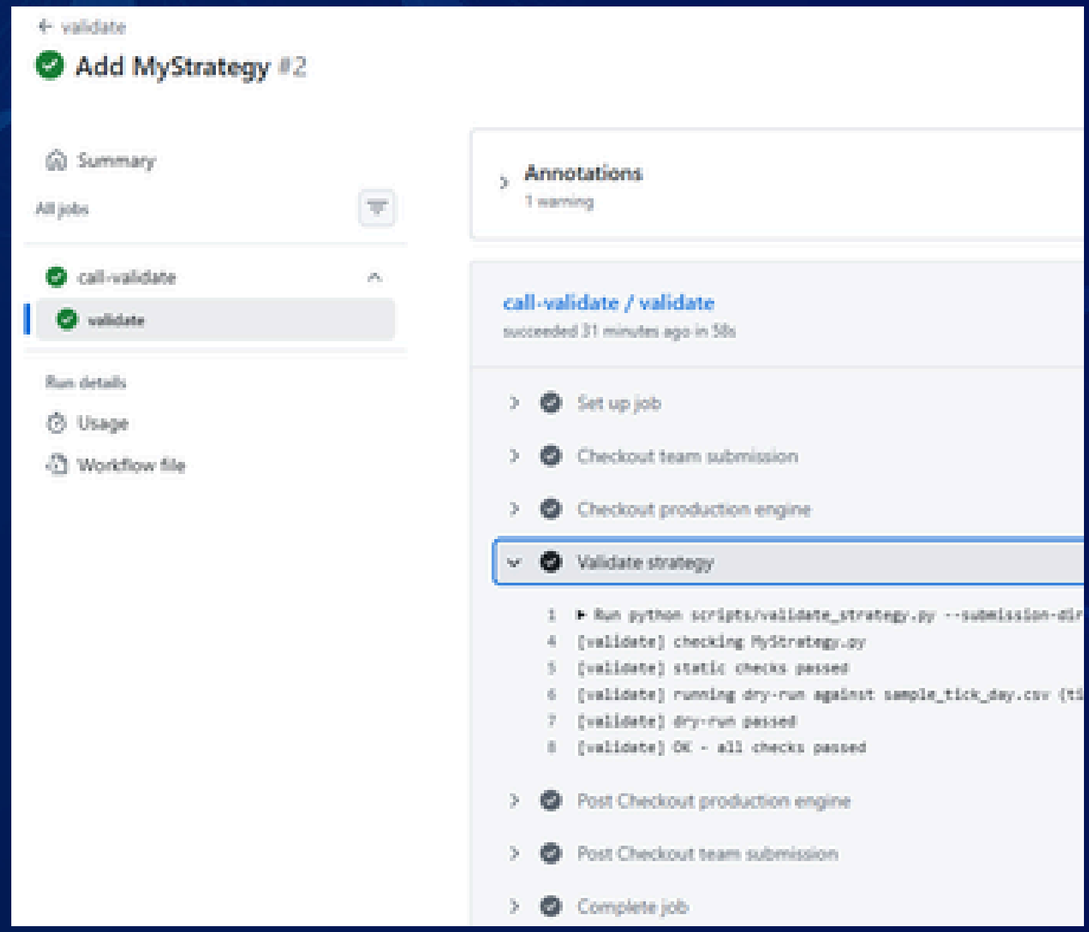
1.ส่ง Strategy

1.เข้าไปที่ Repository ของทีมเปิด Tab Action

2.Click ไปที่ Add MyStrategy ที่เรา Commit ขึ้นไป



3.สามารถเช็คที่ Task Validate strategy ว่าสามารถใช้ในการแข่งได้

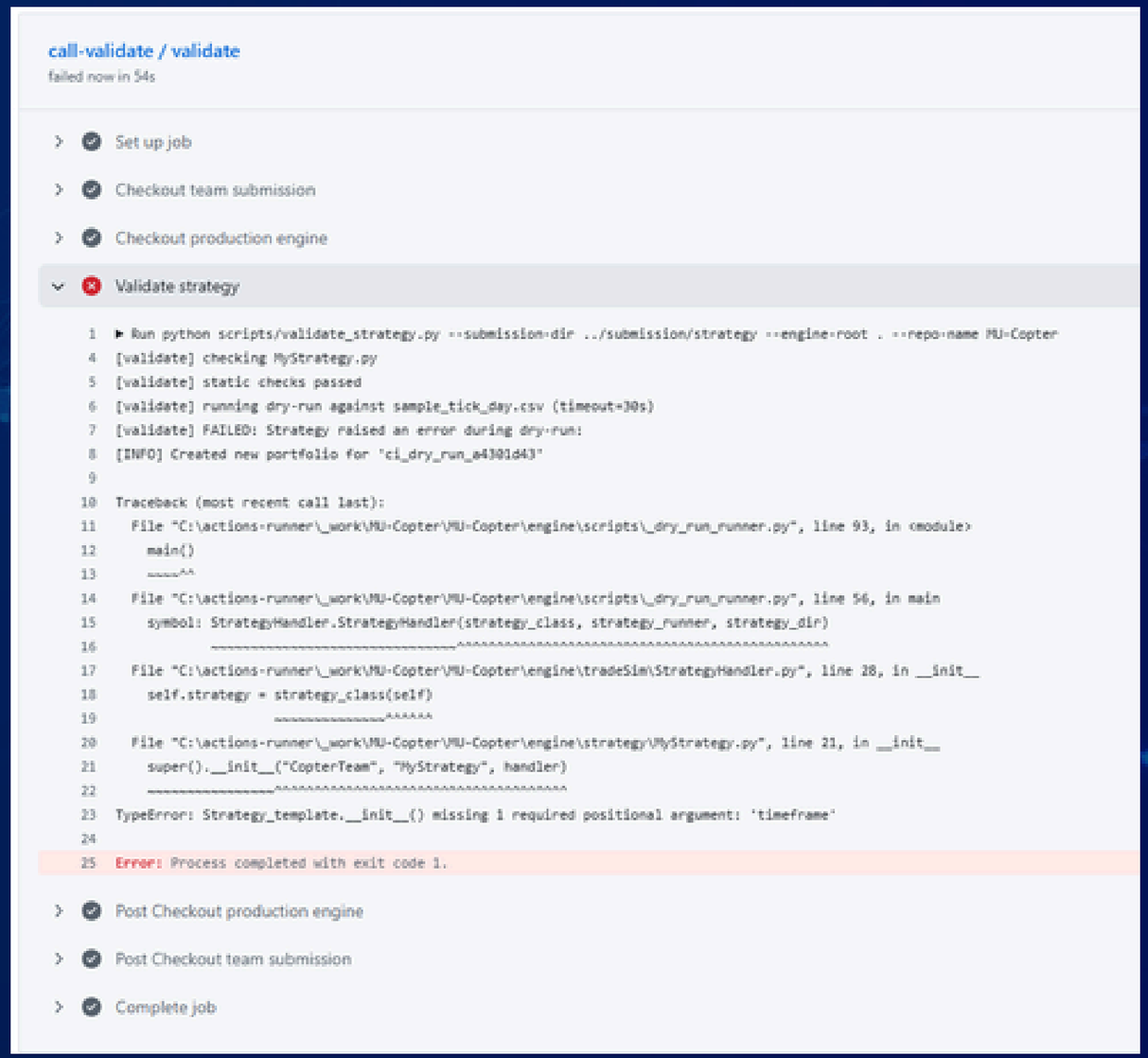


ถ้า File มีปัญหา จะขึ้นยังไง ?

1. เข้าไปที่ Repository ของทีมเปิด Tab Action
2. Click ไปที่ commit ที่เราส่งไป

ต้องทำอะไร ?

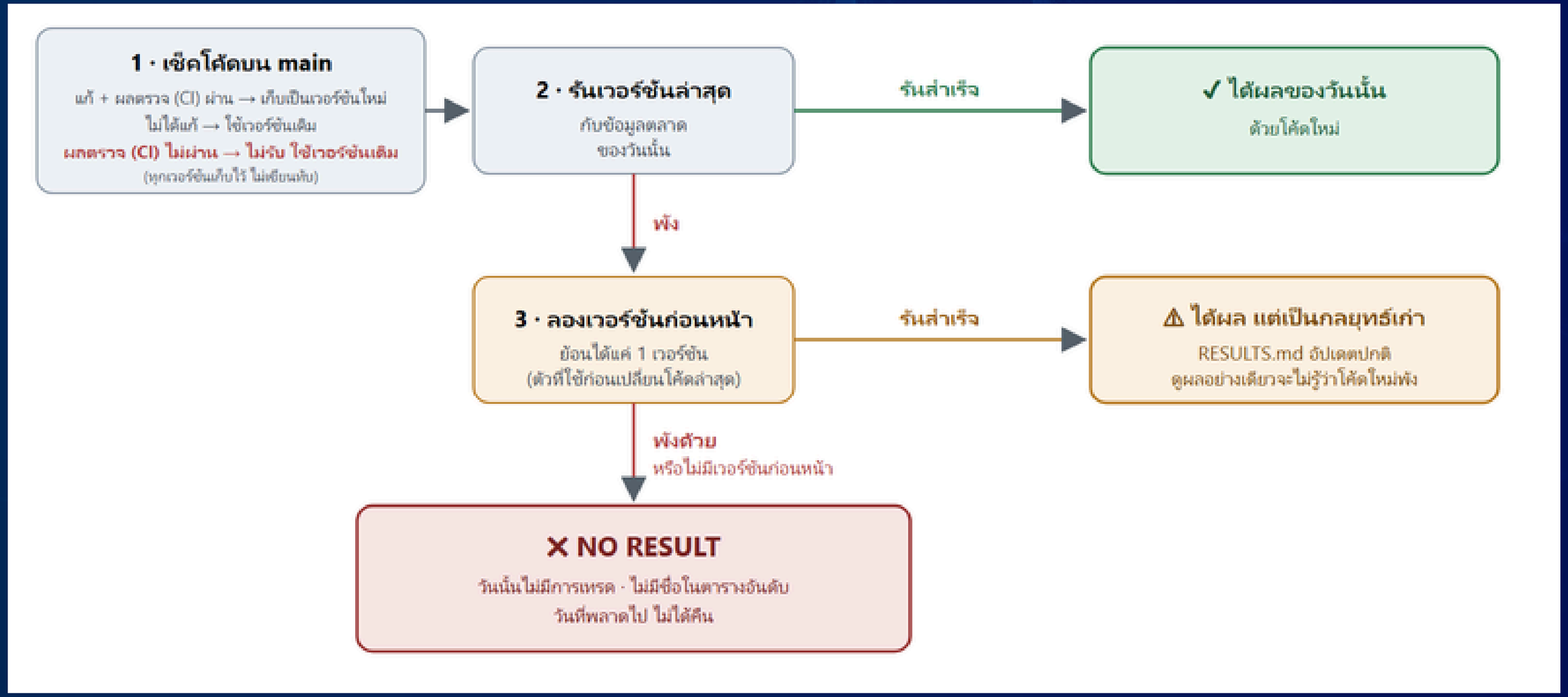
1. แก้ไข code แล้วทดสอบผ่าน TradingSimulation.ipynb
2. commit push ไปใหม่อีกครั้ง
3. เช็ค Action แท็บ อีกครั้ง


call-validate / validate
failed now in 54s
Set up job
Checkout team submission
Checkout production engine
Validate strategy
1 ▶ Run python scripts/validate_strategy.py --submission-dir ../submission/strategy --engine-root . --repo-name MU-Copter
4 [validate] checking MyStrategy.py
5 [validate] static checks passed
6 [validate] running dry-run against sample_tick_day.csv (timeout=30s)
7 [validate] FAILED: Strategy raised an error during dry-run:
8 [INFO] Created new portfolio for 'ci_dry_run_a4301d43'
9
10 Traceback (most recent call last):
11 File "C:\actions-runner_work\MU-Copter\MU-Copter\engine\scripts_dry_run_runner.py", line 93, in <module>
12 main()
13 ~~~~^^
14 File "C:\actions-runner_work\MU-Copter\MU-Copter\engine\scripts_dry_run_runner.py", line 56, in main
15 symbol: StrategyHandler.StrategyHandler(strategy_class, strategy_runner, strategy_dir)
16 ~~~~

ส่งกลยุทธ์ที่พังแล้วไม่ได้แก้ จะเกิดอะไรขึ้นตอนแข่ง?

1. เช็คว่าทีมแก้โค้ดบน branch main หรือเปล่า
    a. แก้ และผลตรวจ (CI) เขียว → เก็บเป็นเวอร์ชันใหม่ (v1 → v2 → v3 ...)
    b. ไม่ได้แก้ → ใช้เวอร์ชันเดิมต่อ
    c. แก้ แต่ผลตรวจ (CI) แดง → ไม่รับ แข่งต่อด้วยเวอร์ชันเดิม
    d. ทุกเวอร์ชันถูกเก็บไว้ ไม่มีการเขียนทับ ถ้า push หลายรอบในวันเดียว นับเฉพาะโค้ดที่อยู่บน main ตอนระบบดึง
2. รันเวอร์ชันล่าสุด กับข้อมูลตลาดของวันนั้น ถ้า error หรือรันเกิน 15 นาที ถือว่า "ไม่สามารถใช้ในการแข่งได้"
3. ถ้าพัง ระบบลองเวอร์ชันก่อนหน้าให้แค่ 1 เวอร์ชัน

ส่งกลยุทธ์ที่พังแล้วไม่ได้แก้ จะเกิดอะไรขึ้นตอนแข่ง?



ส่งกลยุทธ์ที่พังแล้วไม่ได้แก้ จะเกิดอะไรขึ้นตอนแข่ง?

กรณี A · ช่วงตรวจโค้ด 1 สัปดาห์ก่อนแข่ง

1 สัปดาห์ก่อนแข่ง

push แก้ได้เรื่อย ๆ ตลอดสัปดาห์

ต้องได้ผลตรวจ (CI) ผ่าน บน main

อย่างน้อย 1 ครั้ง ก่อนหมดสัปดาห์

ผ่านภายในสัปดาห์

ผลตรวจ (CI) ผ่าน → v1

✓ ได้เข้าแข่ง

เริ่มแข่งด้วย v1

หมดสัปดาห์ยังไม่ผ่าน

ยังไม่เคยผ่านเลย

✕ ไม่ได้เข้าร่วมการแข่งขัน

ไม่มีเวอร์ชันที่ผ่านให้ใช้แข่ง

กรณี B · ระหว่างแข่ง (มีเวอร์ชันที่ผ่านแล้ว v1)

วันที่ 1

ผลตรวจ (CI) ผ่าน → v1

✓ ได้ผล (v1)

วันที่ 2

push แล้วผลตรวจ (CI) **ไม่ผ่าน**

⚠ ไม่รับ แข่งต่อด้วย v1

RESULTS.md ดูปกติ — ต้องดูแท็บ Checks เอง

วันที่ 3

ผลตรวจ (CI) ผ่าน → v2

แต่พังตอนรันจริง

⚠ ย้อนไปใช้ v1

ระบบย้อนให้ 1 เวอร์ชัน

วันที่ 4

ผลตรวจ (CI) ผ่าน → v3

ก็พังตอนรันจริง

✕ NO RESULT

ย้อนได้แค่ v2 ซึ่งพังเหมือนกัน

push แล้วเปิดแท็บ Checks ดูผลตรวจ (CI) ทุกครั้ง — ไม่ผ่าน = โค้ดนี้ไม่ถูกใช้ แก้แล้ว push ใหม่

ผ่านแปลว่า "รันได้กับข้อมูลตัวอย่าง" ไม่ได้แปลว่า "กลยุทธ์ถูก" — รันใน notebook กับข้อมูลจริงให้ผ่านก่อน push ทุกครั้ง

Competition Details

รูปแบบการแข่งขัน

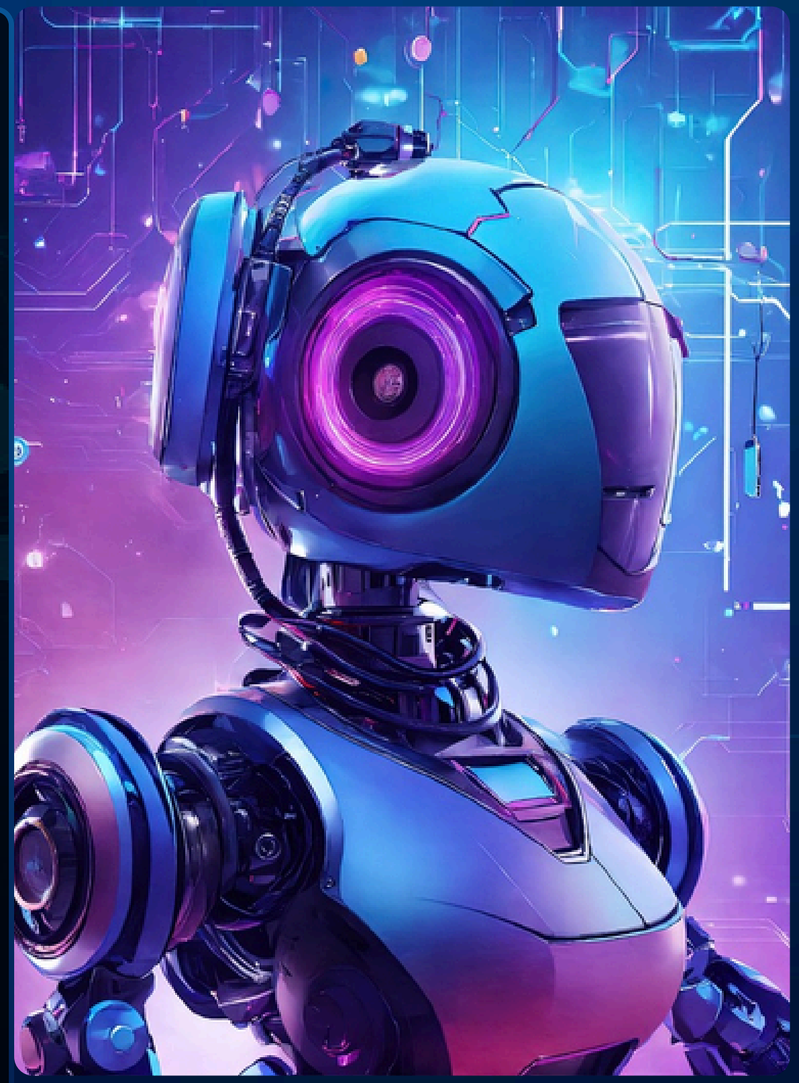
การแข่งขันในครั้งนี้จะเป็นการนำกลยุทธ์ของผู้เข้าแข่งขันมาทำการจำลองการซื้อขายหุ้นกับข้อมูลตลาดจริงโดยนำข้อมูลการซื้อขายของตลาดในวันการแข่งขันมาใช้ในการจำลอง

- ทุกทีมเริ่มที่ 10,000,000 บาท เท่ากัน เทรดได้เฉพาะ SET50
- สิ่งที่ใช้ในการจัดอันดับคือ Excess Return = ผลตอบแทนพอร์ต – ผลตอบแทน SET50 ช่วงเดียวกัน → ชนะตลาดได้เท่าไหร่
- รัน Strategy ตัวอย่าง → เขียนเองจากศูนย์ → เติมจนใช้ indicator ได้ → ส่งงานผ่าน git hub และ validate Strategy ว่าสามารถใช้ในการแข่งขันได้

Competition Details

กติกาการแข่งขัน (Competition Rules)

- สินทรัพย์ที่ใช้ในการแข่งขัน ได้แก่ หลักทรัพย์ที่เป็นองค์ประกอบของดัชนี SET50
- ผู้เข้าแข่งขันต้องพัฒนาและทดสอบกลยุทธ์โดยใช้โปรแกรมและข้อมูลตลาดที่บริษัทฯ จัดเตรียมไว้สำหรับการแข่งขันเท่านั้น
- เงินลงทุนและเงื่อนไขการซื้อขาย (ถูกตั้งค่าในโปรแกรม TradeSim)
  - เงินลงทุนเริ่มต้นสำหรับการแข่งขันจำนวน 10,000,000 บาท
  - กำหนดประเภทบัญชีการซื้อขายเป็น Cash Balance ไม่สามารถส่งคำสั่งแบบ Short Sell ได้
  - คำนวณค่าธรรมเนียมการซื้อขายในอัตรา 0.157%
  - คำนวณภาษีมูลค่าเพิ่ม (VAT) ในอัตรา 7% ตามเงื่อนไขที่กำหนด
  - กำหนดค่าความคลาดเคลื่อนของราคาซื้อขาย (Slippage) เท่ากับ 1 Tick Size
  - กำหนดรูปแบบคำสั่งซื้อขายประเภท Limit Order หรือ Market-to-Limit Order ที่มี validity แบบ IOC เท่านั้น
- การส่งและการปรับปรุงกลยุทธ์
  - ผู้เข้าแข่งขันต้องนำส่งไฟล์กลยุทธ์ .py ภายในวัน เวลา และช่องทางที่บริษัทฯ กำหนด
  - บริษัทฯ จะพิจารณาเฉพาะไฟล์กลยุทธ์ที่นำส่งภายในระยะเวลาที่กำหนด
  - การปรับปรุงหรือเปลี่ยนแปลงกลยุทธ์ระหว่างการแข่งขันสามารถดำเนินการได้ตามเงื่อนไขและกำหนดการที่บริษัทฯ ประกาศ



Competition Details

กำหนดการการแข่งขัน (Competition Timeline)

League	Competition Period
TU	26 OCT – 20 NOV 2026
CU	02 NOV – 27 NOV 2026
MU	02 NOV – 27 NOV 2026
KM	16 DEC 2026 – 15 JAN 2027

1 ช่วงเตรียมความพร้อมก่อนการแข่งขัน (Pre-Competition Period)

- หลังเสร็จสิ้นการสัมมนา ผู้เข้าแข่งขันจะต้องพัฒนากลยุทธ์การซื้อขายในรูปแบบไฟล์ .py
- ผู้เข้าแข่งขันสามารถทดสอบกลยุทธ์กับโปรแกรมและระบบที่บริษัทฯ จัดเตรียมไว้ให้
- กลยุทธ์จะต้องผ่านการทดสอบตามเกณฑ์ที่บริษัทฯ กำหนด จึงจะมีสิทธิ์เข้าสู่การแข่งขันและการจัดอันดับ (Ranking)

2 ช่วงการแข่งขัน (Competition Period)

- บริษัทฯ จะนำไฟล์กลยุทธ์ที่ผ่านการคัดเลือกมาทดสอบกับข้อมูลตลาดจริง 20 วันทำการ ตามช่วงเวลาที่กำหนดของแต่ละมหาวิทยาลัย
- ทีมผู้เข้าแข่งขันสามารถปรับปรุงและนำส่งไฟล์กลยุทธ์ฉบับใหม่ได้ตามวัน เวลา และช่องทางที่บริษัทฯ กำหนด
- ผลการแข่งขันจะพิจารณาจากประสิทธิภาพของกลยุทธ์ตามเกณฑ์การจัดอันดับที่บริษัทฯ กำหนด

3 ช่วงสรุปการแข่งขัน (Post-Competition Period)

- หลังสิ้นสุดการแข่งขัน ทีมที่มีสิทธิ์รับรางวัล จะต้องนำเสนอแนวคิดและกลยุทธ์ที่ใช้ในการแข่งขันในรูปแบบ รายงาน (Report) และ คลิปวิดีโอความยาวไม่เกิน 10 นาที

โดยมีรายละเอียด ดังนี้

- Stock Selection : ปัจจัยและหลักเกณฑ์ที่ใช้ในการคัดเลือกหลักทรัพย์
- Portfolio Management : แนวทางและวิธีการบริหารพอร์ตการลงทุน
- Risk Management : แนวทางการบริหารและควบคุมความเสี่ยงของพอร์ตการลงทุน



Competition Details

คุณสมบัติผู้สมัคร (Eligibility)

1. เป็นนักศึกษาระดับปริญญาตรีหรือปริญญาโท อายุไม่เกิน 35 ปี
2. เป็นนักศึกษาของมหาวิทยาลัยที่เข้าร่วมโครงการ
3. มีพื้นฐานหรือความสนใจในการเขียนโปรแกรม Python
4. มีความสนใจด้านการลงทุนในหลักทรัพย์ไทยและ Algorithmic Trading
5. ยอมรับและปฏิบัติตามเงื่อนไข กติกา และเกณฑ์การตัดสินการแข่งขันที่บริษัทฯ กำหนด
6. ไม่เคยเป็นบุคคลากรทำงานในบริษัทหลักทรัพย์ บริษัทหลักทรัพย์จัดการกองทุน หรือธนาคาร

รูปแบบการแข่งขัน (Competiiton Format)

1. รูปแบบทีม
   - ผู้เข้าแข่งขันแข่งขันในรูปแบบทีม ทีมละไม่เกิน 2 คน
   - สมาชิกในทีมสามารถอยู่ต่างสาขาวิชา คณะ หรือชั้นปีได้
   - สมาชิกในทีมต้องเป็นนักศึกษาจาก มหาวิทยาลัยเดียวกัน
2. รูปแบบการแข่งขัน
   - การแข่งขันจัดขึ้นในรูปแบบออนไลน์ โดยผู้เข้าแข่งขันต้องพัฒนาและนำส่งไฟล์กลยุทธ์ Python (.py) ตามวัน เวลา และช่องทางที่บริษัทฯ

Competition Details

รูปแบบการจัดอันดับ (Competition Ranking)

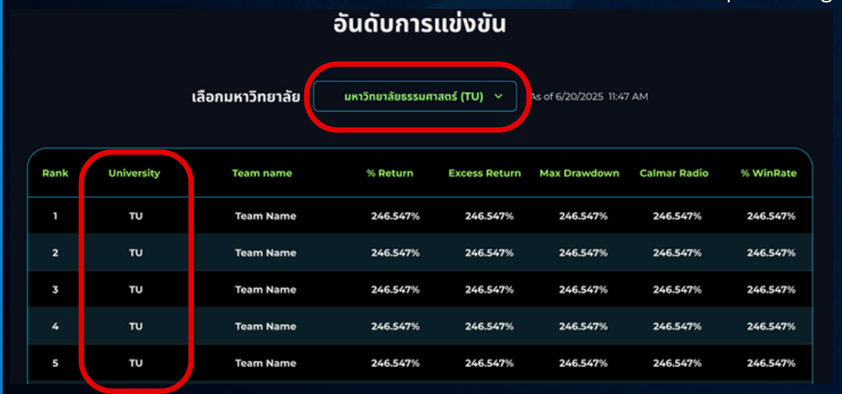
การแข่งขันลีก TU, MU, CU จัดในรูปแบบ **Internal League**

ทีมจากมหาวิทยาลัยเดียวกันจะจัดอันดับแข่งขันกันภายใน โดยพิจารณาจากประสิทธิภาพ

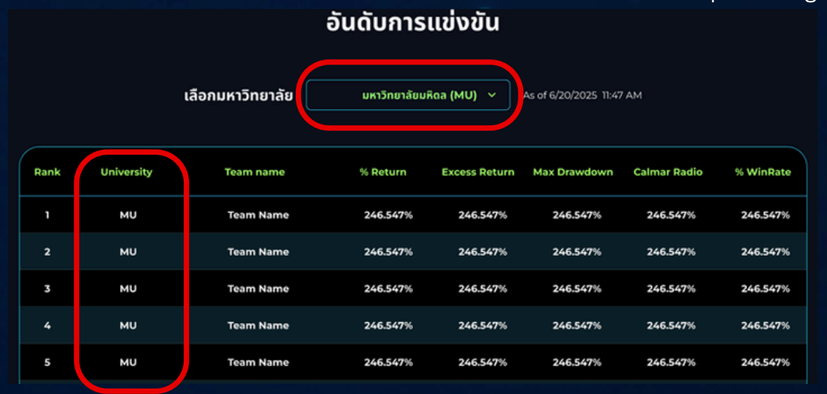
ของกลยุทธ์ Algorithmic Trading ตามเกณฑ์ที่บริษัทฯ กำหนด

จนจบระยะเวลาการแข่งขัน 20 วันทำการ

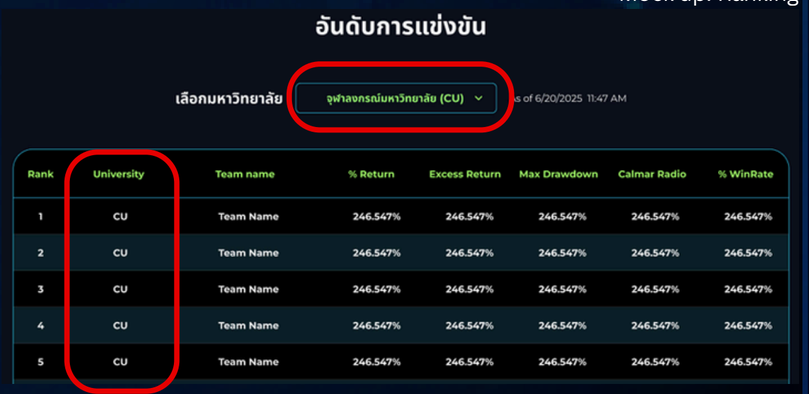
Mock up: Ranking



Mock up: Ranking



Mock up: Ranking



Competition Details

รูปแบบการจัดอันดับ (Competition Ranking)

การแข่งขันลีก 3 พระจอมเกล้าฯ จัดในรูปแบบ **Cross-University League**

โดยนำทีมจาก

- มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี (มจธ.)
- มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ (มจพ.)
- สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง (สจล.)

มาจัดอันดับร่วมกัน โดยพิจารณาจากประสิทธิภาพของกลยุทธ์ Algorithmic Trading

ตามเกณฑ์ที่บริษัทฯ กำหนด จนจบระยะเวลาการแข่งขัน 20 วันทำการ

Mock up: Ranking

อันดับการแข่งขัน

เลือกมหาวิทยาลัย: มหาวิทยาลัยเทคโนโลยีพระเจ้าเกล้า (KMITL x KMUTT x KMUNB) — As of 6/20/2025 11:47 AM

Rank	University	Team name	% Return	Excess Return	Max Drawdown	Calmar Radio	% WinRate
1	KMITL	Team Name	246.547%	246.547%	246.547%	246.547%	246.547%
2	KMUTT	Team Name	246.547%	246.547%	246.547%	246.547%	246.547%
3	KMITL	Team Name	246.547%	246.547%	246.547%	246.547%	246.547%
4	KMITL	Team Name	246.547%	246.547%	246.547%	246.547%	246.547%
5	KMUTT	Team Name	246.547%	246.547%	246.547%	246.547%	246.547%
6	KMUTT	Team Name	246.547%	246.547%	246.547%	246.547%	246.547%
7	KMUTT	Team Name	246.547%	246.547%	246.547%	246.547%	246.547%
8	KMUTT	Team Name	246.547%	246.547%	246.547%	246.547%	246.547%
9	KMUTT	Team Name	246.547%	246.547%	246.547%	246.547%	246.547%
10	KMITL	Team Name	246.547%	246.547%	246.547%	246.547%	246.547%

1 2 3 4 5 6

Competition Details

รางวัลทุนการศึกษา (KGI Scholarship Awards)

Category	Award	Number of Awards	Prize
Main Award	Best Alpha Award (Champion)	1	40,000
Main Award	Best Alpha Award (1st Runner - Up)	1	30,000
Main Award	Best Alpha Award (2nd Runner - Up)	1	20,000
Main Award	Best Risk Control Award	1	10,000
Additional Award*	Most Active Trader Award	5	5,000
Additional Award*	Precision Trader Award	5	5,000

Q & A

Contact Persons

01

คุณ ณัฐดนัย สรรเสริญ (คุณมีน)
รองผู้จัดการฝ่ายธุรกรรมอิเล็กทรอนิกส์
Natdanais@kgi.co.th
02-658-8966



02

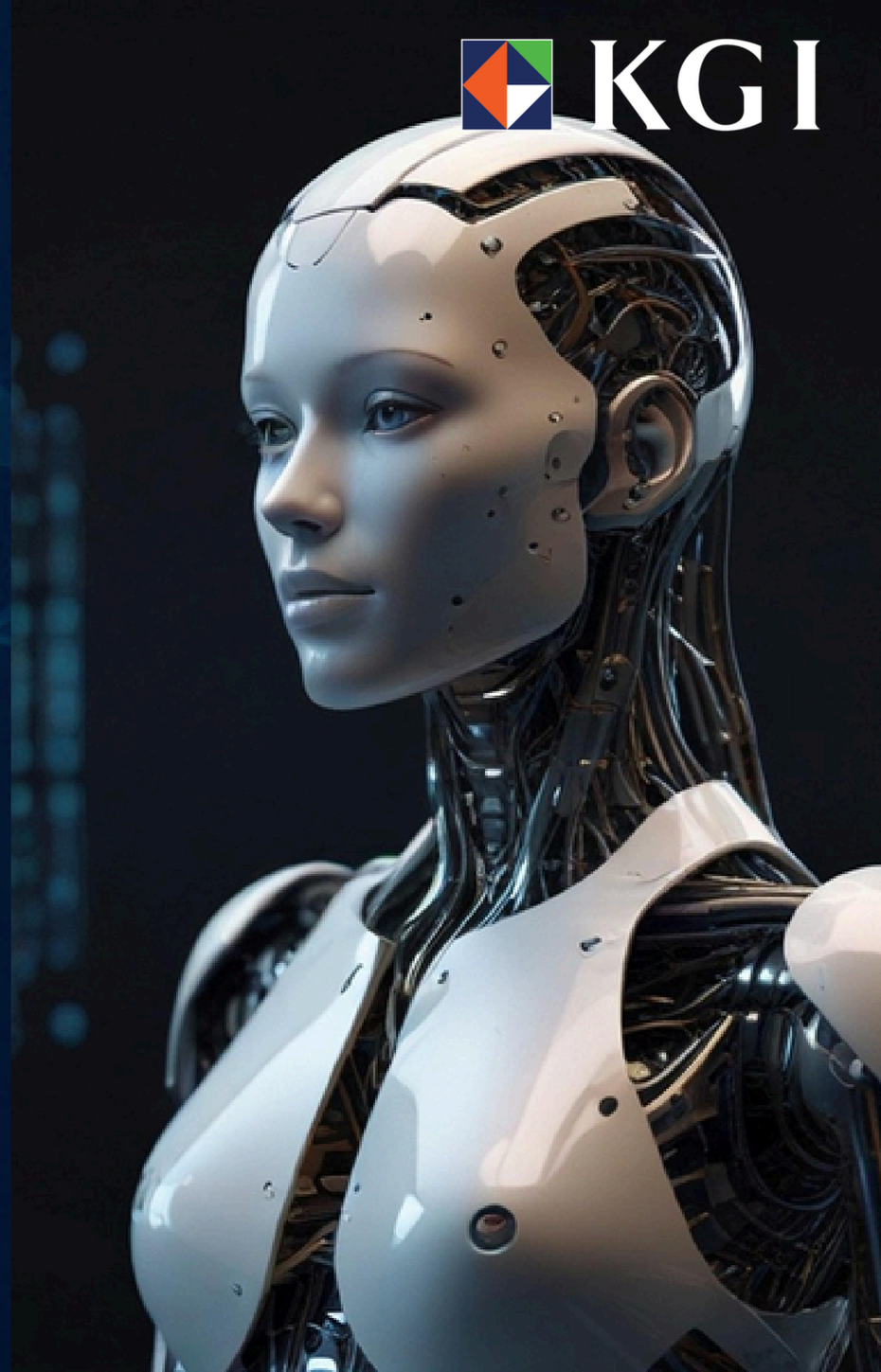
คุณ ปรเมษฐ์ แก้วประดับ (คุณคอปเตอร์)
Programmer / Analysis
Porrametk@kgi.co.th
02-658-8420



03

คุณ ธนัท พิชิตพันธ์พงศ์ (คุณภู)
Programmer / Analysis
Thanatp@kgi.co.th
02-658-8421





Thank You

KGI Securities (Thailand) Public Company Limited
195 One Bangkok Tower 4, 18th –19th Floor,
Wireless Road, Lumpini,
Pathumwan, Bangkok 10330

02-658-8777

ebs@kgi.co.th

https://www.kgieworld.co.th

