ระบบเทรดทองคำอัตโนมัติ

xAuby

คู่มือฉบับสมบูรณ์ — สถาปัตยกรรม การตั้งค่า กลยุทธ์ Market Regime Detector การจัดการความเสี่ยง และการดูแลระบบ

รวบรวมและเรียบเรียงจากเอกสารทั้งหมดของโปรเจกต์
(README, README_DEV, CLAUDE.md, DESIGN.md, PRODUCT.md, docs/*)
อ้างอิงสถานะระบบล่าสุด ณ กรกฎาคม 2026
เอกสารนี้จัดทำเพื่อการศึกษาและใช้งานภายในเท่านั้น

สารบัญ

คำนำ และสถานะระบบล่าสุด

ภาคที่ 1 — รู้จัก xAuby

1.1 xAuby คืออะไร

1.2 ปรัชญาการออกแบบ

1.3 สถานะการทำงานปัจจุบัน (Current Runtime Baseline)

1.4 สิ่งที่บอทนี้ทำ และไม่ทำ

ภาคที่ 2 — เริ่มต้นใช้งาน

2.1 การติดตั้ง

2.2 การตั้งค่า API Key ของ OKX

2.3 รันโหมดจำลองก่อนเสมอ

2.4 คำสั่งใช้งานที่พบบ่อย

2.5 เช็กลิสต์ก่อนเปิดใช้งานจริง (Live)

ภาคที่ 3 — สถาปัตยกรรมระบบ

3.1 ภาพรวมโครงสร้างเลเยอร์

3.2 หนึ่ง Tick ของเอนจิ้นทำอะไรบ้าง

3.3 PairRegistry และ SymbolContext

3.4 กฎการพึ่งพาของระบบ (Dependency Rules)

3.5 ปลั๊กอิน Strategy และ Indicator

3.6 Feature Flags ทั้งหมด (architecture.*)

ภาคที่ 4 — การตั้งค่าระบบ (Configuration)

4.1 ไฟล์ตั้งค่า 3 ไฟล์หลัก

4.2 หลักการ Source of Truth 3 ชั้น

4.3 การตั้งค่า Exchange (OKX ผ่าน CCXT)

4.4 โครงสร้าง Whitelist และคู่เทรดปัจจุบัน

4.5 ตัวแปรแวดล้อม (Environment Variables)

4.6 เมนู Quick Config ในตัว

ภาคที่ 5 — กลยุทธ์และ Market Regime Detector

5.1 กลยุทธ์ (Strategy Plugin) ทั้งหมดในระบบ

5.2 CDC Action Zone — กลยุทธ์หลักที่ใช้งานจริง

5.3 Market Regime Detector คืออะไร ทำงานอย่างไร

5.4 ตัวจับ Regime เชิงสถิติ (GMM Cross-check)

5.5 รายงานความแม่นยำของ Regime Detector
5.6 RegimeRouter — เลือกกลยุทธ์ตาม Regime อัตโนมัติ
5.7 กระบวนการคัดเลือกกลยุทธ์ด้วยข้อมูลจริง (R&D)

ภาคที่ 6 — การจัดการความเสี่ยง

6.1 การคำนวณขนาดโพซิชัน (Position Sizing)
6.2 Stop-Loss / Trailing / Breakeven
6.3 Partial Take-Profit
6.4 Drawdown Guard — สวิตช์ตัดขาดทุนระดับพอร์ต
6.5 การเทรด Short และเงื่อนไขความปลอดภัย
6.6 เช็กลิสต์ความสอดคล้องของค่าความเสี่ยง

ภาคที่ 7 — Backtest และงานวิจัยกลยุทธ์

7.1 แหล่งข้อมูลสำหรับ Backtest
7.2 คำสั่ง Backtest และ Optimizer
7.3 Regime-aware Replay
7.4 Replay Validation หลัง Restart/Incident

ภาคที่ 8 — หน้าจอใช้งาน (TUI / WebUI)

8.1 Textual TUI (แดชบอร์ดในเทอร์มินัล)
8.2 WebUI มือถือ (Read-only)
8.3 ปรัชญาการออกแบบหน้าจอ

ภาคที่ 9 — Telegram

9.1 การติดตั้ง Telegram Bot
9.2 คำสั่งทั้งหมด
9.3 การแจ้งเตือนอัตโนมัติ
9.4 โหมด Semi-auto และ Emergency Pause

ภาคที่ 10 — หลายบัญชี และหลาย Exchange

10.1 รันหลายอินสแตนซ์บนเครื่องเดียว
10.2 Multi-Exchange ผ่าน CCXT
10.3 OKX Perpetual และความปลอดภัยของ Short

ภาคที่ 11 — การดูแลระบบ (Operations)

11.1 การ Deploy บน VPS
11.2 คำสั่งอัปเดตและรีสตาร์ท
11.3 การตรวจสอบ Latency ของ VPS
11.4 เช็กลิสต์ความปลอดภัยฉบับเต็ม

ภาคผนวก

ก. อภิธานศัพท์ (Glossary)
ข. โครงสร้างโปรเจกต์

ค. ตารางสคริปต์อ้างอิงทั้งหมด

ง. คำเตือนความเสี่ยง (Disclaimer)

คำนำ

คำนำ และสถานะระบบล่าสุด

หนังสือเล่มนี้รวบรวมเอกสารทั้งหมดของโปรเจกต์ xAuby — `README.md`, `README_DEV.md`, `CLAUDE.md`, `DESIGN.md`, `PRODUCT.md` และเอกสารทั้งหมดใน `docs/` — มาจัดเรียงใหม่เป็นลำดับการอ่านเดียว แปลและเรียบเรียงเป็นภาษาไทยที่อ่านง่าย ตัดส่วนที่ซ้ำซ้อนออก และ**แก้ไขจุดที่เอกสารต้นฉบับล้าสมัย (doc drift)** ให้ตรงกับสถานะจริงของระบบ ที่อ่านได้จากไฟล์ config จริง (`bot_config.yaml`, `coin_whitelist.json`)

> **หมายเหตุสำคัญเรื่องความคลาดเคลื่อนของเอกสารเดิม:** เอกสารต้นฉบับหลายไฟล์ (โดยเฉพาะ `README_DEV.md`, `docs/architecture.md`, `docs/trading-flow.md`, `docs/configuration.md`, `docs/tui.md`, `docs/webui.md`, `docs/telegram.md`) ยังอธิบายสถานะเก่าของระบบ (Binance.th แบบ Spot, คู่เทรด `XAUTUSDT` + `BTCUSDT` สองคู่พร้อมกัน, `risk_pct 0.45`) ซึ่ง `CLAUDE.md` เองก็ระบุไว้ตรงๆ ว่าเป็น "doc drift warning" หนังสือเล่มนี้ยึด **`coin_whitelist.json` และ `bot_config.yaml` จริงเป็นความจริงหลัก (ground truth)** ตามที่ CLAUDE.md กำหนดไว้ และจะระบุชัดเจนทุกจุดที่เนื้อหาอธิบาย "ความสามารถของระบบ" (เช่น รองรับหลายคู่เทรด, รองรับหลาย exchange) แยกจาก "สิ่งที่กำลังทำงานจริงตอนนี้" (คู่เทรดเดียวคือทองคำบน OKX)

สถานะการทำงานจริงของระบบ ณ ปัจจุบัน (สรุปสั้น)

รายการ	ค่าจริงในระบบ
Exchange	OKX — USDT-settled Perpetual Swap ผ่าน CCXT adapter
คู่เทรดที่ Live	`XAUUSDT` (ทองคำ) เพียงคู่เดียว
กลยุทธ์	`cdc_action_zone` แบบ Long + Short (stop-and-reverse)
Timeframe	หลัก 4 ชั่วโมง (4h) / ยืนยัน 1 วัน (1d)
Leverage	1 เท่า (isolated margin, one-way position)
ความเสี่ยงต่อไม้	`risk_pct = 0.02` (2% ของ Equity)
Max allocation ต่อไม้	25%
Max daily loss	6%
Max open positions	1
Stop-Loss	ปิดใช้งาน (`disable_stop_loss: true`) — เป็นกลยุทธ์ CDC-pure ออกด้วยการกลับโซนเท่านั้น
Partial Take-Profit	ปิด 50% ของโพซิชันเมื่อกำไร +12%
RegimeRouter	ปิดอยู่สำหรับคู่ XAU (`regime_router_enabled: false`)

ตัวเลขเหล่านี้เปลี่ยนแปลงได้ตามการตั้งค่าจริงของผู้ดูแลระบบ — ให้ตรวจสอบ `bot_config.yaml` และ `coin_whitelist.json` เสมอก่อนอ้างอิงในการตัดสินใจจริง

ภาคที่ 1 — รู้จัก xAuby

1.1 xAuby คืออะไร

xAuby คือระบบเทรดอัตโนมัติสำหรับสัญญา Perpetual Swap แบบ single-process รันได้หลายคู่เทรดพร้อมกัน (multi-symbol) บนตลาด **OKX** (`api.okx.com`) เชื่อมต่อผ่านชั้นกลาง (adapter) ที่เป็นกลางกับ exchange ชื่อ **CCXT** ระบบดึงราคาแท่งเทียนและราคาล่าสุด ผ่านทั้ง REST API และ WebSocket, รันแต่ละคู่เทรดผ่านปลั๊กอินกลยุทธ์ของตัวเอง, สามารถส่งผ่านตัวจำแนกสภาพตลาด (Market Regime Classifier) เพื่อเลือกกลยุทธ์อัตโนมัติได้, วางออเดอร์ Swap จริงหรือจำลอง พร้อมระบบ Stop-Loss แบบ ATR / Trailing / Breakeven แล้วบันทึกทุกอย่างลง SQLite และไฟล์ JSONL เพื่อใช้ Replay ตรวจสอบเหตุการณ์ (Incident) และแสดงผลบน Textual Dashboard ผู้ดูแลระบบควบคุมผ่าน CLI, Textual TUI, และ Telegram

ชื่อแพ็กเกจ Python คือ `xauby` รองรับ Python ตั้งแต่ 3.10 ขึ้นไป (README กำหนดเป้าหมายที่ 3.12+ ส่วน container พัฒนาปัจจุบันรันบน 3.11)

1.2 ปรัชญาการออกแบบ

หลักการออกแบบที่ยึดถือมาตลอดทั้งระบบ:

- **Config ขับเคลื่อนพฤติกรรม** — เอนจิ้นไม่ hard-code รายชื่อคู่เทรด, กลยุทธ์, หรือเส้นกราฟใดๆ ทั้งสิ้น ทุกอย่างมาจากไฟล์ตั้งค่า
- **เอนจิ้นไม่รู้เรื่องกลยุทธ์ (Strategy-agnostic)** — ตรรกะสัญญาณซื้อขายและจุดออกทั้งหมดอยู่ในปลั๊กอิน กลยุทธ์ ไม่ใช่ในเอนจิ้นหลัก การเพิ่มกลยุทธ์ใหม่ไม่ต้องแก้โค้ดเอนจิ้น
- **แยกความรับผิดชอบ Config เป็น 3 ชั้น** — Engine / Strategy / Portfolio (อธิบายละเอียดในภาคที่ 4) เพื่อไม่ให้ Backtest กับ Live ให้ผลต่างกัน (parity)
- **ปลอดภัยไว้ก่อน (Fail-safe by default)** — ค่าเริ่มต้นเป็นโหมดจำลองเสมอ, การเทรด Live ต้องผ่าน "ประตู" หลายชั้นพร้อมกัน (ดูหัวข้อ 2.5), Router ที่ยังไม่ผ่านการยืนยันจะถูกบังคับกลับเป็น Sim อัตโนมัติ
- **Isolation ต่อคู่เทรด** — แต่ละคู่เทรดที่ Active มี Strategy instance, Runner, โหมด, และ สถานะ handoff ของตัวเอง แยกจากคู่อื่นโดยสมบูรณ์ ส่วนเงินทุนพอร์ตยังคงใช้ร่วมกันโดยเจตนา
- **Rollback คือการปิดสวิตช์ ไม่ใช่ revert โค้ด** — ฟีเจอร์ใหญ่ๆ (RegimeRouter, Event Bus, GMM Cross-check ฯลฯ) ทุกตัวอยู่หลัง feature flag ใน `architecture:` block พลิกกลับเป็น `false` ได้ทันทีโดยไม่ต้องแก้โค้ด

1.3 สถานะการทำงานปัจจุบัน (Current Runtime Baseline)

ตารางด้านล่างคือค่าจริงที่อ่านจาก `coin_whitelist.json` และ `bot_config.yaml` ของ repository ในปัจจุบัน:

คู่เทรด	โหมด	กลยุทธ์	Primary TF	Confirm TF	ฝั่งที่เปิดได้
`XAUUSDT`	live	`cdc_action_zone`	4h	1d	Long + Short

เอนจิ้นรัน exchange เป็น **OKX Perpetual Swap** (`exchange.provider: ccxt` , `ccxt_id: okx` , `market_type: swap` , `margin_mode: isolated` , `position_mode: one_way`) ที่ **Leverage 1 เท่า** (`derivatives.max_leverage: 1`) พร้อม funding guard (`max_abs_funding_rate`) และ `min_liquidation_distance_pct` ทำงานอยู่ ช่องทองใช้

สัญลักษณ์ `XAUUSDT` (Perpetual) ส่วนงาน Backtest ใช้ `PAXGUSDT` เป็นตัวแทนราคาทอง (proxy) เพราะมีประวัติราคาย้อนหลังยาวกว่าบน Global Binance

คู่เทรดทองคำตัวเดียวที่ Live รันกลยุทธ์ CDC Action Zone แบบ **Long และ Short พร้อมกัน** ในลักษณะ stop-and-reverse (`allowed_sides: [long, short]` , `short_live_enabled: true`) เพราะ adapter ของ Swap ประกาศความสามารถ `swap` / `positions` / `reduce_only` ครบ (`xauby/api/ccxt_client.py`)

รูปแบบการส่งออเดอร์: การเข้าไม้ใช้ออเดอร์ **LIMIT** ที่ราคาล่าสุด (`execution.order_type: limit`) และเมื่อ `execution.entry_market_fallback: true` ส่วนที่ยังไม่ Fill จะถูกเติมด้วยออเดอร์ **MARKET** หลังหมดเวลารอ (แทนที่จะยกเลิก) เพื่อให้ Live ตรงกับผล Backtest ที่จำลองการ Fill แบบ Market ส่วนการออกไม้ใช้ MARKET เมื่อเป็นสถานการณ์เร่งด่วน (CDC แดง / NO_TRADE / บังคับปิด) และใช้ LIMIT ในกรณีอื่น

คู่ทองปัจจุบันเป็นแบบ **CDC-pure** (`disable_stop_loss: true`) จึงไม่มี Stop-Loss ฝั่ง Exchange — การออกจากโพซิชันขับเคลื่อนด้วยการกลับโซนเท่านั้น (zone-flip) และการคำนวณขนาดไม้ ใช้ `position_pct` คงที่ของ Equity แทนระยะห่างจาก SL นอกจากนี้ CDC ยังเปิด Partial Take-Profit แบบ one-shot: `partial_tp_pct: 12.0` , `partial_tp_fraction: 0.5` — เอนจิ้นจะปิด 50% ของโพซิชันเมื่อกำไรถึง +12% บันทึกสถานะ `trade_states.partial_tp_taken` แล้วปล่อยส่วนที่เหลือให้ออกตาม CDC ต่อไป

> **ประตูความปลอดภัยของ Router:** คู่เทรดที่ `mode: live` และเปิด `regime_router_enabled: true` จะถูกบังคับกลับเป็น `sim` โดยอัตโนมัติ เว้นแต่จะมี `regime_router_live_confirmed: true` ด้วย — **ห้ามพลิกค่านี้โดยไม่ผ่าน การอนุมัติจากผู้ดูแลระบบต่อคู่เทรดนั้นๆ โดยตรง** ส่วน Regime ที่เป็น NO_TRADE (`PANIC_SELL` , `BEAR_BREAKDOWN` , `BEAR_TREND_STRONG`) จะปิดกั้นการ BUY ใหม่ทั้งหมด, คงการป้องกันของ Stop ไว้, รัด Trailing ให้แคบลงเหลือ 1 เท่าของ ATR และสามารถบังคับปิดโพซิชันได้หลังครบจำนวนแท่งที่กำหนด (`force_close_candles`)

1.4 สิ่งที่บอทนี้ทำ และไม่ทำ

สิ่งที่บอททำ

- ประเมินสัญญาณกลยุทธ์บนแท่งเทียนที่ตั้งค่าไว้
- เปิด/ปิดโพซิชันจริงบน Exchange หรือโพซิชันจำลอง
- รักษาการป้องกันด้วย Stop-Loss และเกตความเสี่ยงระดับ Local
- บันทึก State/Event สำหรับ Dashboard, Replay, ตรวจสอบ Incident, และ Telegram

สิ่งที่บอทไม่ทำ

- ไม่รับประกันกำไร
- ไม่จัดการเงินในกระเป๋า OKX อื่นที่ไม่เกี่ยวข้องกับบัญชี Swap ที่ตั้งค่าไว้
- ไม่ทำให้ปลั๊กอินกลยุทธ์จากภายนอกปลอดภัยอัตโนมัติ เว้นแต่เปิด Sandbox check
- ไม่ทดแทนการตรวจสอบความเสี่ยง, Leverage, ค่าธรรมเนียม, Funding Rate, และกฎของ Exchange ด้วยตัวเองของผู้ใช้

> **คำเตือนสำหรับผู้ใช้ใหม่:**
> 1. Repository นี้ตั้งค่าไว้สำหรับตลาด Live เดียวคือ OKX `XAUUSDT` USDT-settled perpetual swap (CCXT map เป็น `XAU/USDT:USDT`)

2. การรันโหมดจำลอง (`python run_xauby.py --simulate`) ต้องใช้งานได้ก่อนเสมอ ก่อนพิจารณาออเดอร์ Live ใดๆ
3. การเทรด Live ต้องมี "ประตู" 3 อย่างพร้อมกัน: flag `--live` , `LIVE_TRADING=true` , และคู่เทรดที่ Active ตั้ง `mode: live` ใน `coin_whitelist.json`
4. บอทอ่านเฉพาะบัญชี Trading/Swap ของ OKX ไม่ใช่ทุกกระเป๋าใน OKX — ถ้า Dashboard แสดงพอร์ต `0.00 USDT` ให้ตรวจสอบว่าเงินอยู่ในบัญชีที่ใช้เทรด USDT Swap จริง และ API Key มีสิทธิ์อ่าน/เทรดบัญชีนั้น
5. **ห้ามเปิดสิทธิ์ Withdraw บน API Key เด็ดขาด** บอทต้องการแค่สิทธิ์อ่านและเทรดเท่านั้น
6. รันเอนจิ้นเพียง 1 instance ต่อ 1 runtime root — ถ้าต้องการหลายบัญชี ให้ใช้ค่า `XAUBY_HOME` หรือ `XAUBY_INSTANCE_ID` แยกกัน

ภาคที่ 2 — เริ่มต้นใช้งาน

2.1 การติดตั้ง

สิ่งที่ต้องมี: Python 3.12 ขึ้นไป, API Key ของ Exchange ที่ตั้งค่าไว้แล้ว, Telegram Bot (ไม่บังคับ)

```
git clone https://github.com/iisara555/xAuby.git
cd xAuby

python3 -m venv venv
source venv/bin/activate
pip install -r requirements.txt
pip install -e .

cp .env.example .env
python run_xauby.py --simulate
python health_check.py
```

2.2 การตั้งค่า API Key ของ OKX

ก่อนเทรด Live ให้แก้ไข `.env` และใส่ Credential ของ OKX:

```
OKX_API_KEY=...
OKX_API_SECRET=...
OKX_API_PASSPHRASE=...
LIVE_TRADING=true
```

OKX ต้องการทั้ง API Key, Secret, และ Passphrase (ตัวแปรใดตัวแปรหนึ่งจาก `OKX_API_PASSPHRASE`, `OKX_PASSWORD`, หรือ `OKX_API_PASSWORD`) — **อย่าคอมมิตไฟล์ `.env` เด็ดขาด**

ตรวจสอบคู่เทรดและการตั้งค่า Exchange ก่อนใช้งานจริง:

```
python scripts/evaluate_okx_xau_migration.py --config configs/okx_xau_paper.yaml --whitelist configs/ok
DEFAULT_SYMBOL=XAUUSDT python health_check.py
```

2.3 รันโหมดจำลองก่อนเสมอ

โหมดเริ่มต้นของระบบคือ **Simulation** เสมอ ไม่ว่าจะตั้งค่าอะไรผิดพลาด ระบบจะเอนเอียงไปทางปลอดภัย การสลับไป Live ต้องตั้งค่าอย่างชัดเจนตามหัวข้อ 2.5

Entry point อื่นๆ ที่ใช้ได้:

คำสั่ง	ทำอะไร
`python run_xauby.py --simulate` / `--live`	เริ่มเอนจิ้นเทรดโดยตรง
`python launcher.py`	เมนู Launcher แบบโต้ตอบ (เอนจิ้น + เมนู TUI)
`xauby` / `xauby --sim` / `xauby --live`	Console script ที่ติดตั้งแล้ว (`xauby.cli:main`)
`xauby restart [--live]`	รีสตาร์ทแบบควบคุม (สาย Live รัน preflight ผ่านสคริปต์เฉพาะ)
`xauby update`	ดึงโค้ดจาก GitHub แล้วรีสตาร์ท
`python -m xauby.ui.textual_tui.app`	เปิด TUI แบบเดี่ยว
`python health_check.py`	ตรวจสุขภาพระบบครั้งเดียว
`bin/xauby`	Bash wrapper ที่ cd เข้าโฟลเดอร์โปรเจกต์และเรียก `venv/bin/python`

2.4 คำสั่งใช้งานที่พบบ่อย

คำสั่งแนะนำสำหรับเริ่ม Live หลังผ่านการจำลองและ Health Check แล้ว:

```
env -u DEFAULT_SYMBOL XAUBY_DEFAULT_SYMBOL=XAUUSDT XAUBY_FOCUS_SYMBOL=XAUUSDT \
  SIMULATE_ONLY=false BOT_READ_ONLY=false \
  ./venv/bin/python run_xauby.py --live --pair XAUUSDT
```

WebUI แบบอ่านอย่างเดียวสำหรับดูจากมือถือ/เบราว์เซอร์:

```
./scripts/start_webui.sh
```

คำสั่งที่ใช้บ่อยระหว่างดูแลระบบ:

```
xauby restart        # รีสตาร์ทเอนจิ้น; โหมด Live รัน preflight แบบควบคุม
xauby restart --live # บังคับ restart แบบควบคุมสำหรับ Live
xauby update          # ดึง origin/main จาก GitHub แล้วรีสตาร์ท
```

`xauby update` เป็นทางลัดของคำสั่ง:

```
scripts/deploy_from_github.sh --restart --branch=main
```

เส้นทาง Controlled Restart จะยอมรับโพซิชันที่บอทติดตามอยู่แล้ว (รวมออเดอร์ Stop-Loss ที่ตรงกัน) แต่จะปิดกั้นเมื่อพบยอดคงเหลือที่ไม่รู้จักหรือออเดอร์เปิดที่ไม่ถูกติดตาม

2.5 เช็กลิสต์ก่อนเปิดใช้งานจริง (Live)

การเทรด Live ต้องผ่านทุกเงื่อนไขพร้อมกัน:

1. flag `--live` ตอนรันคำสั่ง

2. ตัวแปรแวดล้อม `LIVE_TRADING=true`
3. `simulate_only: false` ใน YAML
4. คู่เทรดนั้นตั้ง `mode: live` ใน whitelist

โหมด	วิธีเปิด	พฤติกรรม
Simulation	`--simulate`, `simulate_only: true`, หรือ per-asset `mode: sim`	ไม่มีออเดอร์จริง; ใช้บัญชี SimBroker เมื่อเปิดใช้งาน
Live	`--live` + `LIVE_TRADING=true` + `simulate_only: false` + per-asset `mode: live`	ออเดอร์จริงบน Exchange
Read-only	`--read-only` หรือ `read_only: true`	ประมวลสัญญาณอย่างเดียว ไม่ส่งออเดอร์
Semi-auto	`trading.mode: semi_auto`	รอยืนยันผ่าน Telegram ก่อน BUY
Telegram Pause	`/pause` แล้วกด Confirm PAUSE	บล็อกการ BUY ใหม่ทั้งระบบ; `/resume` ปลดล็อก

> รายการเช็กลิสต์ก่อน Live แบบละเอียดทั้งหมด (รวมทั้งเรื่อง Risk, Replay Validation, Regime Router Live-confirm) อยู่ที่หัวข้อ **11.4** ท้ายเล่ม — ควรอ่านให้ครบก่อนเปิดเงินจริง

ภาคที่ 3 — สถาปัตยกรรมระบบ

3.1 ภาพรวมโครงสร้างเลเยอร์

xAuby แบ่งเป็น 5 เลเยอร์ที่ทำงานร่วมกัน:

เลเยอร์	ส่วนประกอบหลัก	บทบาท
Operator	`launcher.py`, `run_xauby.py`, Textual TUI, Telegram	จุดเริ่มต้นและควบคุมจากมนุษย์
Core Runtime	`LiteTradingEngine`, `PairRegistry`, `SymbolContext`, `RegimeRouter`, `StrategyRunner`, Risk/PreTradeGate, Order Executor/SimBroker	แกนกลางของระบบเทรด
Strategy Plane	Strategy plugins, Indicator plugins, Strategy/Portfolio Resolver	ตรรกะสัญญาณและการแสดงผลกราฟ
Data Plane	CCXT REST client, WebSocket, LiteDB/SQLite	ดึงข้อมูลตลาดและเก็บ state
Observability	EventEmitter, JSONL events, State JSON, HealthMonitor	บันทึกเหตุการณ์เพื่อ Replay/ตรวจสอบ/แสดงผล

ลำดับการไหลของข้อมูล: Launcher/CLI เรียก Engine → Engine ดึง Pair จาก Registry → สร้าง SymbolContext ต่อคู่เทรด → ผ่าน RegimeRouter (ถ้าเปิดใช้) → StrategyRunner ประเมินสัญญาณ → ผ่าน Risk/PreTradeGate → ส่งออเดอร์จริงหรือจำลอง → บันทึกลง DB และ Event → อัปเดต State JSON ให้ TUI/WebUI อ่าน

3.2 หนึ่ง Tick ของเอนจิ้นทำอะไรบ้าง

เอกสารนี้อธิบายการทำงานหนึ่ง "Tick" ของ `LiteTradingEngine` สำหรับหนึ่งคู่เทรด โดย Tick ชั้นนอกจะวนลูปทุกคู่เทรดที่ Active จาก PairRegistry

ลำดับขั้นตอนต่อคู่เทรด

1. Sync แท่งเทียนผ่าน REST
2. ตรวจสอบว่าราคาจาก WebSocket เก่าเกินไปหรือไม่ — ถ้าใช่ ใช้ REST ticker แทน, ถ้าไม่ ใช้ค่าจาก WS
3. อ่านสถานะการเทรดปัจจุบัน (Trade State)
4. Resolve การตั้งค่ากลยุทธ์และพอร์ตของคู่เทรดนั้น
5. ถ้า RegimeRouter Active สำหรับคู่นี้: จำแนก Regime + Debounce + เลือกเส้นทาง; ถ้าไม่: ใช้กลยุทธ์ที่ตั้งค่าไว้ตรงๆ
6. ถ้า Regime เป็น NO_TRADE: บล็อกการ BUY ใหม่และรัด Protection ให้แน่นขึ้น; ถ้าไม่: ให้ StrategyRunner ประเมินสัญญาณ
7. จัดการโพซิชันเดิม / recovery / บังคับปิด (ถ้าอยู่ในเงื่อนไข NO_TRADE)
8. ตรวจ Macro Guard ก่อนอนุญาต BUY — ถ้าบล็อก จะ HOLD พร้อมแจ้งเตือน

9. ตามสถานะ+Action: `idle+BUY` → เปิดไม้ / คิว semi-auto, `bought+SELL` → ปิดไม้, `bought+HOLD` → ประมวล Partial TP / Trailing SL / Breakeven
10. บันทึก Trade State + Event ทั้งหมด
11. อัปเดต State JSON ให้ TUI อ่านต่อ

เส้นทางเข้าไม้ (Entry Path)

1. Protection ตรวจจำนวนไม้ต่อวัน, % ขาดทุนต่อวัน, จำนวนแพ้ติดต่อกัน, และจำนวนโพซิชันเปิดสูงสุด
2. PreTradeGate ตรวจ Min Notional และ Slippage เทียบราคาสัญญาณ
3. คำนวณขนาดไม้จาก Config พอร์ตของคู่เทรดนั้น พร้อมเพดาน Deploy ระดับ Global
4. เส้นทางออเดอร์ใช้ Live Executor หรือ SimBroker ตามโหมดของคู่เทรดนั้น
5. วาง Stop-Loss บน Exchange (โหมด Live) หรือติดตาม/จำลองใน Local (โหมด Sim)
6. ส่งรายงานผ่าน Telegram พร้อม Tag สัญลักษณ์และโหมด

เส้นทางออกไม้ (Exit Path)

ทริกเกอร์การออกจากโพซิชันมีได้จาก: สัญญาณ SELL จากกลยุทธ์, Partial Take-Profit แบบ one-shot, Stop-Loss ที่ Local ยืนยันครบจำนวน tick, การตรวจพบ SL Fill บน Exchange, RegimeRouter บังคับปิดเมื่อ NO_TRADE ครบจำนวนแท่ง, หรือ `max_hold_hours`/ตาราง Minimal ROI (ถ้าตั้งค่าไว้) หากการขายล้มเหลว ระบบจะพยายามกู้คืน Stop-Loss และส่ง Telegram แจ้งเตือนระดับวิกฤตถ้ากู้คืนไม่สำเร็จ

เส้นทาง Partial Take-Profit

Partial TP จะทำงาน**หลัง**ที่ pipeline การออกไม้เต็มมีโอกาสขายก่อนเสมอ — ถ้า Tick เดียวกัน เกิด Full Exit ด้วย Full Exit จะชนะ มิฉะนั้น สำหรับสถานะ `bought` ที่ยังไม่เคยเก็บ Partial TP เอนจิ้นจะเทียบราคา Mark กับเกณฑ์ที่ตั้งจากราคาเข้า:

- LONG: `mark ≥ entry × (1 + partial_tp_pct / 100)`
- SHORT: `mark ≤ entry × (1 − partial_tp_pct / 100)`

เมื่อถึงเกณฑ์ `execute_partial_tp()` จะปิดสัดส่วนที่ตั้งไว้ผ่าน Broker Abstraction, ปรับขนาด Stop-Loss ฝั่ง Exchange (ถ้ามี) ให้ตรงกับจำนวนที่เหลือ, ตั้ง `partial_tp_taken=true`, ส่ง Event `partial_tp_executed` แล้วปล่อยส่วนที่เหลือให้ออกตามกลยุทธ์/CDC ต่อไปตามปกติ

การส่งมอบกลยุทธ์ (Strategy Handoff)

เมื่อ RegimeRouter เปลี่ยนกลยุทธ์ขณะมีโพซิชันเปิดอยู่ ระบบออกแบบให้ปลอดภัยเสมอ:

- โพซิชันเดิมยังคงเป็นของกลยุทธ์เก่าจนกว่าจะปิด
- ไม้ใหม่จะถูกบล็อกหรือรอจนกว่า Handoff จะเสร็จสมบูรณ์
- Handoff จะเสร็จสมบูรณ์ก็ต่อเมื่อโพซิชันปิดแล้วเท่านั้น
- เอนจิ้นส่ง Event สำหรับ Handoff เพื่อการตรวจสอบย้อนหลัง

สิ่งนี้ป้องกันไม่ให้กลยุทธ์ใหม่มาบริหารการออกของโพซิชันที่ตัวเองไม่ได้เป็นคนเปิด

การจัดการ NO_TRADE

สำหรับ Regime ที่แม็พไปเป็น `None`: บล็อกการ BUY ใหม่ทั้งหมด, คง Stop Protection เดิมไว้, รัด Trailing Stop ให้แคบลงเหลือ 1 เท่า ATR (ถ้ามี), ต้องผ่านแท่งที่เทรดได้ครบตาม `regime_router.recovery_candles` ก่อนจะกลับมาเทรดได้ปกติ, และสามารถบังคับปิดได้หลังครบ `regime_router.force_close_candles` แท่งที่เป็น NO_TRADE ติดต่อกัน ปัจจุบัน Regime ที่เป็น NO_TRADE คือ `PANIC_SELL`, `BEAR_BREAKDOWN`, และ `BEAR_TREND_STRONG`

การเทรด Short

กลยุทธ์ที่รองรับ Short จะส่งสัญญาณ `OPEN SHORT` / `CLOSE SHORT` แทนที่จะยัดเข้าไปใน BUY/SELL บน USDT Perpetual การเปิด Short จะแม็พเป็น SELL และการปิดแม็พเป็น BUY แบบ Reduce-only Stop-Loss จะอยู่เหนือราคาเข้า และ Fixed Take-Profit จะอยู่ต่ำกว่าราคาเข้า การกระทบยอดโพซิชัน (Reconciliation) ใช้ Endpoint โพซิชันของ Exchange โดยตรง (ไม่ใช่ยอดคงเหลือแบบ Spot) และจะบล็อกการเทรดถ้าพบ โพซิชันกำพร้าหรือฝั่งไม่ตรงกัน

3.3 PairRegistry และ SymbolContext

`PairRegistry` รวมข้อมูลจาก 3 แหล่ง: (1) สินทรัพย์ใน `coin_whitelist.json` (กลยุทธ์, โหมด, Timeframe ต่อเหรียญ), (2) `data.pairs` ใน YAML เมื่อจำเป็นต้องใช้รายชื่อแบบเก่า/ชัดเจน, และ (3) `DEFAULT_SYMBOL` จาก environment (ถ้ามี) เมื่อ `architecture.whitelist_strict: true` Whitelist จะเป็นแหล่งความจริงเดียว และเมื่อ `hot_reload_enabled: true` ระบบจะ poll mtime ของไฟล์ทุก `reload_interval_seconds` วินาที เมื่อไฟล์เปลี่ยน เอนจิ้นจะ re-init SymbolContext และรีเฟรช WebSocket subscription ใหม่

แต่ละคู่เทรดที่ Active จะได้ `SymbolContext` ของตัวเอง ประกอบด้วย:

- Snapshot ราคาจาก WebSocket พร้อม REST fallback
- กลยุทธ์ปัจจุบันและเป้าหมาย Handoff (ถ้ามี)
- Regime ปัจจุบันและ Snapshot ล่าสุด
- โหมดการทำงานต่อคู่เทรด (`sim` หรือ `live`)
- สถานะรอยืนยันของ Semi-auto
- Metadata สัญญาณล่าสุดสำหรับ TUI และ Telegram

คู่เทรดถูกแยกจากกันในระดับ Strategy/Runner/Context แต่เงินทุนพอร์ตและ Guard ระดับ Global ยังใช้ร่วมกันโดยเจตนา

3.4 กฎการพึ่งพาของระบบ (Dependency Rules)

กฎ	เหตุผล
`observability` ห้าม import `engine`	Replay และ Health ต้องไม่ผูกกับเอนจิ้น
TUI อ่านอย่างเดียว (`LiteDB(readonly=True)`)	หน้าจอต้องไม่บล็อกหรือแก้ไขข้อมูลการเทรด
Secret อยู่ใน `.env` เท่านั้น	`bot_config.yaml` คอมมิตได้ปลอดภัยเพราะไม่มีคีย์ใดๆ
เอนจิ้น 1 instance เท่านั้น	`core/.engine.lock` ป้องกันการเทรดซ้ำซ้อน
Strategy plugin กับ Indicator plugin ต้องไปด้วยกันเสมอ	กราฟ/Legend ต้องตรงกับพฤติกรรมกลยุทธ์จริง
เอนจิ้นต้อง Strategy-agnostic	ตรรกะสัญญาณ/ทางออกอยู่ที่ Config/Plugin เท่านั้น

3.5 ปลั๊กอิน Strategy และ Indicator

นี่คือรูปแบบขยายระบบที่สำคัญที่สุด: **Strategy และ Indicator ต้องมาคู่กันเสมอ** PR ที่เพิ่ม Strategy โดยไม่มี Indicator plugin คู่กันและไม่มี mapping ใน `strategy_chart_indicators` ถือว่ายังไม่สมบูรณ์ เพราะ Dashboard จะแสดงผลไม่ตรงกับพฤติกรรมจริง

Strategy Plugin

อยู่ที่ `xauby/strategies/<name>/strategy.py` — สืบทอดจาก `xauby.strategies.base.Strategy`, ประดับด้วย `@register("<name>")`, Implement `analyze(ctx: MarketContext) -> Signal` (และ `default_config()` / `validate_config()` แบบไม่บังคับ) Strategy ต้องเป็น "นักวิเคราะห์บริสุทธิ์" เท่านั้น — **ห้าม**วางออเดอร์, แตะ DB, เรียก Telegram, หรือเรียกเมธอดของเอนจิ้นโดยตรง เอนจิ้นจะโหลดปลั๊กอินอัตโนมัติผ่าน `load_strategy(name, config)`

Indicator Plugin

อยู่ที่ `xauby/strategies/indicators/indicator_<name>.py` — สืบทอดจาก `Indicator` base class, `compute(df, config)` เติมคอลัมน์ที่มีชื่อ (ห้ามมี ANSI code), `display_config` กำหนดโซน/เส้น/เมตริก/ป้ายชื่อ/สี ป้อนเข้ากราฟเทอร์มินัลและ Legend/Checklist ของ TUI

เช็กลิสต์การเพิ่ม Strategy ใหม่

1. Strategy plugin + `@register`
2. Indicator plugin + `@register` (หรือใช้ตัวที่เข้ากันได้อยู่แล้ว)
3. เพิ่ม `bot_config.yaml -> architecture.strategy_chart_indicators.<strategy>`
4. เขียนเทสต์: strategy test, indicator test, และ chart legend coverage

> กลยุทธ์กลุ่ม `*_short` (`donchian_short` , `supertrend_short` , `rsi2_short`) และปลั๊กอินวิจัยอื่นๆ (`rsi2_meanrev` , `vol_breakout`) ถูก tag ว่า `research` — เอนจิ้นจะ**บล็อกอัตโนมัติ**ไม่ให้กลยุทธ์ที่ tag research รันบนคู่เทรดที่ Live เด็ดขาด

> (`_load_strategy_for_symbol`) จึงใช้ได้แค่ในโหมด Sim/Backtest เท่านั้น ห้าม map เข้า `regime_router.mapping` เพื่อ production

3.6 Feature Flags ทั้งหมด (architecture.*)

ฟีเจอร์ระดับสถาปัตยกรรมทุกตัวถูกควบคุมด้วย Flag ใต้ `bot_config.yaml -> architecture:` การ Rollback คือการพลิก Flag กลับเป็น `false` — ไม่ต้อง revert โค้ด

Flag	หน้าที่
`whitelist_strict`	Whitelist เป็นแหล่งความจริงเดียวของคู่เทรด/กลยุทธ์ พร้อม fail-fast validation
`indicator_registry_enabled`	กราฟคำนวณ Indicator ผ่าน Registry แทนการ hard-code
`sync_yaml_pairs_from_whitelist`	ซิงค์ `data.pairs` จาก whitelist อัตโนมัติ
`tui_indicator_registry`	Checklist/Legend ของ TUI มาจาก Registry Adapter
`per_symbol_execution_mode`	แต่ละคู่เทรดเลือก `sim` / `live` แยกกันได้
`sim_broker_enabled`	ออเดอร์จำลองผ่าน SimBroker แบบมี Ledger จริง
`regime_router_enabled`	สวิตช์หลักของ RegimeRouter (ยังต้องเปิดต่อคู่เทรดด้วย)
`regime_confidence_filter`	ระงับการสลับ Regime เมื่อ Confidence ต่ำกว่าเกณฑ์
`regime_statistical_crosscheck`	เปิดตัวจับ Regime เชิงสถิติแบบ GMM คู่ขนาน (advisory เท่านั้น ดูหัวข้อ 5.4)
`event_bus_enabled`	ส่ง Event ของ EventEmitter ให้ Subscriber ภายในโปรเซส
`exchange_plugin_registry_enabled`	Deprecated/ไม่มีผลจริงแล้ว — Gateway ใช้ Registry เสมอ; เก็บไว้เพื่อ backward-compat
`strategy_sandbox_strict`	ปฏิเสธ (ไม่ใช่แค่เตือน) กลยุทธ์ที่ไม่ผ่าน Static Sandbox Scan
`strategy_validate_strict`	ปฏิเสธกลยุทธ์ที่ `config_errors()` ไม่ว่างตอนโหลด

ภาคที่ 4 — การตั้งค่าระบบ

4.1 ไฟล์ตั้งค่า 3 ไฟล์หลัก

ไฟล์	คอมมิตไหม	หน้าที่
`.env`	ไม่	API key, `LIVE_TRADING`, Telegram
`.env.example`	ใช่	เทมเพลตสำหรับติดตั้งใหม่
`bot_config.yaml`	ใช่	Engine, Strategy, Portfolio, การแจ้งเตือน, retention, optimizer
`coin_whitelist.json`	ใช่	คู่เทรด Active, Timeframe, กลยุทธ์ต่อคู่, โหมด, ประตู Router

ไฟล์ Runtime ใต้ `core/` (DB, log, lock, ยอด sim, cache backtest) เป็นไฟล์เฉพาะเครื่อง ห้ามคอมมิตและห้ามถือว่าเป็นค่าตั้งค่าที่ Portable ข้ามเครื่อง เว้นแต่ตั้งใจจะคัดลอกทั้ง runtime root

4.2 หลักการ Source of Truth 3 ชั้น

Config ถูกแบ่งความรับผิดชอบเป็น 3 ชั้น การใส่ค่าผิดชั้นคือสาเหตุบั๊กที่เกิดซ้ำบ่อยที่สุด เพราะ Backtest, Live, Replay, และ Optimizer ทั้งหมด resolve ผ่าน resolver ตัวเดียวกัน — ถ้าค่าไม่ตรงชั้น จะทำให้ผลไม่ตรงกัน (parity พัง)

ชั้น	เป็นเจ้าของ	ห้ามเป็นเจ้าของ
Engine Config (`bot_config.yaml` ระดับบนสุด)	Exchange, Logging, Monitoring, การแจ้งเตือน, Global risk guard, Feature flag	พารามิเตอร์สัญญาณ/ทางออกของกลยุทธ์
Strategy Config (`strategy.config.<id>`, `strategy.symbols.<SYMBOL>`, whitelist `strategy_params`)	RSI/ATR/Volume, SL, Trailing, Breakeven, Timeframe กลยุทธ์, ตรรกะสัญญาณ/ทางออก	Credential หรือขนาดไม้
Portfolio Config (`portfolio.position_sizing`, `portfolio.symbols.<SYMBOL>.position_sizing`)	ขนาดไม้, การจัดสรร, เงินทุน	กฎ Indicator/สัญญาณ

กฎทองคำ: ค่าใดก็ตามที่เปลี่ยนผลลัพธ์ Backtest ได้ ต้องอยู่ใต้ Strategy หรือ Portfolio Config เท่านั้น ห้ามอยู่ใน Engine-only block เอนจิ้นต้อง Strategy-agnostic เสมอ

4.3 การตั้งค่า Exchange (OKX ผ่าน CCXT)

`bot_config.yaml -> exchange:` คือแหล่งความจริงเดียวของการเลือก Exchange และตัวเลขเศรษฐศาสตร์ระดับ Exchange เอนจิ้น, Backtest, Replay, Health Probe, และสคริปต์ปฏิบัติการทั้งหมด resolve ผ่าน Helper ชุดเดียวกัน (`xauby/runtime/exchange_config.py`) ทำให้ Simulation คงความสอดคล้องกับ Live เสมอ

```yaml
exchange:
  provider: ccxt
  ccxt_id: okx
  name: okx
  quote_asset: USDT
  settle_asset: USDT
  fee_pct: 0.0005
  market_type: swap
  margin_mode: isolated
  position_mode: one_way
  api_key_env: OKX_API_KEY
  api_secret_env: OKX_API_SECRET
  base_url_env: OKX_BASE_URL
  params:
    options:
      defaultType: swap
```

- **Credential** อ่านจากตัวแปรแวดล้อมที่ผูกกับ Exchange OKX ต้องมี `OKX_API_KEY`, `OKX_API_SECRET`, และหนึ่งใน `OKX_API_PASSPHRASE` / `OKX_PASSWORD` / `OKX_API_PASSWORD`
- **ลำดับความสำคัญของค่าธรรมเนียม (Fee):** `exchange.fee_pct` → `backtest.fee_pct` → `trading.fee_pct` → `0.001` ค่า `sim_fee_pct` ต่อคู่เทรด (ใน whitelist) จะชนะสำหรับคู่นั้นโดยเฉพาะ
- CCXT คือ Adapter หลักที่ใช้จริงสำหรับ OKX

4.4 โครงสร้าง Whitelist และคู่เทรดปัจจุบัน

```json
{
  "version": 1,
  "quote_asset": "USDT",
  "assets": [
    {
      "symbol": "XAU",
      "enabled": true,
      "primary_timeframe": "4h",
      "confirm_timeframe": "1d",
      "strategy": "cdc_action_zone",
      "mode": "live",
      "allowed_sides": ["long", "short"],
      "leverage": 1.0,
      "short_live_enabled": true,
      "regime_router_enabled": false,
      "regime_router_live_confirmed": false,
      "sim_fee_pct": 0.05
    }
  ]
}
```

ฟิลด์	ค่าที่เป็นไปได้	หมายเหตุ
`symbol`	สินทรัพย์ฐาน เช่น `XAU`	ระบบจะต่อท้ายด้วย `quote_asset` อัตโนมัติ
`enabled`	bool	สินทรัพย์ที่ปิดจะถูกข้ามไป
`primary_timeframe`	เช่น `1h`, `4h`	ใช้ทั้ง Live, Backtest, และกราฟ
`confirm_timeframe`	เช่น `1d` หรือว่าง	ขึ้นกับกลยุทธ์
`strategy`	plugin id	จำเป็นเมื่อ `whitelist_strict: true`
`strategy_params`	object	ค่าปรับแต่งเฉพาะคู่เทรด รวมเข้ากับ Strategy Config
`mode`	`sim`, `live`	ต้องเปิด `per_symbol_execution_mode: true`
`regime_router_enabled`	bool	ประตู RegimeRouter ระดับคู่เทรด
`regime_router_live_confirmed`	bool	ต้องเป็น true ก่อนคู่เทรด Live จะ Route กลยุทธ์ได้
`sim_fee_pct`	number	หน่วยเปอร์เซ็นต์ เช่น `0.05` = 0.05%

4.5 ตัวแปรแวดล้อม (Environment Variables)

ตัวแปร	ค่าเริ่มต้น	คำอธิบาย
`OKX_API_KEY` / `OKX_API_SECRET` / `OKX_API_PASSPHRASE`	-	Credential สำหรับ OKX REST/WS
`LIVE_TRADING`	`false`	ประตูเพิ่มเติมสำหรับออเดอร์จริง
`SIMULATE_ONLY`	จาก YAML	CLI `--live` / `--simulate` override
`BOT_READ_ONLY`	`false`	ข้ามการวางออเดอร์
`TELEGRAM_ENABLED`	`false`	สวิตช์หลักของการแจ้งเตือน
`TELEGRAM_BOT_TOKEN`	-	Token จาก @BotFather
`TELEGRAM_CHAT_ID`	-	Chat ที่ได้รับอนุญาตให้สั่งงาน
`DEFAULT_SYMBOL`	-	Override คู่เทรดเดี่ยว (ถ้ามี)
`XAUBY_HOME`	`core`	โฟลเดอร์ฐานของ Runtime data — ย้ายเพื่อรันหลาย instance คู่กันได้
`XAUBY_INSTANCE_ID`	-	Sub-namespace ใต้ Runtime root (เช่น tenant/account id)

น้ำหนัก Macro ของ Regime Classifier

ตัวจำแนก Regime รวม DXY, อัตราดอกเบี้ย, และข่าวเป็น Macro Bias น้ำหนักปรับได้เพื่อไม่ให้ระบบผูกติดกับทองคำ เพียงอย่างเดียว:

```
macro_sentiment_guard:
  macro_weights:   # ค่าเริ่มต้น (สมมติฐานสำหรับทองคำ)
    news: 1.0
    dxy: 1.0       # ค่าถูก orient ไว้แล้วที่ producer (บวก=bullish) ห้าม invert ซ้ำ
    fred: 1.0
  regime_bias_threshold: 0.3   # |combined_macro| เกินค่านี้จึงพลิกเป็น RISK-ON/RISK-OFF
```

4.6 เมนู Quick Config ในตัว

Launcher menu → **Quick Configuration** (หรือ `xauby --config`) เปิด Editor แบบ Textual: เมนู `OptionList` แบบจัดกลุ่มที่ผลักเข้าสู่ Submenu ของ Toggle, Modal ตัวเลข/ช่วงค่า, และตัวเลือกแบบเลือก (นำทางด้วยลูกศร/Enter/เมาส์ ค่า secret จะถูกซ่อน) การเขียนค่ายังผ่าน `ruamel.yaml` round-trip เสมอ ทำให้ **คอมเมนต์และโครงสร้างใน `bot_config.yaml` ไม่หายไประหว่างแก้ไข**

กลุ่ม	แก้ไขอะไรได้
Trading & Risk	โหมด (sim/live), % ความเสี่ยงต่อคู่, SL ATR, Breakeven, ตัวกรอง D1, กลยุทธ์
Global Trading	`max_open_positions`, `interval_seconds`, timeframe, min order amount, max position per trade
Risk Guards	Drawdown guard, max daily loss %, max consecutive losses
Portfolio	Initial balance, allocation % ต่อคู่เทรด
Strategy Params	RSI min/max, vol min ratio, trailing ATR mult, cool down ต่อคู่เทรด
Regime Router	เปิด/ปิด, confidence threshold, debounce/recovery/force-close, mapping (เลือกจากกลยุทธ์ที่ลงทะเบียนแล้ว)
Multi-Pair & Dashboard	Timeframe, รายชื่อคู่เทรด, focus ของ Dashboard
Telegram Alerts & Schedule	Credential/test, กำหนดการ weekly/daily digest
Macro Guard	Sentiment guard, FRED, AI news
Exchange	Wizard สลับ Exchange แบบมีไกด์, credential ต่อ Exchange, ทดสอบการเชื่อมต่อ
System Features	Architecture flags ของ per-symbol execution และ event bus

ภาคที่ 5 — กลยุทธ์และ Market Regime Detector

5.1 กลยุทธ์ (Strategy Plugin) ทั้งหมดในระบบ

ระบบมีปลั๊กอินกลยุทธ์ทั้งหมด 15 ตัว บางตัวใช้งานจริง (Production) บางตัวเป็นงานวิจัย (Research/Sim เท่านั้น):

Strategy	Indicator plugin	โซนบนกราฟ	เส้นบนกราฟ
`cdc_action_zone` ⭐Live	`cdc_action_zone`	น้ำเงิน/เขียว/แดง/กลาง	EMA12, EMA26
`supertrend_ema200`	`supertrend`, `ema200`	ST Bull/Bear/กลาง	SuperTrend, EMA200
`bbkc_squeeze`	`bbkc_squeeze`	บีบตัว/ทะลุ/กลาง	Bollinger Bands, Keltner Channels
`bbrsi_mean_reversion`	`bbrsi_mean_reversion`	Oversold/กลาง/Overbought	Bollinger Bands
`btc_ema_pullback`	`btc_ema_pullback`	เทรนด์/พูลแบ็ก/ยึดกลับ/กลาง	EMA Fast/Slow/Trend
`ict_lite_strategy`	`ict_lite`	Sweep Low/Reclaim/MSS/กลาง	EMA Fast/Slow, High/Low ล่าสุด
`rsi2_meanrev` (research)	`rsi2_meanrev`	Buy Setup/Oversold/Exit/กลาง	EMA200, SMA5
`sol_ema_pullback`	`sol_ema_pullback`	เทรนด์/พูลแบ็ก/ยึดกลับ/กลาง	EMA20, EMA50
`vol_breakout` (research)	`vol_breakout`	ทะลุ/ATR ขยายตัว/กลาง	Range High, Exit EMA
`xauby_vwap_pullback`	`xauby_vwap_pullback`	เข้า/พูลแบ็ก/เทรนด์/รอ/เสียเทรนด์	EMA20/50/200, VWAP
`donchian_trend`	-	-	Turtle breakout เหนือ EMA200 + ADX filter
`donchian_short` / `supertrend_short` / `rsi2_short` (research)	-	Short-side เฉพาะงานวิจัย ไม่ใช้ Production	

⭐ = กลยุทธ์ที่รันอยู่บนคู่เทรด Live ปัจจุบัน (XAUUSDT)

5.2 CDC Action Zone — กลยุทธ์หลักที่ใช้งานจริง

`cdc_action_zone` เป็นกลยุทธ์เดียวที่รันบนคู่เทรด Live ปัจจุบัน ทำงานแบบ **Stop-and-reverse** คือเปิด Long หรือ Short สลับกันตามการกลับสีของโซน (Zone Flip) ไม่มี Stop-Loss ฝั่ง Exchange (`disable_stop_loss: true`) — การออกจากไม้ทั้งหมดขับเคลื่อน ด้วยการเปลี่ยนโซนเท่านั้น ขนาดไม้คำนวณจาก `position_pct` คงที่ของ Equity (ไม่ใช่ระยะ SL) และมี Partial Take-Profit แบบ one-shot ที่ +12% เก็บ 50% ของโพซิชัน

พารามิเตอร์ (จาก whitelist จริง)	ค่า
`sl_atr_mult`	2.0
`trailing_atr_mult`	1.8
`breakeven_sl_enabled`	false
`require_fresh_zone`	true (fresh_zone_window: 1)
`require_slow_slope`	true (slow_slope_bars: 3)
`position_pct`	0.95
`partial_tp_pct` / `partial_tp_fraction`	12.0 / 0.5
`backtest_data_proxy`	`PAXGUSDT` (ใช้แทนทองในการ Backtest)

5.3 Market Regime Detector คืออะไร ทำงานอย่างไร

Market Regime Detector (`xauby/regime/classifier.py`, ฟังก์ชัน `classify_market()`) คือตัวจำแนกสภาพตลาดแบบ Rule-based ที่วิเคราะห์ตลาดผ่าน 4 แกนหลัก แล้วสังเคราะห์เป็น "Regime" 11 ประเภท ตามแนวคิด Wyckoff (สะสม → ขาขึ้น → กระจาย → ขาลง):

4 แกนที่ใช้วิเคราะห์

- **Trend** — จาก EMA12/EMA26/EMA200 stack + ระยะห่างราคาจาก EMA200 + Momentum 5/20 แท่ง
- **Volatility** — ATR เทียบราคา แล้วจัดอันดับเทียบ**ค่าเฉลี่ยย้อนหลัง 120 แท่ง** (Relative ไม่ใช่ Absolute — จุดสำคัญที่ทำให้ใช้ข้ามสินทรัพย์/ยุคสมัยได้)
- **Liquidity** — Volume Ratio เทียบ SMA 20 แท่ง
- **Macro** — รวม DXY + อัตราดอกเบี้ย (FRED) + Sentiment ข่าว โดย Clamp แต่ละค่าให้อยู่ในช่วง [-1, 1] ก่อนรวม (ป้องกันค่าใดค่าหนึ่ง เช่น News Score จาก AI ที่ไม่ถูก Clamp ต้นทาง มา Dominate ผลลัพธ์)

รายชื่อ Regime ทั้ง 11 ประเภท

Regime	ความหมาย	Phase
`BULL_BREAKOUT`	ขาขึ้นทะลุพร้อม Volume ยืนยันและ Volatility ขยายตัว	MARKUP
`BULL_TREND_STRONG`	ขาขึ้นแข็งแรง Momentum/EMA spread สูง	MARKUP
`BULL_TREND_WEAK`	ขาขึ้นอ่อนแรงหรือ Macro ขัดแย้ง	MARKUP/RECOVERY_RALLY
`BEAR_BREAKDOWN`	ขาลงทะลุพร้อม Volatility ขยายตัว	MARKDOWN
`BEAR_TREND_STRONG`	ขาลงแข็งแรง	MARKDOWN
`BEAR_TREND_WEAK`	ขาลงอ่อนแรง	DISTRIBUTION
`PANIC_SELL`	เทขายตื่นตระหนก — ขาลง+Volatility พุ่ง+Macro หรือ Momentum ลบรุนแรง	CAPITULATION
`LOW_VOL_ACCUMULATION`	Volatility ต่ำ ราคาแนบ EMA200 — ช่วงสะสม	ACCUMULATION
`LOW_VOL_RANGE`	Volatility ต่ำ แต่ไม่แนบ EMA200 พอ	ACCUMULATION
`VOLATILITY_EXPANSION`	Volatility ขยายตัวโดยไม่มีเทรนด์ชัดเจน + Range กว้าง	TRANSITION
`SIDEWAYS_CHOP`	ไม่เข้าเงื่อนไขไหนเลย — แกว่งไร้ทิศทาง	RANGE

Confidence Score

ทุก Regime มาพร้อม Confidence (0.05–0.98) ที่คำนวณแบบ Composite จากหลายปัจจัย: Trend Quality (34%), Macro Alignment (20%), Macro Conviction (14%), Vol Clarity (14%), Liquidity Confirmation (10%), Regime Stability (8%) พร้อมหักคะแนนเมื่อ Macro ขัดแย้งกับ Trend หรือ Transition Risk สูง — ไม่ใช่แค่ค่า 0/1 อย่างระบบพื้นฐานทั่วไป

การป้องกันความผิดพลาดจากข้อมูลไม่พอ

เอนจิ้นป้อนข้อมูล 250 แท่งให้ตัวจำแนกเสมอ (มากกว่า 200 แท่งที่ EMA200 ต้องการ) แต่ตัวจำแนกเองมีการ์ดป้องกัน ไว้ด้วย: หาก EMA200 ที่คำนวณ Fallback มาจากข้อมูลน้อยกว่า 200 แท่งจริง (เช่น กรณีถูกเรียกด้วยข้อมูลสั้นจากที่อื่น) ระบบจะไม่กล้าฟันธง Trend เป็น Bullish/Bearish และบังคับเป็น NEUTRAL พร้อมลด Confidence ลง — ป้องกัน "EMA200 ปลอม" จากข้อมูลไม่พอสร้างสัญญาณเทรนด์ผิดๆ

5.4 ตัวจับ Regime เชิงสถิติ (GMM Cross-check)

เพื่อยกระดับตัวจำแนกแบบ Rule-based (ที่ threshold ตั้งด้วยมือ) ระบบมีตัวจับ Regime แบบ **สถิติล้วน** (`xauby/regime/statistical.py`) ทำงานคู่ขนานเป็น "ความเห็นที่สอง" ที่ไม่มีอคติจาก Threshold ที่คนตั้ง:

- Fit **Gaussian Mixture Model (GMM) แบบ diagonal covariance ด้วย EM** บนพื้นที่ Feature 2 มิติ: Trend (ผลรวม Log-return แบบ Rolling) และ Log Realized Volatility
- เลือก 2 แกนนี้เพราะข้อมูลจริงพิสูจน์แล้วว่า Regime แยกด้วย Volatility ได้ชัดเจนกว่าทิศทาง
- คืนค่า Cluster ของแท่งล่าสุด พร้อม **Posterior Probability** และ **Persistence เชิงประจักษ์ (Stay Probability)**
- ใช้ **numpy pure** ไม่มี scipy/sklearn เพื่อให้ `classifier.py` หลักยังคง pure-stdlib

ผลลัพธ์แนบเป็น Feature `stat_*` เข้ากับ Regime ปกติ เป็น **Advisory เท่านั้น — ไม่เคย Override Regime หรือ Routing ใดๆ** ปิดอยู่โดย Default หลัง Flag `architecture.regime_statistical_crosscheck` ปรับค่าได้ผ่าน Block `regime_statistical:` (`components`, `min_bars`)

5.5 รายงานความแม่นยำของ Regime Detector

"ความแม่นยำ" ของ Regime Detector **ไม่ใช่** "ทายถูกกี่เปอร์เซ็นต์" เพราะไม่มี Ground Truth ว่าตลาด ณ ขณะนั้นอยู่ Regime ไหนจริงๆ ให้เทียบ การวัดที่ถูกต้องมี 3 อย่าง:

1. **Volatility Separation** — Regime แยกสภาพ Volatility จริงได้หรือไม่ (งานหลักที่ขับเคลื่อน Position Sizing และความกว้าง Stop)
2. **Persistence** — Regime อยู่นานพอให้ลงมือทำอะไรได้ หรือสลับไปมาทุกแท่ง (Churn)
3. **Directional Tilt** — Regime กลุ่ม Bull ให้ Drift สูงกว่ากลุ่ม Bear จริงหรือไม่ (คาดหวังว่าจะ*เล็ก*เสมอ เป็นโบนัส ไม่ใช่ตัวชี้วัดหลัก)

วัดด้วย **eta² (สัดส่วนความแปรปรวนของ Return ที่อธิบายได้ด้วย Regime Label)** พร้อม Permutation p-value (ความน่าจะเป็นที่การสุ่มจัดกลุ่มใหม่จะแยกข้อมูลได้ดีเท่ากัน) วัดจากข้อมูลจริง 4 ชั่วโมง (PAXGUSDT ตัวแทนทองคำ ~12,800 แท่ง / BTCUSDT ~17,500 แท่ง)

ตัวชี้วัด	PAXGUSDT (ตัวแทนทองคำ)	BTCUSDT (cross-check)
Volatility separation eta² (p)	**0.041** (p = 0.0005)	**0.034** (p = 0.0005)
Directional separation eta² (p)	0.0004 (p = 0.85 — เท่า Noise)	0.0019 (p = 0.0015)
Persistence (Mean run / Flip rate)	6.6 แท่ง / 15.1%	5.2 แท่ง / 19.2%
ลำดับ Drift (Bull เทียบ Bear)	+1.3 เทียบ −0.0 bps ✓	+5.8 เทียบ +0.5 bps ✓
GMM Cross-check Agreement	50.0%	45.2%

บทสรุป: ตัวจำแนกทำหน้าที่หลักได้ดี — แยก Volatility ได้ชัดเจนและมีนัยสำคัญทางสถิติจริง ($p \leq 0.001$ ทั้งสองสินทรัพย์), Regime นิ่งพอจะเทรด (Mean run > 3 แท่ง แปลว่า Debounce 3 แท่งของระบบตั้งไว้เหมาะสม), และลำดับ Bull/Bear ถูกทิศทาง ส่วน Directional Edge อ่อนตามคาด (ปกติของ Classifier ทุกตัวในตลาดจริง) **ข้อสรุปเชิงปฏิบัติ: ใช้ Regime เพื่อปรับขนาดไม้ตาม Volatility ไม่ใช่ทำนายทิศทางแท่งถัดไป**

> GMM Cross-check เห็นตรงกับ Rule-based เพียง ~45-50% ของเวลา ซึ่ง**ตั้งใจให้เป็นแบบนั้น** — ถ้า Cross-check ให้ผลตรงกันเกือบ 100% แปลว่ามันไม่ใช่ความเห็นที่สองจริงๆ การไม่ตรงกันคือสัญญาณว่าการอ่านตลาด ขณะนั้นกำกวม เป็นข้อมูลที่มีค่าสำหรับผู้ดูแลระบบหรือ Confidence Gate ในอนาคต ไม่ใช่ข้อบ่งชี้ว่าตัวใดตัวหนึ่งผิด

ข้อจำกัดที่ควรรู้

- ค่า eta² เล็กในเชิงสัมบูรณ์ (~0.04) — ตลาดส่วนใหญ่คือ Noise ไม่มี Regime Model ใดอธิบายความแปรปรวนได้ มาก การที่ 0.04 มี $p \leq 0.001$ แปลว่าการแยกนั้น "จริงและมั่นคง" ไม่ใช่ "ใหญ่"
- ประวัติราคาทองใช้ PAXGUSDT เป็นตัวแทน (Proxy) เพราะ XAUUSDT เองมีประวัติสั้นกว่า

- Threshold ยังตั้งด้วยมืออยู่ ยังไม่ได้ Fit ต่อสินทรัพย์ — GMM Cross-check คือก้าวแรกสู่โมเดลที่ขับเคลื่อน ด้วยข้อมูลมากขึ้น
- Forward Return เป็นสถิติเชิงพรรณนาแบบ In-sample ไม่ใช่ผล Backtest ที่พิสูจน์ว่ากลยุทธ์ทำกำไรได้

คำสั่งสร้างรายงานซ้ำ: `python3 scripts/regime_forward_validate.py --csv <candles.csv> --timeframe 4h --horizon 6`

5.6 RegimeRouter — เลือกกลยุทธ์ตาม Regime อัตโนมัติ

RegimeRouter คือระบบที่แม็พ Regime แต่ละแบบไปยังกลยุทธ์ที่เหมาะสม เป็น Opt-in ต่อคู่เทรด:

1. Flag Global `architecture.regime_router_enabled` ต้องเป็น true
2. Flag ต่อสินทรัพย์ `regime_router_enabled` ต้องเป็น true
3. ถ้าเป็นคู่ Live ต้องมี `regime_router_live_confirmed: true` เพิ่มด้วย

ถ้าสินทรัพย์ Live เปิด Router โดยไม่มี Live Confirmation เอนจิ้นจะบังคับกลับเป็น Sim อัตโนมัติ พฤติกรรมอื่นๆ ของ Router: Debounce 3 แท่งก่อนยืนยันการสลับ Regime, NO_TRADE บล็อกการ BUY ใหม่แต่คง Stop Protection ของโพซิชันเดิม, Recovery ต้องผ่าน 3 แท่งที่เทรดได้ก่อนกลับสู่ปกติ, บังคับปิดหลัง 6 แท่ง NO_TRADE ติดต่อกัน, และ Handoff กลยุทธ์ยึดหลักความปลอดภัยตามหัวข้อ 3.2

> **สถานะจริงตอนนี้:** คู่เทรด XAU ปิด RegimeRouter อยู่ (`regime_router_enabled: false`) — หมายความว่าตอนนี้ระบบรัน `cdc_action_zone` ตรงๆ ไม่ผ่านการเลือกกลยุทธ์อัตโนมัติ ตารางด้านล่างคือ Mapping ที่ Config ไว้ในระบบ (พร้อมใช้เมื่อเปิด Flag ในอนาคต):

Regime	กลยุทธ์ที่ Map ไว้	Action
`BULL_BREAKOUT`	`cdc_action_zone`	BUY ได้
`BULL_TREND_STRONG`	`cdc_action_zone`	BUY ได้
`BULL_TREND_WEAK`	`cdc_action_zone`	BUY ได้
`LOW_VOL_ACCUMULATION`	`bbrsi_mean_reversion`	BUY ได้
`LOW_VOL_RANGE`	`bbrsi_mean_reversion`	BUY ได้
`VOLATILITY_EXPANSION`	`supertrend_ema200`	BUY ได้
`SIDEWAYS_CHOP`	`bbrsi_mean_reversion`	BUY ได้
`BEAR_TREND_WEAK`	`cdc_action_zone`	BUY ได้
`PANIC_SELL`	`None`	NO_TRADE
`BEAR_BREAKDOWN`	`None`	NO_TRADE
`BEAR_TREND_STRONG`	`None`	NO_TRADE

เมื่อเปิด Regime Filter สำหรับ Backtest ระบบจะเลียนแบบ Routing นี้ด้วย: บล็อกสัญญาณ BUY ทุกครั้งที่ Regime อยู่นอกกลุ่มเป้าหมายของกลยุทธ์นั้น เพื่อให้ผล Replay ใกล้เคียงกับประตูเข้าไม้ของ Live มากที่สุด

5.7 กระบวนการคัดเลือกกลยุทธ์ด้วยข้อมูลจริง (R&D)

ระบบมี Gatekeeper แบบขับเคลื่อนด้วยข้อมูลสำหรับตัดสินใจเปลี่ยนกลยุทธ์ใน `regime_router.mapping`: กลยุทธ์ผู้ท้าชิงจะแทนที่กลยุทธ์เดิมได้ **ก็ต่อเมื่อ** ชนะ Profit Factor ของกลยุทธ์เดิมทั้งชุด Train (70% เก่าสุด) และ Test (30% ใหม่สุด) ของข้อมูลตลาดจริงหลายปี พร้อมมีจำนวนไม้พอและ Drawdown อยู่ในขอบเขต — **ข้อมูลเป็นคนตัดสิน ไม่ใช่ความเชื่อ**

เกณฑ์ตัดสิน (ต้องผ่านครบทุกข้อ)

1. PF ผู้ท้าชิง > PF เดิม ทั้ง Train และ Test
2. จำนวนไม้ ≥ 20 (Train) และ ≥ 8 (Test)
3. PF_test ≥ 1.0 และ maxDD_test ≤ 1.5 เท่าของเดิม (และ ≤ 15% แบบสัมบูรณ์)
4. Window ล่าสุด ~6 เดือน: PF ≥ 0.8 (Sanity check ว่าไม่พลิกหัวกลับในตลาดปัจจุบัน)

ไม่ผ่านข้อใดข้อหนึ่ง = กลยุทธ์เดิมอยู่ต่อสำหรับ Mapping Key นั้น

> **ผลการตัดสินล่าสุด (มิ.ย. 2026, BTCUSDT 1h):** จากข้อมูลจริง 26,297 แท่ง ไม่มี Candidate ตัวไหนผ่านเกณฑ์ครบ — `donchian_trend` ชนะ Train ชัดเจน (PF 1.16 เทียบ 0.77) แต่พังบน Test (PF 0.32 ต่ำกว่าทั้งเกณฑ์ 1.0 และ PF เดิม) จึง**คงกลยุทธ์เดิมทั้งหมด**ไว้ในทุกกลุ่ม Regime กระบวนการนี้เป็นงานวิจัยบน BTCUSDT ซึ่งปัจจุบันไม่ใช่คู่เทรด Live ของระบบ — ใช้เป็นตัวอย่างระเบียบวิธีสำหรับ การตัดสินใจเปลี่ยนกลยุทธ์ในอนาคต

คำสั่งที่เกี่ยวข้อง: `scripts/fetch_global_klines.py`, `scripts/regime_strategy_eval.py --grid`, `scripts/regime_churn_analysis.py` (วัดความถี่การสลับ Regime และผลกระทบต่อการรัน กลยุทธ์จริง)

6.1 การคำนวณขนาดโพซิชัน (Position Sizing)

ขนาดไม้คำนวณแบบอิงความเสี่ยงและทบต้นอัตโนมัติ:

```
qty = (equity × risk_pct) / sl_distance
```

ถูกจำกัดเพดานด้วย `max_position_per_trade_pct` และ Allocation ต่อสินทรัพย์ (`allocation_pct`) Equity รวมโพซิชันที่เปิดอยู่ ณ ราคา Mark ด้วย ทำให้กำไรที่ทำได้ ไปขยายขนาดไม้ถัดไปโดยอัตโนมัติ (Auto-compounding)

> **ข้อสำคัญ:** `risk_pct` ต้องตรงกันระหว่าง `trading.risk_pct` (Live อ่านจาก `portfolio.position_sizing.risk_pct`) และ `portfolio.position_sizing.risk_pct` (Backtest อ่านจาก `trading.risk_pct`) — ค่าไม่ตรงกันจะทำให้ Live กับ Backtest มีผลไม่ตรงกัน (parity พัง) ค่าปัจจุบันของระบบคือ `0.02` (2%) ทั้งสองจุด ระบบมี Startup guard (`MAX_SANE_RISK_PCT = 0.10`) ที่ปฏิเสธค่า `risk_pct` เกิน 10% ทันที

6.2 Stop-Loss / Trailing / Breakeven

กลไก Exit ที่ระบบรองรับ (ไม่ใช่ทุกตัวเปิดใช้ในทุกกลยุทธ์ — ดูค่าจริงของ CDC ในหัวข้อ 5.2):

- **Initial ATR Stop-Loss** — ระยะเริ่มต้นจาก Entry คูณด้วย `sl_atr_mult`
- **Trailing Stop** — ขยับตาม ATR คูณ `trailing_atr_mult` เมื่อราคาไปในทางที่ดี
- **Breakeven** — เลื่อน SL มาที่จุดเข้าเมื่อกำไรถึงเกณฑ์ที่ตั้ง (ถ้าเปิดใช้งาน)
- **Multi-tick Local SL Confirmation** — สำหรับสายที่ไม่มี SL ฝั่ง Exchange ต้องยืนยันครบ `sl_confirm_ticks` ก่อนตัดสินใจปิดไม้จริง เพื่อกันสัญญาณหลอกจาก Spike ราคาสั้นๆ

6.3 Partial Take-Profit

Strategy Config เปิด Partial Take-Profit แบบ One-shot ได้:

```yaml
strategy:
  params:
    cdc_action_zone:
      partial_tp_pct: 12.0
      partial_tp_fraction: 0.5
```

เมื่อโพซิชันที่เปิดอยู่ถึงผลตอบแทน `partial_tp_pct` จากราคาเข้า เอนจิ้นจะปิด `partial_tp_fraction` ของส่วนที่เหลือด้วย Reduce-only Close บน Swap จริง (หรือผ่าน SimBroker ในโหมด Sim) บันทึกเป็น Partial Closed Trade, ตั้ง `trade_states.partial_tp_taken=1` , และปล่อยส่วนที่เหลือให้ออกตามกลยุทธ์ปกติ ทริกเกอร์นี้เป็นแบบ One-shot ต่อโพซิชัน และคงอยู่แม้ผ่านการ Restart สำหรับ Baseline ทองคำ Live ปัจจุบัน นี่หมายถึงการเก็บ 50% ที่ +12% แล้วปล่อยส่วนที่เหลือขี่ไปจนกว่า CDC จะออก หน้าจอ UI แสดงเป็น `PTP` , `Partial TP` , หรือ `PTP banked` เมื่อดำเนินการแล้ว

6.4 Drawdown Guard — สวิตช์ตัดขาดทุนระดับพอร์ต

`risk.drawdown_guard` คือ Kill-switch ระดับพอร์ตโดยรวม: บล็อกการ BUY ใหม่ (และบังคับปิดโพซิชันถ้าตั้ง `close_positions: true`) เมื่อ Equity ร่วงลงต่ำกว่า `max_drawdown_pct` จากจุดสูงสุดที่เคยบันทึกไว้ (High-water Mark ที่เก็บใน `core/equity_peak.json`)

6.5 การเทรด Short และเงื่อนไขความปลอดภัย

คู่เทรดปัจจุบันเปิด Short ได้จริงบน Live (`allowed_sides: [long, short]`, `short_live_enabled: true`) เพราะ Adapter ของ OKX Swap ประกาศความสามารถ `swap` / `positions` / `reduce_only` ครบ เงื่อนไขความปลอดภัยของ Short:

- ต้องอนุญาตต่อคู่เทรดอย่างชัดเจนผ่าน `allowed_sides`
- Live ต้องเปิด `short_live_enabled` เพิ่มเติม — ไม่งั้นเป็น Paper-only
- Startup ล้มเหลวทันที (Fail closed) ถ้า Adapter ที่เลือกไม่มีความสามารถ Swap/Positions/Reduce-only
- Leverage Default 1 เท่า และเพดานสูงสุด 3 เท่า
- ช่องทาง Streaming (CCXT Pro) ถ้าเสื่อมสภาพจะบล็อก Short ใหม่ทันที; REST Fallback จะไม่เปลี่ยน Exchange/Market Type

6.6 เช็กลิสต์ความสอดคล้องของค่าความเสี่ยง

ควรตรวจสอบให้ตรงกันก่อนเทรด Live:

Key	ค่าปัจจุบันของ XAU
`trading.risk_pct`	0.02
`portfolio.position_sizing.risk_pct`	0.02
`trading.max_position_per_trade_pct`	25.0
`trading.max_daily_loss_pct`	6.0
`trading.max_open_positions`	1
`derivatives.max_leverage`	1

Guard ระดับ Global ใช้กับทุกสินทรัพย์เสมอ ส่วนสถานะความเสี่ยงระดับ Strategy ต่อคู่เทรดถูกแยกจากกัน แต่เงินทุนพอร์ตยังใช้ร่วมกันโดยเจตนา

7.1 แหล่งข้อมูลสำหรับ Backtest

Backtest ดึงข้อมูลจาก **Global Binance** (`api.binance.com`) เป็น Default เพื่อประวัติราคาที่ยาวกว่า (`backtest.data_base_url`, override ผ่าน env `BINANCE_BACKTEST_URL`) ส่วนการเทรด Live ใช้ Endpoint OKX ที่ตั้งค่าไว้จริง Backtest และ Live resolve Config ผ่าน Resolver ตัวเดียวกันเสมอ เพื่อให้ Backtest, Replay, และ Live สอดคล้องกับแหล่งความจริงเดียวกัน

7.2 คำสั่ง Backtest และ Optimizer

```
python scripts/replay_backtest.py --symbol XAUUSDT --config bot_config.yaml
python scripts/optimize_pair_configs.py
```

การเลือกกลยุทธ์เป็น Dry-run โดย Default:

```
python scripts/select_pair_strategy.py --symbol BTCUSDT --candidates a,b,c
python scripts/select_pair_strategy.py --symbol BTCUSDT --candidates a,b,c --apply
```

`--apply` เท่านั้นที่จะเขียนผู้ชนะที่ผ่านเกณฑ์กลับเข้า YAML

7.3 Regime-aware Replay

การตั้ง `use_regime_filter: true` บน Strategy Config ของสินทรัพย์จะเปิด Regime Classifier ภายใน Replay Engine — สัญญาณ BUY จะถูกระงับนอกกลุ่ม `TARGET_REGIMES` ของกลยุทธ์นั้น และประเมินใหม่ทุก `regime_update_bars` แท่ง (Default 24 แท่ง)

7.4 Replay Validation หลัง Restart/Incident

ใช้หลังการ Restart, Incident, หรือเปลี่ยน Config เสมอ:

```
python scripts/replay_validate.py <run_id> --symbol XAUUSDT
```

Replay Validation ตรวจว่า Event `signal_evaluated` ที่บันทึกไว้จาก Live ตรงกับผล Replay ของกลยุทธ์เดียวกันหรือไม่ — ตรวจสอบเฉพาะผลลัพธ์ของกลยุทธ์ ก่อนจะถูก Override โดย Macro Guard, Cooldown, และ Execution

ภาคที่ 8 — หน้าจอใช้งาน

8.1 Textual TUI (แดชบอร์ดในเทอร์มินัล)

xAuby มี UI แบบ Textual ในเทอร์มินัลที่อ่าน State ของเอนจิ้นแบบ Real-time — **ไม่ได้รันลูปเทรดเอง และเปิด Database แบบ Read-only เสมอ**

หน้าจอ	คีย์	หน้าที่
Dashboard	`1` , `d`	กราฟ, ตารางคู่เทรด, Regime, โพซิชัน, ป้ายกลยุทธ์/โหมด
Trade log	`2` , `t`	ไม้ที่ปิดแล้วและประวัติ
Incident explorer	`4` , `i`	Timeline การรันจาก Event store
Menu	Launcher/env	Action ด่วนสำหรับเอนจิ้น, Dashboard, เทสต์, ตัวช่วย Config

คีย์ Global: `[` / `0` คู่เทรดก่อนหน้า, `]` / `9` คู่เทรดถัดไป, `q` ออก

กราฟและ Legend เดินตามกลยุทธ์ที่ Active ของคู่เทรดที่เลือกเสมอ (Source of Truth ตามหัวข้อ 3.5) UI ปรับตามความกว้างเทอร์มินัล: ≥110 คอลัมน์ = สองคอลัมน์ (กราฟซ้าย สถิติขวา), 75-109 = คอลัมน์เดียวเต็มความกว้าง, <75 = แถวแบบมือถือ กราฟสั้นลง

เมื่อมีโพซิชันเปิดและ Config มี `partial_tp_pct` Panel โพซิชันจะแสดงแถว PTP พร้อมสัดส่วนที่จะเก็บ, ราคาทริกเกอร์, และสถานะ (`pending` จนกว่าจะปิดสำเร็จ แล้วเปลี่ยน เป็น `banked`)

รันผ่าน tmux (แนะนำบน VPS):

```
./scripts/start_dashboard_tmux.sh
./scripts/attach_dashboard_tmux.sh
```

8.2 WebUI มือถือ (Read-only)

WebUI คือแดชบอร์ดเบราว์เซอร์ขนาดเล็กแบบอ่านอย่างเดียว สำหรับเช็คบอทจากเดสก์ท็อปหรือมือถือ — **ไม่วาง ไม่ยกเลิก ไม่หยุด และไม่กลับมาเทรดใดๆ ทั้งสิ้น**

```
./scripts/start_webui.sh
```

Bind ค่าเริ่มต้น: `127.0.0.1:8787`

หน้า Home แสดงสถานะ Runtime ของ OKX สด, แท่งเทียน OHLC 24 แท่งล่าสุด, เส้น EMA12/EMA26 ซ้อนทับ, รายละเอียด CDC Action Zone, และสถานะ Partial TP ของโพซิชันที่เปิดอยู่ หน้า Activity มีสโครลของตัวเองสำหรับ Event และไม้ที่ปิดแล้ว

เข้าถึงจาก Windows

```
ssh -L 8787:127.0.0.1:8787 user@your-vps-ip
```

แล้วเปิด `http://localhost:8787`

เข้าถึงจากมือถือ

ตัวเลือกฟรีที่ง่ายที่สุดคือ Tailscale: ติดตั้งบน VPS และมือถือ, Login Tailnet เดียวกัน, ใช้ Tailscale Serve หรือ Bind WebUI ไปที่ IP Tailscale ของ VPS:

```
WEBUI_HOST=<vps-tailscale-ip> WEBUI_PORT=8787 ./scripts/start_webui.sh
```

> ห้าม Bind WebUI ไปที่ `0.0.0.0` บน VPS สาธารณะ เว้นแต่จะเพิ่มระบบยืนยันตัวตนที่เหมาะสม

Endpoint ทั้งหมดเป็น Read-only: `/api/state`, `/api/health`, `/api/recent-events`, `/api/trades`, `/api/candles`

8.3 ปรัชญาการออกแบบหน้าจอ

WebUI ออกแบบให้เหมือน "โต๊ะทองส่วนตัว" — ผู้ใช้คือเจ้าของบอทเพียงคนเดียว เช็คจาก iPhone (Safari, มักผ่าน Tailscale) หรือเดสก์ท็อป บริบทคือการ "เหลือบดู" ไม่ใช่นั่งทำงาน — คำถามเดียวที่ต้องตอบใน 10 วินาที: "บอทยังทำงาน อยู่ไหม, Equity เท่าไหร่, ตอนนี้ถือโพซิชันอะไร, มีอะไรเกิดขึ้นไหม"

หลักการออกแบบ

- **Glanceable ก่อนเสมอ** — ลำดับชั้น: Equity → โพซิชัน → สัญญาณ → อื่นๆ ไม่ต้องเลื่อนจอ
- **สีมีความหมายเท่านั้น** — เขียว/แดง/ส้ม = กำไร/ขาดทุน/เตือน เท่านั้น; สีทอง (Brand) ใช้บอกตัวตน/การเลือกเท่านั้น ไม่เคยบอกสถานะ
- **Read-only แปลว่าสงบ** — ไม่มีองค์ประกอบใดที่บ่งบอกว่า "ควบคุมได้" ผิดปกติจะดังชัด ปกติจะเงียบ
- **มือถือมาก่อน, Safari มาก่อน** — ออกแบบตามจริงของ iOS Safari (Toolbar แบบ Dynamic, Safe Area, ปุ่มสัมผัส ≥44px) เดสก์ท็อปได้ Layout มือถือแบบวางกลาง ไม่ใช่ Layout กว้างแยกต่างหาก
- **ตัวเลขคือ Interface** — ใช้ Tabular figures, ความละเอียดสม่ำเสมอ, คอลัมน์จัดตรงกัน Typography ทำหน้าที่แทนการตกแต่ง

โทนสี (OKLCH)

Token	บทบาท
`--bg-0` / `--surface`	พื้นหลังหน้า / การ์ด-แผง (Dark only — `color-scheme: dark`)
`--gold`	Accent เดียวของแบรนด์ — ใช้บอกตัวตน/การเลือกเท่านั้น
`--pos` / `--neg` / `--warn`	กำไร/Long — ขาดทุน/Short — ระวัง/เสื่อมสภาพ
`--info`	ข้อมูลประกอบ เช่น EMA26, Regime info

Typography ใช้ Font Stack ระบบของ iOS เดียว (ไม่มี Webfont), ตัวเลขทั้งหมดใช้ `font-variant-numeric: tabular-nums` Layout แบบ Phone-first ความกว้าง `min(430px, 100%)`, สามมุมมอง (Home/Signal/Activity) สลับด้วย `body[data-view]`

ภาคที่ 9 — Telegram

9.1 การติดตั้ง Telegram Bot

1. สร้างบอกผ่าน @BotFather
2. หา Chat ID ของตัวเอง เช่น ผ่าน @userinfobot
3. ตั้งค่า `.env`:

```
TELEGRAM_ENABLED=true
TELEGRAM_BOT_TOKEN=...
TELEGRAM_CHAT_ID=...
```

เปิด Polling ใน `bot_config.yaml`:

```yaml
notifications:
  alert_channel: telegram
  telegram_command_polling_enabled: true
```

9.2 คำสั่งทั้งหมด

อนุญาตเฉพาะ `TELEGRAM_CHAT_ID` ที่ตั้งค่าไว้เท่านั้น:

คำสั่ง	คำอธิบาย
`/help`, `/start`	รายการคำสั่ง
`/status`	ทุกคู่เทรด: ราคา, โหมด, โพซิชัน, สัญญาณล่าสุด
`/pnl`	กำไรขาดทุนพอร์ต 7 วัน และ 30 วัน พร้อมแยกต่อคู่เทรด
`/regime`	Regime snapshot ต่อคู่เทรด
`/last`	ไม้ที่ปิดล่าสุดต่อคู่เทรด
`/health`	Snapshot สุขภาพเอนจิ้น: โหมด, สถานะ Pause, คู่เทรด, โพซิชันเปิด, ปัญหา Feed/WS
`/pause`	ยืนยัน Pause ฉุกเฉิน — บล็อกการ BUY ใหม่โดยไม่ปิดโพซิชัน
`/resume`	ยืนยัน Resume — อนุญาต BUY ใหม่อีกครั้ง

9.3 การแจ้งเตือนอัตโนมัติ

เหตุการณ์	ระดับ
เริ่ม/หยุดเอนจิ้น	trade / info
BUY/SELL Fill	trade
Macro Guard บล็อก BUY	info
Regime Score/Label เปลี่ยน	info
RegimeRouter Handoff / NO_TRADE	info / position
Trailing Stop อัปเดต	position
กู้คืน SL ล้มเหลว	position / critical
Daily digest / Weekly review	info

9.4 โหมด Semi-auto และ Emergency Pause

```
trading:
  mode: semi_auto
notifications:
  semi_auto_confirm_timeout_seconds: 60
```

บอทจะส่ง Inline Keyboard (`Confirm BUY` / `Skip`) ก่อนเปิดไม้ทุกครั้ง

`/pause` และ `/resume` เป็นคำสั่งแบบ 2 ขั้นตอน: พิมพ์คำสั่งจะแค่ ถามยืนยัน ปุ่ม Inline ต่างหากที่จะใช้ Control จริง Pause จะบล็อกการ BUY ใหม่ทุกเส้นทาง (สัญญาณ, Manual, Semi-auto) โดยไม่ปิดโพซิชันที่เปิดอยู่ Critical Alert มีปุ่ม `Status` และ `Ack` มาด้วยเสมอ

ภาคที่ 10 — หลายบัญชี และหลาย Exchange

10.1 รันหลายอินสแตนซ์บนเครื่องเดียว

xAuby แยก Config ออกจาก Mutable Data ทำให้รันหลายเอนจิ้น (ต่างบัญชี/Exchange/ชุดคู่เทรด) พร้อมกันบนโค้ด Checkout เดียวกันได้โดยไม่ชนกัน:

ส่วน	ตัวแปร Env	ค่าเริ่มต้น
Config (`bot_config.yaml` , whitelist, `.env`)	`XAUBY_CONFIG_DIR`	โฟลเดอร์ทำงานปัจจุบัน
Mutable data (DB, lock, state, log, ยอด sim)	`XAUBY_HOME` + `XAUBY_INSTANCE_ID`	`core/` (+ `/<id>`)

```
XAUBY_CONFIG_DIR=instances/acct1 XAUBY_INSTANCE_ID=acct1 \
  python run_xauby.py --live
```

ตัวอย่าง systemd template สำหรับหลาย instance:

```
[Service]
Environment=XAUBY_CONFIG_DIR=/root/xAuby/instances/%i
Environment=XAUBY_INSTANCE_ID=%i
WorkingDirectory=/root/xAuby
ExecStart=/root/xAuby/venv/bin/python run_xauby.py --live
```

เปิดใช้ด้วย `systemctl enable --now xauby@acct1` — แต่ละ instance อ่าน `.env` ของตัวเอง ทำให้ Secret ไม่ต้องแชร์ข้ามบัญชี

10.2 Multi-Exchange ผ่าน CCXT

สลับ Exchange ได้ทั้งหมดผ่านเมนู Launcher โดยไม่ต้องแก้ไฟล์: option 4 → Exchange & Flags → Wizard จะเลือก Provider → `ccxt_id` → `quote_asset` → ใส่ Credential (Wizard เขียนตัวแปร .env ที่ถูกต้องให้เอง) → รัน Connection Test → เสนอ Restart

```
exchange:
  provider: ccxt
  ccxt_id: kraken
  api_key_env: KRAKEN_API_KEY
  api_secret_env: KRAKEN_API_SECRET
  quote_asset: USDT
```

Exchange Plugin Registry + WebSocket Streaming

เปิดใช้ด้วย Flag:

```
architecture:
  exchange_plugin_registry_enabled: true
```

เมื่อเปิด: `provider: binance` → Client แบบ Native + BinanceWebSocket, `provider: ccxt` (หรือ ccxt_id ใดก็ได้) → CCXTExchangeClient + CCXTProWebSocket (Streaming จริงสำหรับทุก Exchange ที่ ccxt.pro รองรับ)

เพิ่ม Exchange ใหม่ (Drop-in ไม่ต้องแก้ Factory)

สร้างไฟล์เดียว `xauby/api/exchanges/exchange_<name>.py` ที่ลงทะเบียนทั้ง REST Gateway และ WebSocket Adapter ผ่าน Decorator `@register_exchange` และ `@register_ws` — ระบบค้นพบไฟล์อัตโนมัติผ่าน `pkgutil` (Filename prefix `exchange_`)

10.3 OKX Perpetual และความปลอดภัยของ Short

เส้นทาง Derivatives ที่รับรองแล้วใช้ `market_type: swap`, `margin_mode: isolated`, `position_mode: one_way`, และ Settle Asset เป็น USDT สัญลักษณ์ในเอนจิ้นยังคงกระชับ (`BTCUSDT`) ขณะที่ Adapter resolve เป็นสัญญา Native (`BTC/USDT:USDT`) และแปลงจำนวนฐานเป็น Contract ด้วย `contractSize`

Startup จะล้มเหลวทันที (Fail closed) เมื่อ Adapter ที่เลือกไม่มีความสามารถ Swap, Positions, และ Reduce-only ครบ CCXT Pro รับ Subscribe Ticker, OHLCV, Public Trades, และ Order Book ทุก Market Event มี Identity ของ Exchange/Market กำกับ ช่องทางที่เสื่อมสภาพจะบล็อก Short ใหม่ทันที ส่วน REST Fallback จะไม่เปลี่ยน Exchange หรือ Market Type เด็ดขาด

11.1 การ Deploy บน VPS

```
mkdir -p core/logs
env -u DEFAULT_SYMBOL XAUBY_DEFAULT_SYMBOL=XAUUSDT XAUBY_FOCUS_SYMBOL=XAUUSDT \
  SIMULATE_ONLY=false BOT_READ_ONLY=false \
  ./venv/bin/python run_xauby.py --live --pair XAUUSDT

./scripts/start_dashboard_tmux.sh
./scripts/attach_dashboard_tmux.sh
```

ใช้เอนจิ้นเพียง 1 Instance **ต่อ 1 Runtime Root** เท่านั้น `<root>/.engine.lock` ป้องกันการซ้ำซ้อนภายใน Root เดียวกัน

11.2 คำสั่งอัปเดตและรีสตาร์ท

สำหรับปฏิบัติการปกติบน VPS หลังติดตั้ง CLI แล้ว ให้ใช้:

```
xauby update   # deploy origin/main + controlled restart
xauby restart  # controlled restart โดยไม่ดึงโค้ดใหม่
```

Fallback แบบ Manual:

```
scripts/deploy_from_github.sh --restart --branch=main
scripts/controlled_restart_engine.sh
```

11.3 การตรวจสอบ Latency ของ VPS

ใช้เมื่อบอทรู้สึกช้า, TUI ค้าง, หรือ Request Exchange หมดเวลา:

```
./venv/bin/python scripts/vps_latency_check.py
./venv/bin/python scripts/vps_latency_check.py --json
```

ค่าที่ดู	ความหมาย
curl total	เวลารวมของ Request HTTPS ไปยัง Exchange API
curl ttfb	เวลาถึง Byte แรก — สูงมักแปลว่า Routing/API หน่วง
tcp443	เวลา Connect TCP ดิบ — สูงชี้ไปที่ระยะทาง Network/Routing
clock	`NTPSynchronized=yes` สำคัญเพราะ Request ที่เซ็นลายเซ็นใช้ Timestamp
loadavg	Load สูงต่อเนื่องเกิน CPU ที่มีทำให้บอท/TUI หน่วง
disk_used/free	พื้นที่ต่ำหรือ Disk ช้ากระทบการเขียน SQLite/Event/State

TUI Header แสดง Metric: `REST` (Latency Request ล่าสุด), `WS` (อายุ Tick WebSocket ล่าสุด), `Tick` (ระยะเวลา Tick ทั้งหมด), `Sync` (ระยะเวลา Sync แท่งเทียน), `State` (ระยะเวลา Export State JSON) — ถ้า REST สูงแต่ Tick ต่ำ ปัญหาน่าจะอยู่ที่เส้นทาง VPS/Exchange ถ้า Tick/Sync/State สูง ให้ปรับงาน Local หรือลดความถี่ Poll/Log

11.4 เช็กลิสต์ความปลอดภัยฉบับเต็ม

- ☐ รีวิว Simulation Run แล้วก่อน Live
- ☐ API Key OKX มีสิทธิ์เทรดสำหรับบัญชีที่ตั้งใจ และ**ไม่มี**สิทธิ์ Withdraw
- ☐ ตัวแปร OKX API Key/Secret/Passphrase ครบก่อน Restart แบบ Live
- ☐ เงินอยู่ในบัญชี OKX ที่บอทอ่านสำหรับเทรด USDT Swap; Dashboard อาจแสดง 0.00 ถ้าเงินอยู่ผิดกระเป๋า
- ☐ ทบทวน `risk_pct`, `max_position_per_trade_pct`, และ Guard ระดับ Global ให้เหมาะกับขนาดบัญชี
- ☐ ตั้งเกณฑ์ `risk.drawdown_guard` ให้ตรงกับความเสี่ยงที่รับได้ (Kill-switch)
- ☐ ทบทวนผลประเมินการย้าย Long/Short ของ OKX XAU หลังดาวน์โหลดแท่งเทียน 4H และ Funding History ใหม่
- ☐ `regime_router_live_confirmed: true` ตั้งเฉพาะหลังอนุมัติต่อคู่เทรดโดยตรง
- ☐ `architecture.strategy_sandbox_strict: true` เมื่อรันปลั๊กอินจากภายนอก/ไม่ไว้ใจ
- ☐ รีวิวผล `scripts/select_pair_strategy.py` ก่อนใช้ `--apply`
- ☐ `scripts/replay_validate.py <run_id> --symbol <PAIR>` ผ่านหลัง Restart
- ☐ ทดสอบ Telegram แล้ว: `/status` แสดงครบทุกคู่เทรด, `/health` ไม่มีปัญหาที่ไม่คาดคิด
- ☐ ทดสอบปุ่มยืนยัน Emergency `/pause` และ `/resume` ในโหมด Simulation แล้ว
- ☐ เอนจิ้นถูกดูแลผ่าน tmux/systemd ไม่ใช่ SSH shell ที่ไม่มีคนดูแล

ภาคผนวก

ก. อภิธานศัพท์ (Glossary)

คำศัพท์	ความหมาย
ATR	Average True Range — ค่าเฉลี่ยความผันผวนของราคาต่อแท่ง ใช้กำหนดระยะ SL/Trailing
EMA	Exponential Moving Average — ค่าเฉลี่ยเคลื่อนที่แบบถ่วงน้ำหนักเอนเอียงไปทางข้อมูลล่าสุด
PF (Profit Factor)	อัตราส่วนกำไรรวมต่อขาดทุนรวม — มากกว่า 1 แปลว่าโดยรวมเป็นกำไร
DD (Drawdown)	ระยะที่ Equity ร่วงลงจากจุดสูงสุดที่เคยทำได้
Regime	สภาพตลาดที่ถูกจัดหมวดโดยตัวจำแนก เช่น BULL_TREND_STRONG, PANIC_SELL
Debounce	การรอให้สัญญาณซ้ำกันครบจำนวนแท่งก่อนยืนยันการเปลี่ยนแปลงจริง (กัน Noise)
NO_TRADE	Regime ที่ระบบตัดสินใจไม่เปิดไม้ใหม่ แต่ยังบริหารไม้เดิมต่อ
Handoff	การส่งมอบการบริหารโพซิชันจากกลยุทธ์เดิมไปกลยุทธ์ใหม่อย่างปลอดภัย
Partial TP	การปิดโพซิชันบางส่วนที่กำไรถึงเกณฑ์ที่ตั้งไว้ (One-shot)
Reduce-only	ออเดอร์ที่รับประกันว่าจะลดขนาดโพซิชันเท่านั้น ไม่เปิดโพซิชันใหม่
Sandbox Strategy	การสแกน AST แบบ Static เพื่อปฏิเสธปลั๊กอินที่แตะ Engine/DB/Network หรือใช้ eval/exec
GMM (Gaussian Mixture Model)	โมเดลสถิติที่จัดกลุ่มข้อมูลด้วยการผสมของ Normal Distribution หลายตัว
eta^2 (Eta-squared)	สัดส่วนความแปรปรวนของข้อมูลที่อธิบายได้ด้วยตัวแปรจัดกลุ่ม (ในที่นี้คือ Regime Label)
Permutation p-value	ความน่าจะเป็นที่การสุ่มจัดกลุ่มข้อมูลใหม่จะให้ผลแยกได้ดีเท่ากับของจริง — ต่ำ = นัยสำคัญสูง

ข. โครงสร้างโปรเจกต์

```
xAuby/
|-- run_xauby.py
|-- launcher.py            # thin shim re-exporting the xauby.launcher package
|-- bot_config.yaml
|-- coin_whitelist.json
|-- docs/
|-- scripts/
|-- tests/
`-- xauby/
    |-- launcher/          # interactive launcher: config_io, process, maintenance, quick_config
    |-- engine/            # LiteTradingEngine (mixins), risk, orders, regime routing
    |-- strategies/        # strategy plugins + indicators/
    |-- runtime/            # pair registry, whitelist validation, config resolvers
    |-- regime/            # regime classifier + GMM cross-check
    |-- backtest/          # replay engine, optimizer, metrics
    |-- observability/     # EventEmitter, JSONL store, replay validation
    `-- ui/                # terminal chart + Textual TUI, dashboard, tradelog
```

ค. ตารางสคริปต์อ้างอิงทั้งหมด

คำสั่ง	หน้าที่
`python -m unittest discover -s tests -q`	รันชุดเทสต์ทั้งหมด
`python health_check.py`	ตรวจสุขภาพระบบครั้งเดียว
`python scripts/incident_explorer.py list`	ดูรายการ Incident
`python scripts/replay_validate.py <run_id> --symbol XAUUSDT`	ตรวจสอบ Replay หลัง Restart
`python scripts/evaluate_okx_xau_migration.py`	ประเมินการย้ายมาใช้ OKX XAU
`python scripts/select_pair_strategy.py`	คัดเลือกกลยุทธ์ (Dry-run/--apply)
`python scripts/optimize_pair_configs.py`	Optimizer แบบ Low-CPU
`python scripts/capture_tui_screenshots.py`	สร้างภาพหน้าจอ TUI ใหม่
`python scripts/fetch_global_klines.py`	ดึงแท่งเทียนหลายปีจาก Global Binance
`python scripts/regime_strategy_eval.py --grid`	ประเมินกลยุทธ์ต่อ Regime แบบ Gatekeeper
`python scripts/regime_forward_validate.py`	วัดความแม่นยำ Regime Detector บนข้อมูลจริง
`python scripts/regime_churn_analysis.py`	วัดความถี่การสลับ Regime
`python scripts/vps_latency_check.py`	ตรวจ Latency ของ VPS

ง. คำเตือนความเสี่ยง (Disclaimer)

> ซอฟต์แวร์นี้จัดทำขึ้นเพื่อการศึกษาและวิจัยเท่านั้น การเทรด Cryptocurrency และอนุพันธ์มีความเสี่ยงสูงต่อการสูญเสียเงินทุน ผลการดำเนินงานในอดีตไม่ได้เป็นเครื่องยืนยันผลลัพธ์ในอนาคต ผู้ใช้ต้องปฏิบัติตามข้อกำหนดการใช้งาน API ของ Exchange ที่เลือกใช้และกฎหมาย/ระเบียบข้อบังคับที่เกี่ยวข้องในเขตอำนาจของตนเองเสมอ ผู้จัดทำเอกสารนี้ไม่รับผิดชอบต่อความเสียหาย ใดๆ ที่เกิดจากการใช้งานซอฟต์แวร์นี้กับเงินทุนจริง

— จบเล่ม —