

Kandungan

- **Prakata** 7
- **Cara Menggunakan Buku Ini** 9

BAHAGIAN I – Daripada AI Agent Kepada Trading Automation

- 01** **Evolusi Seorang Trader: Manual, Semi-Auto & Automated** 14
- 02** **Bagaimana Automated Trading System Berfungsi** 22
- 03** **Kenali Broker API: moomoo vs Tradovate** 32
- 04** **Infrastruktur Trading Bot Yang Kita Akan Bina** 41

BAHAGIAN II – moomoo OpenD: Membina Broker Connection

- 05** **Mengenali moomoo OpenD** 51
- 06** **Install moomoo OpenD Pada Windows** 59
- 07** **Running OpenD Pada Ubuntu VPS** 69
- 08** **Python + moomoo API: First Connection** 79
- 09** **Market Data, OHLC & Technical Indicators** 89
- 10** **Account, Positions, Orders & Paper Trading** 100
- 11** **Moomoo Order Lifecycle** 109

BAHAGIAN III – Tradovate API: US Futures Automation

- 12** **Mengenali Tradovate dan Futures Trading API** 120
- 13** **Tradovate API Authentication** 129
- 14** **Tradovate REST API: Accounts, Orders & Positions** 139
- 15** **Real-Time Trading Dengan WebSocket** 149
- 16** **Futures Order Management & Protective Orders** 159

BAHAGIAN IV – Membina Strategy Engine

- 17** **Trading Strategy Bukan Sekadar Buy dan Sell** 171
- 18** **Strategy 01: EMA9 Trend Following** 179
- 19** **Strategy 02: Opening Range Breakout (ORB)** 189
- 20** **Backtesting, Walk-Forward & Paper Trading** 199

BAHAGIAN V – Risk Engine, OMS & Execution

21

Position Sizing & Automated Risk Management

211

22

Build Your Own Broker-Agnostic Trading Engine

223

23

Order Management, Reconciliation & Kill Switch

233

BAHAGIAN VI – Hermes, n8n, Telegram & Trading Dashboard

24

Integrating AI Agents Without Giving Away the Trading Keys

246

25

n8n Trading Automation Workflows

255

26

Telegram & Discord Trading Control Centre

264

27

Build a Premium Trading Automation Dashboard

273

BAHAGIAN VII – Deployment, Failure Modes & Capstone

28

Deploy Trading Automation Pada VPS

283

29

Apa Berlaku Bila Trading Bot Gagal?

292

30

Capstone Project: Your First Automated Trading System

303

A

Cheat Sheet

312

B

Matriks Ciri Broker

316

C

Glosari

323

D

Workbook 7 Hari

329

E

Kesediaan Live – Sebelum Mempertimbangkan Wang Sebenar

332

F

Nota Regulatori Malaysia

336

.

Rujukan

344

01

BAHAGIAN I

Daripada AI Agent Kepada Trading Automation

Sebelum menyambung apa-apa kepada broker, kita bina model mental yang betul: tahap-tahap automasi, kitaran hidup sebuah order, perbezaan broker API, dan infrastruktur yang akan kita bina.

-
- BAB 01 Evolusi Seorang Trader: Manual, Semi-Auto & Automated
 - BAB 02 Bagaimana Automated Trading System Berfungsi
 - BAB 03 Kenali Broker API: moomoo vs Tradovate
 - BAB 04 Infrastruktur Trading Bot Yang Kita Akan Bina

01

Evolusi Seorang Trader: Manual, Semi-Auto & Automated

Kebanyakan trader runcit tidak bermula dengan bot. Mereka bermula dengan carta, alarm harga dan jari di atas tetikus. Sebelum kita menyerahkan satu langkah pun kepada mesin, kita perlu tahu dengan tepat langkah mana yang sedang kita automasikan — dan apa yang akan berlaku apabila langkah itu salah.

HASIL Anda dapat membezakan trading manual, berasaskan alert, semi-automated dan fully automated; meletakkan workflow anda sendiri pada tangga kematangan L0–L5; dan menerangkan lima lapisan sistem yang akan kita bina sepanjang buku ini.

Masa: 45 minit

Tahap: Asas

Prasyarat: Siri 01 atau pengalaman setara; tiada kod diperlukan

APA YANG ANDA AKAN BELAJAR

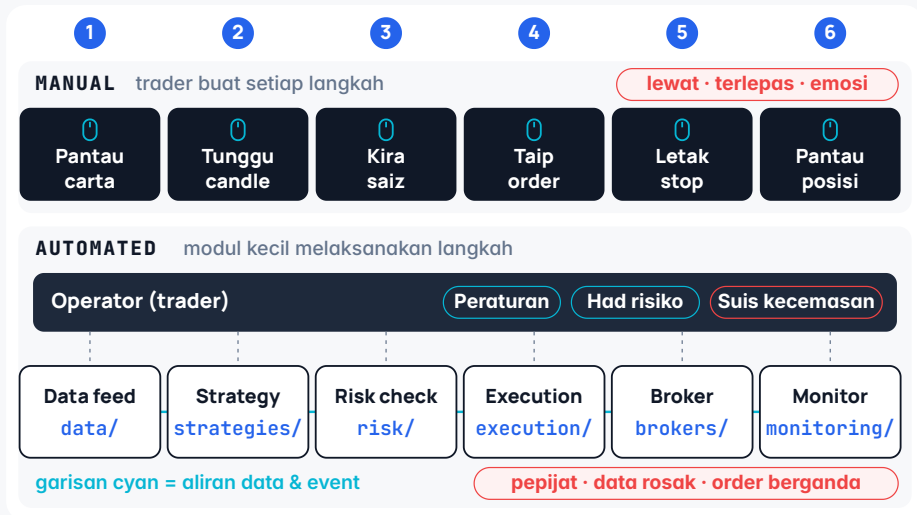
- 01 Membezakan empat cara trader bekerja: manual, berasaskan alert, semi-automated dan fully automated.
- 02 Meletakkan workflow anda sendiri pada **Tangga Kematangan Automasi L0–L5**.
- 03 Membezakan *algorithmic*, *quantitative*, *AI-assisted* dan *Agentic Trading*, serta memahami kenapa trader runcit tidak bersaing dengan HFT.
- 04 Menilai dengan jujur kelebihan dan had automasi — termasuk kenapa **automasi tidak mencipta edge**.
- 05 Menghuraikan lima lapisan sistem: Intelligence, Strategy, Risk, Execution dan Operations.

1.1 Empat Cara Seorang Trader Bekerja

Bayangkan beberapa trader dengan strategi sama: beli apabila MES ditutup di atas EMA9 yang naik. **Trader manual** melakukan setiap langkah dengan tangan — boleh menggunakan konteks yang sukar ditulis sebagai peraturan, tetapi penat, terlepas setup ketika tidur, dan beremosi pada setiap klik. **Trader berasaskan alert** membiarkan mesin *memerhati*; dia masih memutuskan dan melaksanakan. **Trader semi-automated** menerima signal lengkap (entry, stop, target, saiz) lalu menekan butang, atau meluluskan dengan satu ketikan dan sistem yang menghantar. **Sistem fully automated** melakukan semuanya; manusia menjadi *pengendali* yang menetapkan had, memantau dan memegang suis kecemasan.

Dimensi	Trader manual	Trader berasaskan alert	Sistem automated
Keputusan	Manusia, setiap trade	Manusia, selepas alert	Peraturan dalam kod
Pelaksanaan	Manusia (klik)	Manusia (klik)	Kod melalui broker API
Kelajuan	Saat hingga minit	Minit	Milisaat hingga saat
Emosi	Setiap klik	Selepas alert	Berpindah ke reka bentuk dan pemantauan
Mod kegagalan	Lewat, terlepas, salah taip, panik	Alert terlepas, tergesa-gesa	Pepijat, data rosak, sambungan putus, order berganda
Pemantauan	Semasa trading	Semasa alert	Berterusan — sistem, broker, data, log
Kos	Masa dan tenaga	Langganan alert/platform	Infrastruktur, data, akses API, masa ujian

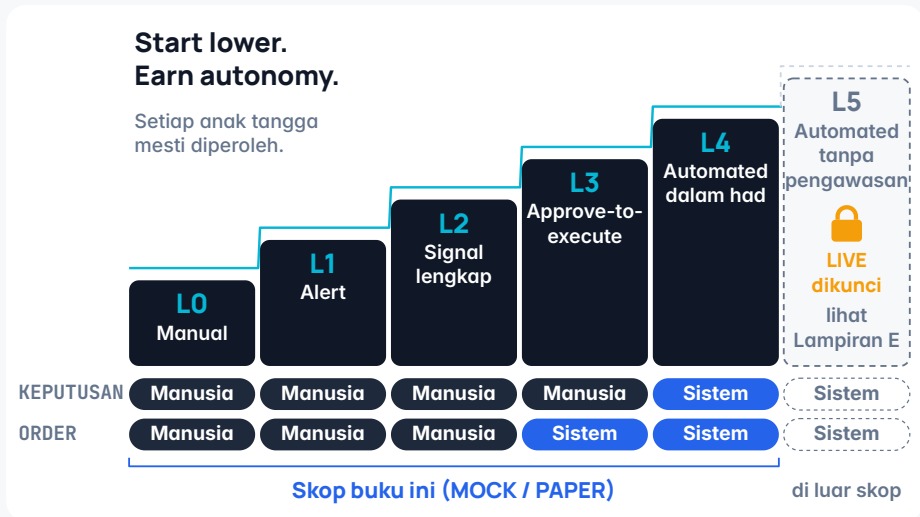
Trader manual gagal secara *perlahan dan kelihatan*; sistem automated boleh gagal secara *pantas dan senyap* — pepijat kecil boleh menghantar order yang sama dua kali sebelum anda sedar. Sebab itu buku ini lebih banyak membincangkan risk engine, reconciliation dan kill switch berbanding strategi.



Rajah 1.1 — Strategi yang sama dilaksanakan secara manual dan secara automated: langkahnya serupa, tetapi siapa yang melaksanakan dan bagaimana ia gagal sangat berbeza.

1.2 Tangga Kematangan Automasi (L0–L5)

Siri 02 menggunakan tangga serupa dengan *Automation Ladder* Siri 01, tetapi untuk **pelaksanaan dagangan**. **L0 Manual** dan **L1 Alert**: manusia memutuskan dan menghantar order. **L2 Signal lengkap**: sistem mencadangkan entry, stop, target dan saiz; manusia memutuskan dan menaip order. **L3 Approve-to-execute**: manusia meluluskan (cth. di Telegram), sistem menghantar. **L4 Automated dalam had**: peraturan dalam kod menghantar order dalam had risiko keras (simulasi/paper), manusia memantau dan memegang kill switch. **L5 Automated tanpa pengawasan** dengan wang sebenar — **tidak dibina dalam buku ini**.



Rajah 1.2 — Tangga Kematangan Automasi. Buku ini membawa anda hingga L4 dalam mod simulasi dan paper; L5 dengan wang sebenar sengaja dikunci.

Naik satu anak tangga ialah keputusan, bukan kewajipan — ramai trader berdisiplin kekal di L2 atau L3. Setiap anak tangga mesti *diperoleh*: naik ke L4 hanya selepas L3 berjalan tanpa kejutan cukup lama, dengan log sebagai bukti.

AGENTIC INSIGHT

L3 masih "automated" pada mata bursa

Order selepas butang "Luluskan" tetap *dijana dan dihantar oleh kod*. CME Rule 536.B (Tag 1028) membezakan order yang dimasukkan terus oleh individu daripada order sistem komputer. Tafsiran berhati-hati penulis (bukan kenyataan CME): order enjin ditanda `isAutomated: true` di Tradovate (Bab 14).

1.3 Algorithmic, Quantitative, AI-Assisted dan Agentic Trading

Algorithmic trading ialah pelaksanaan mengikut peraturan dalam kod, sesederhana "beli apabila harga ditutup di atas EMA9" (EMA9 dan ORB, Bahagian IV). **Quantitative trading** menekankan *cara peraturan ditemui*: hipotesis statistik, ujian di luar sampel, expectancy, drawdown — banyak bot runcit algorithmic tetapi tidak pernah diuji dengan ketat. **AI-assisted trading**

menggunakan AI untuk membantu manusia (Siri 01). **Agentic Trading** dalam siri ini: AI Agent bekerja *di sekeliling* enjin — research, penerangan, pemantauan — manakala pelaksanaan diputuskan peraturan deterministik dan had risiko dalam kod. AI boleh mencadangkan; ia tidak memegang kunci order ([Bab 24](#)).

1.4 HFT vs Automasi Runcit

HFT ialah dunia firma dengan server bersebelahan bursa (*colocation*) dan latensi dalam mikrosaatt. Trader di Malaysia yang menghantar order ke CME melalui VPS dan broker API berada *puluhan hingga ratusan milisaat* jauhnya — jurang fizikal yang tiada kod boleh tutup. Untuk carta 5 minit atau harian, latensi itu hampir tidak relevan; yang penting ialah data betul, candle benar-benar ditutup, saiz tepat, order tidak berganda dan stop yang wujud di broker ([Bab 02](#)). Matlamat kita: **lebih konsisten daripada diri sendiri**, bukan lebih pantas daripada pasaran.

1.5 Kelebihan dan Had Automasi

Empat kelebihan nyata: **konsistensi**, **disiplin** (had kerugian harian dikuatkuasakan kod), **kelajuan** (termasuk sesi US 21:30–04:00 MYT ketika anda tidur) dan **skala**. Setiap satu ada bayangannya: automasi **mempercepat kesilapan** — peraturan salah dilaksanakan dengan sempurna, berulang kali; **overfitting** menjadikan backtest cantik tetapi gagal pada data baharu ([Bab 20](#)); **risiko operasi** — sambungan putus, token tamat, OpenD log keluar, VPS reboot ([Bab 29](#)); dan **kos data serta akses** ([Bab 03](#)).



AGENTIC INSIGHT

Automasi tidak mencipta edge

Edge ialah kelebihan statistik dalam peraturan anda — sebab munasabah kenapa purata hasil trade positif selepas kos. Automasi hanya *melaksanakan* peraturan; tanpa edge, ia menghasilkan kerugian yang lebih konsisten. Dalam [Bab 20](#) EMA9 menunjukkan keputusan positif pada data sintetik yang secara binaan *tiada* edge — pengajaran tentang nasib dan saiz sampel.

1.6 Lima Lapisan Sebuah Sistem Trading

Lapisan	Soalan	Bertanggungjawab	Projek pengiring
Intelligence	Apa konteks pasaran?	Manusia + AI Agent	<code>ai/boundary.py</code> , Hermes (Bab 24)
Strategy	Syarat dipenuhi?	Peraturan deterministik	<code>strategies/</code>
Risk	Dibenarkan? Berapa saiz?	Risk engine — bukan AI	<code>risk/</code>
Execution	Order sampai dengan selamat?	Gateway tunggal + OMS	<code>execution/</code> , <code>orders/</code> , <code>brokers/</code>
Operations	Sistem sihat?	Pemantauan, audit, kill switch, manusia	<code>monitoring/</code> , <code>notifications/</code> , <code>journal/</code>

 SECURITY CHECK

Sempadan paling penting dalam buku ini

Output AI ialah **input yang tidak dipercayai** — boleh salah, lapuk atau dimanipulasi melalui *prompt injection*. AI **tidak pernah** memanggil tool order, memilih akaun live, menukar had risiko atau mematikan kill switch; paling jauh, ia menghasilkan data berstruktur yang disahkan skema.

Siri 01 membina Intelligence dan sebahagian Operations (Hermes, n8n, Telegram) dengan manusia sendiri yang bertindak. Siri 02 menambah Strategy, Risk dan Execution; Hermes kekal pembantu research melalui API baca sahaja, n8n untuk kerja berjadual (bukan gelung order), dan Telegram menjadi pusat kawalan (`/status` , `/pause`).

1.7 Tutorial: Audit Workflow Trading Anda

1

Senaraikan langkah dan siapa melakukannya

Bab ini tiada kod. Ambil trade terakhir anda (atau rekaan), tulis setiap langkah dari "melihat peluang" hingga "trade dicatat", dan tandakan **M** (manusia), **A** (alert/alat) atau **S** (sistem). Tahap anda ialah tahap langkah *pelaksanaan order*.

2 Nilai risiko jika diautomasikan

Bagi setiap langkah M: "Jika kod melakukan ini dan salah, apa kerugian paling teruk?" — **Rendah, Sederhana** atau **Tinggi**.

●●● Contoh audit workflow (diisi)

CONTOH

LANGKAH	SIAPA	RISIKO JIKA AUTO
1 Semak berita & kalendar ekonomi	M	Sederhana
2 Tunggu candle 5 minit ditutup	M	Rendah
3 Semak syarat EMA9 + volume	A	Rendah
4 Kira saiz posisi ikut stop	M	Tinggi
5 Taip order + stop di platform	M	Tinggi
6 Pantau & keluar	M	Tinggi
7 Catat dalam journal	M	Rendah

TAHAP SEMASA: L1 (alert sahaja; order manual)

Contoh ilustrasi. Isikan dengan workflow anda sendiri.

Cara membaca hasil audit

Langkah **Rendah** yang berulang ialah calon automasi pertama. Langkah **Tinggi** (saiz, order, exit) menunggu risk engine ([Bab 21](#)), OMS ([Bab 23](#)) dan berminggu-minggu simulasi. Untung dalam backtest atau demo bukan bukti sedia untuk live ([Bab 20](#), Lampiran E).

TROUBLESHOOT	Sistem automated rugi walaupun strategi "bagus" secara manual
SIMPtom	Strategi yang baik secara manual rugi apabila diautomasikan.
PUNCA	Judgment anda (cth. elak berita besar) tiada dalam kod.
DIAGNOSIS	Bandingkan trade sistem dengan journal manual anda.
PENYELESAIAN	Jadikan setiap sebab peraturan, atau kekal di L2/L3.
PENGESAHAN	Trade simulasi sepadan dengan keputusan manual (≥20).

LATIHAN PRAKTIKAL

Letakkan Diri Anda Pada Tangga

Masa: 15 minit. Tulis tahap semasa (L0–L5) dan sebabnya, sasaran **90 hari** (maksimum satu anak tangga di atas), dan **tiga langkah yang anda TIDAK akan automatisasi lagi** berserta sebab. Simpan untuk [Bab 17](#).

CHAPTER CHECKPOINT

Tandakan semua sebelum ke bab seterusnya

- ☐ Saya boleh membezakan empat cara trader bekerja, tahu tahap semasa dan sasaran 90 hari saya.
- ☐ Saya boleh menamakan lima lapisan, menerangkan kenapa AI tiada dalam laluan order, dan kenapa automasi tidak mencipta edge.

RUMUSAN BAB

Automasi mengubah *siapa* yang melaksanakan peraturan, bukan *kualiti* peraturan. Kita mendaki satu anak tangga pada satu masa, dengan risk engine deterministik sebagai penjaga pintu. Bab 02 menjejaki sebuah signal hingga ke journal.



02

Bagaimana Automated Trading System Berfungsi

Apabila orang berkata "bot saya beli MES tadi", sebenarnya sebelas perkara berbeza telah berlaku – dan mana-mana satu daripadanya boleh gagal secara senyap. Bab ini membuka enjin itu dan menjejaki satu signal, langkah demi langkah, dari data pasaran hingga ke journal.

HASIL Anda dapat menerangkan kitaran hidup sebuah order dalam sistem automated, membezakan submitted, acknowledged dan filled, memahami idempotency, dan menjejaki satu signal simulasi melalui audit log selepas menjalankan capstone dengan 400 bar.

Masa: 60-75 minit

Tahap: Asas-Pertengahan

Prasyarat: Bab 01; projek pengiring dipasang dan ujian lulus (Bab 04 menerangkan pemasangan penuh – jika belum, baca bab ini dahulu dan kembali ke tutorial selepas Bab 04)

APA YANG ANDA AKAN BELAJAR

- 01 Menghuraikan kitaran hidup sebuah signal, dari data pasaran hingga journal.
- 02 Memahami seni bina *event-driven*, perbezaan OMS dan EMS, serta peranan broker gateway, portfolio dan risk engine.
- 03 Membezakan **submitted** ≠ **acknowledged** ≠ **filled** menggunakan enum `OrderStatus` sebenar.
- 04 Menerangkan idempotency: kenapa signal yang sama tidak boleh mencipta dua order.
- 05 Menjalankan ujian state machine order dan capstone simulasi, kemudian menjejaki satu signal melalui audit log.

2.1 Sebuah Signal, Sebelas Langkah

Satu trade simulasi bermula dengan sebatang **bar 5 minit** yang sudah *ditutup* dan berakhir dengan satu baris journal. Di antaranya ada sebelas langkah: **(1) market data** disahkan (urutan, duplikasi, jurang, umur, OHLC) → **(2) strategy** menilai peraturan → **(3) signal** (arah, harga rujukan, stop, target, sebab) → **(4) order intent**, apabila perancang mengira saiz → **(5) risk check** dengan semua semakan bernama → **(6) execution** melalui gateway tunggal → **(7) broker acknowledgement** ("working") → **(8) fill**, sebahagian atau penuh → **(9) position update** daripada fill, bukan niat → **(10) reconciliation** dengan broker → **(11) journal** dengan P&L dan R. Langkah 1–3 milik lapisan Strategy, 4–5 Risk, 6–8 Execution, 9–11 Operations.

Perancang (`risk/planner.py`) mencipta `OrderIntent` sebelum risk check kerana risk engine perlu menilai sesuatu yang konkrit. Tetapi seperti kata kod, intent "**bukan order sehingga risk meluluskannya**".



Rajah 2.1 — Aliran signal-ke-pelaksanaan dalam enjin buku: sebelas langkah, dua cabang kegagalan, dan audit log yang merekod setiap keputusan penting.

2.2 Seni Bina Event-Driven

Sistem kita **event-driven**: ia bertindak balas terhadap *peristiwa* — bar ditutup, acknowledgement, fill, sambungan terputus. **Feed** membekalkan data (fixture berlabel atau broker). **Signal engine** ialah strategi yang tidak tahu apa-apa

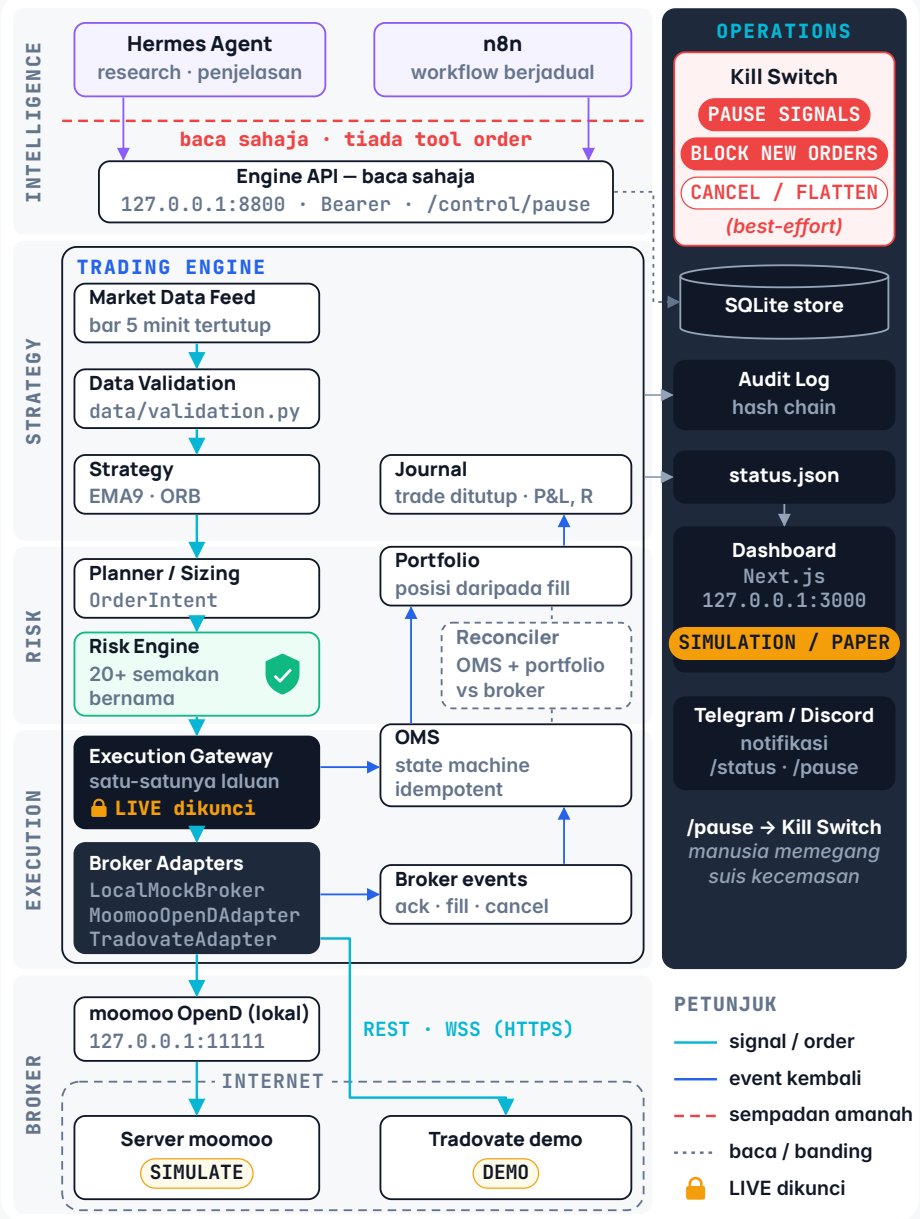
tentang broker. **Risk engine** deterministik menamakan *setiap* semakan yang gagal. **OMS** (*Order Management System*: `orders/oms.py` , `orders/state.py`) ialah buku rekod order, status dan fill. **EMS** (*Execution Management System*) bercakap dengan venue dan mengendalikan ralat, timeout dan had kadar — di sini `ExecutionGateway` bersama **adapter broker** (`brokers/moomoo.py` , `tradovate.py` , `mock.py`). **Portfolio** dikira *daripada fill*, dan **telemetry** terdiri daripada audit log berantai hash, `status.json` dan notifikasi.



AGENTIC INSIGHT

Kenapa satu laluan sahaja ke broker

`ExecutionGateway` ialah **satu-satunya** kod yang memanggil `submit_order` — bukan strategi, Telegram atau AI. Di situ dikuatkuasakan: `RiskDecision` yang diluluskan untuk intent yang *sama*, LIVE dikunci, persekitaran broker sepadan dengan mod (`MOCK` → `MOCK` ; `PAPER` → `SIMULATE` atau `DEMO`), dan satu intent menjadi satu order.



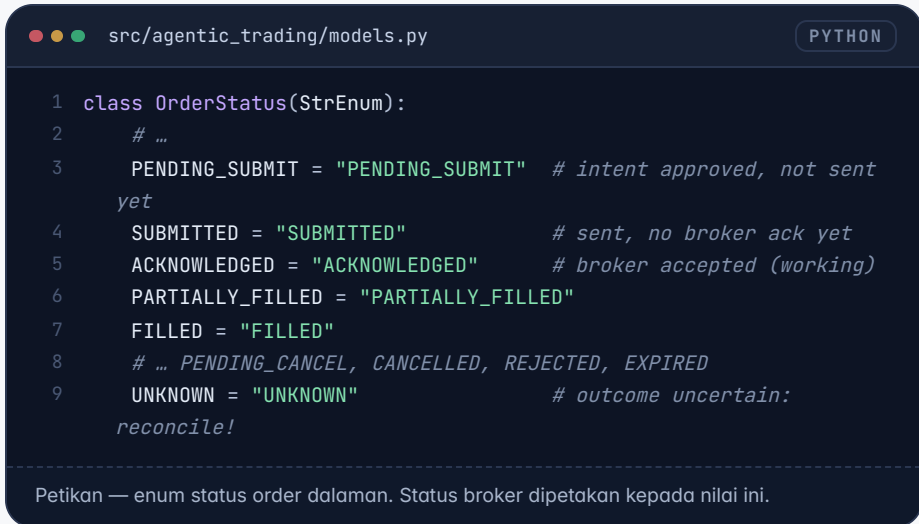
Rajah 2.2 — Seni bina penuh sistem Agentic Trading Siri 02. AI dan n8n hanya membaca melalui API baca sahaja; setiap order melalui risk engine dan execution gateway tunggal.

Latensi strategi 5 minit bukan masalah; masalah sebenar ialah **data lapuk** (risk engine menolak data lebih tua daripada `max_data_age_seconds` , lalai 120 saat dalam `config/default.yaml`), **bar belum ditutup** (enjin hanya memproses bar

tertutup), dan **broker tidak menjawab** — isu *kepastian*, dikendalikan dengan status `UNKNOWN`.

2.3 Submitted ≠ Acknowledged ≠ Filled

Apabila `place_order` atau `placeorder` pulang tanpa ralat, itu **tidak** bermaksud anda memegang posisi:



```

src/agentic_trading/models.py PYTHON
1 class OrderStatus(StrEnum):
2     # ...
3     PENDING_SUBMIT = "PENDING_SUBMIT" # intent approved, not sent
    yet
4     SUBMITTED = "SUBMITTED"           # sent, no broker ack yet
5     ACKNOWLEDGED = "ACKNOWLEDGED"    # broker accepted (working)
6     PARTIALLY_FILLED = "PARTIALLY_FILLED"
7     FILLED = "FILLED"
8     # ... PENDING_CANCEL, CANCELLED, REJECTED, EXPIRED
9     UNKNOWN = "UNKNOWN"              # outcome uncertain:
    reconcile!

```

Petikan — enum status order dalaman. Status broker dipetakan kepada nilai ini.

SUBMITTED: dihantar. **ACKNOWLEDGED:** broker menerima ("working"). **FILLED:** dipadankan — barulah wang bergerak (**PARTIALLY_FILLED** di antaranya).

UNKNOWN: "tidak pasti — tanya broker dahulu". Adapter memetakan status broker (moomoo `FILLED_PART`; Tradovate `Working`, `Filled`) kepada enum ini (Bab 11, Bab 14). Push moomoo boleh tiba **sebelum** `place_order` pulang; `orders/state.py` memastikan status tidak bergerak "ke belakang" dan fill sentiasa diaplikasikan (dinyahduplikasi melalui `fill_id`).

2.4 Idempotency: Satu Signal, Satu Order

Enjin restart dan membaca semula bar terakhir, atau kod "cuba lagi" selepas sambungan putus — anda kini memegang dua kontrak. **Idempotency:** signal sama diproses dua kali = **satu** order, melalui ID **deterministik**:

```

src/agentic_trading/models.py PYTHON
1 def deterministic_id(*parts: object, prefix: str = "") -> str:
2     """Stable id from inputs: the same signal always maps to one
   id."""
3     raw = "|".join(str(p) for p in parts)
4     digest = hashlib.sha256(raw.encode()).hexdigest()[:16]
5     return f"{prefix}-{digest}"
6 # ...
7 @property
8 def client_order_id(self) -> str:
9     # Same intent -> same client id -> broker/OMS can dedupe.
10    return f"AT-{self.intent_id[:20]}"

```

Petikan — ID stabil dan client order id.

`signal_id` (strategi, simbol, masa bar, arah) → `intent_id` → `client_order_id`. Gateway memanggil `claim_intent` (SQLite); intent yang sudah dituntut pulang sebagai `DUPLICATE`. Client order id dihantar ke broker (`remark` moomoo, `clOrdId` Tradovate) supaya selepas timeout kita *mencari semula* order ([Bab 23](#)).

Semuanya bermula di `TradingEngine.on_bar` bagi setiap bar tertutup:

```

src/agentic_trading/engine.py PYTHON
1 def on_bar(self, bar: Bar) -> ExecutionResult | None:
2     self.kill_switch.refresh()
3     # ... report = validateBars(...)
4     if not report.ok:
5         return None # ...
6     if not self.kill_switch.signals_allowed:
7         return None
8     if not self.broker_healthy:
9         return None
10    # ...
11    signal = self.strategy.on_bar(self.history, ctx)
12    # ...
13    return self.handle_signal(signal, data_ts=bar.ts +
    self.interval)

```

Petikan — beberapa baris ditinggalkan (# ...). Fail penuh dalam projek.

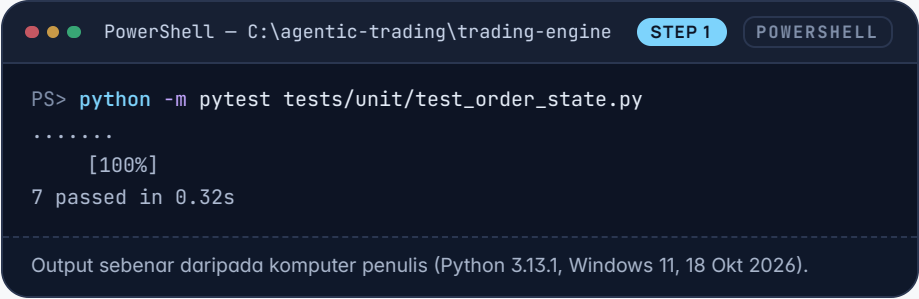
Apa yang kod ini buat?

Empat pintu: **kill switch** (dibaca semula dari fail supaya `/pause` serta-merta), **data sah, signal dibenarkan, broker sihat**. Kemudian `handle_signal` memanggil `self.risk.evaluate(...)`, merekod keputusan, dan jika lulus, `self.gateway.execute(...)`.

2.5 Tutorial: Lihat Kitaran Hidup Berjalan

Di komputer anda, folder `trading-engine`, venv aktif (Bab 04); tiada broker atau Administrator.

1 Uji state machine order



```
PowerShell - C:\agentic-trading\trading-engine STEP 1 POWERSHELL

PS> python -m pytest tests/unit/test_order_state.py
.....
      [100%]
7 passed in 0.32s
```

Output sebenar daripada komputer penulis (Python 3.13.1, Windows 11, 18 Okt 2026).

Apa yang command ini buat?

Tujuh ujian state machine (fill sebelum ack, peristiwa berganda, restart). **Pulih:** `ModuleNotFoundError` = salah folder atau venv belum aktif.

2

Jalankan capstone simulasi dengan 400 bar

```

PowerShell - C:\agentic-trading\trading-engine STEP 2 POWERSHELL

PS> python -X utf8 scripts/run_capstone.py --fresh --bars 400
[1] start · mode=MOCK live_trading_enabled=False
[3] data: 1560 bars · DATA SIMULASI – UNTUK TUJUAN PENDIDIKAN SAHAJA
[14] reconcile on start: RECONCILED: internal state matches broker
[10] journal: 8 closed simulated trades · net P&L (simulated) 1.33 ·
    var/journal.md

...
    audit chain valid: True
DATA SIMULASI – keputusan ini bukan prestasi dagangan sebenar.

-----
Output sebenar daripada simulator (DATA SIMULASI), dipendekkan (...). --fresh memadam
folder var/ dahulu.
  
```

Apa yang command ini buat?

400 bar MES sintetik (`fixtures/sim_mes_5m.csv`) disuap kepada enjin dengan EMA9 dan `LocalMockBroker`. **Amaran:** `--fresh` memadam `var/` — salin dahulu jika perlu. **Pengesahan:** `audit chain valid: True`. P&L ini tidak bermakna apa-apa.

3

Jejaki signal pertama melalui audit log

`scripts/trace_signal.py` membuka `var/trading.db` (baca sahaja) dan `var/audit.jsonl`, lalu memaparkan rekod audit bagi satu signal.

```

PowerShell - C:\agentic-trading\trading-engine STEP 3 POWERSHELL

PS> python -X utf8 scripts/trace_signal.py
SIGNAL SIG-dd3dd3739ec64d5a (1/18)
strategy=ema9-trend LONG MES bar=2026-09-01T15:35:00Z
entry=5001.25 stop=4978.25 target=5046.75
keputusan: APPROVED
15:47:14Z risk_decision APPROVED
15:47:14Z order_submit BUY 1 MES MARKET mode=MOCK
15:47:14Z fill BUY 1 @ 5001.50
15:47:14Z order_cancel_request AT-INT-0b347aa2b65f2321-ST
15:47:14Z order_cancel_request AT-INT-0b347aa2b65f2321-TA
audit chain valid: True
DATA SIMULASI — bukan dagangan sebenar.

Output sebenar daripada simulator (DATA SIMULASI). Masa audit ialah jam komputer
semasa larian; masa bar ialah masa simulasi (UTC).

```

Signal LONG (bar 15:35 UTC = 23:35 MYT) diluluskan; fill pada 5001.50 ialah **satu tick lebih buruk** daripada 5001.25 kerana `slippage_ticks: 1`. Kaki bracket `-ST` dan `-TA` dibatalkan kemudian, apabila EXIT menutup posisi — *sebelum* keluar, supaya stop tertinggal tidak membuka posisi baharu.

! SECURITY CHECK

Audit log ialah bukti

`var/audit.jsonl` dirantai hash SHA-256 — **tamper-evident**, bukan kalis gangguan. Kunci seperti `password`, `token`, `api_key`, `sec`, `cid` disamarkan (`***`); jangan simpan rahsia dengan nama lain.

✓ DISAHKAN DENGAN DOKUMENTASI

- `trading-engine/ 0.2.0`, `run_capstone.py`, `trace_signal.py`, `test_order_state.py`, `test_trace_signal.py` · 18 Okt 2026 · Python 3.13.1 · Windows 11 · MOCK · **TESTED** (simulasi; tiada broker sebenar).
- Push sebelum `place_order` pulang: <https://openapi.moomoo.com/moomoo-api-doc/en/trade/place-order.html> (18 Okt 2026) — **DOC-VERIFIED**.

TROUBLESHOOT `--fresh` gagal dengan WinError 32**SIMPtom** `PermissionError: [WinError 32]` pada `var\trading.db`.**PUNCA** Proses lain (API, bot Telegram) masih membukanya.**DIAGNOSIS** `Get-Process python` menyenaraikan proses Python.**PENYELESAIAN** **Ctrl+C** dalam tetingkap proses itu; ulang `--fresh`.**PENGESAHAN** Capstone selesai dengan `audit chain valid: True`.**LATIHAN PRAKTIKAL****Jejaki Signal Yang Tidak Menjadi Order**

Masa: 20 minit. Jalankan `trace_signal.py --n 2` dan `--n 13`. Kenapa yang kedua `NO ORDER: risk budget smaller than one unit at this stop?` ($\text{USD}25,000 \times 0.5\% = \text{USD}125$; 1 MES = \$5 setiap mata.) Tandakan langkah (1–11) bagi setiap rekod audit.

CHAPTER CHECKPOINT

Tandakan semua sebelum ke bab seterusnya



Saya boleh menerangkan sebelas langkah, status order dan ID deterministik, dan telah menjejaki dua signal.

RUMUSAN BAB

Signal menjadi intent, intent menjadi order hanya selepas risk engine meluluskannya, dan order menjadi posisi hanya apabila fill tiba. ID deterministik, audit log dan status UNKNOWN menjadikan setiap langkah idempotent, boleh disemak dan jujur. Bab 03 melihat moomoo dan Tradovate.

