Stock Trading Agent

股票交易智能体

Design Document v4 | 设计文档 v4

Yang Liu · Feb 2026

1. Overview

1. 概述

An AI-powered trading agent for US tech stocks, focused exclusively on earnings report events. When a major tech stock beats earnings expectations and surges, the agent monitors the surge and waits — when the surge slows down (dual validation), it shorts the stock expecting a pullback. After the pullback, it covers the short in 3 batches to close the position and take profit.

美股科技股 AI 交易智能体，只做财报行情。当主要科技股财报超预期并暴涨时，智能体监控涨势，等待涨势放缓（双验证通过）——然后做空股票，预期价格回调。回调后分 3 批平仓获利了结。

Strategy summary: Short selling after post-earnings surge slows. Borrow shares at high price, return them at lower price after pullback — profit is the difference.

策略总结：财报暴涨放缓后做空。高价借券卖出，回调后低价买回还券——差价即利润。

Broker: Interactive Brokers (IBKR) via Client Portal API. Requires Margin Account with short selling permission enabled. All credentials stored in environment variables, never hardcoded.

券商：盈透证券（IBKR），使用 Client Portal API。需要 Margin 账户并开通融券做空权限。所有账户信息存储在环境变量中，不写入代码。

Monitored Stocks

监控股票列表

TSLA	AAPL	NVDA	META	GOOGL	MSFT
AMZN	AMD	QCOM	WDC	CRM	PANW

2. Tech Stack

2. 技术栈

All code written in Python 3.11+. Python is chosen because all core libraries are Python-native and the author has 10 years of ML engineering experience in Python.

全部代码使用 Python 3.11+ 编写。选择 Python 是因为所有核心库都是 Python 原生支持，且作者有 10 年 Python ML 工程经验。

Library / 库	Version / 版本	Purpose / 用途

langgraph	latest	Task orchestration / 任务编排
anthropic	latest	Claude AI analysis / AI 分析判断
yfinance	latest	Real-time stock price data / 实时股价数据
ib_insync	latest	IBKR order execution / 盈透证券下单执行
twilio	latest	SMS notification + MFA / 短信通知+动态验证
chromadb	latest	Vector search for memory / 记忆系统向量检索
asyncio	built-in	Async SMS waiting + price polling / 异步等待短信和价格轮询
python-dotenv	latest	Load .env credentials / 加载环境变量账户信息
pyyaml	latest	Load config.yaml preferences / 加载用户偏好配置
requests	latest	Web search for earnings news / 搜索财报新闻

---

3. Startup Configuration

3. 启动配置

Before running the agent for the first time, the user must configure credentials and preferences. All sensitive information is stored in a .env file — never in the code.
首次运行智能体前，用户需要配置账户信息和偏好设置。所有敏感信息存储在.env 文件中——不写入代码。

Required Configuration (.env file)

必填配置（.env 文件）

- IBKR_USERNAME=your_username # IBKR account username
IBKR_USERNAME=你的用户名 # 盈透证券账户用户名
- IBKR_PASSWORD=your_password # IBKR account password
IBKR_PASSWORD=你的密码 # 盈透证券账户密码
- IBKR_ACCOUNT_ID=your_account_id # IBKR account ID
IBKR_ACCOUNT_ID=你的账户 ID # 盈透证券账户 ID
- TWILIO_ACCOUNT_SID=xxx # Twilio SMS service account SID
TWILIO_ACCOUNT_SID=xxx # Twilio 短信服务账户 SID

- TWILIO_AUTH_TOKEN=xxx # Twilio auth token
TWILIO_AUTH_TOKEN=xxx # Twilio 认证 Token
- USER_PHONE=+1xxxxxxxxxx # Your phone number for SMS confirmation
USER_PHONE=+86xxxxxxxxxx # 接收短信确认的手机号
- ANTHROPIC_API_KEY=xxx # Claude API key for AI analysis
ANTHROPIC_API_KEY=xxx # Claude API 密钥，用于 AI 分析

User Preferences (config.yaml)
用户偏好（config.yaml）

- max_short_position_per_stock: 10000 # Max $ value to short per stock
max_short_position_per_stock: 10000 # 每只股票最大做空金额（美元）
- price_guard_min_gain: 40 # Min $ gain above pre-earnings price before shorting
price_guard_min_gain: 40 # 做空前股价必须比财报前高出的最小金额（美元）
- max_days_to_wait_cover: 7 # Max days to wait for pullback before force-cover
max_days_to_wait_cover: 7 # 等待回调的最长天数，超过则强制平仓
- daily_loss_limit: 2000 # Max daily loss in $ before agent pauses
daily_loss_limit: 2000 # 每日最大亏损额，超过则暂停当天操作（美元）

---

3. Full Agent Flow
3. 完整流程

All 10 steps orchestrated by LangGraph with conditional routing. Security layer wraps every operation. Flow loops back to Step 1 after each completed trade.
全部 10 个步骤由 LangGraph 编排，条件路由控制每步走向，安全层包裹每个操作。完成一次交易后回到第 1 步。

[STEP 1: Earnings Calendar Collection]
[第 1 步：财报日历采集]

Runs continuously in background
持续后台运行

Every day: fetch earnings dates for all 12 stocks, write to daily log
每天：抓取全部 12 只股票的财报日期，写入日志

No upcoming earnings → keep watching, write to log
无即将到来的财报 → 继续监控，写入日志

Upcoming earnings detected within 1 week → trigger Step 2
发现 1 周内有财报 → 触发第 2 步

↓

[STEP 2: Earnings Result Detection]

[第 2 步：财报结果采集]

After earnings release: search web for news about this stock's earnings
财报发布后：搜索网页获取该股票的财报新闻

Analyze: did it beat expectations? By how much?
分析：是否超预期？超出幅度是多少？

EPS beat < 5% → skip, risk/reward not worth it
EPS 超预期幅度 < 5% → 跳过，风险回报不足

EPS beat > 10% → strong signal, continue to Step 3
EPS 超预期幅度 > 10% → 强信号，继续第 3 步

Miss expectations → stop, wait for next stock
未超预期 → 停止，等待下只股票

↓

[STEP 3: Surge Detection]
[第 3 步：暴涨判断]

Real-time price data via Yahoo Finance API (every 5 minutes)
Yahoo Finance API 实时价格数据（每 5 分钟）

Also check market environment: SPY/QQQ daily change
同时检查大盘环境：SPY/QQQ 当日涨跌幅

Market down > 2% → pause, do not trade today
大盘跌幅 > 2% → 暂停，今日不操作

Stock not surging → keep watching
股票未暴涨 → 持续监控

Stock surging significantly → continue to Step 4
股票暴涨 → 继续第 4 步

↓

[STEP 4: Slowdown Detection — Short Selling Trigger — Dual Validation]
[第 4 步：涨势放缓判断——做空触发信号——双验证]

Day 1, 2, N: surge still ongoing → keep watching, do NOT short yet
第 1、2...N 天：涨势仍在 → 持续观察，不做空

Hard Rules — at least 2 of 3 must be met:
硬规则——满足 3 条中的 2 条：

· 5-min price increase < 0.3%
· 5 分钟涨幅 < 0.3%

· Volume down 40% vs prior 30 minutes
· 成交量比前 30 分钟下降 40%

· Price pulled back > 1.5% from today's high
· 距今日最高点回落超过 1.5%

AI Confidence: Claude analyzes real-time data → confidence score 0-100, must be > 70%
AI 置信度：Claude 分析实时数据 → 置信度 0-100 分，必须 > 70%

Price Guard: short price must be $40+ above pre-earnings price
价格保护：做空价格必须比财报前高$40 以上，确保足够利润空间

Dynamic Stop Loss (auto-calculated per stock based on 30-day volatility):

动态止损（根据每只股票 30 日波动率自动计算）：

· High volatility e.g. TSLA (volatility > 3%): stop loss = 8% above short price
· 高波动如 TSLA（波动率>3%）：止损线 = 做空价上方 8%

· Medium volatility (volatility 2-3%): stop loss = 6%
· 中波动（波动率 2-3%）：止损线 = 6%

· Low volatility e.g. AAPL (volatility < 2%): stop loss = 5%
· 低波动如 AAPL（波动率<2%）：止损线 = 5%

NOT MET → keep watching
未满足 → 继续监控

ALL MET → trigger short sell signal, continue to Step 5
全部满足 → 触发做空信号，进入第 5 步

↓

[STEP 5: ReAct Self-Verification (fixed 2 iterations)]
[第 5 步：ReAct 自我验证（固定 2 次，避免死循环）]

Iteration 1: Prompt Enhancement → enrich intent → execute full analysis
第 1 次：Prompt 增强 → 丰富意图理解 → 执行完整分析

Iteration 2: Verify iteration 1 result is correct → only output if verified
第 2 次：验证第 1 次结果是否正确 → 通过后才输出

Fixed at 2 iterations — no infinite loop risk
固定 2 次——无死循环风险

↓

[STEP 6: SMS Notification + MFA]
[第 6 步：短信通知 + 动态验证]

Send SMS to user with full analysis:
发送短信给用户，包含完整分析：

· Stock name, current price, pre-earnings price, surge %
· 股票名称、当前价格、财报前价格、暴涨幅度

· AI confidence score
· AI 置信度分数

· Dynamic stop loss price for this stock
· 该股票动态止损价格

· Program calculates suggested short batch size based on account balance
· 程序根据账户余额自动计算建议做空批次

· Example: NVDA @ $875, pre-earnings $820, up $55. Short 30 shares, stop loss @ $945
· 示例：NVDA @ $875，财报前$820，涨$55。建议做空 30 股，止损价$945

User replies: YES / NO / CHANGE [number]
用户回复：YES / NO / CHANGE [数量]

NO → cancel entire operation
NO → 取消本次操作

YES or CHANGE → proceed to Step 7
YES 或 CHANGE → 进入第 7 步

↓

[STEP 7: Short in 3 Batches]
[第 7 步：分批做空（3 次）]

Each batch: send SMS with details → wait YES → execute short via IBKR API
每批：发短信说明详情 → 等 YES → IBKR API 执行做空（借券卖出）

Batch 1: ~30% of planned short position → SMS confirm → execute
第 1 批：约 30%计划做空仓位 → 短信确认 → 执行

Batch 2: ~30% of planned short position → SMS confirm → execute
第 2 批：约 30%计划做空仓位 → 短信确认 → 执行

Batch 3: remaining ~40% → SMS confirm → execute
第 3 批：剩余约 40% → 短信确认 → 执行

Stop loss auto-monitored throughout — if triggered, send SMS alert immediately
全程自动监控止损线——触达立即发短信提醒，等用户决定是否平仓

↓

[STEP 8: Wait for Pullback]
[第 8 步：等待回调]

After shorting: monitor price every 5 minutes
做空后：每 5 分钟监控一次价格

Day 1-N: price still high or rising → hold short position, keep watching
第 1-N 天：价格仍高或继续涨 → 持有空仓，继续观察

Stop loss triggered at any time → send SMS alert immediately
任何时候触达止损线 → 立即发短信提醒

Day 7+: no pullback detected → send SMS: recommend covering to limit risk
第 7 天以上无回调 → 发短信：建议平仓控制风险

Pullback detected (price drops significantly from short price) → trigger Step 9
检测到明显回调（价格从做空价显著下跌）→ 触发第 9 步

↓

[STEP 9: Cover Short in 3 Batches — Close Position, Take Profit]
[第 9 步：分批平仓（3 次）——买回股票还券，获利了结]

Send SMS with cover analysis: current price, profit per share, total profit estimate
发短信含平仓分析：当前价格、每股利润、预计总利润

Each batch: send SMS → wait YES → execute buy-to-cover via IBKR API
每批：发短信 → 等 YES → IBKR API 执行买入平仓（买回股票还给券商）

Batch 1: cover ~30% of short position → profit locked on this portion
第 1 批：平仓约 30%空仓 → 这部分利润已锁定

Batch 2: cover ~30% of short position
第 2 批：平仓约 30%空仓

Batch 3: cover remaining ~40% — all positions closed, trade complete
第 3 批：平仓剩余约 40% — 全部仓位关闭，本次交易结束

↓

[STEP 10: Memory + Reflection]
[第 10 步：记忆 + 反思]

Write to short-term memory (current session log):

写入短期记忆（当前会话日志）：

- Full trade timeline: every price check, every SMS, every execution
- 完整交易时间线：每次价格检查、每条短信、每次执行

Write to long-term MEMORY.md:
写入长期记忆 MEMORY.md：

- Trade record: stock, short price, cover price, profit/loss, days held
- 交易记录：股票、做空价、平仓价、盈亏、持仓天数
- Stock profile: win rate, average post-earnings surge, average pullback days
- 股票画像：胜率、平均财报后涨幅、平均回调天数
- User preferences: any CHANGE commands the user made (position size adjustments)
- 用户偏好：用户发过的 CHANGE 指令（仓位调整记录）

Write reflection:
写反思：

- Why did I short at this price? Was the timing right?
- 为什么在这个价格做空？时机对吗？
- How many days did the pullback take? Match historical average?
- 回调用了几天？和历史平均值吻合吗？
- What would I do differently next time?
- 下次有什么可以改进的？

Memory retrieval: vector search + keyword matching, recent trades weighted higher
记忆检索：向量搜索+关键词匹配，近期交易权重更高

Memory expiry: 6 months → lower weight, 1 year → archived
记忆过期：6 个月 → 降权，1 年以上 → 归档不参与检索

Clear short-term memory, loop back to Step 1
清空短期记忆，回到第 1 步等待下次财报

---

4. LangGraph Task Orchestration

4. LangGraph 任务编排

All 10 steps connected by LangGraph with conditional routing. Flow loops back to Step 1 after each completed trade cycle.
全部 10 个步骤由 LangGraph 串联，条件路由控制每步走向，完成交易后回到第 1 步。

```
# Entry point — runs continuously / 入口——持续运行
workflow.set_entry_point('earnings_calendar')

# Step 1 → 2: trigger only when earnings found / 发现财报才触发
workflow.add_conditional_edges('earnings_calendar', check_earnings_upcoming, {
    'found':   'earnings_result',   # earnings found / 有财报
    'waiting': 'earnings_calendar'  # no earnings / 无财报，继续监控
```

```
})

# Step 2 → 3: only strong beat proceeds / 只有大幅超预期才继续
workflow.add_conditional_edges('earnings_result', check_beat_expectations, {
    'strong_beat': 'surge_detection',  # beat > 10% / 超预期>10%
    'weak_beat':   END,                # beat < 5% / 超预期<5%，跳过
    'miss':        END                 # miss / 未超预期，结束
})

# Step 3 → 4: market check + surge check / 大盘检查+暴涨检查
workflow.add_conditional_edges('surge_detection', check_surge, {
    'surging':     'slowdown_detection',  # surging / 暴涨
    'market_down': END,                   # market down > 2% / 大盘跌>2%
    'waiting':     'surge_detection'      # not yet / 未暴涨
})

# Step 4 → 5: dual validation must pass / 双验证通过才做空
workflow.add_conditional_edges('slowdown_detection', check_dual_validation, {
    'confirmed': 'react_verify',       # pass / 通过 → 做空
    'waiting':   'slowdown_detection'  # not yet / 未通过，继续监控
})

# Step 5 → 6: after verification send SMS / 验证通过发短信
workflow.add_edge('react_verify', 'sms_notify')

# Step 6 → 7: execute only after user YES / 用户 YES 才执行
workflow.add_conditional_edges('sms_notify', check_user_reply, {
    'yes': 'short_batch',  # user YES / 用户 YES → 开始做空
    'no':  END             # user NO / 用户 NO → 取消
})

# Step 7: short in batches loop / 分批做空循环
workflow.add_conditional_edges('short_batch', check_short_complete, {
    'more': 'short_batch',         # more batches / 还有批次
    'done': 'pullback_detection'   # all shorted / 做空完毕，等回调
})

# Step 8 → 9: cover when pullback detected / 回调后平仓
workflow.add_conditional_edges('pullback_detection', check_pullback, {
    'pullback':   'sms_cover_notify',   # pullback / 回调 → 发短信
    'stop_loss':  'sms_stop_loss',      # stop loss hit / 止损触达
    'timeout':    'sms_cover_notify',   # day 7+ / 超时建议平仓
    'waiting':    'pullback_detection'  # not yet / 继续等
})
```

```
# Step 9: cover in batches loop / 分批平仓循环
workflow.add_conditional_edges('cover_batch', check_cover_complete, {
    'more': 'cover_batch',  # more batches / 还有批次
    'done': 'memory'        # all covered / 全部平仓，写记忆
})

# Step 10 → Step 1: loop back / 完成后回到起点
workflow.add_edge('memory', 'earnings_calendar')
```

---

5. Three Core Principles
5. 三个核心原则

1. Prompt Enhancement
1. Prompt 增强

Before every node executes, the agent enriches the intent — adding context, breaking down sub-tasks, considering edge cases — then executes based on true intent, not literal instruction.
每个节点执行前，智能体先丰富意图——添加上下文、拆解子任务、考虑边界情况——然后基于真实意图执行，而不是字面指令。

2. Cross-session Memory (OpenClaw-style)
2. 跨会话记忆（OpenClaw 方式）

Two-tier memory: short-term for current session (full trade timeline), long-term MEMORY.md for trade history, stock profiles, and user preferences. Retrieval uses vector search + keyword matching, recent memories weighted higher. Records older than 6 months lose weight, older than 1 year are archived.
两层记忆：短期记忆用于当前会话（完整交易时间线），长期 MEMORY.md 记录交易历史、股票画像和用户偏好。检索使用向量搜索+关键词匹配，近期记忆权重更高。6 个月以上记录降权，1 年以上归档。

3. Multi-tool Coordination
3. 多工具协调

Each tool does what it does best: Yahoo Finance for real-time price data, web search for earnings news, Claude AI for analysis and judgment, Twilio for SMS confirmation, IBKR for order execution. LangGraph coordinates all tools without manual handoffs.
每个工具做最擅长的事：Yahoo Finance 提供实时价格，网页搜索获取财报新闻，Claude AI 分析判断，Twilio 发送短信确认，IBKR 执行下单。LangGraph 协调所有工具，无需人工交接。

6. Security Layer
6. 安全层

Security is not a separate module — it wraps every single operation in the flow.
安全层不是独立模块——它包裹流程中的每一个操作。

Physical Isolation: Docker Sandbox
物理隔离：Docker 沙箱

- All code execution runs inside a Docker container
所有代码在 Docker 容器内执行
- No external network except Yahoo Finance, IBKR, Twilio, and Anthropic APIs
无外网访问，只允许 Yahoo Finance、IBKR、Twilio、Anthropic API
- Read-only filesystem except /workspace
只读文件系统，除/workspace 外
- Non-root user, all Linux capabilities dropped
非 root 用户，禁用所有 Linux 能力
- Container destroyed after each session
每次会话结束后销毁容器

Guardrails: Per-operation AI Safety Check
反作弊：每步 AI 安全检查

- Before every operation, Claude checks: is this safe to execute?
每个操作前，Claude 检查：这个操作安全吗?
- Dangerous patterns blocked immediately (e.g. short entire account balance)
危险操作立即拦截（如做空全部账户余额）
- Credentials never passed to AI context — environment variables only
账户凭证不传给 AI 上下文，只用环境变量

MFA: Human in the Loop
动态验证：人工审批

- No trade executes without user SMS confirmation
没有用户短信确认，任何交易都不执行
- Every single batch requires a fresh YES reply
每一批次都需要新的 YES 回复
- User can reply NO or CHANGE at any point to cancel or adjust

用户随时可以回复 NO 取消或 CHANGE 调整数量

---

7. Risk Control
7. 风险控制

Dynamic Stop Loss
动态止损

- Stop loss calculated per stock based on 30-day historical volatility
根据每只股票 30 日历史波动率自动计算止损线
- High volatility e.g. TSLA (> 3%): stop loss = 8% above short price
高波动如 TSLA（>3%）：止损线 = 做空价上方 8%
- Medium volatility (2-3%): stop loss = 6%
中波动（2-3%）：止损线 = 6%
- Low volatility e.g. AAPL (< 2%): stop loss = 5%
低波动如 AAPL（<2%）：止损线 = 5%

Daily Loss Limit
每日亏损上限

- If total daily loss exceeds user-defined limit → stop all operations for the day
当日总亏损超过用户设定上限 → 停止当天所有操作
- Send SMS: 'Daily loss limit reached. Agent paused until tomorrow.'
发短信：当日亏损上限已触达，智能体暂停至明日

Market Environment Check
大盘环境检查

- Check SPY/QQQ daily change before every trade
每次操作前检查 SPY/QQQ 当日涨跌幅
- Market down > 2% → pause all operations
大盘跌幅 > 2% → 暂停所有操作

Timeout Protection
超时保护

- Day 7+: no pullback → send SMS recommending cover to limit risk
第 7 天以上无回调 → 发短信建议平仓控制风险

- User decides whether to cover or continue holding

由用户决定是否平仓

---

8. Pending Confirmation

8. 待确认事项

Item / 事项	Status / 状态	Notes / 备注
Broker API / 券商	IBKR confirmed / 已确认	Client Portal API, Margin account required
SMS Provider / 短信服务	TBD / 待定	Twilio recommended / 推荐 Twilio
Max short position / 最大做空仓位	TBD / 待定	Set in config.yaml / 在 config.yaml 中设置
Batch split ratio / 批次比例	30/30/40 confirmed / 已确认	Adjustable / 可调整
$40 price guard / $40 价格保护	Confirmed / 已确认	Adjustable per stock / 每只股票可调
Max wait days / 最长等待天数	7 days confirmed / 已确认	Adjustable in config / 可在配置中调整

我自己的 prompt：

对啊我现在跟你的讨论是 设计部分 设计完善 就是你把这个 agent 写出来 我要 put 到 github 上 这个设计包括 记忆 包括 输入我要做特斯拉 包括我的账号 包括 先做所有数据采集 先采集 一个股票的最近上涨了很多 科技股尤其是科技股吧 在财报发出以后 我只要做这个 财报发出去以后才会有突然上涨 或者大下跌操作 所以我们只做财报前后的大上涨 然后 这些数据采集 。分析这些数据 因为美股科技股财报还是在一个固定时间 首先你要采集比较出名的科技股 他们财报时间。然后财报报道以后的时候是不是报账了 这个也需要用搜索网页的方式 获取 。 然后是暴涨了 就判断用两个方式判断 第一个是 AI 判断 第二个方式是我自己写函数判断 上涨的比较平缓了。这个平缓 已经判断 yes 的时候 就卖出。同时这个卖出价格 也要明显高于上涨之前（财报之前）的价格比如涨了 40 美刀以上。 然后这个过程也是一个工具对吧。 然后再做 给用户发送动态验证 。 给用户发送短信分析结果 。然后再操作盈透账号进行卖出操作。 卖出以后 再上涨 再卖出 不是一次交易所有资金都用上。 分三次卖出。 这样操作完 等待买入时机 分三次买入 。 这个使整个 agent 过程 。我要加入记忆。万一记录每个股票 盈利多少 亏损多少。然后同时 加入反思 。长短记忆和 openclaw 的方式一样做记忆 。另外再做用 langgraph 把我上面说的几个过程做任务编排 。然后再做安全问题，第一做 Docter 沙漏 隔离 另外 每个操作 都组织 AI prompt 判断这个操作 是否安全 。做完一次以后

再进行 react 方式自我验证一遍 做的结果对不对 。如果对就输出。 react 如果有不容易退出循环问题 就固定做两次 第二次主要验证 第一次把 prompt 增强做好 。我不是说了三点吗 都用上了 我的需求够具体么