/Options Fix Plan

API-Validated Implementation Guide

February 26, 2026

Generated by: 16-Persona Expert Brainstorm Session

Groups 2A · 2B · 2C · 2D

17 **P0 Critical Bugs**	**10 Phases** **8 Working Days**	**33.5 hrs** **Total Effort**	**3 APIs** **ORATS · Finnhub · Tradier**

Section 1 · Executive Summary

Overview

This document is the definitive action guide for fixing the **/Options** trading platform. All 17 P0 (Critical) bugs have been identified, root-caused, and fix-planned through a 16-persona expert brainstorm session. Every proposed fix has been validated against live API testing of ORATS, Finnhub, and Tradier (sandbox) to confirm that the required data exists and the API calls work as described.

Critical Findings

Finding	Impact	Fix Complexity
P0-1: return trapped in for-loop (options_analyzer.py)	Only 1st strike of 1st expiry ever returned — scanner produces near-zero results	S (dedent 8 lines)
P0-2: Skew score always returns 50 (use_schwab gate)	15% of scoring weight wasted; all tickers appear identical in skew dimension	M (new ORATS method)
P0-3: Earnings risk check hardcoded False	Weekly options placed across earnings events without warning or penalty	M (Finnhub integration)
P0-5: FLASK_DEBUG defaults True	RCE exposure — Werkzeug interactive debugger accessible on any traceback URL	S (1 character)
P0-6/7: OCO key mismatch + market stop type	Bracket orders silently fail; stop-loss is a market order (catastrophic gap fills)	L (cancel+recreate)
P0-10: _get_username() auth bypass to 'demo'	Unauthenticated requests access all demo account data	S (3 lines)
P0-14: Bookend snapshots record stock price, not option price	All MFE/MAE calculations corrupted ($242 stock price stored instead of $3.50 option)	M (refactor)

Key API Discoveries (Verified Live)

API	Discovery	Impact on Fixes
ORATS hist/cores	Has ivPctile1y, daysToNextErn, impliedEarningsMove, divDate, contango (340 fields). T-1 delay. Returns 4817 rows — must cache.	Enables P0-3 earnings, P1-A2 IVP regime, P1-A3 ex-div check
ORATS live/summaries	Has real-time skew: rSlp30, skewing, contango, dlt25Iv/dlt75Iv per timeframe (129 fields).	Fixes P0-2 skew calculation (replaces dead use_schwab path)
ORATS live/cores	FORBIDDEN (premium). Cannot use for real-time IV rank.	Confirms hist/cores is the only IV percentile source
Tradier VIX	GET /markets/quotes?symbols=VIX WORKS — returns last=18.63 real-time.	Enables XC-1 VIX regime filter
Tradier OCO modify	CANNOT modify OCO orders — cancel + recreate only. Indexed keys required: quantity[0], stop[1], price[1].	Requires P0-6 to implement cancel+recreate with naked-position fallback
Tradier stop_limit	type[N]='stop_limit' with both stop[N] + price[N] accepted. type[N]='stop' is a dangerous market order.	Directly fixes P0-7
Finnhub /calendar/earnings	WORKS — real-time earnings dates with EPS estimates and hour (bmo/amc/dmh).	Enables P0-3 earnings risk check
Finnhub /calendar/economic	FORBIDDEN (premium). VIX also blocked via Finnhub.	Confirms Tradier as VIX source

Implementation Summary

- 10-phase implementation over 8 working days, ~33.5 hours total effort
- All 17 P0 fixes shipped by end of Day 5 (Phase 6)

- Phases 0–2 (Days 1–2): Emergency security + critical one-line fixes + risk controls
- Phases 3–6 (Days 2–5): New API methods + scanner data fixes + broker order fixes + monitor fixes
- Phases 7–9 (Days 5–8): Architectural improvements, code quality, retry resilience
- Every PR ships with regression tests — no merge without tests
- God class (hybrid_scanner_service.py, 1823 lines) decomposition deferred to Sprint 2 to avoid merge conflicts

Section 2 · API Capability Summary

Endpoint Status — Verified Against Live APIs

API	Endpoint	Status	Data Provided
ORATS	live/strikes	**WORKS**	Full chain: greeks, bid/ask, OI, volume — 44 fields/strike
ORATS	live/summaries	**WORKS**	IV term structure + skew: rSlp30, skewing, contango, dlt25Iv/dlt75Iv — 129 fields
ORATS	live/monies/implied	**WORKS**	Per-expiry IV surface: slope, atmiv, earnEffect, vol0-vol100 — 44 fields
ORATS	hist/cores	**WORKS (T-1)**	ivPctile1y/1m, daysToNextErn, impliedEarningsMove, divDate, contango — 340 fields
ORATS	hist/dailies	**WORKS**	OHLCV price history for technical indicators
ORATS	hist/summaries	**WORKS**	Historical IV data — 128 fields
ORATS	live/cores	**FORBIDDEN**	Premium tier — cannot use for real-time IV rank
ORATS	live/ivrank	**FORBIDDEN**	Premium tier — IV rank/percentile real-time
Finnhub	/calendar/earnings	**WORKS**	Real-time earnings dates, EPS estimates, hour (bmo/amc/dmh)
Finnhub	/news-sentiment	**WORKS**	Bullish/bearish %, buzz score, company news score
Finnhub	/stock/metric	**WORKS**	132 financial metrics: ROE, PE, margins, beta, 52-wk hi/lo
Finnhub	/stock/recommendation	**WORKS**	Analyst buy/hold/sell consensus
Finnhub	/stock/dividend	**WORKS**	Ex-dividend dates, amounts, pay dates
Finnhub	/calendar/economic	**FORBIDDEN**	Premium tier — Fed/macro calendar blocked
Finnhub	/stock/upgrade-downgrade	**FORBIDDEN**	Premium tier — individual upgrades blocked
Tradier	GET /markets/quotes	**WORKS**	Real-time quotes including VIX (last=18.63 confirmed)
Tradier	GET /markets/options/chains	**WORKS**	Options chain with greeks (delta, gamma, theta, vega, smv_vol)
Tradier	POST /accounts/{id}/orders	**WORKS**	All order types: equity, option, OCO, OTOCO
Tradier	PUT /accounts/{id}/orders/{id}	**PARTIAL**	Can modify simple orders ONLY — OCO cannot be modified
Tradier	DELETE /accounts/{id}/orders/{id}	**WORKS**	Cancel any order type

Data Source Assignments

Data Need	Primary Source	Endpoint / Field	Fallback
IV Percentile	ORATS	hist/cores → ivPctile1y (T-1)	Use 50 (neutral) if unavailable
Real-time Skew	ORATS	live/summaries → rSlp30, skewing	Existing calculate_skew() on chain data
Earnings Date	Finnhub	/calendar/earnings → date, hour (real-time)	ORATS hist/cores → daysToNextErn (T-1)
Implied Earnings Move	ORATS	hist/cores → impliedEarningsMove (T-1)	None available
VIX Level	Tradier	/markets/quotes?symbols=VIX → last	Default 20 (moderate regime)
Ex-Dividend Date	ORATS	hist/cores → divDate (T-1)	Finnhub /stock/dividend
Sentiment Score	Finnhub	/news-sentiment → bullishPercent, buzz	Neutral (50)
Analyst Consensus	Finnhub	/stock/recommendation → buy/hold/sell	Not used if unavailable
Option Quote (live)	ORATS	live/strikes → bid/ask/mark per contract	trade.current_price (last known)
OCO Order Execution	Tradier	POST /accounts/{id}/orders (class=oco)	No fallback — manual intervention

Section 3 · Implementation Timeline

Phase-by-Phase Schedule

Phase	PR	Day(s)	Est. Hours	Fixes Covered	Risk Level
0: Emergency Security	#1	Day 1 AM	0.5	P0-5, P0-9, P0-10	**NONE**
1: Critical One-Line Fixes	#2	Day 1 PM	2.0	P0-1, P0-11, P0-12, P0-15, QW-8, QW-9, QW-10	**LOW**
2: Risk Controls	#3	Day 2 AM	3.0	P0-4, P0-8, P0-16	**MEDIUM**
3: New API Methods	#4	Day 2-3	3.0	orats.get_summary, orats.get_hist_cores, finnhub.get_earnings_calendar, QW-5	**NONE**
4: Scanner Data Fixes	#5	Day 3	4.0	P0-2, P0-3, P0-13, P1-A10	**MEDIUM**
5: Broker / Order Fixes	#6	Day 4	6.0	P0-6, P0-7	**HIGH**
6: Monitor Data Fixes	#7	Day 5 AM	3.0	P0-14 + data migration	**LOW**
7: Scan Direction + VIX	#8	Day 5 PM	4.0	P0-17, XC-1	**MEDIUM**
8: Code Quality Cleanup	#9	Day 6-7	4.0	QW-6, QW-7, QW-11, QW-12, QW-13, XC-5, P1-A12, XC-7	**NONE**
9: Retry + Resilience	#10	Day 7-8	4.0	P1-A7 (retry decorator), P2-A3 (scheduler max_instances)	**LOW**
TOTAL	**10 PRs**	**8 days**	**33.5 hrs**	**17 P0 + 7 P1 + all QW**	

Risk Matrix by PR

PR	Files Changed	Merge Conflict Risk	Rollback Difficulty	Sandbox Test Required?
#1	config.py, paper_routes.py (2)	None	Trivial	No
#2	options_analyzer.py, hybrid_scanner_service.py (2)	Low	Trivial	No
#3	paper_routes.py, app.py (2)	None	Easy — monitor lifecycle_sync first run	No
#4	orats.py, finnhub.py (2)	None	Trivial	**YES — verify new endpoints live**
#5	hybrid_scanner_service.py (1, God class)	Medium	Moderate — compare scan results before/after	**YES**
#6	tradier.py, monitor_service.py (2)	Low	**HARD — order state implications**	**YES — mandatory Tradier sandbox**
#7	monitor_service.py + migration (2)	Low	Easy	No
#8	hybrid_scanner_service.py (1, God class)	Medium	Moderate — verify put scans	**YES**
#9	5 files	Low	Easy	No
#10	4 files	Low	Easy	No

Section 4 · P0 Fix Specifications

17 P0 Critical findings. For each: root cause, fix summary, API data source, test case, feasibility verdict, phase assignment, and effort.

P0-1 · _process_expiration — return trapped inside for-loop

Phase: Phase 1 / PR #2

File: options_analyzer.py L252 | **Feasibility: CONFIRMED** | **Effort: S**

Root Cause	The return opportunities statement and debug block are indented inside the for strike_str, option_list loop, causing Python to return after the first strike of the first expiry, discarding all remaining strikes and expiry dates.
Fix Summary	Dedent the debug summary block and return statement by one level — outside the for-loop, inside the method. Remove the duplicate return at L70 of parse_options_chain (dead code). Total change: 1 indentation fix.
API / Data Source	N/A — logic bug, no API involved
Test Case	Given: AAPL chain with 4 expiries, 20+ strikes each. When: parse_options_chain(). Expect: results from all 4 expiry dates (>=10 opportunities). Currently broken: returns max 1 result (first strike only).
PR Note	Unblocks P0-17 (LEAP puts). Most impactful single fix — 10-20x more results.

P0-2 · Skew score dead — always returns 50 (use_schwab gate)

Phase: Phase 4 / PR #5 (after PR #4 adds get_summary)

File: hybrid_scanner_service.py L423 | **Feasibility: NEEDS MODIFICATION** | **Effort: M**

Root Cause	calculate_skew() is fully implemented in options_analyzer.py but the call site in hybrid_scanner_service.py is gated by if self.use_schwab and options_data:. Since use_schwab = False (hardcoded), skew_score = 50 is always used. 15% of scoring weight is permanently wasted.
Fix Summary	Replace the use_schwab gate with use_orats. Add a new OratsAPI.get_summary(ticker) method that calls live/summaries to retrieve rSlp30 (30-day risk-neutral slope). Normalize rSlp30 to 0-100: score = max(0, min(100, 50 + rSlp30 * 5)). Cache the live/summaries response for 5 minutes (changes slowly).
API / Data Source	ORATS live/summaries → rSlp30 (30-day risk-neutral slope), skewing (persistence), contango. Fallback: existing calculate_skew() on chain data.
Test Case	Given: NVDA with known put-heavy skew. When: scan NVDA. Expect: skew_score between 30-45 (bearish), not 50. Verify against ORATS dashboard rSlp30 for NVDA.
Validation Note	Proposed method get_skew() does not exist — use live/summaries.rSlp30 instead. New OratsAPI.get_summary() method required first.
PR Note	Scores shift ±7.5 pts after fix — expected and correct.

P0-3 · Earnings risk check hardcoded False — never fires

Phase: Phase 4 / PR #5 (after PR #4 adds get_earnings_calendar)

File: hybrid_scanner_service.py L1031 | **Feasibility: CONFIRMED** | **Effort: M**

Root Cause	The earnings check block was removed during "Strict Mode" migration (see L1032 comment). The replacement Finnhub integration was never wired in. has_earnings_risk = False is hardcoded permanently.
Fix Summary	Add a new FinnhubAPI.get_earnings_calendar(symbol, _from, to) method. Query for earnings within the next 7 days at scan time. Set has_earnings_risk = True when a match is found. Apply -20 score penalty for WEEKLY/0DTE positions, -5 for LEAPs. Also fetch ORATS hist/cores.impliedEarningsMove for the expected move size.
API / Data Source	Finnhub /calendar/earnings → date, hour (bmo/amc/dmh), epsEstimate (real-time). ORATS hist/cores → impliedEarningsMove, daysToNextErn (T-1 fallback).
Test Case	Given: NVDA with earnings on March 5. When: weekly scan on Feb 27. Expect: has_earnings_risk=True, earnings_date=2026-03-05, score penalty applied. Currently broken: has_earnings_risk=False, no warning.
Validation Note	New FinnhubAPI.get_earnings_calendar() method required. Batch the earnings query (all tickers at once) to stay within Finnhub 60 req/min limit.
PR Note	Rate limit: 50-ticker scan = 50 Finnhub calls. Recommend single batch query at scan start.

P0-4 · Daily loss limit stored but never enforced

Phase: Phase 2 / PR #3

File: paper_routes.py L334-348 | **Feasibility: CONFIRMED** | **Effort: M**

Root Cause	daily_loss_limit is stored in UserSettings and shown in the UI. The place_trade() route queries UserSettings for max_daily_trades but never queries realized P&L.; No daily P&L; check exists in the trade placement flow.
Fix Summary	Insert a SQLAlchemy ORM query for sum(realized_pnl) of CLOSED trades today immediately after the max_daily_trades check and before trade creation. Use func.coalesce(func.sum(PaperTrade.realized_pnl), 0) with filter on username, status=CLOSED, and date(closed_at)=today(). Return HTTP 403 (not 429) if limit is breached.
API / Data Source	Internal PostgreSQL — paper_trades table, realized_pnl and unrealized_pnl columns. Verify index exists on (username, status, closed_at).
Test Case	Given: User with daily_loss_limit=$500, two closed trades today totaling -$550. When: place_trade() called. Expect: HTTP 403, error message includes today P&L.; Currently broken: trade placed successfully.
Validation Note	Use SQLAlchemy ORM (not raw SQL). Count realized losses of CLOSED trades only. Use HTTP 403 not 429.
PR Note	Add index on (username, status, closed_at) if missing.

P0-5 · FLASK_DEBUG defaults True — RCE exposure

Phase: Phase 0 / PR #1

File: config.py L56 | **Feasibility: CONFIRMED** | **Effort: S**

Root Cause	config.py L56: FLASK_DEBUG = os.getenv('FLASK_DEBUG', 'True') == 'True'. Any deployment without an explicit FLASK_DEBUG=False env var runs the Werkzeug interactive Python debugger, enabling arbitrary code execution on any traceback URL.
Fix Summary	Change default from 'True' to 'False': FLASK_DEBUG = os.getenv('FLASK_DEBUG', 'False') == 'True'. Also add a startup guard in app.py that raises RuntimeError if SECRET_KEY matches the default dev value in non-debug mode. Update .env.example.
API / Data Source	N/A — configuration fix
Test Case	Given: No FLASK_DEBUG env var. When: App starts. Expect: app.debug==False, no debugger accessible. Currently broken: app.debug==True, debugger on all traceback URLs.
Validation Note	DEPLOY FIRST before all other fixes. Also verify no app.run(debug=True) in app.py overrides config.
PR Note	Highest-priority security fix. One character change.

P0-6 · OCO adjust_bracket() silently fails — key mismatch + wrong modify approach

Phase: Phase 5 / PR #6

File: monitor_service.py L851-861 | **Feasibility: NEEDS MODIFICATION** | **Effort: L**

Root Cause	monitor_service.py constructs OCO with keys 'qty', 'stop_price', 'limit_price' but tradier.py reads 'quantity', 'stop', 'price'. Additionally, Tradier OCO orders CANNOT be modified via PUT — must cancel and recreate. The current code uses the wrong keys AND the wrong API method.
Fix Summary	Rewrite adjust_bracket() as a cancel+recreate flow: (1) Cancel old OCO, (2) Recreate with correct indexed Tradier keys (quantity[N], stop[N], price[N]), (3) Update DB with new order ID inside a transaction. Include emergency fallback: if recreate fails after 3 retries, place a simple stop-market order and log CRITICAL alert.
API / Data Source	Tradier DELETE /accounts/{id}/orders/{id}, then POST /accounts/{id}/orders with class=oco, indexed keys: quantity[0], type[0], price[0] (limit leg), quantity[1], type[1]='stop_limit', stop[1], price[1] (stop leg).
Test Case	Given: Open trade with SL=$2.50, TP=$5.00. When: Adjust bracket to SL=$3.00, TP=$6.00. Expect: Old OCO cancelled, new OCO placed. Currently broken: KeyError silently caught, DB updated but Tradier never updated.
Validation Note	Must use cancel+recreate (not modify). Requires failure-recovery logic for naked-position window. See risk matrix.
PR Note	Highest-risk PR. Mandatory Tradier sandbox test before production.

P0-7 · OCO stop-loss uses market order type "stop" not "stop_limit"

Phase: Phase 5 / PR #6 (with P0-6)

File: tradier.py L415 | **Feasibility: CONFIRMED** | **Effort: S**

Root Cause	tradier.py L415 sets type[0]: "stop". A Tradier stop order becomes a market order when triggered. In illiquid options (wide spreads), a market sell can fill at the bid or worse — potentially 50-99% below the intended stop price.
Fix Summary	Change type[N]: "stop" to type[N]: "stop_limit" and add a price[N] limit price at 5% below the stop trigger (stop_price * 0.95). This ensures the order will not fill at catastrophic prices below the limit floor.
API / Data Source	Tradier POST /accounts/{id}/orders — type[N]: "stop_limit" with stop[N] (trigger) and price[N] (limit floor). Both fields confirmed working in sandbox.
Test Case	Given: OCO with SL stop=$3.50, option gaps to $2.00 overnight. Expect (after): Stop-limit activates at $3.50, limit floor is $3.33 — order sits if no fill above floor. Currently: Market order triggers, fills at $0.10 bid.
Validation Note	Verify leg index in actual code (leg 0 vs leg 1). Apply same change in P0-6 recreate logic since they share OCO construction.
PR Note	Configurable buffer (sl_buffer_pct) is a future enhancement.

P0-8 · lifecycle_sync() implemented but never scheduled

Phase: Phase 2 / PR #3

File: app.py (APScheduler registration) | **Feasibility: CONFIRMED** | **Effort: S**

Root Cause	app.py init_scheduler() registers 4 APScheduler jobs but never registers lifecycle_sync(). The method is fully implemented in monitor_service.py L903-990 with advisory lock support (lock ID 100003) but has no trigger.
Fix Summary	Add scheduler.add_job(func=monitor.lifecycle_sync, trigger=IntervalTrigger(seconds=60), id="lifecycle_sync", max_instances=1) before scheduler.start(). The max_instances=1 prevents overlapping executions.
API / Data Source	N/A — scheduler registration. Method already implemented.
Test Case	Given: Trade with status=PENDING and tradier_order_id that is filled at Tradier. When: 60s elapse. Expect: lifecycle_sync runs, trade transitions to OPEN. Currently broken: Trade stays PENDING forever.
Validation Note	On first activation, all accumulated PENDING trades are processed. Monitor logs for initial burst. lifecycle_sync and sync_tradier_orders operate on different state transitions so they do not conflict.
PR Note	Monitor first run in staging — stale PENDING trades from before fix will all process.

P0-9 · log vs logger NameError crashes trade placement

Phase: Phase 0 / PR #1

File: paper_routes.py L389 | **Feasibility: CONFIRMED** | **Effort: S**

Root Cause	paper_routes.py L33 defines logger = logging.getLogger(__name__). L389 uses log.warning(...). 'log' is undefined. When the context-capture try block at L378-387 catches any exception, NameError fires in the except clause, aborting the entire place_trade() function.
Fix Summary	Change log.warning to logger.warning on L389. Also grep all log. references in the file to find any others.
API / Data Source	N/A — typo fix
Test Case	Given: ORATS API timeout during context capture. When: User places trade. Expect: Warning logged, trade still succeeds. Currently broken: NameError aborts trade placement, user sees 500 error.
Validation Note	Verified: paper_routes.py L33 is logger, L389 is log. One-character fix.
PR Note	Deploy immediately — blocking trade placements whenever ORATS context capture fails.

P0-10 · _get_username() auth bypass — silent fallback to demo user

Phase: Phase 0 / PR #1

File: paper_routes.py L47 | **Feasibility: CONFIRMED** | **Effort: S**

Root Cause	paper_routes.py L47: return session.get('user', 'demo'). Unauthenticated requests (no session, expired session, direct API call) silently operate as the 'demo' user with full read/write access to all demo account data.
Fix Summary	Remove the 'demo' default. Replace with: user = session.get('user'); if not user: abort(401). Add from flask import abort if not already imported. Existing watchlist routes already follow this pattern.
API / Data Source	N/A — auth fix
Test Case	Given: No session cookie. When: GET /api/paper/trades. Expect: HTTP 401 {error: Authentication required}. Currently broken: HTTP 200 with all demo trades returned.
Validation Note	Test suite may need session mocking update since tests may rely on demo fallback.
PR Note	Security fix — deploy immediately with P0-5.

P0-11 · current_price reset to 0 after valid history price — aborts scan

Phase: Phase 1 / PR #2

File: hybrid_scanner_service.py L282 | **Feasibility: CONFIRMED** | **Effort: S**

Root Cause	L253 sets current_price from df[Close].iloc[-1] (valid). L282 resets current_price = 0, discarding the valid price. If ORATS real-time quote then fails, current_price remains 0 and the scan aborts at L294 even though a perfectly valid historical price was available.
Fix Summary	Preserve the history price as fallback before overwriting: history_price = current_price. Then attempt ORATS quote. If quote fails or returns None, fall back to history_price. Log which source was used. Only abort if both sources fail.
API / Data Source	Primary: ORATS live/strikes → stockPrice. Fallback: ORATS hist/dailies → clsPx (from DataFrame T-1 close).
Test Case	Given: AAPL with valid history (current_price=$242). When: ORATS quote times out. Expect: scan continues with $242. Currently broken: current_price=0, scan aborted.
Validation Note	History price is previous close — acceptable for LEAP scans, note staleness for 0DTE. Add NaN check on history_price.
PR Note	Simple isolated fix in God class.

P0-12 · Target Friday formula uses constant 3 (Thursday) instead of 4 (Friday)

Phase: Phase 1 / PR #2

File: hybrid_scanner_service.py L858

Feasibility: CONFIRMED

Effort: S

Root Cause	L858: target_friday = today + timedelta((3-today.weekday()) % 7). Python weekday() is Monday=0...Friday=4. Constant 3 targets Thursday. On every weekday: Mon→Thu(+3), Tue→Thu(+2), Wed→Thu(+1), Thu→Thu(today), Fri→Next Thu(-1+7=+6).
Fix Summary	Change constant 3 to 4: target_friday = today + timedelta((4 - today.weekday()) % 7). Note: L1025 already uses the correct formula (4 - today.weekday() + 7) % 7. Bug only at L858.
API / Data Source	N/A — date math fix
Test Case	Parametric: Mon(0)→+4, Tue(1)→+3, Wed(2)→+2, Thu(3)→+1, Fri(4)→+0 (today). All must produce a weekday==4 (Friday).
Validation Note	L858 bug affects log output and early-exit paths. L1025 (actual filtering) is already correct. Still fix L858 for consistency.
PR Note	Dangerous because wrong expiry date passes silently — system does not crash.

P0-13 · calculate_base_score() anchors at ~50 — missing context keys

Phase: Phase 4 / PR #5

File: hybrid_scanner_service.py L764 + reasoning_engine.py L246

Feasibility: CONFIRMED

Effort: S

Root Cause	reasoning_engine.py reads context['technicals']['score'] and context['sentiment']['score']. The scanner builds context without these keys — only 'trend', 'ma_signal', 'rsi', etc. Both defaults to 50. AI base_score = 50 for every ticker regardless of actual technicals or sentiment.
Fix Summary	Before the AI call, add two lines to populate the required keys: context['technicals']['score'] = technical_score (0-100 from calculate_technical_score()) and context['sentiment'] = {'score': sentiment_score} (0-100 from Finnhub pipeline). Apply in BOTH scan_ticker() and scan_weekly_options().
API / Data Source	Internal — technical_score from TechnicalIndicators.calculate_technical_score(), sentiment_score from Finnhub pipeline. Both already computed before the AI call.
Test Case	Given: NVDA with technical_score=78, sentiment_score=65. When: AI reasoning calculates base score. Expect: base_score ~73 (50 + (78-50)*0.6 + (65-50)*0.4). Currently broken: base_score ~50.
Validation Note	Both scores are already available at the point of AI call. Apply in both LEAP and WEEKLY scan paths.
PR Note	Score distribution will spread after fix. Thresholds may need recalibration.

P0-14 · Bookend snapshots record underlying stock price, not option price

Phase: Phase 6 / PR #7

File: monitor_service.py L607 | **Feasibility: CONFIRMED** | **Effort: M**

Root Cause	monitor_service.py L607 calls orats.get_quote(ticker) which returns the underlying STOCK price. The snapshot stores this (~$270 for AAPL) in mark_price instead of the option premium (~$3.50). All MFE/MAE calculations built on these snapshots are corrupted.
Fix Summary	For each open trade during bookend, call orats.get_option_quote(ticker, strike, expiry, option_type) to get the option-level bid/ask/mark. Store the option mark in mark_price and the stock price in the separate underlying field. Use trade.current_price as fallback if option quote fails.
API / Data Source	ORATS live/strikes → filtered by expirDate + strike → callBidPrice/callAskPrice or putBidPrice/putAskPrice (via existing get_option_quote() method).
Test Case	Given: AAPL 230C at entry_price=$8.50, AAPL stock at $242. When: Bookend at 4:05 PM. Expect: snapshot.mark_price ~$8.75 (option mid), snapshot.underlying=$242. Currently broken: mark_price=$242 (stock), unrealized_pnl=$23,350 (wrong).
Validation Note	Entire historical bookend dataset is corrupted. Mark old snapshots as legacy. Remediate open trades using intraday snapshots if available.
PR Note	Data migration required to tag legacy snapshots. MFE/MAE for closed trades cannot be recovered.

P0-15 · W_PROF weight mismatch — code is 20%, spec says 5%

Phase: Phase 1 / PR #2

File: options_analyzer.py L426 | **Feasibility: CONFIRMED** | **Effort: S**

Root Cause	L426: W_PROF = 0.20 for LEAP scoring. Architecture spec says 5%. Combined weights sum to 1.15 (already over 100%). High-premium OTM options (300% profit potential) dominate rankings over quality ITM setups.
Fix Summary	Change W_PROF = 0.20 to W_PROF = 0.05. Redistribute the 15% to W_SENT (+5% → 0.25) and add W_FUND = 0.10 (placeholder for fundamental quality score). Add an assertion: assert abs(sum(weights) - 1.0) < 0.001. Verify weekly/0DTE weight schemes do not have the same bug.
API / Data Source	N/A — weight constant fix. Fundamental score already available from FMP get_rating() at L303-312.
Test Case	Option A: 60-delta, 50% profit, 75 tech, 60 sent. Option B: 25-delta, 300% profit, 75 tech, 60 sent, low liquidity. Expect: A scores higher (after fix). Currently: B scores higher (300% * 0.20 dominates).
Validation Note	Weights were already summing to 1.15 — bug compounded by the overage.
PR Note	"Top 3" ticker rankings will change after fix — expected, correct behavior.

P0-16 · Max positions not enforced server-side

Phase: Phase 2 / PR #3

File: paper_routes.py (place_trade route) | **Feasibility: CONFIRMED** | **Effort: S**

Root Cause	max_positions is stored in UserSettings but place_trade() never checks the count of open + pending positions before creating a new one. The frontend may warn but any direct API call bypasses it.
Fix Summary	Insert a SQLAlchemy count query for OPEN+PENDING positions before trade creation. Use func.count(PaperTrade.id) with filter on username and status IN (PENDING, OPEN). Return HTTP 403 if count >= max_positions.
API / Data Source	Internal PostgreSQL — count of paper_trades with non-terminal status.
Test Case	Given: User with max_positions=5, currently 5 OPEN. When: place_trade() for 6th. Expect: HTTP 403 "Maximum positions reached (5)". Currently broken: 6th trade placed successfully.
Validation Note	Include PENDING in count (committed capital). Race condition possible but acceptable for paper trading.
PR Note	Deploy with P0-4 (daily loss limit) for complete risk enforcement.

P0-17 · LEAP scan bull-only — all put opportunities suppressed

Phase: Phase 7 / PR #8

File: hybrid_scanner_service.py L260-262 | **Feasibility: CONFIRMED** | **Effort: M**

Root Cause	L260-262 applies a universal SMA-200 filter that returns None for any ticker below its 200-day SMA, before option direction is determined. Long puts on downtrending stocks — the most natural bearish LEAP trade — are discarded unconditionally.
Fix Summary	Accept a scan_direction parameter (call/put/both). For downtrending tickers in "both" mode, switch to put-only. For call-only requests, block downtrending tickers. Never unconditionally return None based solely on the SMA filter. Propagate scan_direction to parse_options_chain() to filter the chain.
API / Data Source	N/A — logic fix. SMA-200 already calculated from ORATS hist/dailies.
Test Case	Given: INTC at $22, SMA-200=$28 (downtrend). When: scan_ticker("INTC", scan_direction="both"). Expect: Put LEAP opportunities (e.g., INTC $25P, 180 DTE) returned. Currently broken: return None — ticker discarded.
Validation Note	Requires P0-1 (return-in-loop fix) to be effective. Apply direction param approach as MVP. Doubles scan scope for "both" direction — monitor API usage.
PR Note	Highest merge-conflict risk in God class. Apply last after other scanner fixes.

Section 5 · P1 and Cross-Cutting Fixes

All P1 and cross-cutting (XC) items, in table format for quick reference.

ID	Description	Data Source	Phase / PR	Effort
P1-A1	Profit calc intrinsic-only — ignores time value. Add delta-gamma approximation: P ≈ delta*dS + 0.5*gamma*dS².	ORATS live/strikes → delta, gamma per strike	PR #5 or later	M
P1-A2	IVP threshold static — not regime-adjusted. IVP scoring should invert in high-VIX regime.	ORATS hist/cores → ivPctile1y; Tradier VIX (via XC-1)	PR #8 (after XC-1)	M
P1-A3	No ex-dividend date check for early exercise risk on ITM calls.	ORATS hist/cores → divDate; Finnhub /stock/dividend	PR #5	M
P1-A4	Open Interest threshold too low for LEAP liquidity. LEAP min OI should be 100, not 10.	N/A (threshold constant)	PR #2 or #5	S
P1-A5	No per-trade max loss check. User could risk 50% of account in one trade.	Internal: UserSettings.account_size, entry_price	PR #3	S
P1-A6	No OCO confirmation loop. 200 OK can contain rejected order in Tradier response body.	Tradier GET /accounts/{id}/orders/{id} → status	PR #6	S
P1-A7	No retry logic on external API calls. Transient failures abort entire scan.	N/A (resilience improvement)	PR #10	M
P1-A9	No sector concentration limit. 10/10 tech positions possible.	Internal: ticker_cache sector field	Sprint 2	M
P1-A10	sma_5 lookup crashes on IPO/ETF tickers (<200 bars of history). Defensive None check needed.	N/A (defensive coding)	PR #5	S
P1-A11	AI score explanation not linked to score drivers. score_breakdown missing from response.	Internal: reasoning_engine.py score components	Sprint 2	S
P1-A12	Fernet ENCRYPTION_KEY not validated at startup. Wrong format crashes on first broker decrypt.	N/A (startup validation)	PR #9	S
P1-A13	No CSRF protection on state-changing routes (trade placement, bracket adjustment).	N/A (flask-wtf or X-Requested-With header)	Sprint 2	M
P1-QW5	No timeout on ORATS API calls. Hung calls block Flask worker threads indefinitely.	N/A (requests timeout param)	PR #4	S
P1-QW6	Duplicate analyze_volume() call in technical_indicators.py L322-323.	N/A (dead code)	PR #9	S
P1-QW7	Dead scoring formula block in options_analyzer.py L439-445 (uses W_SKEW for sentiment).	N/A (dead code)	PR #9	S
P1-QW8	min_profit_potential double assignment in options_analyzer.py __init__.	N/A (dead code)	PR #9	S
P1-QW9	ma_signal key mismatch: 'pullback bullish' (space) vs 'pullback_bullish' (underscore).	N/A (string constant)	PR #2	S
P1-QW10	Put break-even formula wrong: strike + premium should be strike - premium for puts.	N/A (formula fix)	PR #2	S
P1-QW11	print() calls scattered in production paths — should be logger.debug().	N/A (logging)	PR #9	S
P1-QW12	Unreachable second return in parse_options_chain L70 (dead code).	N/A (dead code)	PR #9	S
P1-QW13	get_db() yield pattern — verify session closes correctly after caller.	N/A (generator pattern)	PR #9	S

XC-1	No VIX regime filter in scan pipeline. Same thresholds used at VIX=12 and VIX=40.	Tradier GET /markets/quotes ?symbols=VIX → last (18.63 live)	PR #8	M
XC-2	HybridScannerService God Class (1823 lines). Extract DataFetcher, ScoreCalculator, OptionFilter, CalendarUtils.	N/A (refactor)	Sprint 2	L
XC-3	Zero test coverage for scanner, ORATS client, and options analysis.	N/A (test infrastructure)	Sprint 2	L
XC-4	Missing input validation on all API routes. Ticker/quantity/price not validated.	N/A (validation utility)	Sprint 2	S
XC-5	calculate_hv_rank mutates shared DataFrame. Add df = df.copy() at top of method.	N/A (defensive coding)	PR #9	S
XC-6	AI reasoning disconnected from quantitative scores. Covered by P0-13 plus incremental enrichment.	Internal: skew, VIX, earnings context	PR #5 (initial)	M
XC-7	No rate limiting on login or API routes. Add flask-limiter (5 req/min on /login).	N/A (flask-limiter)	PR #9	S

Section 6 · Quick Wins Checklist

15 quick wins (S-effort, isolated, high-confidence). Total estimated time: ~42 minutes.

QW#	Fix ID	File	Line(s)	Change Description	Time
QW-1	P0-9	paper_routes.py	L389	log.warning → logger.warning (typo)	1 min
QW-2	P0-12	hybrid_scanner_service.py	L858	Friday formula: constant 3 → 4	1 min
QW-3	P0-5	config.py	L56	FLASK_DEBUG default 'True' → 'False'	1 min
QW-4	P0-8	app.py	L558-559	Register lifecycle_sync APScheduler job (5 lines)	5 min
QW-5	P1-A7 partial	orats.py	L82, L109, L161, L227	Add timeout=(5, 30) to all requests.get() calls	5 min
QW-6	P1	technical_indicators.py	L322-323	Delete duplicate analyze_volume() call (1 line deletion)	1 min
QW-7	P1	options_analyzer.py	L439-445	Delete dead scoring formula block (uses W_SKEW for sentiment)	2 min
QW-8	P1	options_analyzer.py	L8-10	Remove min_profit_potential double assignment (1 line)	1 min
QW-9	P1	hybrid_scanner_service.py	L1129-1130	Fix ma_signal key: 'pullback bullish' (space) → 'pullback_bullish' (underscore)	5 min
QW-10	P1	options_analyzer.py	L221	Put break-even: strike + premium → strike - premium	5 min
QW-11	XC	Multiple files	Various	Replace print() with logger.debug() in production paths	10 min
QW-12	XC	hybrid_scanner_service.py	L70	Remove unreachable second return in parse_options_chain	1 min
QW-13	P1	models.py	L86-92	Verify get_db() uses yield (not return) for session lifecycle	5 min
QW-14	P0-10	paper_routes.py	L47	_get_username(): remove 'demo' fallback, add abort(401)	2 min
QW-15	P0-15	options_analyzer.py	L426	W_PROF = 0.20 → 0.05 + weight sum assertion	5 min
TOTAL				**15 fixes across 6 files**	**~42 min**

Section 7 · New API Methods Required

Three new methods must be added to existing API client classes before Phase 4 scanner fixes can be applied. These are in PR #4 (Phase 3) and carry zero risk since they are additive (new code only).

Method 1: OratsAPI.get_summary(ticker)

File: Options/backend/api/orats.py | **Required by:** P0-2 (skew) | **PR:** #4

Calls ORATS **live/summaries** endpoint. Returns real-time IV term structure and skew metrics. Should cache response for 5 minutes since skew changes slowly intraday.

Field	Type	Description	Used For
rSlp30	float	30-day risk-neutral slope (~-10 to +10). Negative = put skew (bearish demand).	Skew score normalization
skewing	float	Skew persistence indicator	Skew quality signal
contango	float	Term structure shape (positive = normal, negative = inverted)	Regime context
dlt25Iv30d	float	25-delta put IV, 30-day. Higher = expensive downside protection.	Put/call skew comparison
dlt75Iv30d	float	25-delta call IV, 30-day (75-delta calls = 25-delta puts by parity)	Put/call skew comparison
atmiv30d	float	At-the-money IV, 30-day	IV baseline for term structure

Normalization formula: **skew_score = max(0, min(100, 50 + rSlp30 * 5))** (rSlp30 of ±10 maps to 0 or 100; ±0 = neutral 50)

Method 2: OratsAPI.get_hist_cores(ticker, trade_date=None)

File: Options/backend/api/orats.py | **Required by:** P0-3 (earnings), P1-A2 (IVP), P1-A3 (ex-div) | **PR:** #4

Calls ORATS **hist/cores** endpoint. Returns T-1 (previous trading day) fundamental option analytics. **Critical:** Full response is 4,817 rows (~2MB). Must cache per-ticker per-day. Use tradeDate parameter to fetch only the latest row if ORATS supports it (verify before implementation).

Field	Type	Description	Used For
ivPctile1y	float	IV percentile vs. last 1 year (0-100). T-1 delay.	P1-A2 regime-adjusted IVP scoring
ivPctile1m	float	IV percentile vs. last 1 month (0-100). T-1 delay.	Short-term IV context
daysToNextErn	int	Days to next earnings announcement. T-1 delay.	P0-3 fallback if Finnhub down
impliedEarningsMove	float	Expected stock move at earnings (e.g., 0.08 = ±8%). T-1.	P0-3 earnings move display
divDate	str	Next ex-dividend date (YYYY-MM-DD format). T-1.	P1-A3 early exercise risk check
contango	float	IV term structure shape (also in live/summaries)	VIX proxy if Tradier down
slope	float	IV skew slope (historical). Also in live/summaries.	Historical skew context
wksNextErn	float	Weeks to next earnings. T-1.	P0-3 weeks-based earnings check

Caching strategy: Store latest row in-memory dict keyed by ticker+date. TTL = current trading day (T-1 data does not change intraday). Memory cost: ~340 fields * 8 bytes * 50 tickers = ~136KB (negligible).

Method 3: FinnhubAPI.get_earnings_calendar(symbol, _from, to)

File: Options/backend/api/finnhub.py | **Required by:** P0-3 (earnings) | **PR:** #4

Calls Finnhub **/calendar/earnings** endpoint. Real-time earnings dates. Rate limit consideration: Finnhub free tier is 60 calls/minute. Recommended: single batch query at scan start for all tickers (no symbol filter), then check against returned list locally for each ticker.

Field	Type	Description	Used For
earningsCalendar	array	Array of earnings entries for the date range.	Outer container

symbol	str	Ticker symbol. Must filter by exact match (not partial).	Ticker matching
date	str	Earnings date (YYYY-MM-DD). Real-time.	has_earnings_risk date
hour	str	'bmo' (before market open), 'amc' (after close), 'dmh' (during market hours)	Risk timing display
epsEstimate	float	Consensus EPS estimate. Can be null.	Context display
epsActual	float	Actual EPS (null if not yet reported)	Post-earnings verification

Verified working: GET https://finnhub.io/api/v1/calendar/earnings?from=2026-02-26&to;=2026-03-05&symbol;=NVDA returns earningsCalendar array with date, hour, epsEstimate fields confirmed.

Section 8 · Risk Assessment

Risk 1: OCO Cancel+Recreate — Naked Position Window (P0-6)

The most dangerous fix in the plan. Between canceling the old OCO and placing the new one, the position is unprotected. Network errors or Tradier rate limits during this window leave a naked long option with no stop-loss or take-profit.

Scenario	Cancel	Recreate	Position State	Resolution
Happy path	Success	Success	Protected (new bracket)	Normal — update DB with new order ID
Cancel fails	Fails	Not attempted	Old bracket still active	Log warning, retry next monitoring cycle
Cancel OK, Recreate fails	Success	**FAILS**	**NAKED POSITION**	**3 retries + emergency stop-market + CRITICAL alert**
Cancel OK, Recreate timeout	Success	Unknown	Unknown	Check order status; place emergency stop if uncertain
Both timeout	Unknown	Not attempted	Unknown	Check order status next lifecycle_sync cycle

Required Mitigations:

- Wrap cancel + recreate + DB update in a single transaction with rollback on failure
- Retry recreate up to 3 times with exponential backoff (0.5s, 1s, 2s)
- If all retries fail: place a simple stop-market order as emergency fallback
- Log CRITICAL alert with trade_id, position details, and failure reason
- Monitor: add metric for naked_position_events — alert if > 0
- All values (symbol, quantity, side) come from the database, never from user input

Risk 2: ORATS API Rate Limits for 50-Ticker Scans

The proposed fixes increase ORATS calls from 3 to 5 per ticker. For a 50-ticker scan, total calls go from 150 to 250. The critical new calls are get_summary() (skew) and get_hist_cores() (IV percentile, earnings). Both can and must be cached.

API Call	Current (per ticker)	After Fixes	50-Ticker Total	Mitigation
get_history()	1	1	50	No change
get_quote()	1	1	50	No change
get_option_chain()	1	1	50	No change
get_summary() (NEW)	0	1 — P0-2 skew	50	Cache 5 min — only 50 on first scan, 0 on subsequent scans within TTL
get_hist_cores() (NEW)	0	1 — IVP+earnings	50	Cache per-ticker per-day — only 50 on first daily scan
TOTAL	**3/ticker**	**5/ticker**	**250 max**	**With caching: first scan 250, subsequent scans 150 (same as today)**

hist/cores payload note: Full hist/cores response for AAPL is ~4,817 rows (~2MB). Verify if ORATS supports tradeDate parameter to fetch only the latest row. If yes, use it. If not, fetch full history and take data[-1] (most recent). Cache the result per-ticker per-day.

Risk 3: God Class Merge Conflict Strategy

hybrid_scanner_service.py (1823 lines) is touched by 9 of 17 P0 fixes. Concurrent development on this file will produce merge conflicts.

Fix	Lines Changed in God Class	Merge Conflict Risk	Sequencing Rule

P0-11 (price fallback)	L282 — 5 lines	Low	Apply in PR #2 (Day 1 PM)
P0-12 (Friday formula)	L858 — 1 line	None	Apply in PR #2 (Day 1 PM)
P0-2 (skew)	L423-431 — 15 lines	Low (isolated block)	Apply in PR #5 (Day 3)
P0-3 (earnings)	L1030-1032 — 12 lines	Low (isolated block)	Apply in PR #5 (Day 3)
P0-13 (AI context)	L764 — 2 lines	Low (additive)	Apply in PR #5 (Day 3)
P1-A10 (sma_5 crash)	L1169 — 2 lines	None	Apply in PR #5 (Day 3)
P0-17 (put scan direction)	L260-262 — 15 lines	**Medium (control flow)**	Apply last, in PR #8 (Day 5 PM)
XC-1 (VIX regime)	New method + 30 lines	**Medium**	Apply in PR #8 with P0-17

Rule: Never have two developers working on different PRs that touch the God class simultaneously. Sequence: PR #2 merges before PR #5 starts. PR #5 merges before PR #8 starts. God class decomposition (XC-2) is deferred to Sprint 2 — decompose after all P0 fixes are merged.

Risk 4: hist/cores Caching Implementation

hist/cores returns T-1 data that does not change during the trading day. This makes it safe and efficient to cache.

- Cache structure: Python dict keyed by (ticker, trade_date) → {field: value, ...}
- TTL: current trading day string (e.g., '2026-02-26'). Invalidate at midnight or market open.
- Memory cost per ticker: ~340 fields × 8 bytes ≈ 2.7KB. For 50 tickers: ~135KB (negligible).
- Cache misses trigger a full hist/cores fetch (~2MB response, ~1-3s). First scan of the day is slower.
- If ORATS supports tradeDate parameter: pass yesterday's date to get only 1 row instead of 4817 rows.
- Verification needed: test GET /hist/cores?ticker=AAPL&tradeDate;=2026-02-25 before assuming tradeDate works.
- Thread safety: use threading.Lock or rely on Python's GIL if dict operations are atomic (they are).

Document generated February 26, 2026. Based on: 16-persona expert brainstorm (Groups 2A+2B+2C+2D), live API testing of ORATS, Finnhub, and Tradier sandbox, and direct code review of the /Options codebase. All API fields verified against actual responses. All code references verified against source files in /Options/backend/.