## Trading Bot v1.0.2

This document describes how to set up, run, and tune the EMA+advisors candle bot for Coinbase Advanced Trade.

Overview

• Exchange: Coinbase Advanced Trade (REST + WebSocket).

• Strategy: EMA crossover (short vs long) on fixed-interval candles, with a small dead-band to reduce churn.

• Advisors (optional): RSI and MACD act as one-sided vetoes — EMA is the “captain,” advisors only block clearly bad entries.

• Order style: Prefers post-only maker limits with per-asset offsets; will fall back to market orders if configured.

• Risk: Daily BUY spend cap, per-product cooldown, optional hard stop in basis points, dry-run mode for simulation.

• State: Lightweight JSON/CSV in a .state directory (overridable).

**Repository Layout**

main.py — process entrypoint (logging, env, lifecycle).

bot/tradebot.py — WebSocket loop, candle aggregation, indicators, order placement, fills/KPIs, persistence.

bot/config.py — defaults for products, candles, EMA, advisors, risk and maker behavior.

bot/strategy.py — advisor thresholds & veto logic.

bot/indicators.py — RSI, EMA, MACD implementations.

bot/orders.py — price/size rounding and maker limit computation.

bot/constants.py — paths and filenames for state.

bot/persistence.py — JSON I/O, daily spend tracking, last-trade cooldown, portfolio store, fill de-dup.

bot/utils.py — thin re-exports for constants and persistence helpers.

**Requirements**

• Python 3.10+ recommended.

• Packages: coinbase Advanced Trade SDK (RESTClient/WSClient), python-dotenv, and standard library modules.

• Network: outbound HTTPS/WebSocket access to Coinbase.

**Environment & Secrets**

Create an .env file (default name: APIkeys.env) with:

```
COINBASE_API_KEY=...
COINBASE_API_SECRET=...
PORTFOLIO_ID=...          # optional
```

You can set ENV_PATH to point at a different dotenv file at launch.

Optional: BOT_STATE_DIR to override the default .state location.

**Quick Start**

1) Install dependencies.

2) Put API keys into APIkeys.env (see above).

3) Adjust bot/config.py if needed (dry_run, products, thresholds, etc.).

4) Run:  python main.py

- Ctrl+C (SIGINT) or SIGTERM will shut down cleanly and log session P&L.

**How It Trades**

Candles

• Interval: 5m by default. You can subscribe to WS candles or locally aggregate from tickers.

• Warmup: waits for min_candles before evaluating signals; confirm_candles must align with the crossover before entry.

**EMA Crossover**

• Default: short_ema=40, long_ema=120 (tuned for 5m). A dead-band (ema_deadband_bps) reduces flip-flops.

**Advisors (RSI/MACD)**

• One-sided RSI veto: BUYs allowed only if RSI ≤ rsi_buy_max; SELLs allowed only if RSI ≥ rsi_sell_min.

• MACD veto: uses normalized histogram in basis points; BUY requires ≥ macd_buy_min, SELL requires ≤ macd_sell_max.

**Orders**

• By default, the bot prefers post-only maker limit orders. Limit price is computed off best bid/ask with a per-asset offset (bps).

• Price and size rounding uses the product's increments from the REST product metadata.

• SELLs are clamped to the position size; optional hard_stop_bps can force a market exit below cost basis.

**Risk Controls**

• dry_run: if True, simulates P&L and spend without touching the exchange.

• usd_per_order & daily_spend_cap_usd: BUYs stop after the daily cap; SELLs continue.

• per_product_cooldown_s: suppress re-entries too quickly after a trade.

**Key Defaults (bot/config.py)**

Products (sample): ETH, SOL, LINK, XRP, DOGE, ADA, AVAX, DOT, ARB, FIL, NEAR, ATOM, ALGO, XLM, HBAR, CRO, SUI, LTC.

Candles: candle_interval='5m', min_candles=120, confirm_candles=3, use_backfill=True, warmup_candles=200.

EMA: short_ema=40, long_ema=120.

Advisors: enable_advisors=True, rsi_period=14, rsi_buy_max=60, rsi_sell_min=40, macd 12/26/9, macd_buy_min=+3 bps, macd_sell_max=−3 bps.

Ops/Risk: dry_run=True, usd_per_order=20, daily_spend_cap_usd=160, per_product_cooldown_s=900, hard_stop_bps=120.

Maker: prefer_maker=True (sells can be configured separately), maker_offset_bps default 5.0 + per-product overrides.

Repricing: reprice_each_candle=True, reprice_if_unfilled_candles=1, max_reprices_per_signal=2, reprice_jitter_ms≈1500.

TTF Targets (5m): asset-specific candle targets (e.g., Tier A ≤1, Tier B ≤2, Tier C ≤3).

Misc: lookback_hours=48, processed_fills_max=10000, ema_deadband_bps=8.0, log_level=INFO.

**State, Logs & KPIs**

Default state dir: .state (overridable via BOT_STATE_DIR).

Files:

• daily_spend.json — BUY spend per UTC day (used to enforce daily cap).

• last_trades.json — timestamps for per-product cooldowns.

• portfolio.json — running positions, cost basis, realized P&L.

• processed_fills.json — de-duplication of fills to prevent double-count.

• trade_log.txt — human-readable trade lines + session footers (P&L, runtime).

• trades.csv — per-fill ledger with size, price, fee, liquidity flag, slippage (abs/bps), intent price, hold time, etc.

**Lifecycle & Reconciliation**

• On startup, indicators can be backfilled from REST candles to warm them up.

• During runtime, immediate fills are fetched and portfolio state is updated; a CSV line is appended per fill.

• On shutdown (or daily cap reached), the bot logs session P&L and runtime duration.

• reconcile_recent_fills() can backfill fills over a lookback window (default 48h) to fix any gaps.

**Tuning Tips**

• If fills lag: raise maker_offset_bps for that asset and/or allow repricing every candle.

• If you see churn: widen ema_deadband_bps and require more confirm_candles.

• If you want more selectivity: lower rsi_buy_max, raise rsi_sell_min, and tighten MACD thresholds in bps.

• Adjust usd_per_order and daily_spend_cap_usd to fit risk budget; keep dry_run=True while testing.

**Safety & Disclaimers**

This bot is for educational purposes only. Live trading involves risk, including possible loss of principal. Use dry_run mode first, review your keys/permissions, and operate only within your risk tolerance.