**Tradingbot – Workflow & Dokumentation (Steg 1–4)**

**Vad vi vill uppnå**

Bygga ett **stabilt, stegvis workflow** som:

* skannar hela Capital-universet och tar fram **handelsbara** instrument,
* rankar **topp-10** på pris-/volymbaserat momentum,
* gör en **edge-analys** (trend + momentum + VWAP + range) och producerar **ENTRY/SL/TP**,
* lägger **pending BUY STOP-orders** med SL/TP där varje trade riskerar **100 USD eller 1000 SEK** — oberoende av hävstång.

**Filer & kommandon**

**Script (körordning)**

1. capital\_scan\_tradeable.py
2. python capital\_scan\_tradeable.py

➜ Output: scan\_tradeable\_current.csv

1. rank\_top10\_momentum.py
2. python rank\_top10\_momentum.py

➜ Output: top10\_momentum\_current.csv

1. analyze\_edge\_score.py
2. python analyze\_edge\_score.py

➜ Output: orders.csv (endast DEC=BUY)

1. place\_pending\_orders.py
2. python place\_pending\_orders.py

➜ Lägger pending BUY STOP med SL/TP, loggar till open\_trades.csv

**Hur delarna hänger ihop (översikt)**

[Capital API: markets]

│

▼

(1) capital\_scan\_tradeable.py

│ → filter: TRADEABLE + låg spread

└──► scan\_tradeable\_current.csv

│

▼

(2) rank\_top10\_momentum.py

│ → pris/volym-signal: M1/M5/M15, RangePos, VWAP, RSI

└──► top10\_momentum\_current.csv

│

▼

(3) analyze\_edge\_score.py

│ → EDGE-gate (DEC=BUY/WATCH/SKIP), ENTRY/SL/TP

└──► orders.csv (endast BUY)

│

▼

(4) place\_pending\_orders.py

→ Pending BUY STOP + SL/TP

→ open\_trades.csv (logg)

**Block 1 – Universe Scan (sammanfattning)**

* Loggar in mot Capital.com (DEMO), hämtar alla marknader.
* Filtrerar på **TRADEABLE**, rimlig spread (≤ ~0,15 %), relevanta typer (SHARES/INDICES/ETF/COMMODITIES).
* Berikar med bl.a. spread%, namn och (om möjligt) **leverage**.
* **Output:** scan\_tradeable\_current.csv – vår startlista.

**Block 2 – Momentum Top-10**

* Läser scan-filen och hämtar intradagsdata.
* Räknar **momentum** (M1/M5/M15), **range-position**, **avstånd till VWAP**, **RSI-14**.
* Viktar ihop till **score** och väljer **Top-10**.
* **Output:** top10\_momentum\_current.csv (inkl. leverage).
* **Kommande feature:** nyheter/insiders/sentiment/volatilitet för ännu bättre ranking.

**Block 3 – Edge-analys & Orderplan**

* Läser Top-10, hämtar prisdata för **MA20/MA200**.
* Räknar **ENTRY** (buffert över pris), **SL/TP** (t.ex. ±3 %).
* **DEC (Decision)-logik:**
  + **BUY** om: **Score-tröskel** (mål 70+) **och** pris > **MA200** **och** **M5% > 0** **och** **VWAP ≥ 0**.
  + WATCH/ SKIP annars (filtreras bort).
* **Output:** orders.csv med **endast BUY** och kolumner: **EPIC, ENTRY, SL, TP, LEVERAGE**.
* **Viktigt:** Steg 3 är **analys-only** (lägger **inga** ordrar).

**Block 4 – Ordermotor (Pending BUY STOP)**

**Vad skriptet ska göra**

* Läsa orders.csv.
* För varje rad: lägga **pending BUY STOP** på **ENTRY** med **SL** och **TP** (standard ±3 % från ENTRY).
* **Risk per trade:** exakt **100 USD** eller **1000 SEK**, **oberoende av hävstång**.  
  ➜ Storlek beräknas via **risk till SL**, inte positionsvärde.
* Robusthet:
  + **401 → relogin**, **429/5xx → backoff/retry**.
  + **Idempotens/dedupe** via clientOrderId (t.ex. EPIC|ENTRY|SL), kolla även mot befintliga working orders.
  + Respektera **minStopDistance**, **tick size/price decimals** och **minsta kontraktssteg** innan ordern skickas.
* Loggning: skriv/append till open\_trades.csv (tid, EPIC, side, ENTRY/SL/TP, size, leverage/margin, clientOrderId, status, dealId).

**Viktiga justeringar (identifierade gap)**

* **Storleksberäkning (prio):**  
  *Nuvarande kod använder positionsvärde*. Ska ändras till:  
  size = risk\_ccy / ((ENTRY - SL) \* value\_per\_unit)  
  (Leverage påverkar **margin**, inte PnL per punkt.)
* **Valuta för risk:**  
  Stöd **100 USD** eller **1000 SEK** (antingen kontovaluta eller enkel FX-faktor).
* **Marginvalidering:**  
  Kontrollera att required\_margin ≈ entry \* size \* margin\_fraction är rimlig mot equity / definierad max-andel.
* **Marknadsregler:**  
  Använd marketinfo för **minStopDistance**, **minSize**/**sizeStep**, **tickSize/decimals**, **forceOpen**, ev. **timeInForce**.
* **Stabilitet & dubbletter:**  
  Implementera clientOrderId, lokalt och mot servern, och skriv försök/retry-logg.

**Status idag (Steg 4)**

* Skriptet skickar pending-orders, men:
  + **Risk ≠ 100 USD / 1000 SEK** ännu (fel storleksformel).
  + Saknar robust dedupe/relogin/backoff.
  + Saknar rigorös kontroll av **minStopDistance** och **size-kvantisering**.
* Detta är morgondagens fixar — nu har vi en tydlig specifikation att koda mot, steg för steg.