**Binance USDⓈ-M 선물(테스트넷↔실계좌 토글) 실매매 프로그램 — 상세 설계서(v1)**

**목표**

* **테스트넷 우선** 설계로 즉시 시뮬·E2E 점검 → **토큰만 교체하면 실계좌**로 전환.
* **BTCUSDT, ETHUSDT** 2심볼 지원(확장 용이).
* **Discord 4채널**: #분석, #컨트롤패널, #대시보드, #알림.
* **ENV 최소화**: 필수만 .env로, 나머지는 **컨트롤패널 UI(슬라이더/버튼)** 로 실시간 변경 & **영속 저장**.
* **분석 엔진은 “현재”가 아니라 “미래 k-바 수익률” 예측**에 초점. 단일 지표가 아닌 **지표 상호작용과 레짐(상태) 기반** 점수화.
* **순차 매수/매도(스카웃→스케일)**, **리스크 타게팅(ATR 유닛)**, **중복진입/리부팅 안정성** 내장.
* **알림은 최소·핵심**(진입/청산/리스크/목표수익).

**전체 아키텍처(모듈)**

* **Orchestrator**: 이벤트 루프, 스케줄, 장애 복구.
* **Exchange Connector**: Binance USDⓈ-M (테스트넷/실계좌 토글), REST+WS.
* **Market Data Service**: 멀티TF 캔들/마크프라이스/계정/포지션 스트림.
* **Feature & Indicator Pipeline**: 지표/마켓마이크로 구조/상관·베타 등 **공통 스냅샷** 생성.
* **Regime + Forecast Engine**: 레짐 분류(트렌드/변동성) → **k-바 앞** 방향/강도 확률 예측 → **종합 점수화**.
* **Risk Engine**: ATR 기반 사이징, 데일리 손실컷, 상관 캡, 레버리지 관리.
* **Execution(게이트키퍼 포함)**: 라우팅 검사 → 게이트 → 주문 전송/체결 추적 → **순차 스케일/부분청산/트레일링**.
* **Position & PnL Ledger**: 포지션 하이드레이션, 수수료·실현/미실현 PnL, 일일 집계.
* **Discord Agent**: 슬래시커맨드/버튼/슬라이더, 대시보드 실시간 편집, 컨트롤 패널.
* **Persistence**: sqlite(trades, positions, pnl, config, patches, events), config.json, patch.txt.
* **Env Loader**: token.env 분리, load\_env\_chain().

**공통 규칙 준수**: 전역 **현재가 스냅샷 공유**, **Mark Price 기준** 트리거/SL/TP, **idem\_hit(동캔들 중복 진입 차단)**, **부팅 시 포지션 하이드레이션**, 보호체크는 (hit, reason) 계약.

**데이터·지표·예측 설계**

**타임프레임 & 스트림**

* **Symbols**: BTCUSDT, ETHUSDT
* **TF**: 1m(실행), 5m/15m(시그널), 1h/4h(레짐)
* **스트림**: markPrice, aggTrades(선택), kline, account/position

**피처(예시)**

* **가격·변동성**: r₁, r₅, r₁₅, ATRₙ, RV(실현변동성), 파라볼릭SAR, Donchian breakout
* **추세**: EMA(20/50/200) 기울기/스프레드, MACD comp, ADX
* **평균회귀**: z-score(가격–EMA), Bollinger band width/퍼센타일
* **시장구조**: 고저폭/진폭, realized skew/kurtosis, range location
* **파생·선물 전용**: 펀딩률 변화, OI 변화(가능 시), 주문서 불균형, 최근 강제청산 집계
* **크로스에셋**: ETH↔BTC **상관(ρ)**, **리드-래그**, 베타(ETH→BTC, BTC→ETH)
* **메타**: 최근 모델 성능(rolling hit-rate), 슬리피지/수수료 추정

**레짐 + 포캐스트(핵심)**

1. **Regime Classifier**
   * trend ∈ {up, down, flat}, vol ∈ {low, high} → 3~4 레짐.
   * 휴리스틱(EMA 스프레드+RV 백분위)로 경량 구현.
2. **Directional Forecast (k-bar ahead)**
   * Horizon k: 기본 5~15분(슬라이더로 조정).
   * **앙상블 3축**
     + *Trend* 모델: EMA slope, Donchian, ADX로 P(up\_k)
     + *Mean-reversion* 모델: z-score, band touch 리버설로 P(down\_k)
     + *Cross-asset* 모델: 선행심볼 변화(예: ETH가 선행)로 Δp\_target\_k
   * **온라인 가중치 업데이트**: 최근 N시그널 성과에 따라 모델 가중치 **W** 적응(가중치 소실 방지 ε 적용).
   * **최종 점수** S = W·[trend, mr, cross] → 레짐별 캘리브레이션 테이블로 진입/증액/축소 의사결정.
3. **시나리오 스코어링 → 실행**
   * S\_enter ≥ θ\_enter(Regime) → **스카웃**(소량).
   * 유리한 **MAE/MFE** 진전 시 단계적 **스케일-인**, 역으로 **축소/BE 승격/트레일링**.

**리스크 & 포지션 정책**

* **ATR-unit 사이징**: qty ∝ target\_R / ATR; 변동성 급등 시 자동 축소.
* **상관 캡**: 동일 방향 동시노출 제한: adj\_notional = notional × (1 - ρ\_to\_BTC).
* **데일리 리스크 가드**: 일손실/최대DD 컷, 당일 재진입 금지 옵션.
* **레버리지**: 심볼 기본 레버리지만, **실노출은 Risk Engine 기준**으로 제한.
* **스케일링**:
  + **스카웃 → Add1 → Add2 …** (각각 조건: MFE 진전, S 유지, RV 안정)
  + **부분청산(1R/2R/3R)** + **손익분기점(BE) 승격** + **트레일 스탑**
* **중복·오류 방지**: 같은 캔들 재트리거 방지(idem\_hit), 주문/체결 일치 검사, 재부팅 시 **포지션/미체결 복원**.

**Discord UX**

**채널 역할**

* **#분석**:
  + “현재 레짐/예측 요약 카드” + 핵심 차트(선택) + 이유 로그(간결)
  + 일정 주기(예: 1m) **업데이트 편집(Edit)**, 과도한 채팅 스팸 방지
* **#대시보드**:
  + **총자본**, **미실현/실현 PnL**, **수익률(누적/당일)**, **사용 증거금**, **수수료 누계**, **심볼별 노출**
  + **실시간 편집 갱신**(포지션이나 가격 변동 시)
* **#알림**:
  + **진입/증액/축소/청산 체결**
  + **위험 경보**(증거금비율↑, 강제청산거리↓, 일손실/목표수익 도달)
  + **중요 시스템 이벤트**(재시작, 네트워크 지연, API 오류)
* **#컨트롤패널**(슬래시 명령 + 버튼/슬라이더)
  + /mode paper|testnet|live (live 전환은 “ARM\_LIVE” 타이핑+확인 버튼 2단계)
  + /auto on/off, /close all|BTC|ETH, /reverse BTC, /flat BTC
  + **슬라이더**: target\_R%, horizon\_k, max\_adds, corr\_cap, risk\_stop\_day%
  + **토글**: SCOUT\_ONLY, STRONG\_SIGNAL\_BYPASS, TRAIL\_ENABLE
  + **수동 주문 버튼**: Buy/Sell(시장), 지정가 입력 모달
  + **설정 저장/불러오기**: Save Config, Load Last

**ENV 최소 셋(나머지는 컨트롤패널에서)**

# [ENV:CORE]

MODE=testnet # paper|testnet|live

SYMBOLS=BTCUSDT,ETHUSDT

TF\_EXEC=1m

TF\_SIGNAL=5m,15m,1h,4h

# [ENV:EXCHANGE]

BINANCE\_TESTNET\_API\_KEY=...

BINANCE\_TESTNET\_API\_SECRET=...

BINANCE\_LIVE\_API\_KEY=...

BINANCE\_LIVE\_API\_SECRET=...

# [ENV:RISK]

RISK\_TARGET\_PCT=0.30 # 기본값, 패널에서 즉시 변경 가능

DAILY\_MAX\_LOSS\_PCT=3.0

CORR\_CAP\_PER\_SIDE=0.65

# [ENV:DISCORD]

DISCORD\_BOT\_TOKEN=...

CHAN\_ANALYSIS\_ID=...

CHAN\_DASHBOARD\_ID=...

CHAN\_PANEL\_ID=...

CHAN\_ALERTS\_ID=...

# [ENV:MISC]

DB\_PATH=./runtime/trader.db

CONFIG\_PATH=./runtime/config.json

PATCH\_LOG=./patch.txt

실계좌 키는 token.env로 분리, .gitignore 적용.

**파일 구조(초안)**

ftm2/

app.py # Orchestrator(메인)

core/

env.py # load\_env\_chain()

state.py # 전역 스냅샷/상태 버스

scheduler.py

persistence.py # sqlite, config.json, patch.txt

exchange/

binance.py # REST/WS, testnet/live 토글

data/

streams.py # WS 구독, kline/mark/account

features.py # 지표/피처 파이프라인

signal/

regime.py # 레짐 분류

forecast.py # k-bar 예측 앙상블

scoring.py # 최종 점수/임곗값

trade/

risk.py # ATR-unit, 상관캡, 데일리컷

gatekeeper.py # 라우팅 검사→게이트 순서

execution.py # 스카웃/스케일/부분청산/트레일

positions.py # 하이드레이션/동기화

fees.py # 수수료/슬리피지 모델

discord\_bot/

bot.py # 런처

views.py # 버튼/슬라이더 UI

panel.py # 슬래시 커맨드

dashboards.py # 메시지 편집 업데이트

tests/

e2e\_sim.py

unit/

test\_features.py

test\_risk.py

파일 내 주요 섹션은 # [ANCHOR:BINANCE\_CLIENT] 등 **앵커 주석**으로 구분.

**DB 스키마(요약)**

* trades(id, ts, symbol, side, qty, px, type, fee, order\_id, link\_id)
* positions(symbol, qty, avg\_px, u\_pnl, r\_pnl, leverage, liq\_px)
* pnl\_daily(date, realized, fees, net, max\_dd)
* config(key, val, updated\_at) ← 패널 변경사항 영속
* events(ts, level, source, message)
* patches(ts, version, title, note) ← **patch.txt와 동기화**

**실행 플로우 & 게이트 순서**

1. **라우팅 검사**: 모드/심볼 가능 여부, 네트워크/레이턴시/쿨다운
2. **게이트키퍼**:
   * AUTO=on?
   * idem\_hit(동캔들 중복 차단)
   * SCOUT\_ONLY/STRONG\_SIGNAL\_BYPASS/TARGET\_SCORE\_WAIT(옵션)
   * 데일리 손실컷/추가 트레이드 금지 여부
3. **리스크 체크**: ATR-unit 사이즈, 상관 캡, 레버리지, 증거금 여력
4. **주문 전송**: 테스트넷/실계좌 라우팅, link\_id 부여
5. **체결 처리**: 포지션/대시보드/알림 업데이트, 보호체크(트레일/BE/부분청산)

**대시보드 카드(예시, 실시간 편집)**

* **총자본(추적)**, **미실현/실현 PnL**, **일수익률/누적수익률**
* **수수료 누계**, **증거금 사용률**, **강제청산 거리**
* **심볼별**: 포지션, 평균가, MFE/MAE, 스케일 단계, 다음 트리거
* **상태 뱃지**: MODE, AUTO, SCOUT\_ONLY, STRONG\_SIGNAL\_BYPASS

**알림 정책(간결 & 필수만)**

* **ENTRY/ADD/REDUCE/EXIT**: symbol side qty px (reason=S, regime, k)
* **리스크**: 증거금비율↑, liq\_px 근접, 일손실컷, 네트워크 지연
* **성능**: 목표R 도달, Trailing 체결, 데일리 리포트 1회

**테스트 플랜(E2E)**

* **시뮬/테스트넷**에서 24h 스트림 재생(리플레이) → 진입/스케일/부분청산 검증
* **재시작 복원**: 미체결/포지션 하이드레이션, idem\_hit 정상 동작
* **컨트롤패널**: 슬라이더/버튼 실시간 반영 & DB 저장 → 재시작 후 복원
* **안전장치**: live 전환 2단 확인, 키 미입력 시 주문 차단 로그

**보안·실수 방지**

* 키는 token.env에만, 커밋 금지
* **live 전환**: /mode live → “ARM\_LIVE” 입력 모달 → 최종 확인 버튼
* **살수/매수 실수 방지**: 사이즈/레버리지 상한, 극단 주문 차단
* **레이트 리미트/재시도** 표준화, REST/WS 자동 재연결

**patch.txt 운영**

* **기능 추가**는 반드시 patch.txt에 **추가** 기록(버그픽스 제외).
* 형식(예):
* 2025-09-03 v0.1.0
* - 초기 골격 구축: Exchange/Streams/Features/Regime/Forecast/Risk/Execution/Discord/DB
* - 테스트넷 E2E 루프 가동, 대시보드 실시간 편집 업데이트
* - 컨트롤패널: AUTO/R%/k/SCOUT\_ONLY/STRONG\_SIGNAL\_BYPASS/DAILY\_CUT
* patches 테이블에도 동기 삽입 → UI에서 **/patch log** 조회 가능.

**“예측 중심” 의사결정 로직(간단 수식)**

* P\_up = σ(a·EMA\_slope + b·Donchian + c·ADX …)
* MR\_edge = f(zscore, band\_touch, reversion\_speed)
* Cross\_edge = g(lead\_lag(ETH↔BTC), dβ/dt)
* **온라인 가중치**: W\_{t+1} = (1-λ)·W\_t + λ·normalize(perf\_last\_N)
* **최종 점수**: S = W·[P\_up, MR\_edge, Cross\_edge]
  + 레짐별 임계치 {θ\_enter, θ\_add, θ\_reduce, θ\_exit}
  + k, λ, N은 **슬라이더**로 조정.

**초기 마일스톤(1~3주 내 완성 기준 설계)**

1. **M1 — 골격 & 테스트넷 주문 라운드트립**
   * Exchange/WS, DB, ENV 최소, Discord 연결, /mode, /auto, 대시보드 카드
2. **M2 — 레짐·포캐스트·리스크 타게팅**
   * Feature 파이프라인, 레짐 분류, k-bar 예측 앙상블, ATR-unit 사이즈, 상관캡
3. **M3 — 스케일링 & 보호체크 & 알림 최적화**
   * 스카웃/스케일/부분청산/BE/트레일, idem\_hit, 하이드레이션, 위험/체결 알림

**Codex 요청서 포맷(추후 코드 작성 시 예시)**

* **앵커**: [ANCHOR:ORDER\_ROUTER]
* **검색어**: class OrderRouter
* **정확한 교체/추가 지점**: def submit\_order(...): 아래
* **.env 토글**: RISK\_TARGET\_PCT, DAILY\_MAX\_LOSS\_PCT
* **요청 내용(요약)**:

ATR-unit 사이징 반영하여 주문 수량 산출, 상관캡 적용, idem\_hit 체크 후 전송. 체결 시 보호체크 스케줄 등록. 실패 시 재시도.

**바로 착수 산출물(오늘 생성 가능)**

* ftm2/ **프로젝트 스캐폴드**(위 구조)
* .env.sample, token.env.sample, config.json 기본값
* patch.txt **v0.1.0** 초기 기록
* Discord **슬래시 커맨드/버튼/슬라이더** 기본 세트
* **테스트넷 마켓데이터 수신 + 더미 포캐스트 → 스카웃 1단계 주문**까지

필요하면 이 설계 그대로 **코드 뼈대부터 생성**해드릴게요. 원하는 우선순위(예: “대시보드 먼저”, “스케일링 먼저”)가 있으면 말씀만 주세요.