cTrader Bot Project Manual (Phases 0 to 370)

This manual covers the complete cTrader bot project from Phases 0 to 370. It includes an overview, phase descriptions, setup instructions, deployment steps, and usage notes. This document is intended for developers and traders.

Overview

Overview: The cTrader bot project is designed to provide a modular, scalable, and robust trading system. It includes signal logic, reinforcement learning, adaptive SL/TP tuning, CAB logging, MetaVvAI feedback, and cloud-ready deployment features. The bot is split into three main modules: CoreBot, RLBot, and DeploymentBot. CoreBot (Phases 333–343): - Signal logic combining Envelope, Engulfing, and Boilinger signals. - Weighted scoring system. - Adaptive SL/TP tuning based on signal strength. - CAB logging and chart markers. - MetaVvAI reinforcement learning feedback. RLBot (Phases 344–350): - Reinforcement learning overlays. - Adaptive lot sizing. - Session-aware SL/TP tuning. - Predictive overlays and RL scoring heatmaps. DeploymentBot (Phases 351–370): - Telemetry integration. - Cloud logging hooks. - Error recovery mechanisms. - Secure API key handling. - Cloud deployment script. - Hardened exception handling.

Phase Descriptions

Phase Descriptions: Phases 0–332: Placeholder for early logic (ready for future backfill). Phases 333–343: CoreBot - Signal logic, scoring, SL/TP tuning, CAB, MetaVvAI. Phases 344–350: RLBot - Reinforcement learning, overlays, lot sizing. Phases 351–370: DeploymentBot - Telemetry, cloud hooks, error handling, packaging. Detailed Phase Breakdown: Phase 333: TripleConfirmationSignal logic. Phase 334: Weighted scoring system. Phase 335: CAB logging + chart markers. Phase 336: MetaVvAI RL feedback. Phase 337: Reward decay + normalization. Phase 338: Session overlays. Phase 339: CAB-MetaVvAI sync. Phase 340: Reward confidence bands. Phase 341: CAB timeline overlays. Phase 342: Adaptive SL/TP with CAB overlays. Phase 343: CAB + MetaVvAI fusion logic. Phase 344: Predictive overlays. Phase 345: RL scoring heatmaps. Phase 346: CAB-MetaVvAI signal fusion. Phase 347: Adaptive lot sizing. Phase 348: Session-aware SL/TP tuning. Phase 349: Export-ready packaging. Phase 350: Final integration and testing. Phase 351: Cloud Logging Hook (optional HTTP POST). Phase 352: Journal Sync Engine (local + remote logs). Phase 353: Remote Config Fetch (cloud presets). Phase 354: Error Telemetry Hook (webhook alerts). Phase 355: Cloud Dashboard Sync (live stats). Phase 356: Secure API Key Handling. Phase 357: Cloud-Ready Build Mode (toggle local/cloud). Phase 358: Journal Integrity Checker. Phase 359: Cloud Deployment Script. Phase 360: Hardened Exception Handling.

Setup Instructions

Setup Instructions: 1. Extract the ZIP file containing the cTrader bot project. 2. Open the `cTraderBot.sln` file in Visual Studio. 3. Build the solution to compile all phases. 4. Ensure all dependencies are installed (e.g., .NET framework). Configuration: - Configure API keys and cloud endpoints in the `DeploymentBot` module. - Adjust SL/TP presets and risk levels in the `CoreBot` module. - Enable or disable cloud logging and telemetry in the `DeploymentBot` module. Testing: - Use the provided test harness to simulate trades and validate logic. - Check logs and overlays for accuracy and consistency.

Deployment Steps

Deployment Steps: 1. Ensure the bot is compiled and tested locally. 2. Configure cloud endpoints and API keys. 3. Deploy the bot to a cloud-hosted cTrader instance using the provided deployment script. 4. Monitor telemetry and logs to ensure proper functioning. Cloud Deployment: - Use the `Cloud Deployment Script` in the `DeploymentBot` module. - Monitor live stats and error reports via the cloud dashboard. - Validate journal consistency using the `Journal Integrity Checker`. Local Deployment: - Deploy the bot to a local cTrader Automate environment. - Monitor logs and overlays for accuracy. - Adjust configurations as needed based on trading performance.

Usage Notes

Usage Notes: - The bot is designed to be modular and scalable. - Each module can be deployed independently or as part of the unified bot. - Use the provided test harness to simulate trades and validate logic. - Monitor telemetry and logs to ensure proper functioning. - Adjust configurations based on trading performance and market conditions. Best Practices: - Regularly update API keys and cloud endpoints. - Validate journal consistency and flag anomalies. - Monitor live stats and error reports via the cloud dashboard. - Adjust SL/TP presets and risk levels based on market conditions.