Clouty / Sabi — Quantitative Engineer Technical Assessment

Candidate: Olaoluwa

This assessment evaluates your ability to architect and implement an execution bot for Clouty's event-contract exchange. Demonstrate practical skills in quantitative system design, market microstructure understanding, and execution logic. Use C++ for this exercise.

Goal

Build a simplified execution bot that automatically quotes and updates prices for event contracts across multiple categories — economic, political, sports, and cultural. The bot should maintain market state, manage exposure, and respond dynamically to order flow.

Structure

- Part 1 – Design (60–90 min): Outline architecture, logic, and risk safeguards.
- Part 2 – Build (5–8 hours): Code the MVP execution bot in C++.
- Part 3 – Explain (30 min): Summarize and defend design choices.

Part 1 – Design Brief

Write a 1–2 page design note describing:

- Architecture and data flow (orders → state → quotes).
- How mid, spread, and size are determined for a binary event contract.
- How quotes adapt to imbalanced order flow.
- Risk management logic (inventory caps, circuit breakers).
- How the bot maintains stability after disconnects or data gaps.

Part 2 – Build the MVP

Requirements:

- Support at least three event markets (e.g., inflation >20%, election outcome, sports result).
- Implement a state engine per market with mid_prob, spread, inventory, exposure, and fees.
- Generate two-sided quotes and update dynamically based on flow.
- Handle incoming orders, simulate fills, update exposure and PnL, and re-quote.

Integrate adaptive logic that widens spreads or shades the mid when flow is one-sided.

- Enforce risk caps (inventory and exposure limits).
- Log metrics: fill count, notional, PnL, max drawdown, average spread, final inventory.

Extra Credit: async I/O, pytest unit tests, or a simple dashboard.

Part 3 – Debrief

After completion, provide a short written or live explanation covering your pricing logic, risk design, trade-offs, and scalability considerations.

Evaluation Rubric (50 Points)

- Architecture & Clarity (10): Modular, readable structure.
- Market-Making Logic (12): Responsive, realistic quoting behavior.
- Risk Controls (10): Clear limits and protective behavior.
- Correctness & Tests (8): Deterministic, validated logic.
- Performance & Resilience (5): Handles multiple markets smoothly.
- Product Sense (5): Metrics usable by a live trading desk.

Submission

Submit code, configuration files, and notes in a Git repo or zipped folder. Include a README explaining how to run simulations, interpret results, and where you'd optimize further.

Good luck — show how you think, build, and execute like a market architect.