DATE: October 17, 2025
TO: Dr. Neeraja Yadwadkar
FROM: Sree Duggirala

SUBJECT: Individual Risk Reduction – Latency Measurement, Staleness Detection, and Network Health Monitoring

For my individual risk-reduction task, I focused on verifying and enforcing the system's latency requirements. In algorithmic market making, even small delays in order acknowledgments or market-data updates can cause the strategy to operate on stale information, increasing the likelihood of adverse selection. To mitigate this risk early in development, I designed and implemented the system that measures order round-trip latency (ORTT), tracks WebSocket health, and detects data-feed staleness.

Technical Implementation

My work centered on two modules: (1) a timestamping and monitoring framework for quotes and acknowledgments, and (2) a HeartbeatMonitor for network-health tracking.

Order Latency Instrumentation

I extended the order-lifecycle tracking in crates/oms to generate precise timestamps at three points:

- created_ms — when the strategy decides to place an order
- sent_ms — when the adaptor dispatches the HTTP/WebSocket request
- recv_ms — when the exchange acknowledgment is parsed

Using these fields, I compute the **Order Round-Trip Time (ORTT)**:

ORTT = recv_ms – created_ms

This metric allows us to verify that p99 acknowledgement latency stays under our 200 ms threshold. Because high-frequency strategies depend on live, accurate order states, exceeding this threshold would make the system vulnerable to quote-aging and unnecessary fills.

Heartbeat-Based Staleness Detection

To detect feed degradation without sending real orders, I implemented the HeartbeatMonitor in crates/adapters. It sends periodic application-layer pings over the WebSocket and measures the elapsed time until the corresponding pong is received. The module keeps track of last_message_received and exposes a staleness flag if no packets arrive for a configurable timeout.

This mechanism allows the system to detect:

- network lag
- silent WebSocket failures
- partial disconnections

and triggers a reconnection event before the quoting system becomes blind.

Experimental Validation

To test these mechanisms safely, I simulated degraded network conditions.

Order Latency Results

Using exchange testnet endpoints:

- p99 ORTT measured **150 ms**, below our **200 ms** requirement
- the system reliably captured latency spikes and exposed them via logs

Heartbeat & Staleness Results

I manually severed the network connection during active market-data streaming:

- staleness detection fired in **~1.5 seconds**
- reconnection (performed by Praneel's reconnection loop) completed in **< 1 second**
- overall downtime: **2.3 seconds**, well within the **5 second** limit

This confirmed that the system can quickly detect and recover from data-feed failures before the inventory model becomes unstable.

Summary

My risk-reduction work ensures that our system operates with reliable, real-time market visibility. By validating ORTT performance and implementing robust staleness detection through the HeartbeatMonitor, we significantly reduced the risk posed by network delays and silent data-feed failures. This foundation is essential for safe algorithmic trading and improves the overall dependability of our market-making engine.