


Comprehensive Error Handling and Recovery Strategy

Introduction

Purpose

This document outlines the sophisticated error handling and recovery mechanisms designed for the real-time stock data streaming system, ensuring robust, resilient, and reliable performance.

 error_handling_diagram

Error Classification Taxonomy

1. Connection Errors

- **WebSocket Disconnection**
 - Unexpected connection termination
 - Network instability
 - External API unavailability
- **Authentication Failures**
 - Invalid API credentials
 - Token expiration
 - Security-related disconnections

2. Parsing Errors

- **Message Structure Violations**
 - Malformed JSON
 - Unexpected data formats
 - Missing critical fields
- **Data Type Mismatches**

- Incorrect data type conversion
- Precision loss
- Unexpected data representations

3. Processing Errors

- **Threading Complications**
 - Deadlock prevention
 - Race condition mitigation
 - Resource contention handling
- **Queue Management Issues**
 - Message backlog management
 - Overflow prevention
 - Prioritization strategies

4. Resource Constraint Errors

- **Memory Exhaustion**
 - Prevent out-of-memory scenarios
 - Intelligent memory management
 - Dynamic resource allocation
- **CPU Overutilization**
 - Thread pool throttling
 - Computational load balancing
 - Graceful performance degradation

5. External API Errors

- **Rate Limiting**
 - Handle API quota restrictions
 - Implement intelligent backoff
 - Manage request frequency
- **Data Inconsistency**

- Validate external data sources
- Implement data integrity checks
- Provide fallback mechanisms

Error Response Strategies

1. Automatic Reconnection

- **Exponential Backoff Algorithm**
 - Incremental wait times between reconnection attempts
 - Prevent overwhelming the system
 - Adaptive retry mechanism
- **Connection Retry Limits**
 - Maximum reconnection attempts
 - Fallback to alternative data sources
 - Comprehensive logging

2. Message Retry Mechanism

- **Transient Error Handling**
 - Identify retrievable errors
 - Implement smart retry logic
 - Preserve message integrity
- **Message Queue Management**
 - Temporary message storage
 - Intelligent reprocessing
 - Prevent data loss

3. Fallback Data Processing

- **Degraded Mode Operation**
 - Maintain partial system functionality
 - Alternative data sources
 - Reduced processing capabilities

- **Cached Data Utilization**
 - Use previously processed data
 - Minimize service interruption
 - Provide continuous insights

4. Resource Reallocation

- **Dynamic Thread Pool Adjustment**
 - Adaptive thread management
 - Resource optimization
 - Performance maintenance
- **Computational Load Balancing**
 - Distribute processing load
 - Prevent single point of failure
 - Maintain system responsiveness

5. Graceful Degradation

- **Selective Ticker Processing**
 - Prioritize critical stock tickers
 - Suspend non-essential processing
 - Maintain core system functionality
- **Partial Data Delivery**
 - Provide available data
 - Clear error communication
 - Transparent system state

Recovery Mechanisms

1. Websocket Reconnection

- Intelligent connection restoration
- Minimal service interruption
- Comprehensive connection state management

2. Thread Pool Reset

- Safe thread termination
- Resource cleanup
- Predictable system recovery

3. Cache Recovery

- Persistent data storage
- State restoration capabilities
- Minimize data loss

4. State Restoration

- Capture system state before failure
- Intelligent rollback mechanisms
- Comprehensive context preservation

5. Logging and Alerting

- Detailed error documentation
- Real-time notification systems
- Forensic analysis support

Monitoring and Reporting

Error Rate Tracking

- Comprehensive error metrics
- Historical performance analysis
- Trend identification

System Health Dashboard

- Real-time system status
- Detailed error visualizations
- Actionable insights

Performance Impact Analysis

- Quantify error-related overhead
- Optimization recommendations
- Continuous improvement tracking

Root Cause Investigation

- Detailed error context
- Systematic problem identification
- Preventative strategy development

Continuous Improvement Feedback

- Error pattern recognition
- Machine learning-enhanced prediction
- Proactive system enhancement

Conclusion

The error handling and recovery strategy provides:

- Robust system resilience
- Minimal service disruption
- Intelligent error management
- Continuous system improvement

Key Principles

- Anticipate potential failures
- Provide intelligent responses
- Maintain system integrity
- Ensure data reliability

Author : Shriniwas Kulkarni

- PCCOE 2026 BTech CSE(AIML)
- Email: kshriniwas180205@gmail.com
- Phone: +91 [8999883480]
- GitHub: github.com/Shriniwas18K