Project feedback generator - Project Feedback Summary - Project Feedback Summary

Milestone 1.pdf
PDF

Please give feedback on my project attached

Feedback on Milestone 1 Project Submission

Strengths:

1. **Comprehensive workflows**: Your project provides detailed workflows for various subdomains like cross-traded cryptocurrency identification, arbitrage opportunity identification, and trading strategy execution. These workflows clearly define the inputs, outputs, and processing steps, reflecting a structured approach to the business processes.
2. **Clear use of Ubiquitous Language**: The naming conventions align well with the domain-driven design (DDD) principles. Events, commands, and types such as `CrossTradedCryptocurrenciesIdentified`, `ArbitrageOpportunitiesIdentified`, and `TradeCompleted` are concise and reflective of the domain requirements.
3. **Core subdomain focus**: Key subdomains like `Trading Strategy` and `Order Buy and Sell` are treated as core areas of the application, and you maintain functional purity by isolating side effects in specific contexts (such as the use of external systems for order processing).
4. **Correct handling of trading process steps**: The progression from identifying arbitrage opportunities to placing orders and tracking their fulfillment is logically laid out, ensuring that each step of the workflow has a clear start and stop condition, as recommended in functional programming and DDD.

Weaknesses:

1. **Event use without definitions**: Some events, such as `CrosstradedCryptocurrenciesIdentified` and `ArbitrageOpportunitiesIdentified`, are used within workflows but lack clear data structure definitions. For instance, while `CurrencyPair` is mentioned, there is no detailed breakdown of its content. You need to explicitly define the types involved in these events to avoid ambiguity.
2. **Missing or incomplete workflows**:

  - The workflow for order status retrieval (`Order Status Update`) is incomplete. It should include how the status is retrieved and updated, specifically when the order is fulfilled or partially fulfilled. Additionally, the stop conditions for this workflow are not clearly addressed.
  - There is no clear step for comparing market data in real-time to identify arbitrage opportunities. The current design does not mention how the latest price spread values will be maintained or compared for identifying arbitrage potential.
3. **Unmanaged stop conditions**: The overall stop conditions, particularly for the trading processes, are not well-defined. For example, in the "Real-time market data feed check" workflow, it is unclear how trading would stop when certain parameters are breached (e.g., minimal price spread or maximal transaction values).
4. **Lack of error handling**: While you have included some command workflows, domain errors such as `OrderNotFound` or issues like network timeout or API failures are not consistently handled. Specifically, the `Stop Trading Strategy` workflow should account for errors or interruptions in market data feed subscriptions.

Specific Areas for Improvement:

1. **Define all used events and commands**: Ensure that all events and commands, such as `CrosstradedCryptocurrenciesIdentified` and `TradeCompleted`, have clearly defined data structures that are referenced within the workflows.
2. **Complete the workflow for order status retrieval**: Ensure that your `Order Status Update` workflow handles all possible order outcomes (fully fulfilled, partially fulfilled, or canceled). This includes defining how the system will retrieve and persist these updates.
3. **Clarify real-time market data handling**: Introduce a specific mechanism to handle real-time comparison of bid/ask prices, ensuring that price spreads are maintained and consistently evaluated across exchanges. You need to manage how arbitrage opportunities are identified and acted upon in the system.
4. **Add error handling mechanisms**: Introduce workflows or steps that deal with domain errors, such as incomplete order fulfillment or market data retrieval errors. The absence of these may cause the system to behave unpredictably in edge cases.

Your project shows solid alignment with DDD principles, but refining these workflows will significantly improve its completeness and correctness.