Please give feedback on my project attached.

Feedback for "Milestone I: System Design"
Strengths:

Comprehensive Coverage of Workflows: The document contains a well-structured set of workflows, covering various essential aspects of the system, such as real-time data feed subscription, processing arbitrage opportunities, executing trading strategies, and retrieving order statuses. These workflows are consistent with the functional requirements outlined in the project specification.

Appropriate Use of Domain-Driven Design (DDD) Principles: The bounded contexts are clearly defined, and the system leverages DDD elements effectively. The naming conventions for events and commands are clear and aligned with the principles of ubiquitous language. This makes the document easy to follow and contributes to clarity in understanding the system behavior.

Real-Time Trading Workflow: The "Real-Time Data Feed Subscription" and subsequent workflows like "Process New Quote for Arbitrage Opportunity" and "Execute Trading Strategy" accurately capture the real-time trading activities of the system. This aligns well with the project requirements of continuously identifying arbitrage opportunities and acting upon them in a real-time environment.

Inclusion of Error Handling: Your workflows contain explicit error handling (e.g., InsufficientFundsError, SpreadCalculationError), which is essential in financial applications like arbitrage trading. This is a good step toward robustness in your design, as it accounts for potential failures during the execution of commands and workflows.

Weaknesses:

Incomplete Stop Conditions: The document does not clearly specify the stop conditions for when trading should halt automatically. While there is a manual stop workflow ("Stop Real-Time Trading"), the system should have automatic triggers to stop trading under specific conditions (e.g., reaching trading limits or a specified time period). This is crucial to avoid risks like over-trading or exceeding user-defined constraints.

Unclear Market Data Comparison: In the workflow "Process New Quote for Arbitrage Opportunity," there is a step for calculating the spread and checking arbitrage opportunities. However, it is unclear how the comparison of real-time market data is managed and how the latest values to compare with are stored or maintained. This is a key functionality for identifying arbitrage opportunities and should be more explicitly defined.

Event and Command Redundancies: Some events, such as OrderStatusUpdated and TransactionPersisted, are defined but lack clear connections to the workflows. Additionally,

there are some redundancies where multiple events seem to handle similar outcomes without clear differentiation. This could lead to confusion or inefficiency in the system's event handling.

Pseudocode Notation: While the use of pseudocode is mostly appropriate, there are a few instances where longer sentences are used, particularly in the steps describing logic (e.g., "emit event OrderEmitted for both BuyOrder and SellOrder"). Shorter, more concise statements would improve clarity and adhere more closely to the pseudocode style required by the course.

Weaknesses Restated in Specific Workflows:

Stop Conditions: The system should manage stop conditions for trading more explicitly. For example, in the workflow "Execute Trading Strategy," once arbitrage opportunities are identified, the system should check whether user-defined trading limits have been reached and stop trading accordingly. This functionality is not fully addressed.

Market Data Comparison: In the workflow "Process New Quote for Arbitrage Opportunity," ensure that there is a mechanism to compare real-time data (from the latest quotes across exchanges). It is not clear how this data is maintained, and without this, the system cannot reliably identify arbitrage opportunities.

Redundancies in Event Handling: The OrderStatusUpdated and TransactionPersisted events should be refined. You can consolidate these to streamline event handling and ensure that each event has a clear purpose and is used efficiently within workflows like "Retrieve Order Status and Update."

Improvement of Pseudocode: Consider reworking some of the pseudocode to eliminate long, sentence-like descriptions. This can be seen in workflows like "Execute Trading Strategy," where the logic for emitting orders could be more concise and aligned with a more formal pseudocode notation.

Conclusion:
Your submission demonstrates a strong grasp of Domain-Driven Design principles, especially in terms of structuring workflows and defining commands and events. However, addressing the stop conditions and market data handling more clearly would greatly improve the coherence of the system. Additionally, refining the event structure and ensuring proper use of pseudocode will enhance clarity and efficiency.

We acknowledge the suggestion regarding incomplete stop conditions and agree that implementing automatic stop triggers would enhance trading safety. While we haven’t incorporated these changes into the current pseudocode, we plan to implement automatic stop conditions, such as trading limits or specific time periods, in a future update to prevent risks like over-trading or exceeding user-defined constraints.