Work Trial Task: Strategy Backtesting Framework for Smart Order Routing (SOR)

Objective:

The goal of this task is to assess your ability to design and implement a backtesting framework for a Smart Order Router (SOR). This trial evaluates your conceptual understanding of backtesting and your practical engineering skills in creating a robust testing environment.

Part 1: Methodology and Framework (2-3 Pages)

You will write a **2-3 page PDF** outlining your proposed backtesting framework for an SOR, detailing the following:

1. Data Pipeline

- How to source, process, and handle market/order book data.
- Address challenges related to missing data, bid/ask updates, and trade data synchronization.

2. Execution Strategies

- Description of simplified execution strategies, such as TWAP (Time-Weighted Average Price) and VWAP (Volume-Weighted Average Price).
- Highlight how different market conditions could affect these strategies.

3. Performance Metrics

- Explain the key metrics for strategy evaluation, including execution cost, slippage, and fill rates.

4. Simulation Logic

- Design the logic for simulating multi-venue routing decisions and how to accurately model order placements across venues.
- Include multi-leg trades and the potential for more complex order types.

5. Extensibility and Scalability

- Propose methods to scale the framework for multi-leg strategies and advanced SOR configurations.
- Discuss how the system can be extended to different asset classes.

Evaluation Criteria for Part 1:

- **Clarity:** Structure, logical flow, and explanations.
- **Feasibility:** Practicality and scalability of the proposed framework.
- **Alignment:** How well your framework aligns with real-world backtesting requirements.

Part 2: Code Implementation

You will implement a **basic backtesting simulator** in Python to demonstrate your framework.

Deliverables:

1. **Python Script:**
    - Simulate trade execution using a TWAP strategy.
    - Generate synthetic data for prices, volumes, and timestamps.
    - Calculate and output the following metrics:
        - **Execution Cost:** Difference between executed price and benchmark price (e.g., VWAP).
        - **Slippage:** Difference between expected execution price and actual price.
2. **Report (1 Page):**
    - A concise explanation of your implementation approach and the results of your backtesting simulation.

Sample Research Papers:

To help you design your framework, you can reference the following research papers:

1. **Combining Deep Learning on Order Books with Reinforcement Learning for Profitable Trading**
    - Link to Paper
    - Focus: Temporal-difference learning models for return forecasting.

2. **Multi-Agent Reinforcement Learning in a Realistic Limit Order Book Market Simulation**
    - Link to Paper
    - Focus: Agent-based simulation using Double Deep Q-Learning (DDQL).
3. **Interpretable ML for High-Frequency Execution**
    - Link to Paper
    - Focus: Modeling the fill probability function with state dependence for execution backtesting.
4. **Deep Reinforcement Learning for Market Making Under a Hawkes Process-Based Limit Order Book Model**
    - Link to Paper
    - Focus: DRL-based controller for optimizing order execution under stochastic conditions.

Submission Guidelines:

1. **Format:**
    - **Part 1:** PDF (methodology and framework).
    - **Part 2:** Python code (backtesting implementation) + 1-page report.
2. **GitHub Repository:**
    - Provide a public GitHub link containing the full project with clear documentation and structure.
3. **Deadline:**
    - **5 days** from task assignment.
4. **Submission:**
    - Submit Github link and Latex PDF to submission link provided

---

Evaluation Criteria:

1. **Methodology and Documentation:**
    - Quality of the 2-3 page framework document.
    - Clear explanation of data handling, execution strategies, and simulation design.
2. **Implementation:**
    - Accuracy of the backtesting simulation and output metrics.
    - Code clarity, structure, and adherence to Python best practices.
3. **Analysis and Reporting:**

    - Clear communication of results and implementation details in the 1-page report.
4. **Innovation and Scalability:**
    - Evidence of creative solutions to enhance the backtesting framework's realism and extensibility.
5. **Bonus Points:**
    - Incorporating cross-asset dependencies, adaptive strategies, or advanced machine learning models in the simulation.

Optional Bonus Task:

- Extend your implementation to evaluate more complex strategies, such as VWAP with dynamic rebalance intervals or reinforcement-learning-based smart routing.