Backend Architect Technical Assessment

Objective:

This assessment evaluates your ability to design and implement a trading chart service in Go that reads tick data from a data stream, aggregates the data into OHLC candlesticks, broadcasts the current bar to a streaming server API, and stores complete bars into a database. The service should be deployed to a Kubernetes cluster using Terraform.

Estimated Time to Complete:

6-8 hours (with the option to extend to a maximum of 10 hours)

Problem Statement

You are tasked with building a trading chart service that connects to the Binance API, fetches tick data for specified symbols, aggregates the tick data into OHLC candlesticks, broadcasts the candlesticks to a streaming server, and persists the data to appropriate datastore(s).

This service will form a critical part of the trading platform and should be designed to be reliable, scalable, and maintainable.

Requirements

1. Data Ingestion and Aggregation

- Fetches real-time tick data for the following symbols:
  - BTCUSDT
  - ETHUSDT
  - PEPEUSDT
- Aggregates data into 1-minute OHLC candlesticks.

2. Streaming Server API

- Implements a gRPC API to broadcast current candlestick data on each new tick received from the Binance API.

3. Data Persistence

- Persist complete candlestick data into datastore(s) of choice.

4. IaC

- Use Terraform to define and manage infrastructure for the service.
- Services will be deployed to K8s cluster(s).

5. Testing

- Appropriate test suite(s) for key functional requirements and/or service components.

6. Documentation

- Include a README with:
    - A high-level overview of the architecture.
    - Instructions on how to build, run, and test the service locally and/or in K8s.

Time Management Guidance

We understand that the requirements are ambitious for a 6-8 hour timeframe. You are encouraged to spend no more than 10 hours on this project. Focus on delivering a functional, high-quality implementation for the key requirements. If you are unable to complete all parts within the time limit, prioritize what you believe to be the most critical aspects of the project and document what could be improved or added with more time.

Deliverables

1. **Code**:
    - A GitHub repository (or ZIP file) containing the source code
        1. Go service
        2. Terraform code for infrastructure provisioning
        3. K8s manifests

2. **Design Document**:
    - A brief document explaining your design decisions, with a focus on:
        1. How the service handles data ingestion, aggregation, and streaming.
        2. Data design.
        3. Choice of K8s resources and deployment strategy.

3. **Instructions**:
    - A `README.md` file with setup, run, and test instructions.

Evaluation Criteria

1. **Go Expertise**:
    - Code clarity, maintainability, and proper use of Go idioms.
    - Knowledge of multithreading and/or synchronization.
    - Data pipelining.
    - Future-proof code.

2. **Systems Design**:
    - Soundness and scalability of the service architecture.
    - Robust error handling, edge-case management, and resilience to issues.
    - Ability to work with obscure requirements.

3. **IaC**
    - Effective use of Terraform to define and manage infrastructure.
    - Proper design and configuration of K8s resources.

4. **Testing**:
    - Coverage and effectiveness of unit tests.
    - Simplicity and clarity in running the tests.

5. **Documentation**:
    - Clarity and completeness of the documentation.

6. **Time Management**:
    - How well you managed the 10-hour limit, and the quality of work delivered within this timeframe.