Technical Assessment: Binance Go Integration

Overview

In this assessment, you'll be creating a Go application which interacts with Binance's WebSocket API. Your solution should be containerized using Docker. Please also provide a README that explains how to set up and run the application, any design decisions you made, performance considerations, and any limitations of your design. Use idiomatic Go standards and project layouts where possible.

Tasks

1. Connect to Binance's WebSocket API and subscribe to the Diff. Depth Stream.
   - https://developers.binance.com/docs/binance-spot-api-docs/web-socket-streams#diff-depth-stream.
   - Process incoming data on multiple symbols, using appropriate data structures to maintain all orderbook levels in memory. Use the standard library JSON package (V1) to perform the unmarshalling.
   - On each update, print the top level prices to stdout in the following format:
   - "{"symbol":"BTCUSDT","ask":100001.00,"bid":100000.00 "ts":1672515782136}".
2. Propose alternate strategies for the unmarshalling step above in order to improve performance.
   - Your strategies should consider: heap allocations and speed.
   - Provide verification via testing and benchmarks.

Submission

- All work must be completed in a GitHub repository.
- Submission is completed by sharing the repository link.
- Task deadline is 1 week from reciept of this document.
- Do not hesitate to ask for extra time if needed.

Deliverables

- Go source code implementing the tasks.
- Dockerfile for containerization.
- Setup and execution instructions.
- Explanations of development decisions.
- Details of any limitations.