Event Statistics API

Welcome to your take-home assignment! This project involves implementing a simple but thoughtful REST API. It is meant to test your ability to design, implement, and test a small service with real-world concerns like time filtering, concurrency, and correctness. Please reach out with any follow-up questions if anything is unclear, otherwise, document your assumptions. The exercise shouldn't take more than 10 hours.

Task Overview

You will implement a RESTful API with two endpoints:

1. `POST /event` — Accepts timestamped float values and stores them in memory.
2. `GET /statistics` — Returns aggregate statistics (min, max, avg, count) over events from the **last 1 hour**.

API Requirements

`POST /event` Accepts a JSON payload:

```
{
  "timestamp": "2025-06-26T14:30:00Z",
  "value": 12.34
}
```

- `timestamp`: UTC ISO 8601 format (e.g., with Z suffix).
- `value`: a float
- Stores the event in memory.

`GET /statistics` Returns:

```
{
  "count": 4,
  "min": 1.0,
  "max": 42.0,
  "mean": 12.3
}
```

- Only events from the **last hour** should be included
- If no events are recent, return count: 0 and min, max, mean: null

  Be mindful of the time filter and ensure older events are not included

Evaluation Criteria

Your solution will be evaluated based on:

- Correctness of results
- Proper time window filtering
- Code readability and structure
- Thread safety (concurrent reads/writes)
- Unit tests and test coverage
- Documentation of any trade-offs, reasoning, assumptions, and instructions on how to execute

- Your attention to detail

Bonus: Stretch Goals (Optional)

If you have time or want to go the extra mile:

- **Docker Support:** Add a working Dockerfile and optionally docker-compose.yml
- **Advanced Stats:** Add sum, standard deviation, or value distribution

Submission

Please zip your solution with all the files necessary to execute and run the project and the relevant documentation and send the file back to us.