Rate Limiter Project - Take Home Task

Project Overview

Design and implement a rate limiter service that can be used to control the rate of traffic sent to or received from a network service. This project will test your ability to work with concurrency, design scalable systems, and write clean, maintainable code.

Task Description

Create a rate limiter that limits the number of requests a client can send to an API within a specific time window. The rate limiter should be flexible enough to handle different rate-limiting algorithms and be easily integrated into a web service.

Requirements

Functional Requirements

1. Implement a rate limiter that can limit requests based on a fixed window (e.g., 100 requests per minute).
2. The rate limiter should be able to handle multiple clients, each with their own limit.
3. Provide a way to configure the rate limit (requests per time window) for each client.
4. The service should be thread-safe and able to handle concurrent requests.

Technical Requirements

1. **Use Go** as the primary programming language.
2. Include **unit tests with at least 80%** code coverage.
3. Provide **clear documentation** on how to use the service.
4. Include a simple HTTP server that demonstrates the use of the rate limiter.
    a. You don't need to use a database for this server, using a key-value store is enough. What we want is to see the rate limiter algorithm and implementation.
5. Use appropriate error handling and logging.

Bonus Features (Optional)

- Implement a distributed rate limiter using Redis or a similar distributed cache.
- Add support for rate limiting based on custom attributes (e.g., IP address, API key).
- Implement a mechanism to handle burst traffic.

Deliverables

1. Source code for the rate limiter service.
2. Unit tests for the service.
3. A simple HTTP server demonstrating the use of the rate limiter.
4. README file with:
   - Instructions on how to run the project
   - Overview of your design decisions
   - Explanation of the rate-limiting algorithms implemented
   - Any assumptions or limitations of your implementation
5. (Optional) Docker configuration for easy setup and testing.

Guidelines

- You will have 7 days to complete this task or you can send it early.
- If you don't complete all requirements, please explain what you would do with more time.
- Feel free to use any external libraries, but be prepared to explain your choices.
- If you make any assumptions, please document them in your README.
- If you have any questions, don’t hesitate to ask to andika@doitpay.co

Submission

Please submit your project as a Git repository (e.g., GitHub, GitLab). Make sure to include all necessary files and instructions for running your project. Send it to andika@doitpay.co.

Good luck, and we look forward to reviewing your implementation!