Reverse Polish Notation Service

Pendar's internship final project

September 2024

1 Introduction

Suppose you have a service exposed to clients that gets an arithmetic expression string formatted with reverse polish notation and returns the computed output of it. The following contraints holds true for the input of the service:

- The valid operators are '+', '-', '*', and '/'.
- Each operand may be an integer or another expression.
- The division between two integers always truncates toward zero.
- There will not be any division by zero.
- The input represents a valid arithmetic expression in a reverse polish notation.
- The answer and all the intermediate calculations can be represented in a 32-bit integer.

Note that your service MUST be authenticated and not everyone should be able to request to your service. To make this happen, you need to have a signup endpoint where users provide a unique username and a password and have their user registered on the service. Also, there is a login method in which users provide the username/password and get back a JWT token, which is mandatory to call reverse polish notation service endpoint.
Additionally, users' requests to the service MUST be rate limited. Each user must have a subscription to the service that indicates the total number of requests it can make and the expiry date of subscription. If the total number of user requests exceeds the subscription's hard cap or the requests are made after expiry date, the requests must be get rejected. Each user can have at most one ACTIVE subscription at any given point, meaning you can issue a new subscription for a user only when that user doesn't have any unexpired subscription with remaining requests.
Subscription issuance and renewal happens through a dedicated endpoint available to only an admin account, which its username/password defined through the service configuration. The admin account must login to the service all the same. The admin should state the username for which it requesting a subscription and the capacity and expiry date of the subscription.

2 Requirements

The following requirements MUST be respected:

- The service must be implemented in Go
- The state management must be through Postgres
- The project must be fully Dockerised and ready to ship
- The project must have a CI pipeline via Github Actions

Have Fun Coding :)