K8s-like API Server

Snapp Team

November 22, 2022

Contents

1 Introduction

Design and implement a service with Golang programming language to work like a minimal k8s API server.

Successful HTTP call is indicated by an HTTP OK *2XX* status code and the failed call is indicated by a *non-2XX* status code. So stick with this standard during your development.

2 Database

Feel free to use any kind of database you want (e.g. MongoDB, etcd, etc.).

3 RESTFul API

3.1 Authentication

Use JWT Tokens for authenticating users. Feel free to decide on JSON structure of the token.

3.2 Users

This application needs simple user management and authentication so that users can register and receive a token in order to be able to use other endpoints to create containers, update containers, get the stats of the containers, etc.

3.2.1 Users Endpoints

- Create a new user (public endpoint)
- Generate Token for the user (auth-required)

3.3 Authorization

There should be a mapping between users and the resource and verbs which a user have access to. For example, user *x* has access to resource *y* with verbs *get*, *create*, *update* and *delete* in namespace *z*. There should be a default *admin* user which **only** this user can register new rbacs. The rbac itself is not required to have an rbac.

- Register (CRUD) a new rbac (auth-reuquired, only admin)

3.4 Container

Each User must be able to create, update, delete and get container definitions as below:

```yaml
apiVersion: v1
kind: Container
metadata:
  labels: map[string]string (optional)
  name: string (required)
  namespace: string (required)
spec:
  name: string (required)
  image: string (required)
  args: []string (optional)
  env: map[string]string (optional)
  ports: []int (optional)
```

Validation:

- The container spec should be validated.
- container name should be unique in each namespace.
- User access should be checked in each request with his/her JWT token.

API:

- Create/Update/Delete a Container (auth-required)
- List Containers by namespace and user (auth-required)

3.5 Quota

Each user have a quota for container count and port count, and user can not create more containers than the quota, or request more total count ports in all created containers above the limit.

The quota for each user can be set only by *admin* user, so it does not require any *rbac*.

- Set quota for user (auth-required, admin only)

4 Instructions

Please create a GitHub repository in snappcloud-learning organization and share it with us as soon as you started working on this assignment.

- Tell us about the Trade-Offs and the important decisions which you made during the design and development.
- Provide docker-compose file so we can easily run and see how your application works
- Use concise and good commit messages
- Design the API with RESTful standards
- We don't need 100% test coverage but do your best to write some tests for important parts of the application
- Write some documentation about your service in a file called README.md

Good Luck!