Golang API Challenge

This is where you get to show us what you can do. Write us a Golang API for GCP that fulfils the spec below and meets the criteria listed.

We know tech tests have a habit of taking a whole weekend, but that's not necessary here. You should spend a couple of hours on this (that's what we're expecting). Of course, if you want to spend a weekend and build The World's Greatest Tech Test we'd be thrilled to review it.

We also use this same test for all of our developers - junior, senior, principal etc - and we'll be keeping the level of the role in mind when reviewing. So junior devs will get a little more slack, while principal devs should come prepared to make us wonder if we're even qualified to review your work.

Upload your solution to a Git repository. Document your solution and notes on how to run it in a README.md markdown file. Use Github projects to plan your work. Create issues, break them into tasks and track relationships, making sure to use git appropriately. Feel free to modify and refactor the existing code to best achieve your solution.

Objective:

Create a Golang API hosted on Google Cloud Platform (GCP) that meets the provided specifications. The API should allow users to interact with cake data, including listing all cakes, searching for cakes by yumFactor and/or name, adding a new cake, and deleting an existing cake.

Specifications:

- The API should return an OpenAPI or Swagger spec.
- Users should be able to fetch a cake by ID.
- Users should be able to list all cakes.
- Users should be able to search for a cake by yumFactor and/or name.
- Users should be able to add another cake.
- Users should be able to delete an existing cake.

Cake Data Structure:

id: <number> (required)
name: <string> (required, max 30 characters)
comment: <string> (required, max 200 characters)
imageUrl: <string> (required)
yumFactor: <number> (required, value between 1 and 5 inclusive)

Deployment Criteria:

- Utilise Google Cloud (GCP) for deployment.
- Use Google Cloud Functions for serverless execution.
- Implement Google Cloud Build for CI/CD pipeline.
- Store cake data in Google Datastore or Firestore.
- Optionally, integrate Memcached or Redis for caching.
- Ensure the API can be easily downloaded and run with a simple git clone.
- Implement authentication for added security.

Implementation Approach:

HTTP Server: Utilise the standard **net/http** package to implement the HTTP server. This package provides all the necessary functionalities for creating HTTP endpoints, handling requests, and managing responses without the need for additional frameworks.

Routing: Use the **ServeMux** from the **net/http** package to define routes and handle different API endpoints. The ServeMux allows for straightforward routing without introducing unnecessary complexity.

Middleware: Leverage the built-in middleware capabilities of the net/http package for implementing **authentication** and other cross-cutting concerns. Middleware functions can be seamlessly integrated into the request-response lifecycle.

Error Handling: Implement error handling using idiomatic Go practices. Return appropriate HTTP status codes along with informative error messages to clients.