Backend Engineer Take-home Challenge

As part of the hiring process for backend engineering at Oden, we like to assess a candidate's technical abilities by challenging them to build something that will demonstrate how well they handle vague requirements, approach problems, and produce code.

Guidelines

Before we dive into the problem, let's discuss some guidelines.

You don't need to "finish" the project.
This is an open-ended challenge and we expect it to take around two hours to complete, but we don't expect you to spend more than four hours on it. The purpose of this exercise is not for you to write a perfect solution, but for us to see what decisions you make, to get a feel for your code composition, and to find some topics for discussion later in the interview process. So even if you don't finish, you should make sure that the path to completion -- the steps you *would* take if you had more time -- is clear, either in your code or in the README.

Make it easy on yourself.
The Oden backend team is predominantly a Go, Python, and Java shop. That being said, you may use any language and libraries you believe suit the problem. We suggest you use whatever you're most familiar with.

Make it easy on us.
However, please avoid any libraries that generate large amounts of code, as this will make it harder for us to review your project. Also, keep data local to your service; relying on a third party database to do the hard work of this assignment won't give us a very good idea of your ability to write code.

Treat this like a real project.
Treat this project as if it will eventually go into production and your co-workers will have to maintain it. Please document all components just as you would with a normal project.

Assessment

We will be reviewing your submission for:

- Code structure and modularity
- Code syntax and readability
- Runtime performance
- App usability
- Output correctness

Challenge

At Oden we deal with a lot of time-series data. We process millions of IoT metrics every day and have to make sure our end users can visualize the data in a timely manner. One common use case for our users is to monitor metrics that are indicators of product quality in real-time. For our cable extrusion customers, this happens to be the cable diameter.

http://takehome-backend.oden.network/?metric=cable-diameter

The API returns the current value for the metric.

Your task is to assemble an application that polls the Oden API, calculates a one minute moving average of the Cable Diameter, and exposes that moving average via an HTTP service when issued a GET request to localhost:8080/cable-diameter.

Example:

```
curl localhost:8080/cable-diameter
11.24
```

Your moving average should be updated, if possible, once per second and, after each update, the new moving average is logged to STDOUT.

Submission

We ask that you return your solution to us within 1 week of receiving this prompt by emailing us a zip of your working directory. We understand that everyone has different schedules and commitments; if you need more time, just let us know and we'll try to accommodate you.

We review takehomes anonymously to avoid any implicit bias. **Please make sure your name does not appear in your code or Readme.**

Along with your solution, please include answers to the following questions in your README:

1. How should we run your solution?
2. How long did you spend on the take home? What would you add to your solution if you had more time and expected it to be used in a production setting?
3. If you used any libraries not in the language's standard library, why did you use them?
4. If you have any feedback, feel free to share your thoughts!

*Please be sure **not** to upload your solution to a public repository!* We love public examples that let your code shine, but we want to make sure your solution remains uniquely your own.

We look forward to reviewing your submission!