Remote Assignment V3

Thank you for your interest in Gawkbox and congratulations for having been invited through to the second round as a Software Engineer on the Platform team!

Objectives

This assignment is designed to:

- Test your ability to read, understand & engage in product specification
- Understand how you learn and engineer a technical solution given specific requirements
- Evaluate your reasoning and implementation of technical product solutions

Specification

Overview:

Gawkbox is the platform content creators and live streamers use to monetize.

As a software engineer at Gawkbox, some of your main responsibilities are to understand how APIs work, and then build systems using those APIs. Sometimes those APIs are provided by 3rd parties, but most times we end up building our own.

Gawkbox is highly integrated with Twitch and uses the Twitch API to fetch stream data in order to further engage streamers and their fans. We would like to be able to query the API in order to get a particular streamer's channel stats.

Technical Requirements:

Attached to this e-mail is a .zip directory structure with a main.go as a starting point for your application.

In order to accomplish the aforementioned mission we need you to build an HTTP Server as an API with various endpoints where the request contains the streamer's username and returns an HTTP response. We would like to access this API to receive the following information:

- user's channel's **# of views**
- user's channel's **# of followers**

- user's channel's **game**
- user's channel's **language**
- if the user is **currently streaming**
- user's **display name**
- user's **bio**
- user's **account creation date**

Upon completion please provide a link to your GitHub repository with the solution.

Evaluation

A successful solution will comprise the following:

- An HTTP Server written in **Golang** as the deliverable
- **your own** Twitch API integration package (ie. without the use of a 3rd party package)
- HTTP routing using the Golang standard library's **net/http** package (no frameworks please!)
- a **README** containing a "How-To" section describing how to build & run your server with a sample request/response flow

****Bonus:** Build your application as a docker container and provide instructions to spin it up locally.

Please feel free to reach out if you have any questions!