ForgeRock

Cloud SRE/DevOps Challenge

This is a simple exercise that should take you no more than 4-5 hours to complete. This is intended to be fun. If you enjoy this exercise, you will like working with the Cloud teams at ForgeRock.

This exercise should be achievable no matter whether your background is more on the software engineering or operations end of the SRE spectrum. Part 1 is more software engineering focused whereas Part 2 is more operational - please make an attempt on both parts, but if you find one part taking too much of your time, play to your strengths.

If you have any questions feel free to reach out to us for assistance.

Part 1: Stock Ticker

Write a web service that looks up a fixed number of closing prices of a specific stock. We prefer golang but if you are more comfortable in another language, we are also polyglot and will accept a language you are most proficient with.

Guidance:

- In response to a GET request, the service should return the last `NDAYS` days of data along with the average closing price over those days. The structure of the response is up to you.
- The stock SYMBOL (the symbol to look up) and `NDAYS` (the number of days) are environment variables provided to your program.
- Use this free quote service:
  - Sample query: https://www.alphavantage.co/query?apikey=C227WD9W3LUVKVV9&function=TIME_SERIES_DAILY&symbol=MSFT
  - **Note**: You should be able to use the `apikey="C227WD9W3LUVKVV9"`, but you may need to create your own API key if that one has expired.
  - The API has a quota per key, so you will need to bear this in mind.
- Create a docker image that runs your web service.
- Publish your docker image, your code, and provide instructions on how to build the image and run it.
- Code should exhibit good hygiene. If you are running short of time you can demonstrate intent without being exhaustive.

Part 2: Kubernetes

Create a Kubernetes manifest that deploys your web service, creates a service for it, and exposes it as an ingress.

Use a configmap to pass in all environment variables. For the exercise use `SYMBOL=MSFT` and `NDAYS=7`

- Use a secret to pass in the api key `APIKEY=C227WD9W3LUVKVV9`
- Publish your manifests, etc. to git (github, bitbucket, gitlab, etc) , and send us the link along with instructions on how to deploy it.

The sample provided should run on a vanilla Kubernetes environment (minikube, for example).

Part 3: Resilience

Think about the resilience of the application you have written. What would it take to support it in production? We do not necessarily expect you to have time left to actually implement changes for this, but if you do, great.