Operability Take Home Exercise

Welcome to the start of our recruitment process for Operability Engineers. It was great to speak to you today regarding an opportunity to join the Equal Experts network!

Please write code to deliver a solution to **only one** of the 3 problems outlined below. We have provided 3 possible problems for you to focus on, please select **only one** to complete. We appreciate that your time is valuable and do not expect any of these exercises to take **any more than 90 minutes**.

If you think these will take longer than that, I **strongly** encourage you to please get in touch with me to ask any clarifying questions.

Submission guidelines

Do

Provide a README file in text or markdown format that provides a concise way to setup and run the provided solution.

Take the time to read any applicable API or service docs, it may save you significant effort

Make your solution simple and clear. We aren't looking for overly complex ways to solve the problem since in our experience simple, clear solutions to problems are also generally the most maintainable and extensible solutions.

Provide your solution in a zip or compressed tar archive.

Don't

Expect the reviewer to dedicate a machine for the purpose of reviewing the test by

- Installing software globally that **may** conflict with system software
- Requiring changes to system wide configurations

Provide overly complex solutions that need to spin up a ton of unneeded supporting dependencies. We aspire to keep our dev experiences as simple as possible (but no simpler)!

Include identifying information in your submission. We are endeavouring to make our review process anonymous.

Exercises

These problems are not particularly difficult but do vary. It’s up to you which **one** you choose to spend your time on. Please don't submit a solution for more than one exercise.

If you have any questions we encourage you to get in touch and we'll answer as soon as possible.

Github API

Write a script that uses the Github API to query a user's publicly available gists. When the script is first run, it should display a listing of all the user's publicly available gists. On subsequent runs the script should list any gists that have been published since the last run. The script may optionally provide other functionality (possibly via additional command line flags) but the above functionality **must** be implemented.

Minikube

Write a simple hello world application in any **one** of these languages: Python, Ruby, Go. Build the application within a Docker container and then load balance the application within minikube. You are not required to automate the installation of minikube on the host machine.

Vagrant CI

Using a configuration management language of your choosing (Ansible, Puppet, Salt, Chef) setup a Continuous Integration server of your choosing on a Vagrant Box. You are not required to setup Vagrant or its requirements on the host machine.

Best of luck,

Equal Experts