Liatrio Apprenticeship Interview Exercise

Overview

Welcome! The goal of this interview is to get an idea of how well you will fit into our practice, and if the way that we work is the right fit for you.

First, let's be clear - a successful interview does NOT mean that you finish the exercise without help, or even necessarily finish it. We are going to see how you identify and solve problems, how you communicate, how and where you ask for help, and finally how you demonstrate what you did.

Some of the items of the interview are going to be purposely vague for 2 reasons:

1. To see how well you can figure things out on your own
2. How you search, find, and ask for help

A perfect way to ask for help would be something like: "I am struggling with doing XXX. I found this issue (YYY), and researched a bit about ZZZ. Can you point me in the right direction?" Whenever possible include error messages or screen shots to give context to what you are asking about.

You are free to use any information needed to work this out. We all use Google, Stack Overflow, and our friend sitting right next to us every day. It is critically important that you understand what you are using and doing. We will expect to hear about how you accomplished these tasks and what resources you needed at demonstration time.

If you are stuck, or if something is unclear, communicate this in the Slack channel.

Remember, at Liatrio, **communication, collaboration, transparency and sharing** are key pillars of our work culture.

Interview Exercise

For this interview, we are going to ask you to create a simple web application which exposes a HTTP API endpoint and use a CI/CD pipeline to containerize and deploy it to a cloud platform.

We have this exercise broken down into general sections, and while there are many ways to accomplish each, we note some tools you should use. It is possible that you may have experience using other tools, but you should stick to our suggestions when and where possible.

As noted, **communication is key**, so please ask questions and provide updates as you make progress.

GitHub Repository

- Create a public GitHub repository to store the code and files for your application. You'll also be using this repository to host a GitHub Actions workflow later.
- Share this repository in Slack and push up changes frequently so we can keep track of your progress.

Golang Application

- Install Golang and use Fiber to build a simple single endpoint web application that returns the following JSON object. The blank space should be your name, and the timestamp should be dynamically generated. The API should return a minified JSON object, the following example is formatted for readability.

```
{
      “message”: “My name is ___”,
      “timestamp”: 12312344
}
```

Docker Containerization

- Write a Dockerfile that will build your application
- Make sure the Dockerfile is hosted in your repository

GitHub Actions

- Create GitHub Actions Workflow(s) that:
    - Builds your application's Docker image
    - Verifies the application functionality using Liatrio's GitHub apprentice-action
    - Pushes your image to an OCI compliant repository (Docker Hub as an example)
    - Uniquely version your image each successful workflow run

Cloud Deployment

- Deploy your app to a cloud platform of your choice (AWS, Google Cloud Platform,

Azure) using the image from Docker Hub.
  - Note: Many cloud platforms have student credits for educational purposes. Use the free credits so that you aren't paying to deploy your application. Also, you only have to run your application when testing or demoing.

Deployment Workflow (extra credit)

- Add a GitHub Workflow which automatically deploys your versioned application to the cloud platform when changes are made to the main branch of your repository.
  - Ensure you are deploying your versioned image from your OCI repository
- Add a field to the JSON output of your application and verify the change is deployed

Wrap Up, Demo, Follow-Up

Once you have completed all of the items above, please communicate with us in Slack. We will want you to conduct a demonstration where you will present what you did, what you learned, what you may have struggled with, and any additional ideas or items that you may think of. The demo will be given over a video conference platform with a screen share for our folks who cannot make it to the Chico office. You will be giving this demo in the Chico office unless you cannot present in-person. One of our consultants will meet with you prior to the demo to go over expectations and help you prepare.

Demo Fundamentals

Below is a write-up from our employee hand-book of how we want Liatrio consultants/engineers to perform a demo; please use this as a guideline as a framework for your demo.

Liatrio Demo: Expectations

Demos are a very essential and critical part of the Liatrio way. We firmly believe that at any given time, all of the work should be in a working state (see definition below) and that you should be confident to conduct a demo.

Liatrio teams normally do demo's frequently to their respective team/account members ranging from quick 5 mins to 15 mins maximum.

At Liatrio, 'working state' means the work is at a stage where it's functional and can be shown off or tested. It's not about being finished or perfect; it's about progress that can be demonstrated. The point is to ensure continuous progress and the ability to gather feedback early and often.

Demo Prep:

Prepping for a demo is a very essential trait and it's never a good idea to come to a demo without having a prep session prior. This is not intensive but definitely deliberate. Sometimes the prep can be for as little as 5 mins but it essentially sets the tone for what you are about to present.

1. Demo prep involves taking the time to be ready for the demo. Be in a good place to share screen; log into the meeting a few minutes prior, close out all of the non-essential applications and notifications, and mentally have a checklist of the flow of the demo.

2. Make sure you write up something - a few bullets of what it is that you will demo. Not having a context for your audience is a sure way to render it useless.

3. Make sure you have some way to take notes.

4. Plan for all of the windows/screens you will need. Doing something that others cannot see is a waste of time.

5. Plan to show everything. The goal of the demo is to show how it's done - not what the end product is.

Actual Demo:

1. Start with a quick introduction of the demo - “Hi, my demo today will showcase how to stand-up a Golang application in a Docker container and both build and deploy the application using Github Actions”. Concise and precise.

2. Share your screen preferably with something that outlines the flow of what you are going to show. A task list, a Trello board, an Evernote list, or anything that can tell whoever is seeing what to expect is helpful.

3. Start the demo with a clean slate and showcase a slow/step-by-step execution. Cater to the lowest possible denominator in your audience - non-technical business folks who would not have a context but can follow a logic.

4. Speak slowly, clearly, and using language that is understandable to a layman. Don’t shift around the screen too much as it confuses folks. The goal of a demo is not to showcase your technical mastery of the work you have done but to make your audience understand the work you have completed and get a sense of what it is.

5. If your time allotment is 15 mins; plan to introduce for 2 mins, demo for 8 mins, and leave time for questions. This is very important as time management is a critical aspect of good meeting etiquette.