Home Assignment: SRE Engineer

Hello,

Thank you for your interest in the SRE Engineer position with Opus Major!

The goal of this home assignment is to assess your ability to design and deploy a simple, production-like environment using modern DevOps and SRE practices. We are also evaluating your understanding of the requirements and your ability to prioritize the aspects most relevant to the role. If anything is unclear, please don't hesitate to ask questions.

Objectives

Your company operates a simple API that stores and retrieves player data for a multiplayer game.

You've been tasked with deploying a minimal version of this service to a Kubernetes cluster using Infrastructure as Code (IaC).

The project should be capable of running in-cluster with ArgoCD for deployment management.

You may use Minikube for local testing. While ArgoCD must be installed for deployment, its installation and configuration do not need to be included in the project or documentation.

Deliverable

Your project should include the following components:

- **Golang Service:**

  A very simple Go application exposing a single HTTP GET endpoint at /player-data The API returns static data, this is just a dummy endpoint that we will expose later.

- **Containerization:**

  The service must be containerized using Docker. Include a Dockerfile in the repository and publish the built image to a public docker repository.

- **Deployment with ArgoCD:**

  Create an **ArgoCD Application** that deploys the service using **Kustomize**.

- **Service Exposure:**

  Expose the application so it can be accessed locally after deployment via ArgoCD. You may use a **Kubernetes Service**, **Ingress**, or **Gateway**.

- **Documentation**

  Write a concise README.md (max 1 page) explaining:
  - The overall architecture
  - How to deploy everything
  - Any trade-offs or assumptions due to time limits

Criteria (How we'll evaluate)

Your submission will be evaluated based on the following criteria:

- Reasoning and Prioritization

Understanding the requirements, identifying and determining what the priorities are and spending the time on topics that have the most impact.

- Functionality

The service should build, deploy, and run successfully on a Kubernetes cluster. The /update-player-data endpoint should respond correctly when accessed.

- Infrastructure as Code (IaC) Quality

Use of declarative configuration and Kustomize best practices will be assessed. We'll look for clear structure, reusability, and adherence to Kubernetes conventions.

- ArgoCD Integration

The ArgoCD Application should correctly deploy and manage the service. We'll verify that synchronization, health status, and deployment automation work as expected.

- Documentation & Clarity

Instructions for building, deploying, and testing the application locally should be clear and concise. A short README explaining how to run the project will be expected.

- Bonus Points

Additional attention to observability (e.g., basic logging, health checks) or good use of Kubernetes patterns will be viewed positively but are not mandatory.

Duration

- You have five days, excluding weekends and holidays, to complete this assignment. Please inform us of any personal commitments that may affect your ability to meet this deadline.

- Estimated time: 2–4 hours. Please do not exceed 4 hours. We respect your time. If you hit time limits, include a short note on what you'd do next and why in your README file.

Submission Instructions

Create a GitHub repository and share the link.

Final step

Email your submission link or file to sylvain@opusmajor.io and jared@opusmajor.io , cc careers@opusmajor.io

Non-Usage Disclaimer

We assure you that the content created within this assignment will not be used by our organization for internal or commercial purposes without your explicit written agreement.

OPUS MAJOR
SAS au capital de 3.091,55 €

Registered Office:
15 rue des Halles
75001 PARIS FRANCE

N° Immatriculation RCS Paris :
922 671 268