Nxtgenhub Engineer Coding Challenge

Summary

This coding challenge is presented to candidates applying for a DevOps Engineer position at nxtgenhub. It is intended to be completed before a team interview, allowing the candidate to work at their own pace and present the solution during the interview.

Expected Results

The candidate will complete the challenge described below within 2-4 hours and provide a link to a GitHub repository containing their solution. During the interview, the candidate will walk through each file in the repository, explaining how each contributes to the solution and discussing design choices, challenges faced, and any trade-offs made.

Challenge

**Objective**

Deploy a Web Server that prints “Hello, World!” using the specified technologies and best practices.

**Technologies Required**

* ● **Kubernetes**: Use a Kubernetes cluster for deployment.
* ● **Helm**: Utilize Helm for managing the Kubernetes manifests.
* ● **Ingress Controller**: Preferably use Traefik, but other options are acceptable.
* ● **LoadBalancer Service**: Ensure the Ingress controller Service is of type  
     
  LoadBalancer.  
     
  **Requirements**    
  1. **KubernetesClusterSetup**

● Deploy a Kubernetes cluster. You may use any local setup like Minikube, Kind, or Docker Desktop, or a cloud provider like GKE, AKS, or EKS.

2. **WebServerDeployment**

* ● Deploy a web server that serves the text “Hello, World!”. You can use an  
     
  existing Docker image or create your own simple web server image.
* ● Create a Helm chart for deploying the web server.  
     
  3. **IngressConfiguration**

* ● Configure an Ingress resource to route traffic from **http://localhost:80** (or  
     
  a custom domain name of your choice) to your web server.
* ● Ensure the Ingress controller Service is of type LoadBalancer.
* ● The Ingress controller should be part of the Helm chart deployment, either  
     
  as a Helm dependency or included in the templates. 4. **RunbookCreation**

● Create a runbook outlining the steps to administer and maintain the "Hello, World!" web server. This should include common operational tasks such as scaling, logging, monitoring, and troubleshooting.

**Bonus Points (Optional)**

If you have extra time and wish to further demonstrate your skills, consider implementing any of the following for additional credit. These are not required and are purely optional.

● **Prometheus Metrics**

* ● Stand up a Prometheus instance in the Kubernetes cluster.
* ● Introduce Prometheus metrics for the web server and/or Ingress controller.  
     
  Consider using The Four Golden Signals.
* ● Reference these metrics in your runbook as a way to troubleshoot or  
     
  identify issues.  
     
  **● Alerting**

* ● Stand up an Alertmanager instance in the Kubernetes cluster.
* ● Create alerts for the Prometheus metrics to proactively monitor the health  
     
  and performance of the application.
* ● Include alerting runbooks to guide on-call engineers in troubleshooting  
     
  and resolving issues.

● **Security Enhancements**

* ● Make the web server secure with a certificate (HTTPS).
* ● Utilize a service like cert-manager to automate the process of creating  
     
  certificates for secure communication for the web server.

**Deliverables**

● A GitHub repository containing:

* ● Helm chart that includes the web server and ingress-controller.
* ● Dockerfile (if you create your own web server image).
* ● Runbook documentation.
* ● Any additional scripts or configuration files.  
     
  Discussion Points for the Interview

* ● **Architecture and Design Choices** 
  + ● Explain the architecture and design choices you made.
  + ● Justify your choice of Ingress controller and LoadBalancer setup.
* ● **Security and Reliability** 
  + ● Discuss the security measures you implemented.
  + ● Explain how you ensured the reliability and scalability of the web server.
* ● **Production-readiness** 
  + ● Describe the steps you would take to make the application and its backing  
       
    infrastructure production-ready.
  + ● Identify the biggest vulnerabilities in the current infrastructure and suggest  
       
    mitigation strategies.
  + ● Discuss potential improvements if you had more time, such as CI/CD  
       
    integration, advanced monitoring, automated scaling, and enhanced security measures.  
       
    Submission Instructions

* ● Complete the challenge within 2-4 hours.
* ● Provide a link to the GitHub repository containing your solution.
* ● Be prepared to share your screen and walk through your solution during the  
     
  interview