Architecture Assignment

Xin Yan

May 4, 2023

1 Introduction

This document presents a solution for the scenario outlined in the assignment. To achieve this, the rest of the document is structured as follows: Section 2 provides an overview of two different architectures designed during various project phases. In Section 3, I illustrate how to ensure security in the proposed solution, while Section 4 outlines the necessary tasks that should be included in the CICD pipeline.

2 Architecture

2.1 Initial Version

For a brand-new project, the main concern is meeting the basic requirements and ensuring the architecture is simple enough. Therefore, the initial version I designed is shown in Figure 1.

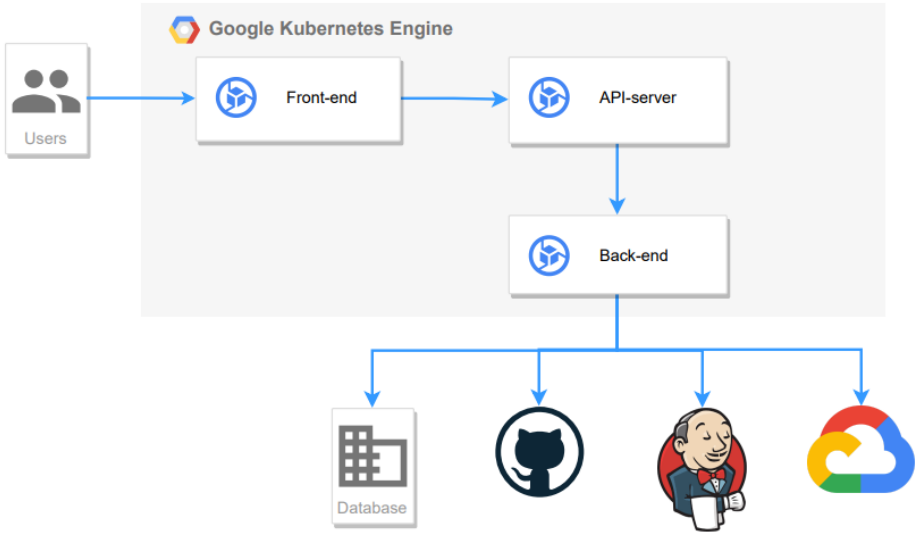

Figure 1: Initial version

Here are the functionalities of each component:

- Front-end: provides a service for user interaction.
- API-server: exposes the necessary APIs, authenticates the API's caller, and audits the behaviour of the API's caller.
- Back-end:
  1. create GitHub repository and GCP project
  2. store relevant information in a database
  3. triggered by GitHub and call Jenkins to run the CI/CD pipeline.

2.2 Future Version

Thanks to the simplicity of the initial version, we can conveniently test and monitor the solution to ensure that it meets the necessary basic requirements. Once we've established that the system is functioning correctly, we can consider decoupling the functionality to make it more extensible. Figure 2 shows the updated version of the architecture, which consists of several components that work independently to achieve the desired functionality. This design enables us to add new features and scale the system more efficiently while maintaining its reliability and performance.

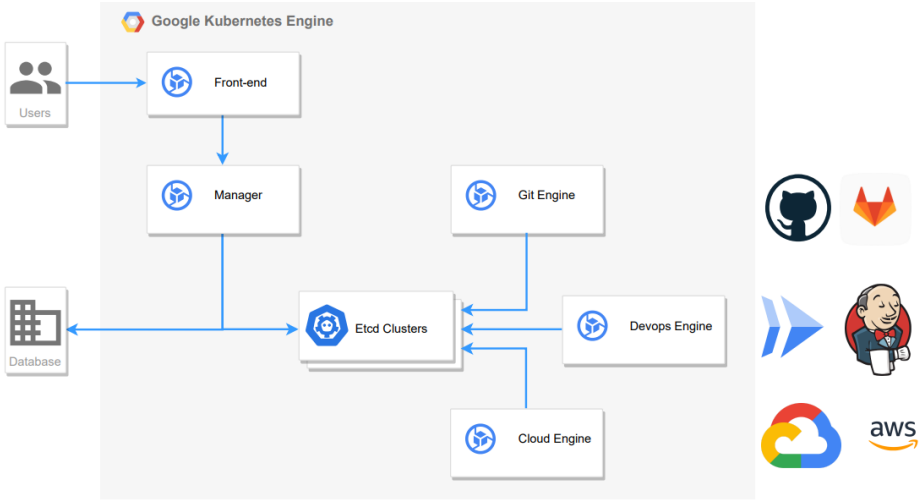

Figure 2: More extensible version

In this version of the architecture (shown in Figure 2), the Manager component is responsible for recording the user's request directly into both the database and Etcd and responding to the user immediately. The three Engines, on the other hand, continuously watch the Etcd node for any changes. When a change is detected, each Engine performs its respective task and records the result back into the Etcd. Additionally, the Manager component watches the Etcd and updates the database accordingly, ensuring that the information in both components is consistent. Moreover, The user can access a web portal to track the progress and obtain the result of the user's request.

3 Security

To ensure the security of the resources running on GCP and achieve a zero-trust network within the Kubernetes cluster, I implement security policies in IAM and network policies, as depicted in Figure 3. Specifically, IAM policies enable us to control access to resources based on the principle of least privilege. Meanwhile, network policies ensure that only authorized traffic is allowed within the Kubernetes cluster. Additionally, I use ingress and egress gateways to ensure that there is only one entrance and one exit from the Kubernetes cluster, respectively. By implementing these security measures, we can reduce the risk of unauthorized access and potential security breaches, enhancing the overall security and reliability of the solution.

4 CICD

There are at least three jobs that should be included in CICD pipeline, just as shown in Figure 4. I should check the resources required in terraform configure file before I actually deploy it. And in the Deployment job, I run a container which uses the self-defined image to execute terraform instructions.

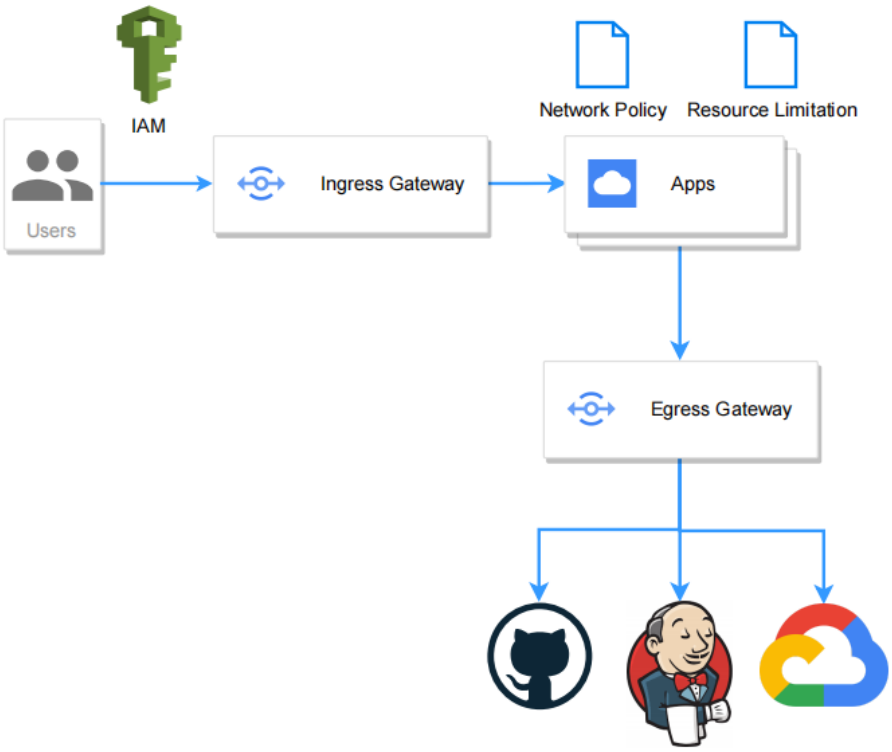

Figure 3: Security Control

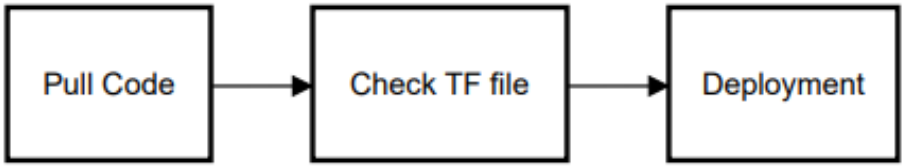

Figure 4: Security Control

5 Conclusion

In this document, I briefly introduce the architecture design and security design for the solution. And I will provide more detailed explanations during the later interview.