**Ingress inginx:**

We started by deploying an EKS cluster. Next, we deployed the ingress-nginx Helm chart. The process involved:

1. Creating a namespace specifically for the ingress controller, named ingress-controller.
2. Deploying the ingress Helm chart within this namespace.

Each Helm chart comes with a default values.yaml file, which can be modified to customize the deployment according to our needs.

**default values file:**

**Internal Loadbalancer line no 537:** we can enable the internal load balancer and add the necessary annotations to configure it properly.

**Tls Cerfificate:** we can also include a certificate annotation in the annotations section.

**servicetype-line no 552:** Here, we can change the service type from ClusterIP to LoadBalancer so that the service can be accessed from outside the cluster.

**Servicetype line no 466:** Here, we can switch the external service type from LoadBalancer to ClusterIP because we're using an internal load balancer.

**Admissionwebhook line no 728:** Disable the webhook here why because, when webhook is enabled, it automatically generate dynamic admission control on api request. It will not allow to delete or update anything related to external secrets. Webhook will locks the current configuration.

**Jenkins:**

we deployed the Jenkins Helm chart in the EKS Cluster  
 1. Creating a namespace specifically for the jenkins, named ingress-controller.

1. Deploying the Jenkins Helm chart within this namespace.

**Default values file**

**Path based routing line no 173:** For Jenkins path based routing we can add “/jenkins”

**Disable the ingress line no 654:** disable the default ingress and deploy the ingress yaml file because of pathbased routing.

**SonarQube:**

we deployed the SonarQube Helm chart in the EKS Cluster  
 1. Creating a namespace specifically for the SonarQube, named ingress-controller.

1. Deploying the SonarQube Helm chart within this namespace.

**Default Values file:**

**Path based routing:** For sonarqube path based routing Update these values at the end of the file.

extraEnv:

sonar.web.context: /SonarQube

line no-179: readinessProbe: sonarWebContext: /sonarqube/

line no-190: livenessProbe: sonarWebContext: /sonarqube/

line no-201: startupProbe: sonarWebContext: /sonarqube/

**Disable the ingress line no 102:** Disable the default ingress and deploy the ingress yaml file because of path based routing.

**Update the RDS section line no 438:** Here we configure the SonarQube with AWS RDS

**Disable the Defualt Database line no 454:** here we disable the default database

**JFrog:**

we deployed the Jfrog Helm chart in the EKS Cluster

1. Creating a namespace specifically for the Jfrog, named ingress-controller.

1. Deploying the Jfrog Helm chart within this namespace.

**Default values file:**

**Add keys line no 17,18:** For security reasons, we can generate the master and join keys and specify these keys in the values file.

**Enable the ingress line no 112:** Activate path-based ingress for JFrog

**ingress-nginx,enabled false line no: 1276 :**

**Service,type line no 1555 :**

**Database line no 1727:** Configure the jfrog with AWS RDS.

**Postgresql line no 1684:** Disable the Default Database