**Designing and Implementing a Microservice-based Application using Helm and Jenkins CI/CD Pipeline.**

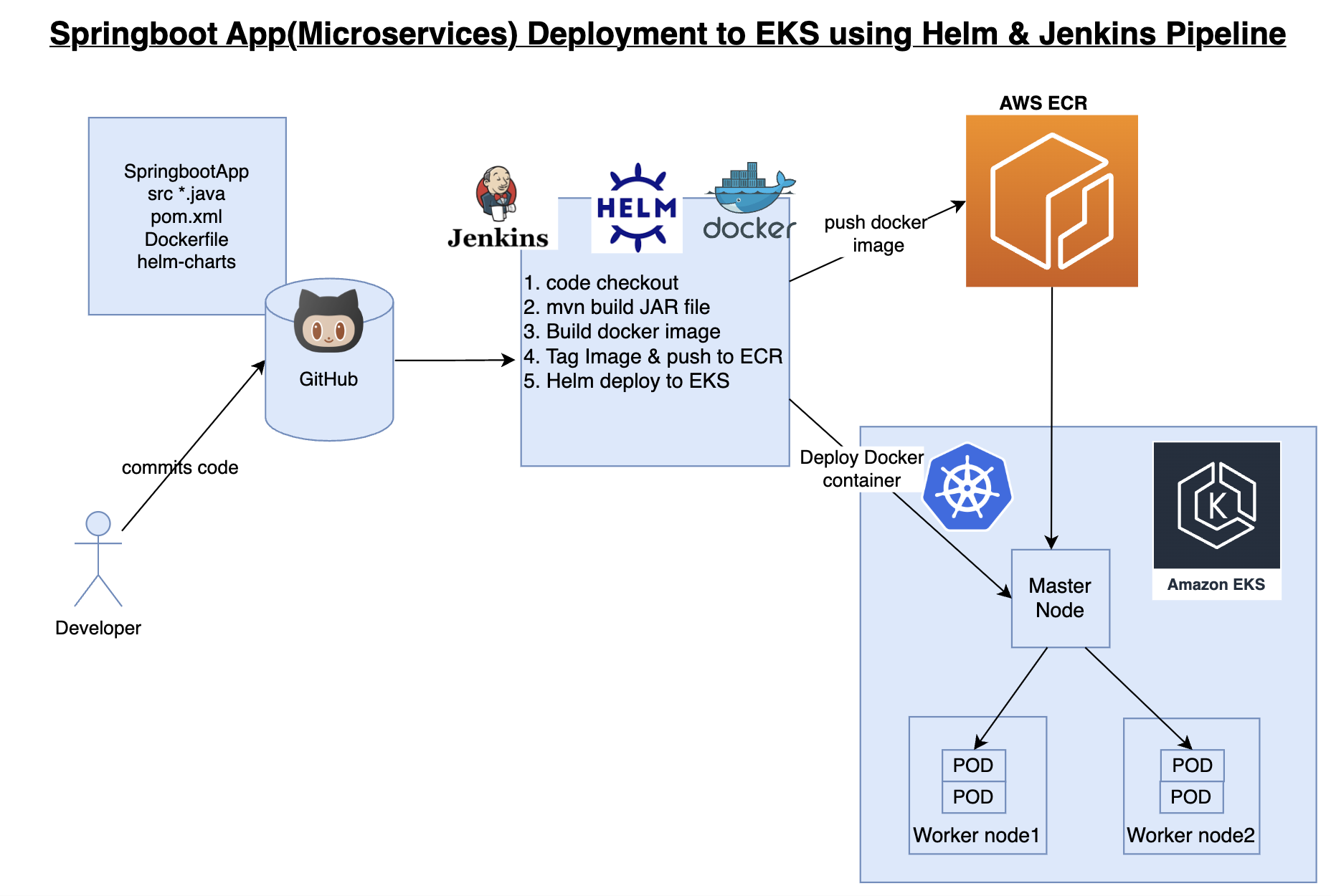
**What is Helm?**

Helm is a package manager for Kubernetes. **Helm** is the K8s equivalent of yum or apt. It accomplishes the same goals as Linux system package managers like APT or YUM: managing the installation of applications and dependencies behind the scenes and hiding the complexity from the user.

I have created a sample Springboot App setup in GitHub.

**Jenkins pipeline will:**

- Automate maven build(jar) using Jenkins  
- Automate Docker image creation  
- Automate Docker image upload into Elastic container registry(ECR)  
- Automate Springboot docker container deployments into Elastic Kubernetes Cluster using Helm charts



**Pre-requisites:**

1. EKS cluster needs to be up running. Click [here](https://www.coachdevops.com/2022/02/create-amazon-eks-cluster-by-eksctl-how.html) to learn how to create Amazon EKS cluster.

2. Jenkins instance is up and running

3. Install [AWS CLI](https://www.coachdevops.com/2020/10/install-aws-cli-version-2-on-linux-how.html) on Jenkins instance

4. Helm installed on Jenkins instance for deploying to EKS cluster

5. Install Kubectl on Jenkins instance

6. Install eksctl on Jenkins instance

7. Install Docker in Jenkins and Jenkins have proper permission to perform Docker builds

8. Make sure to Install Docker, Docker pipeline



9. Create ECR repo in AWS

10. Dockerfile created along with the application source code for springboot App.

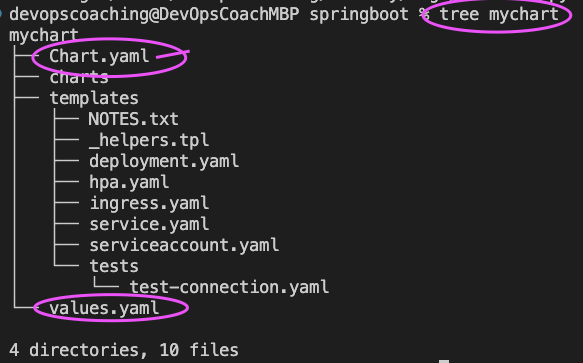
11. Namespace created in EKS cluster

**Create Helm chart using helm command**

helm create mychart

C:\Users\Bluechip\Downloads\Screenshot 2023-04-29 at 11.06.51 PM.png

tree mychart



Execute the above command to see the files created.

Add Docker image details to download from ECR before deploying to EKS cluster

Open mychart/values.yaml.

Enter service type as LoadBalancer

And also

Open mychart/templates/deployment.yaml and change containerPort to 8080

**Step 1. Create Maven3 variable under Global tool configuration in Jenkins**

Make sure you create Maven3 variable under Global tool configuration.

**Step 2. Create a namespace in EKS**

e.g kubectl create ns helm-deployment

**Step 3. Create a pipeline in Jenkins**

Create a new pipeline job.

**Step 4. Create a job pipeline code/script**

**Step 5. Build the pipeline**

**Step 6. Verify deployments in EKS**

Execute the below command to list the helm deployments:

Helm ls –n helm-deployment

Kubectl get pods –n helm-deployment

C:\Users\Bluechip\Downloads\Screenshot 2023-05-13 at 5.24.01 PM.png

**Step 7. Access Springboot App Deployed in EKS cluster**

Once deployment is successful, go to browser and enter above load balancer URL mentioned above