DevOps Certification Training

Certification Project – Medicure

Healthcare Domain

Medicure is a super specialty hospital based in New York, USA and provide world class treatment and surgery including Heart, Liver, Kidney transplants and first robotic surgery center. The chain is owned and managed by Global Health Limited.

The Medicure would centrally like to manage all the doctor's and patient's data across the Medicure hospitals in various cities. They have developed an microservice, which offers these services. In order to reduce unnecessary maintenance cost and manual labor, they would like to automate their application build and deployment process using DevOps. They are fine to use any one of the (**AWS**, **Azure**, **GCP**) cloud platform as their primary cloud service provider.

The company's primary goal is to deliver the product updates frequently to production with High quality & Reliability. They also want to accelerate software delivery speed, quality and reduce feedback time between developers and testers. They would like to use Kubernetes to manage their container deployments, scaling and descaling of containers etc.

Initially, Medicure is facing multiple problems, because of involved complexity in application as well as Infrastructure.

Following are the problems:

- ✓ Building Complex builds is difficult
- ✓ Manual efforts to test various components/modules of the project
- ✓ Incremental builds are difficult to manage, test and deploy
- ✓ Creation of infrastructure and configure it manually is very time consuming
- ✓ Continuous manual monitoring the application is quite challenging.

In order to implement a POC, you are requested to develop a mavenized microservice using spring boot and in memory h2 database.

1. a microservice which exposes below mentioned endpoints as APIs and uses in memory h2 database to store the data.

   a. /registerDoctor (HTTP Method : POST) (Request Body : JSON)
   b. /updateDoctor/{doctorRegNo} (HTTP Method : PUT) (Request Body : JSON)
   c. /searchDoctor/{doctorName} (HTTP Method : GET) (No Request Body)
   d. /deletePolicy/{doctorRegNo} (HTTP Method : DELETE) (No Request Body)

2. Write necessary Junit testcase.
3. Generate HTML report using TestNG.
4. Push your code into your GitHub Repository.

Note : Preload some data into the database.

Later, you need to implement Continuous Integration & Continuous Deployment using following tools:

- ✓ Git - For version control for tracking changes in the code files
- ✓ Jenkins - For continuous integration and continuous deployment
- ✓ Docker - For containerizing applications
- ✓ Ansible - Configuration management tools
- ✓ Selenium - For automating tests on the deployed web application
- ✓ Terraform - For creation of infrastructure.
- ✓ Kubernetes – for running containerized application in managed cluster.

This project will be about how to test the services and deploy code to dev/stage/prod etc, just on a click of button.

Business challenge/requirement

As soon as the developer pushes the updated code on the GIT master branch, the Jenkins pipeline should be triggered and code should be checkout, compiled, tested, packaged and containerized. A new test-cluster should be provisioned and configured automatically with all the required software's and as soon as the cluster is healthy and available, the application must be deployed to the test-server automatically using Kubernetes.

The deployment should then be tested using a test automation tool, and if the build is successful, it should be deployed to the prod server/cluster using Kubernetes. All this should happen automatically and should be triggered from a push to the GitHub master branch.

Kubernetes cluster must contain at least 2 servers and must be monitored continuously using Prometheus and dashboard must be visualized using Grafana.

Link for the solution of Medicure project code is attached below, You can use it to validate your solution.
https://github.com/StarAgileDevOpsTraining/star-agile-health-care.git

Note : To have a detailed information about running the application and exposed APIs, Input/Output format, Refer to the README.md in the GitHub repository.