Kubernetes for App Developers (LFD459)

Modality: Virtual Classroom

Duration: 4 Days

About this course:

The course teaches on the topic of Kubernetes for developers. Kubernetes program is a high-end open-source container-orchestration system. This program allows for a smooth computerized operation of deploying or installing, scaling and managing the applications. In this course, this is exactly what will be taught. You will get to learn on the matters of learning how to deploy, scale and manage applications using Kubernetes.

This course is designed in such a way that it automatically covers all those topics that are tested by the Cloud Native Computing Foundation Certified Kubernetes Application Developer (CKAD) Program. In this way, this course proves to be an advantageous asset, as it largely increases the chances for the student to become a certified developer.

The course will teach you on the methods of encapsulating or containerizing, hosting, configuring, and deploying applications on a multi-node cluster. You will start off with a simple Python script in which you will have to define application resources and then use the basic techniques of creating, regulating and troubleshooting the scalable applications through the usage of Kubernetes. You will be trained in and introduced to different features used to deploy an application in a production environment by using network plugins, security and cloud storage.

The learning specifics of this course are vendor-agnostic and distribution-agnostic. Hence, this will give you the margin as well as the opportunity for universal application of the learned skills.

The course will be available to you for an entire year, from the day of purchase, or in the case of preorders, availability, and will be accessible even if you finish the course early. The course hours are approximately 30-35 hours in total. The course has the feature of self-paced so you will be able to complete it at your own ease.

On average, a Kubernetes Engineer earns $143,283 per annum.

Learning Objectives:

The course has the following learning objectives:

- Containerizing or encapsulating, and deploying a new Python script
- Configuring the deployment of the application through the usage of ConfigMaps, Secrets and SecurityContexts
- Gaining comprehension of multi-container pod design
- Setting and managing of probes for measuring pod health
- Making updates and rolling back an application
- Incorporating services and NetworkPolicies

- Making use of PersistentVolumeClaims for state persistence
- And many other skills

Audience:

This course is suitable for and designed for app developers, Kubernetes engineers, and Linux developers.

Requirements:

The requirements of this course have been set in such a way that you get to fully reap the benefits of this course. Having prior knowledge of basic Linux command line and file editing skills will be beneficial for your future learning in this course. Additionally, you must be familiar with using a programming language like Python, Node.js, and Go. You should also be having knowledge of the concepts of Cloud Native application and architectures (similar to which is taught in our free introduction of Kubernetes edX MOOC).

Course Outline:

Introduction

- Objectives
- Who You Are
- The Linux Foundation
- Linux Foundation Training
- Preparing Your System
- Course Registration
- Labs

Kubernetes Architecture

- What Is Kubernetes?
- Components of Kubernetes
- Challenges
- The Borg Heritage
- Kubernetes Architecture
- Terminology
- Master Node
- Minion (Worker) Nodes
- Pods
- Services
- Controllers
- Single IP per Pod
- Networking Setup
- CNI Network Configuration File
- Pod-to-Pod Communication
- Cloud Native Computing Foundation
- Resource Recommendations

- Labs

Build

- Container Options
- Containerizing an Application
- Hosting a Local Repository
- Creating a Deployment
- Running Commands in a Container
- Multi-Container Pod
- readinessProbe
- livenessProbe
- Testing
- Labs

Design

- Traditional Applications: Considerations
- Decoupled Resources
- Transience
- Flexible Framework
- Managing Resource Usage
- Multi-Container Pods
- Sidecar Container
- Adapter Container
- Ambassador
- Points to Ponder
- Labs

Deployment Configuration

- Volumes Overview
- Introducing Volumes
- Volume Spec
- Volume Types
- Shared Volume Example
- Persistent Volumes and Claims
- Persistent Volume
- Persistent Volume Claim
- Dynamic Provisioning
- Secrets
- Using Secrets via Environment Variables
- Mounting Secrets as Volumes
- Portable Data with ConfigMaps
- Using ConfigMaps
- Deployment Configuration Status
- Scaling and Rolling Updates
- Deployment Rollbacks

- Jobs
- Labs

Security

- Security Overview
- Accessing the API
- Authentication
- Authorization
- ABAC
- RBAC
- RBAC Process Overview
- Admission Controller
- Security Contexts
- Pod Security Policies
- Network Security Policies
- Network Security Policy Example
- Default Policy Example
- Labs

Exposing Applications

- Service Types
- Services Diagram
- Service Update Pattern
- Accessing an Application with a Service
- Service without a Selector
- ClusterIP
- NodePort
- LoadBalancer
- ExternalName
- Ingress Resource
- Ingress Controller
- Labs

Troubleshooting

- Troubleshotting Overview
- Basic Troubleshooting Steps
- Ongoing (Constant) Change
- Basic Troubleshooting Flow: Pods
- Basic Troubleshooting Flow: Node and Security
- Basic Troubleshooting Flow: Agents
- Monitoring
- Logging Tools
- Monitoring Applications
- System and Agent Logs
- Conformance Testing

- More Resource
- Labs

Closing and Evaluation Survey