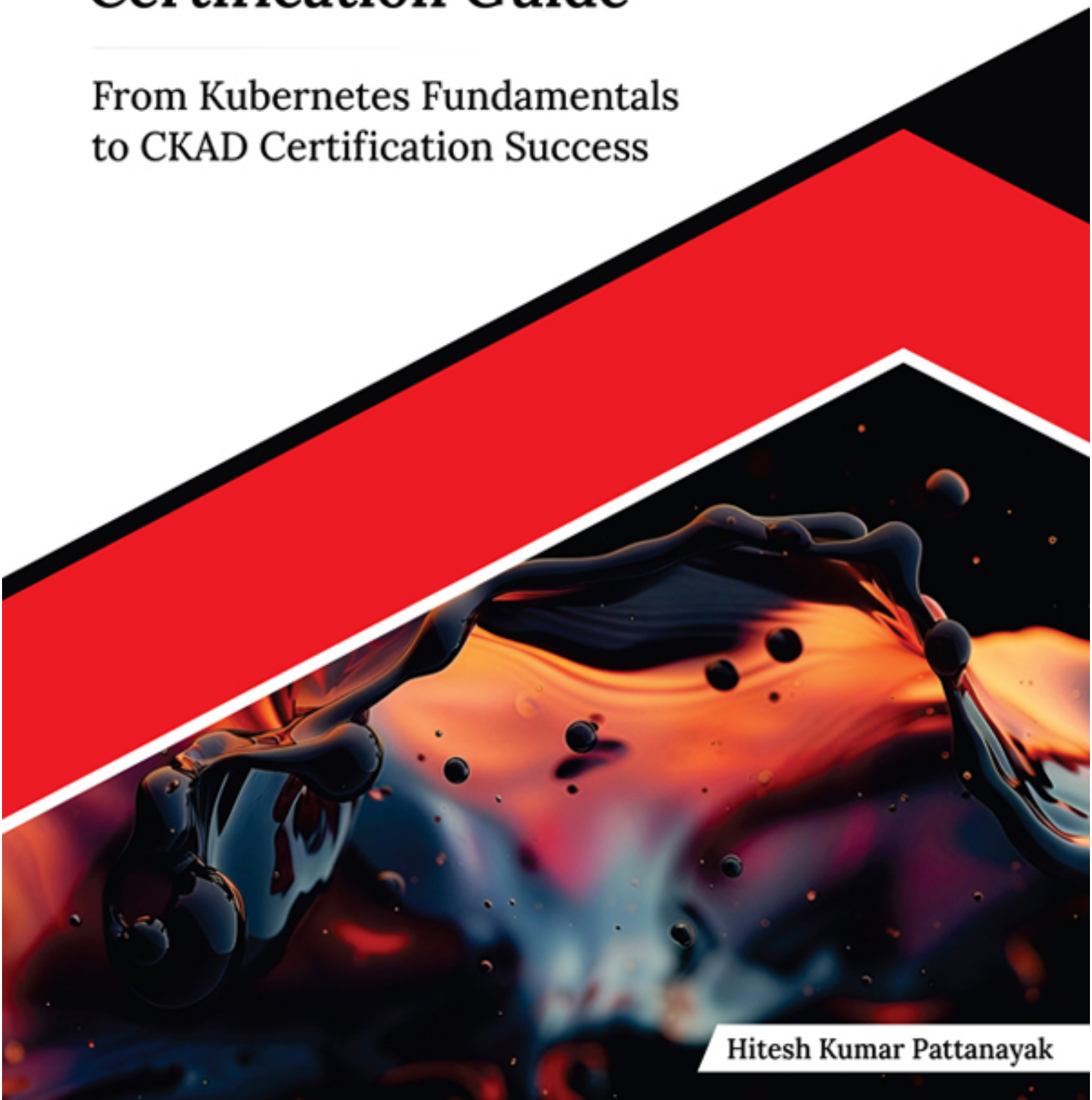
ULTIMATE

Certified Kubernetes Application Developer (CKAD) Certification Guide

From Kubernetes Fundamentals to CKAD Certification Success



Hitesh Kumar Pattanayak

First Published: March 2026

Published by: Orange Education Pvt Ltd, AVA®

Address: 9, Daryaganj, Delhi, 110002, India

275 New North Road Islington Suite 1314 London,
N1 7AA, United Kingdom

ISBN (PBK): 978-93-49887-13-8

ISBN (E-BOOK): 978-93-49887-10-7

Scan the QR code to explore our entire catalogue

www.orangeava.com

Table of Contents

1. Understanding the CKAD Certification
- Introduction
- Structure
- Overview of CKAD Certification
- Purpose of being CKAD Certified
- How to Get Started with CKAD
  - *Key Skills Tested*
  - *Exam Domain and Weightage*
- Preparing for the Exam: Environment and System Requirements
- Exam Format
- Preparation Strategy
    - *Strategy to Attempt Questions*
    - *Prepare on Simulator*
    - *Sample Preparation Schedule*
- Mock Tests and Practice Exercises
- Conclusion
- Multiple Choice Questions
- Answers

2. Container Image Management
- Introduction
- Structure
- Kubernetes 1.24 Updates
- Setting Up Your Local Environment
  - *Installing Docker on Your System*
    - *Step 1: Install docker based on OS*
    - *Step 2: Hands-on Setup and Verification*
- Building Custom Container Images
  - *Building Custom Container Images*
    - *Basic Dockerfile Structure*
    - *Step-by-Step Walkthrough*
- Working with Docker and Container Registries
    - *Using Docker Hub*

Modifying and Optimizing Container Images
Build Time Optimization
Reducing Image Size with Multi-Stage Builds
Optimization Tips for Dockerfiles
Best Practices for Container Image Security
Conclusion
Multiple Choice Questions
Answers

3. Workload Resources and Scheduling
Introduction
Structure
Introduction to AvaKart Codebase
System Architecture
Communication Flow
Build and Push Docker Images
Kubernetes Architecture
Kubectl Command Flow
Understanding Pods with AvaKart
Explaining Pod Lifecycle
Best Practices Implementation
Pod Configuration Deep Dive
Create Pods
ReplicaSets: Ensuring Service Availability
Understanding ReplicaSets
Detailed 'Books' ReplicaSet Configuration
Implementation in AvaKart
Key Components Explained
Deployments: Managing Application Updates
Understanding the Deployment Abstraction
Implementation in AvaKart
Key Components Explained
Benefits of a Deployment Manifest
Jobs and CronJobs
Jobs
Implementation in AvaKart: Order Processing Batch Job
Key Components Explained

CronJobs
Key Components Explained
Scheduling in Kubernetes
Node Selection
Pod Scheduling
AvaKart Implementation
Best Practices
Conclusion
Multiple Choice Questions
Answers

4. Multi-Container Pod Design
Introduction
Structure
Introduction to Multi-Container Pod Patterns
Use Cases of Multi-Container Pods
Choosing the Right Pattern
Sidecar Pattern
Implementation in the Storage Service
Architecture Overview
Sidecar Container Configuration
Deployment Definition
Working of the Sidecar Pattern
Init Container Pattern
Core Principles
Use Cases of Init Containers
Implementation in the Books Service
Working of the Init Container Pattern
Ambassador Pattern
Implementation in the Order Processor Service
Architecture Overview
Ambassador Container Configuration
Deployment Definition
Working of the Ambassador Pattern
Use Cases of the Ambassador Pattern
Conclusion
Multiple Choice Questions

Answers

5. Managing Volumes
Introduction
Structure
Types of Volumes
EmptyDir
EmptyDir Implementation Details
Kubernetes Manifest
Application Code
EmptyDir Benefits and Use Cases
EmptyDir Disadvantages
Persistent Volumes and Claims
PV PVC Implementation Details
Kubernetes Manifest
Application Code
PV, PVC Benefits and Use Cases
PV and PVC Disadvantages
Storage Class
Storage Class Implementation Details
Kubernetes Manifests
Application Code
Storage Class Benefits and Use Cases
Storage Class Disadvantages
Volume Snapshots
Volume Snapshots Implementation Details
Kubernetes Manifests
Volume Snapshots Benefits and Use Cases
Volume Snapshots Disadvantages
Conclusion
Multiple Choice Questions
Answers

6. Kubernetes Deployment Strategies
Introduction
Structure
Need for Deployment Strategies

Rolling Updates
Implementation Details
Health Checks for Safe Deployments
Applying and Testing Rolling Updates
Blue/Green Deployments
Implementing Blue/Green Deployments
Applying and Testing Blue/Green Deployments
Canary Releases
Feature Toggles for Canary Testing
Applying and Testing Canary Releases
Advanced Deployment Using Ingress
Implementing Advanced Traffic Routing
Other Deployment Patterns
Best Practices for Zero-Downtime Deployments
Implement Health Checks
Implement Graceful Shutdown
Use Resource Limits and Requests
Use Proper Deployment Strategy for Your Application
Monitor Deployments
Automate Rollbacks
Conclusion
Multiple Choice Questions
Answers

7. Rolling Updates and Rollbacks
Introduction
Structure
Advanced Rolling Update Configurations
Key Configuration Parameters
Revision History Management
Change Tracking with Annotations
Version Labeling
Rollback Mechanisms and Strategies
Viewing Deployment History
Rolling Back to the Previous Version
Rolling Back to a Specific Revision
Monitoring Rollback Status

Rollback Demonstration
Comparing Deployment Versions
Ensuring Consistency during Updates and Rollbacks
Environment Variables for Version Management
Resource Management
Health Checks
Advanced Health Checking with Probes
Types of Health Probes
Probe Configuration Parameters
Conclusion
Multiple Choice Questions
Answers

8. Helm and Kustomize
Introduction
Structure
Introduction to Helm
Helm Concepts
Chart Structure
Template Files
Using Helm Charts for Application Deployment
Installing a Chart
Upgrading a Release
Rolling Back a Release
Creating a Chart for AvaKart Frontend
Using Helm Dependencies
Introduction to Kustomize
Kustomize Concepts
Kustomize Structure
Base Resources
Managing Kubernetes Configurations with Kustomize
Patches
ConfigMap and Secret Generation
Variants
Applying Kustomizations
Comparing Helm and Kustomize
Kustomize Templates for AvaKart

Key Features of AvaKart Kustomize Templates
Helm Charts from AvaKart
Key Features of AvaKart Helm Charts
Deployment Flow
Customization Options
Conclusion
Multiple Choice Questions
Answers

9. Monitoring and Debugging Applications
Introduction
Structure
Kubernetes Native Logging and Monitoring
Basic Log Retrieval
Log Collection Strategies
Structured Logging in Kubernetes
Benefits of Structured Logging
Implementing Structured Logging
Filtering and Analyzing Structured Logs
Best Practices for Logging
Debugging Application Issues
Checking Pod Status
Describing Resources
Checking Events
Monitoring Resource Usage
Port Forwarding
Executing Commands in Containers
Copying Files
Troubleshooting Common Kubernetes Issues
Pod Startup Issues
Service Connectivity Issues
Resource Constraints
Using Debug Containers
Purpose of Debug Containers
Creating a Debug Container
Creating a Debug Pod
Debug Pod Example

Comprehensive Debugging Script
Conclusion
Multiple Choice Questions
Answers

10. Probes and Health Checks
Introduction
Structure
Understanding Kubernetes Probes
Implementing Liveness Probes
Basic Liveness Probe
When to Use Liveness Probes
Liveness Probe Parameters
Implementing Readiness Probes
Basic Readiness Probe
When to Use Readiness Probes
Readiness Probe Parameters
Implementing Startup Probes
Basic Startup Probe
When to Use Startup Probes
Startup Probe Parameters
Advanced Probe Configurations
HTTP Probe with Headers and Parameters
TCP Socket Probe
Exec Command Probe
Implementing Health Check Endpoints
Books Service Health Endpoints
Storage Service Health Endpoints with Dependency Checks
Health Check Best Practices
Comprehensive Health Check Implementation
Best Practices for Health Checks
General Guidelines
Probe-Specific Best Practices
Liveness Probes
Readiness Probes
Startup Probes
Implementation Examples

Simple HTTP Service
Slow-Starting Application
Common Pitfalls and How to Avoid Them
Conclusion
Multiple Choice Questions
Answers

11. Managing API Deprecations
Introduction
Structure
Understanding Kubernetes API Versioning
API Version Formats
Deprecation Policy
Identifying Deprecated API Versions
Using kubectl to Check for Deprecated APIs
Example of a Deprecated API in a Deployment
Migrating to Newer API Versions
Migrating Deployments
Migrating Ingress Resources
Migrating CronJob Resources
Tracking Kubernetes Release Notes
Key Changes in Recent Releases
Common Deprecated APIs and Their Replacements
Implementing a Deprecation Check Tool
Best Practices for API Version Management
Development Practices
Operational Practices
Using kubectl Convert for API Migration
Handling Special Cases
PodSecurityPolicy Removal
Ingress Path Types
Conclusion
Multiple Choice Questions
Answers

12. ConfigMaps and Secrets
Introduction

Structure
Understanding ConfigMaps
Purpose of ConfigMaps
Use Cases for ConfigMaps
Creating and Managing ConfigMaps
Creating ConfigMaps from Literal Values
Creating ConfigMaps from Files
Creating ConfigMaps from Multiple Files
Consuming ConfigMaps in Pods
Using ConfigMap as Environment Variables
Using Individual ConfigMap Values as Environment Variables
Using All ConfigMap Values as Environment Variables
Using ConfigMap as Volume
Understanding Kubernetes Secrets
Purpose of Secrets
Use Cases for Secrets
Creating and Managing Secrets
Creating Secrets from Literal Values
Creating Secrets from Files
Creating Secrets for TLS
Consuming Secrets in Pods
Using Secrets as Environment Variables
Using Secrets as Volumes
Implementing Configuration in AvaKart Services
Books Service Configuration Handling
Storage Service Configuration from Files
Order Processor Configuration and Secrets
Best Practices for ConfigMaps and Secrets
ConfigMap Best Practices
Secret Best Practices
General Best Practices
Example: Implementing Configuration Reloading
AvaKart-Specific Best Practices
Conclusion
Multiple Choice Questions
Answers

13. Security Best Practices
- Introduction
- Structure
- Understanding Service Accounts
  - *Purpose of Service Accounts*
  - *Creating Service Accounts*
    - *Basic Service Account*
    - *Service Account with Image Pull Secrets*
    - *Service Account with Annotations*
  - *Using Service Accounts in Pods*
  - *Verifying Service Accounts*
- Implementing Security Contexts
  - *Purpose of Security Contexts*
  - *Pod-Level Security Contexts*
  - *Container-Level Security Contexts*
  - *Verifying Security Contexts*
- Role-Based Access Control (RBAC)
  - *Purpose of RBAC*
  - *Creating Roles and RoleBindings*
  - *Creating ClusterRoles and ClusterRoleBindings*
  - *Verifying RBAC Configurations*
- Network Policies
  - *Purpose of Network Policies*
  - *Default Deny Policy*
  - *Service-Specific Network Policies*
  - *Verifying Network Policies*
- Implementing Security in AvaKart Services
  - *Secure Books Service Deployment*
  - *Secure Storage Service Deployment*
- Security Best Practices for Kubernetes
  - *Container Security*
  - *Pod Security*
  - *Authentication and Authorization*
  - *Secrets Management*
  - *Network Security*
  - *Audit and Compliance*
- Applying Security Configurations

Conclusion
Multiple Choice Questions
Answers

14. Resource Management and Quotas
Introduction
Structure
Understanding Resource Requests and Limits
Resource Requests
Resource Limits
Resource Constraints Behavior
Quality of Service (QoS) Classes
Guaranteed QoS
Burstable QoS
BestEffort QoS
Resource Management in Deployments
Understanding Resource Quotas
ResourceQuota Object
Quota Scope
Quota Enforcement
Implementing Resource Quotas
Basic ResourceQuota
Scoped ResourceQuota
Understanding LimitRanges
Default Values
Min/Max Constraints
Max/Min Ratios
Resource Types
Enforcement
Implementing LimitRanges
Container LimitRange
Storage LimitRange
Monitoring Resource Usage
kubectl top
Metrics Server
Kubernetes Dashboard
Prometheus and Grafana

Implementing Resource Management in AvaKart Services
Books Service with Resource Management
Storage Service with Resource Management
Order Processor with Resource Management
Managing Resource Contention in Multi-Tenant Environments
Namespace Creation for Multi-Tenancy
Resource Quotas for Teams
LimitRanges for Teams
Best Practices for Resource Management
Setting Resource Requests and Limits
Resource Quotas and Namespace Management
Application Design for Resource Efficiency
Conclusion
Multiple Choice Questions
Answers

15. Authentication, Authorization, and Admission Controllers
Introduction
Structure
Understanding Authentication in Kubernetes
Working with Service Accounts
Creating a Service Account
Creating a Token for a Service Account
Using a Service Account in a Pod
Understanding Authorization in Kubernetes
Implementing RBAC for Authorization
Creating a Role
Creating a RoleBinding
Creating a ClusterRole
Creating a ClusterRoleBinding
Understanding Admission Control
Working with Built-in Admission Controllers
PodSecurity Admission Controller
ResourceQuota Admission Controller
Implementing Webhook Admission Controllers
Implementing ValidatingAdmissionPolicy
Implementing Security for AvaKart Services

Secure Deployment for Books Service
Troubleshooting Authentication and Authorization Issues
Common Authentication Issues
Invalid or Expired Service Account Tokens
Missing Service Account in Pod Specification
Common Authorization Issues
Missing RBAC Roles or Bindings
Testing Permissions
Using a Debug Pod
Best Practices for Kubernetes Security
Authentication Best Practices
Authorization Best Practices
Admission Control Best Practices
Conclusion
Multiple Choice Questions
Answers

16. Services and Ingress
Introduction
Structure
Understanding Kubernetes Services
Implementing ClusterIP Services
Books Service ClusterIP
Storage Service ClusterIP
Order Processor Service ClusterIP
Testing ClusterIP Services
Working with NodePort Services
Frontend NodePort Service
Books Service External Access
NodePort Considerations
Configuring LoadBalancer Services
Frontend LoadBalancer Service
API Gateway LoadBalancer Service
Using Headless Services for StatefulSets
Database Headless Service
StatefulSet with Headless Service
Introduction to Ingress Resources

Setting Up NGINX Ingress Controller
NGINX Controller RBAC
NGINX Controller Deployment
NGINX Controller Service
Creating Basic Ingress Rules
Frontend Ingress
API Services Ingress
Implementing SSL/TLS Termination
Creating TLS Secrets
API TLS Secret
Certificate Management Best Practices
Advanced Ingress Features
Canary Deployments
Rate Limiting
Multi-Tenant Routing
Service Discovery and DNS
DNS-Based Service Discovery
Environment Variable Discovery
Testing Service Discovery
Troubleshooting Services and Ingress
Service Connectivity Issues
Service Not Accessible
DNS Resolution Failures
Load Balancing Not Working
Ingress Troubleshooting
Ingress Not Routing Traffic
SSL/TLS Certificate Issues
Path-Based Routing Problems
Debugging Commands
Conclusion
Multiple Choice Questions
Answers

17. Network Policies
Introduction
Structure
Understanding Network Policies in Kubernetes

- *Key Concepts*
- *Functioning of Network Policies*
- *Basic Network Policy Structure*

Prerequisites and CNI Plugin Requirements
- *Checking CNI Plugin Support*
- *Supported CNI Plugins*
- *Installing Calico for Network Policy Support*

Implementing Default Deny Policies
- *Default Deny All Traffic*
- *Default Deny Ingress Only*
- *Default Deny Egress Only*
- *Testing Default Deny Policies*

Creating Ingress Network Policies
- *Allowing Frontend Access to Books Service*
- *Allowing External Access to Frontend*
- *Applying Ingress Policies*

Implementing Egress Network Policies
- *Allowing Books Service Database Access*
- *Allowing External API Access*
- *Important DNS Considerations*

Namespace-Based Network Policies
- *Allowing Cross-Namespace Communication*
- *Denying Cross-Namespace Communication*
- *Namespace Labeling*

Advanced Network Policy Patterns
- *Multi-Tier Application Policy*
- *Complete AvaKart Policy Set*

Testing and Validating Network Policies
- *Automated Testing*
- *Manual Testing*
- *Connection Testing Patterns*

Monitoring and Troubleshooting
- *Common Issues*
- *Troubleshooting Commands*
- *Monitoring Setup*
- *Performance Impact*

Best Practices for Production

Production Deployment
Conclusion
Multiple Choice Questions
Answers

18. Custom Resource Definitions
Introduction
Structure
Understanding Custom Resource Definitions
Purpose of CRDs
Key Concepts
CKAD Exam Relevance
Creating Basic CRDs
AvaKart BookCatalog CRD
Understanding CRD Structure
Installing the CRD
Working with Custom Resources
Creating BookCatalog Resources
Managing Custom Resources
Custom Columns in Action
Advanced CRD Features
Order Management CRD
Validation Features
Creating Order Resources
Status Subresource
Understanding the Operator Pattern
How Kubernetes Operators Work: Step-by-Step Operators follow a control loop pattern:
Controller Pattern
Operator Examples
CKAD Relevance
CKAD Exam Scenarios and Troubleshooting
Common Exam Tasks
Essential Commands
Validation Error Troubleshooting
Common Issues and Solutions
CRD Not Found Error

Validation Errors
Custom Resource Stuck in Pending
Conclusion
Multiple Choice Questions
Answers

19. Mock Tests and Practice Tests
Introduction
Structure
Understanding the CKAD Exam Format
Mock Test 1: Application Design and Build
Question 1: Multi-Container Pod Design (5 minutes)
Question 2: Job and CronJob Implementation (8 minutes)
Question 3: Custom Container Image Build (7 minutes)
Mock Test 2: Application Deployment
Question 4: Rolling Update Deployment (8 minutes)
Question 5: Blue-Green Deployment Strategy (10 minutes)
Question 6: Helm Chart Deployment (7 minutes)
Mock Test 3: Application Observability and Maintenance
Question 7: Health Probes Implementation (6 minutes)
Question 8: Debugging and Troubleshooting (8 minutes)
Question 9: Container Logs and Monitoring (6 minutes)
Mock Test 4: Application Environment, Configuration, and Security
Question 10: ConfigMaps and Secrets Management (10 minutes)
Question 11: RBAC and ServiceAccounts (10 minutes)
Question 12: SecurityContext and Resource Management (10 minutes)
Mock Test 5: Services and Networking
Question 13: Service Types and Load Balancing (8 minutes)
Question 14: Ingress Configuration (10 minutes)
Question 15: Network Policies (7 minutes)
Full-Length Mock Exam
Common Exam Scenarios and Solutions
Scenario 2: Service Discovery Issues
Scenario 3: Configuration Management
Scenario 4: Security Hardening
Scenario 5: Scaling and Updates

Exam Tips and Best Practices
kubectl Efficiency Techniques
Quick Resource Creation
Resource Management
Debugging and Troubleshooting
Useful Aliases and Shortcuts
Conclusion

20. Tips and Tricks For Success
Introduction
Structure
CKAD Exam-Taking Strategies
Pre-Exam Preparation Strategy
Essential Bookmarks for Exam Day
Exam Day Strategy
Command-Line Efficiency and Productivity Tricks
Essential Aliases for Speed
Auto-Completion Setup
Power User kubectl Techniques
Efficient YAML Manipulation
Context and Namespace Management
Setting Up Your Exam Environment
Terminal Configuration
Browser Setup for Documentation
Practice Environment Setup
Time Management and Question Prioritization
Question Analysis Framework
Execution Strategy
Time-Saving Techniques
Common Pitfalls and How to Avoid Them
Technical Pitfalls
Strategic Pitfalls
Verification Checklist
Open Source and Personal Projects
Conclusion

Index

CHAPTER 1

Understanding the CKAD Certification

Introduction

In this chapter, we will explore the Certified Kubernetes Application Developer (CKAD) certification program, developed by the Cloud Native Computing Foundation (CNCF) in collaboration with the Linux Foundation. This program is designed to validate your skills in designing and building applications that run on Kubernetes, a leading container orchestration platform.

Structure

In this chapter, we will discuss the following topics:

- Overview of CKAD Certification
- Purpose of being CKAD Certified
- How to Get Started with CKAD
    - Key Skills Tested
    - Exam Domains and Weightage
- Preparation of Exam: Environment and System Requirements
- Exam Format
- Preparation Strategy

Overview of CKAD Certification

The CKAD certification is a credential that demonstrates your ability to design, build, and manage applications in a Kubernetes environment. It emphasizes practical, hands-on skills that are essential for modern cloud-native development. As Kubernetes adoption continues to grow exponentially—visible through trends like those on Google—the demand for certified professionals is rising. By obtaining this certification, you can set yourself apart in the competitive job market, showcasing your ability to deploy and manage cloud-native applications with confidence.

Purpose of being CKAD Certified

In the current technological landscape, it is crucial for professionals to establish their credibility and demonstrate their value in the marketplace. The CKAD certification helps you stand out by proving that you have mastered the foundational skills necessary for Kubernetes application development. Whether you are looking to enhance your current role or seeking new opportunities, this certification serves as a mark of excellence.

By becoming CKAD certified, you also show that you are committed to continuous learning and adapting to the latest cloud-native practices. With Kubernetes quickly becoming the standard for container orchestration, attaining this certification will position you at the forefront of this rapidly evolving field.

How to Get Started with CKAD

Preparing for the CKAD certification begins with understanding the exam's structure and curriculum. The certification is designed to test your ability to perform real-world tasks in a Kubernetes environment, focusing on core topics that are critical to application development and deployment.

Key Skills Tested

The CKAD exam is designed to measure not only theoretical knowledge but also the **ability to apply** that knowledge in a Kubernetes environment, making it a **hands-on, practical assessment**. Success in the exam relies on demonstrating these key skills through task completion under time constraints.

Exam Domain and Weightage

Let us break down the exam domains and their respective weightage:

1. **Application Design and Build (20%)**
    a. Understand how to define, build, and modify container images.
    b. Choose the appropriate Kubernetes workload resource (such as Deployment, DaemonSet, StatefulSet, or CronJob) based on your use case and requirements.
    c. Implement multi-container pod design patterns.
    d. Utilize persistent and ephemeral volumes to manage data storage.
2. **Application Deployment (20%)**
    a. Use Kubernetes primitives to implement deployment strategies such as blue/green deployments and canary releases.

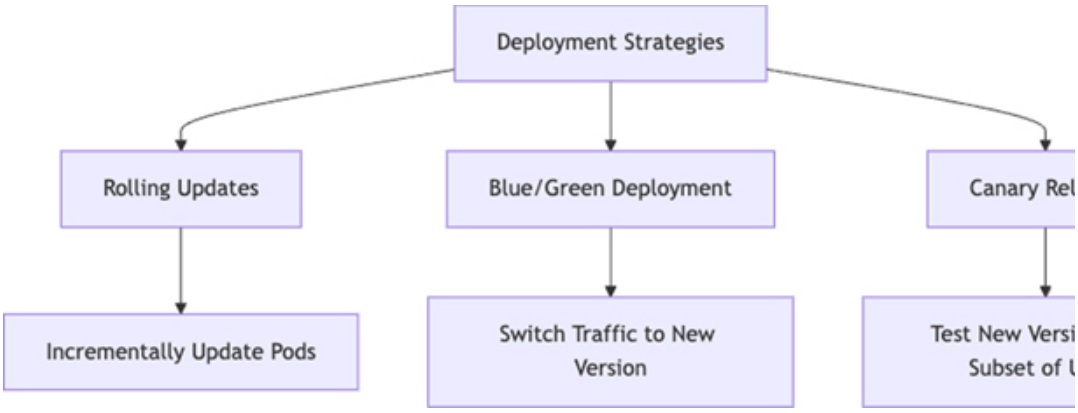


Figure 1.1: *Deployment Strategies*

b. Master rolling updates and rollback mechanisms to restore faulty deployments.
c. Create and use Helm packages for managing Kubernetes applications.
d. Employ Kustomize to manage Kubernetes resources.

3. **Application Observability and Maintenance (15%)**

a. Stay informed about API deprecations and their impact on application functionality.
b. Implement probes and health checks to ensure the stability of your applications.
c. Use logs and resource events to debug applications running in Kubernetes.

4. **Application Environment, Configuration, and Security (25%)**

a. Create and manage custom resources using Custom Resource Definitions (CRDs) and Operators.
b. Understand Kubernetes authentication, authorization, and admission control processes.
c. Define resource requests, limits, and quotas to manage application resources efficiently.
d. Work with ConfigMaps, Secrets, and ServiceAccounts to manage application configurations and secure sensitive data.
e. Implement security contexts and capabilities to ensure that your applications run securely.

5. **Services and Networking (20%)**

a. Understand and implement NetworkPolicies for controlling network traffic.
b. Troubleshoot application access issues using Kubernetes services.
c. Use Ingress rules to expose applications to external users effectively.

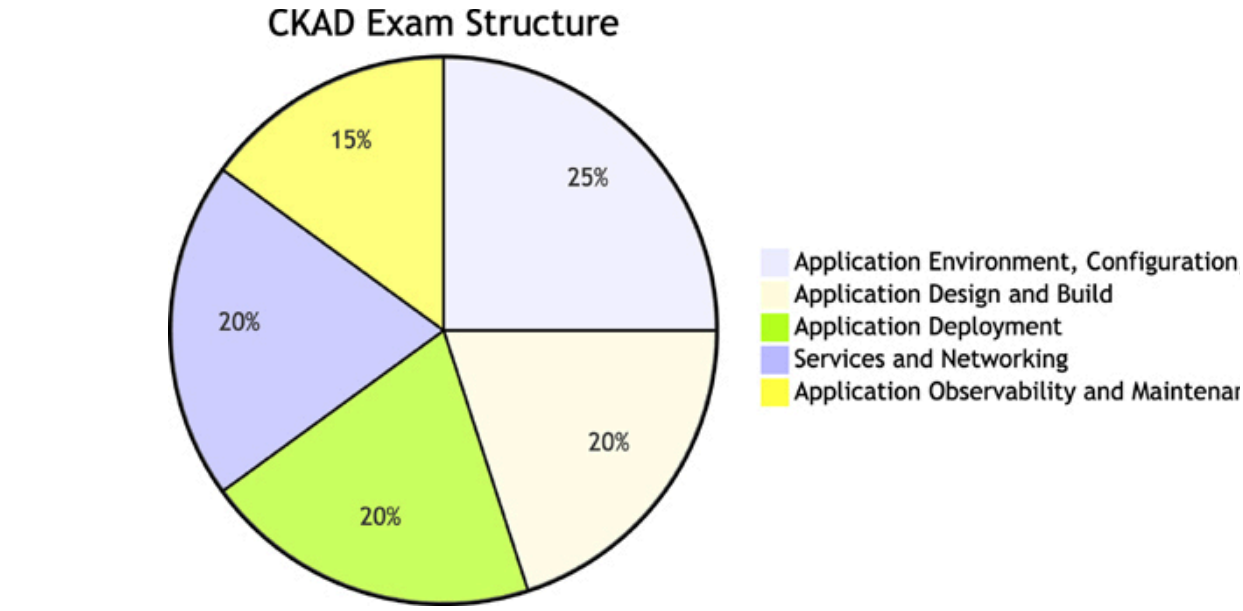


Figure 1.2: *CKAD Exam Structure*

For more information on the certification, please visit CNCF CKAD Certification: http://cncf.io/certification/ckad.

Preparing for the Exam: Environment and System Requirements

To ensure a smooth exam experience, it is important to be aware of the environmental and system requirements. The CKAD exam is conducted online, and can be taken from the comfort of your home, but there are specific guidelines you need to follow.

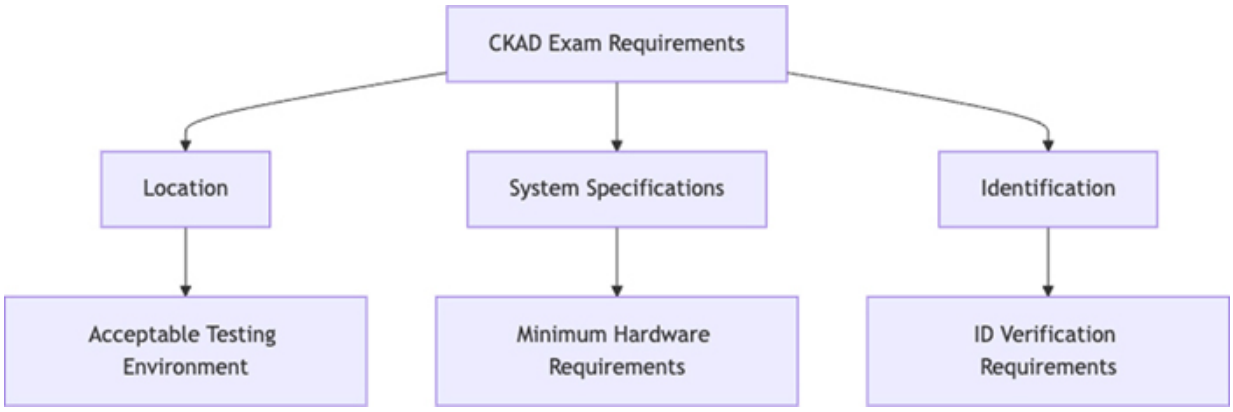


Figure 1.3: *CKAD Exam Requirements*

- **Room Setup**: Make sure your room meets the acceptable norms for the exam. For details, please review the Linux Foundation's guidelines:

https://docs.linuxfoundation.org/tc-docs/certification/tips-cka-and-ckad#acceptable-testing-location.

- **System Requirements**: Your system must meet specific hardware and software criteria to run the exam smoothly. Review the detailed requirements here: **https://docs.linuxfoundation.org/tc-docs/certification/tips-cka-and-ckad#system-requirements-to-take-the-exam.**
- **Network Connectivity**: Ensure that you have a stable internet connection to avoid disruptions during the exam.
- **Identification**: You will need valid identification to take the exam. For more information, please see the identification requirements: **https://docs.linuxfoundation.org/tc-docs/certification/tips-cka-and-ckad#id-requirements-to-take-the-exam.**

Exam Format

CKAD exam is not your typical multiple-choice exam. It is a hands-on, performance-based assessment where you will be required to complete practical tasks using Kubernetes. The focus is on real-world application, so memorization of commands is less important than understanding how Kubernetes works and being able to apply that knowledge.

You will have access to the official Kubernetes documentation (https://kubernetes.io/docs/home/) throughout the exam. Throughout this book, we will emphasize how to effectively use the documentation to find the right information quickly and efficiently.

The exam consists of **15-20** tasks that need to be completed within a two-hour window. As of today, the exam fee is **$395** USD and includes one free retake. The exam is proctored online, meaning you can take it anytime, anywhere, while being monitored by a live proctor.

You will receive your exam results via email within 24 hours of completion.

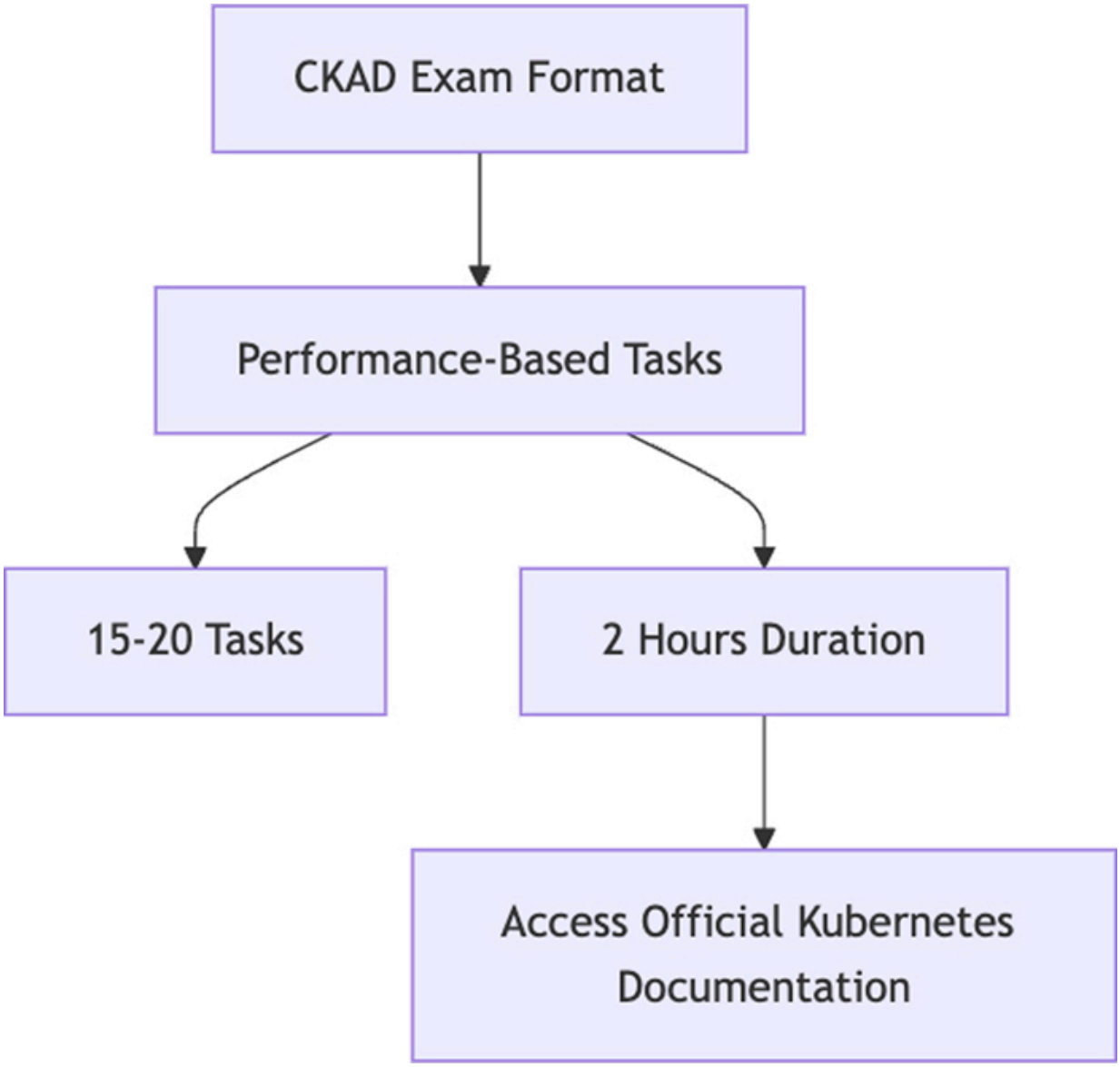


Figure 1.4: *CKAD Exam Format*

Preparation Strategy

Preparing for the Certified Kubernetes Application Developer (CKAD) exam requires a focused and practical approach. Since the exam emphasizes real-world application of Kubernetes, the preparation strategy should blend hands-on practice with a solid understanding of key concepts.

1. **Understand the Exam Structure and Domains**
    - Start by reviewing the exam curriculum to understand the weight of each domain.

- Prioritize your study time according to the weight of these sections. Focus more on high-weighted topics such as Application Environment, Configuration, and Security, but do not neglect other domains.

2. **Set Up a Kubernetes Practice Environment**
   - Use a local setup such as Minikube or Kind (Kubernetes in Docker) to create your own Kubernetes cluster for practice. Alternatively, use a cloud provider (for example, GKE, EKS, AKS) to gain exposure to cloud-managed Kubernetes.
   - Explore Kubernetes concepts by working with real deployments, services, and configurations. Practice the commands required during the exam.
   - For this book, we would stick to the usage of Minikube in local system (no cloud provider).
3. **Use the Kubernetes Documentation**
   - The Kubernetes official documentation is your best friend during preparation and the exam itself. Since the documentation is available during the exam, practice navigating through it quickly and efficiently.
   - Understand where to find information on common tasks such as creating deployments, managing services, configuring secrets, and more. Bookmark the most relevant sections for easier access.

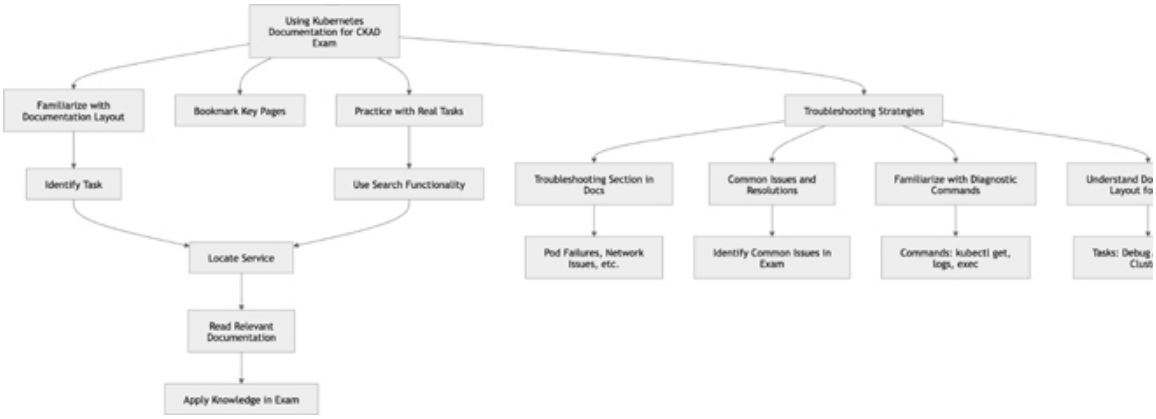


Figure 1.5: *Kubernetes Official Docs Usage*

4. **Focus on Key Topics**
   - **Workload Management**: Practice creating and managing deployments, DaemonSets, StatefulSets, and Jobs. Be comfortable with modifying resources during runtime.
   - **Container Image Management**: Learn how to define, build, and modify container images. Practice using kubectl run or kubectl create commands to deploy containers.

- **Deployment Strategies**: Implement different deployment strategies such as rolling updates, blue/green deployments, and canary releases.
- **Pod Design**: Get familiar with multi-container pods and patterns such as sidecar containers.
- **ConfigMaps and Secrets**: Learn to create and use ConfigMaps and Secrets to manage configuration data and sensitive information within applications.
- **Network and Services**: Practice setting up services, understanding ClusterIP, NodePort, and LoadBalancer types, and using Ingress to expose applications. Also, focus on implementing NetworkPolicies for securing traffic.

5. **Hands-On Practice**

- **Task Automation**: Automate repetitive Kubernetes tasks using kubectl and YAML manifests. Create your own YAML files to define deployments, services, and configurations.
- **Debugging and Troubleshooting**: Learn how to debug issues by examining pod logs, resource events, and using probes (readiness, liveness).
- **Performance Testing**: Test resource management by configuring requests, limits, and quotas for applications. Get familiar with commands such as kubectl top to monitor resource usage.

6. **Time Management During Practice**

- The CKAD exam gives you **2 hours** to complete around **15-20** tasks. Practice completing tasks within time limits.
- Create aliases for frequently used commands and practice using them regularly to ensure that you are comfortable applying them during the exam.

  The following list outlines the aliases used throughout this book. You are encouraged to customize them to fit your personal workflow or preferences.

```
alias k='kubectl'
alias kc='kubectl create'
alias kcfg='kubectl config'
alias kd='kubectl describe'
alias kdep='kubectl get deploy'
alias kdp='kubectl describe pod'
alias ke='kubectl explain'
alias kg='kubectl get'
```

```
alias kgaa='kubectl get all --show-labels'
alias kgpa='kubectl get pod --all-namespaces'
alias kns='kubectl get namespaces'
alias kpod='kubectl get pod'
alias krem='kubectl delete'
alias krep='kubectl replace'
alias ksetimg='kubectl set image deploy'
alias ksvc='kubectl get svc'
```

> ***Note**: Many of these commands may not make sense right now, but they will become clearer as we progress through the chapters.*

- Simulate exam conditions by timing yourself while completing hands-on labs. This helps develop speed and accuracy under pressure.
- Skip tasks that are taking too long and return to them later, just like you would in the exam.

7. **Mock Exams and Practice Tests**
   - Take mock exams or use online practice platforms such as **Killer.sh** (official CKAD simulator) to get a feel of the exam format. These simulators offer practice questions that closely resemble the CKAD exam.
   - Focus on problem-solving and understanding why each solution works.
8. **Stay Current with Kubernetes Updates**
   - Kubernetes is constantly evolving, and new versions may deprecate older APIs or introduce new features. Ensure that you are studying with the most recent version of Kubernetes (usually the exam is based on the most recent stable release).
   - Keep track of API deprecations and new features in each Kubernetes release, as this could affect exam content.
9. **Join the Kubernetes Community**
   - Engage with the Kubernetes community by joining forums such as Reddit, Slack channels, or the Kubernetes Discussion Forum. Here you can learn from others' experiences, gain valuable insights, and ask questions.
10. **Plan for Exam Day**
    - Make sure your environment and system are ready for the exam. Test your camera, microphone, and internet connection before the exam day.
    - Have a clean, quiet space where you would not be disturbed during the exam.

- Review the exam guidelines and ensure that you have all necessary identification documents ready.
- Familiarize yourself with the online proctoring process and system requirements as outlined by the Linux Foundation.

Strategy to Attempt Questions

The exam will feature questions with varying weightages, such as 4%, 6%, and 8%. Begin with the higher-weighted questions, rather than answering them sequentially. Avoid spending excessive time on any single question. Keep in mind that there will likely be a tricky question—flag it and move on to the next higher-weighted question.

Prepare on Simulator

When you feel prepared for the exam, take advantage of the free simulator test available upon registration here: https://training.linuxfoundation.org/certification/certified-kubernetes-application-developer-ckad/. Following is a screenshot showing where to access the mock simulator:

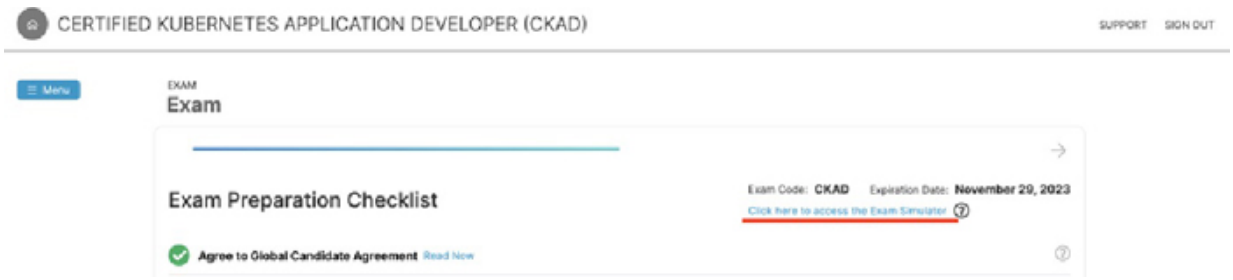


Figure 1.6: *CKAD Exam Mock Simulator Link*

The simulator offers a near-real exam experience, which is crucial for practice. Even basic tasks such as copying and pasting function differently in the actual exam environment, as it uses an Ubuntu OS.

Sample Preparation Schedule

The exam preparation is self-paced and will vary depending on the individual. Experience level also plays a significant role. However, the following schedule can serve as a reference.

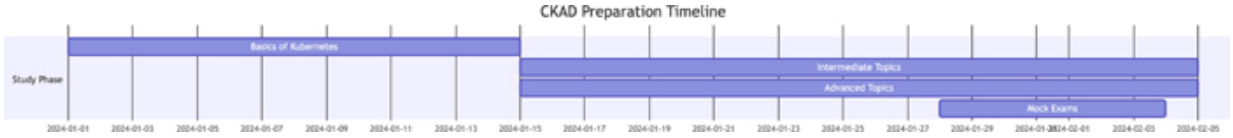


Figure 1.7: *Sample CKAD Preparation Timeline*

- **Week 1-2**: Learn the basics of Kubernetes, set up a practice environment, and go through key concepts such as Pods, Deployments, Services, and Storage.
- **Week 3-4**: Dive into more advanced topics such as multi-container pod patterns, Helm, Kustomize, and security contexts. Practice creating and troubleshooting workloads.
- **Week 5**: Take mock exams, review your performance, and work on areas where you need improvement. Focus on time management and documentation navigation.

Thus, by the end of this preparation strategy, you should feel comfortable with both the Kubernetes concepts and the exam format.

Mock Tests and Practice Exercises

There are limited quality CKAD exam practice tests available, but fortunately, they are sufficient for honing skills. Even better is the online Kubernetes environment provided by **Killercoda**, which allows users to experiment with practice tests and explore different Kubernetes objects.

- The official Kubernetes documentation at https://kubernetes.io/docs/tutorials is an excellent resource, offering comprehensive information. It is highly recommended to start from there.
- If additional guidance is needed, here is a direct link to a basic practice tutorial: https://kubernetes.io/docs/tutorials/kubernetes-basics/deploy-app/deploy-intro/.
- **KillerCoda CKAD practice test**:
- **CKAD exercises on GitHub**:
    - https://github.com/dgkanatsios/CKAD-exercises#contents
    - https://github.com/HiteshRepo/Kubernetes-101

Conclusion

In this chapter, we gained a comprehensive understanding of the Certified Kubernetes Application Developer (CKAD) certification, including what it entails, why it is valuable, and how to prepare for the exam. As Kubernetes continues to grow in popularity, becoming certified provides a significant advantage, showcasing your ability to design, build, and deploy cloud-native applications in this powerful platform. We also explored the exam's structure, key skill areas, and practical guidelines for success.

With the right preparation and practice, the CKAD certification will not only validate your expertise but also enhance your credibility in the competitive job market. Remember, this is a hands-on exam that tests your ability to work in real-world

Kubernetes environments, so focus on practical knowledge and efficient use of Kubernetes documentation.

In the next chapter, we will explore Container Image Management, covering how to define, build, and optimize images, best practices for security, and how Kubernetes integrates images into deployments—a foundational skill for Kubernetes application development.

Multiple Choice Questions

1. What is the CKAD certification designed for?
    a. System administrators
    b. Kubernetes cluster operators
    c. Kubernetes application developers
    d. Network administrators
2. Which organization developed the CKAD certification in collaboration with CNCF?
    a. Microsoft
    b. The Linux Foundation
    c. Red Hat
    d. AWS
3. What is the expected duration of the CKAD exam?
    a. 1 hour
    b. 2 hours
    c. 3 hours
    d. 4 hours
4. The CKAD exam focuses on:
    a. Memorizing Kubernetes commands
    b. Multiple-choice questions
    c. Performance-based tasks with hands-on skills
    d. Writing Kubernetes documentation
5. What percentage of the CKAD exam curriculum is dedicated to Application Design and Build?
    a. 10%

b. 20%

c. 25%

d. 30%

6. Which of the following is NOT part of the CKAD exam domains?

    a. Application Deployment

    b. Services and Networking

    c. Troubleshooting Kubernetes Clusters

    d. Application Observability and Maintenance

7. Which tool is commonly used to package Kubernetes applications as mentioned in the CKAD curriculum?

    a. Docker Compose

    b. Kustomize

    c. Terraform

    d. Ansible

8. Which Kubernetes concept is used for controlling network traffic between pods as covered in the CKAD exam?

    a. Ingress

    b. NetworkPolicies

    c. ServiceAccount

    d. ConfigMap

9. How many tasks can candidates expect to complete during the CKAD exam?

    a. 10-15

    b. 15-20

    c. 20-25

    d. 25-30

10. During the CKAD exam, candidates can refer to which of the following resources?

    a. Kubernetes official documentation

    b. Any internet resource

    c. Personal notes

    d. External Kubernetes tutorials

Answers

1. c
2. b
3. b
4. c
5. b
6. c
7. b
8. b
9. b
10. a