CKAD

Bulut yerel (cloud-native) uygulama geliştirmenin yükselişiyle birlikte, uygulamaları Kubernetes üzerinde dağıtma, yapılandırma ve sorun giderme becerileri yazılım geliştiriciler için vazgeçilmez hale geldi. İşte tam da bu ihtiyaca yönelik olarak Certified Kubernetes Application Developer (CKAD) sertifikası öne çıkıyor.

CKAD, Cloud Native Computing Foundation (CNCF) ve The Linux Foundation tarafından sunulan, geliştiricilerin Kubernetes üzerinde bulut yerel uygulamaları tasarlama, inşa etme, dağıtma ve sorun giderme konusundaki yetkinliklerini doğrulayan bir sertifikadır. CKA gibi, CKAD de tamamen uygulamalı (hands-on) bir sınavdır. Yani, çoktan seçmeli sorular yerine, size verilen bir Kubernetes ortamında, geliştirici odaklı gerçek dünya senaryolarına dayalı görevleri tamamlamanız beklenir. Bu sayede teorik bilginizin yanı sıra pratik yetenekleriniz de ölçülür.

CKAD

KURUMSAL ÖZEL EĞİTİM

Bu konuyu istediğiniz gibi özelleştirebilirsiniz. Kullandığınız teknolojiye özel uygulamalar ile anlatılmasını, projenize uygun içerik haline getirilmesini ...

Vebende

CKA EĞİTİMİ

Neden CKAD Sertifikası?

- **Geliştirici Odaklı Uzmanlık:** CKAD, bir Kubernetes yöneticisi olmaktan ziyade, Kubernetes'i bir uygulama geliştirici bakış açısıyla kullanma becerilerinizi tesciller. Bu da sizi, Kubernetes ekosisteminde çalışan geliştiriciler arasında öne çıkarır.
- **Artan İş Talebi:** Günümüzde birçok şirket, uygulamalarını Kubernetes üzerinde geliştirip dağıtıyor. CKAD sertifikası, bu alandaki yetkinliğinizi kanıtlayarak iş bulma ve kariyer ilerleme fırsatlarınızı artırır.
- **Daha İyi Uygulama Tasarımı:** Kubernetes'in çalışma prensiplerini ve nesnelerini daha iyi anlayarak, uygulamalarınızı bu platforma en uygun şekilde tasarlamanızı ve geliştirmenizi sağlar.
- **Sorun Giderme Becerisi:** Kendi uygulamalarınızda veya Kubernetes ortamında çıkan sorunları daha hızlı ve etkin bir şekilde teşhis edip çözme yeteneği kazanırsınız.

HAZIRLANMA YÖNETİMİ

- **kubectl Hakimiyeti:** kubectl komutlarını hızlı ve etkin bir şekilde kullanmayı öğrenin. Özellikle imperative (emirsel) komutlar (--dry-run=client -o yaml ile manifest oluşturma) sınavda büyük zaman kazandırır.
- **YAML Bilgisi:** Kubernetes manifestlerini YAML formatında okuma, oluşturma ve düzenleme konusunda pratik yapın.
- **Resmi Dokümantasyon:** Sınav sırasında kullanabileceğiniz kubernetes.io/docs adresini iyi tanıyın. Hızlı arama ve navigasyon becerisi kritiktir.
- **Pratik Ortamlar:** Minikube, KIND gibi yerel Kubernetes ortamlarında veya Killer.sh gibi CKAD simülatörlerinde bolca pratik yapın. Özellikle simülatörler, sınav deneyimine alışmanız için birebirdir.

EĞİTİM İÇERİĞİ

Modül 1: Kubernetes'e Giriş ve Temel Mimarisi

Konteyner Orkestrasyonunun Rolü ve Kubernetes

- Neden konteyner orkestrasyonuna ihtiyaç duyarız? Mikroservisler ve bulut yerlisi (cloud-native) paradigmada Kubernetes'in konumu.

Kubernetes Mimarisine Genel Bakış

- Control Plane (Master) ve Worker Node bileşenleri. API Server, etcd, Scheduler, Controller Manager, Kubelet, Kube-proxy gibi temel bileşenlerin görevleri ve etkileşimleri.

Modül 2: kubectl Komut Satırı Aracı ve Kaynak Yönetimi

kubectl Kurulumu ve Temel Kullanım

- Kubernetes kümesiyle etkileşim kurmak için kullanılan ana aracın tanıtımı. Konfigürasyon dosyaları ve farklı kümelere bağlanma.

Kubernetes Nesnelerini Yönetme

- kubectl get, describe, apply, delete gibi temel komutlarla Pod, Deployment, Service gibi nesnelerin listelenmesi, detaylarının incelenmesi ve yaşam döngülerinin yönetilmesi.

Modül 3: Pod Kavramı ve Yaşam Döngüsü Yönetimi

Kubernetes'in En Küçük Birimi: Pod'lar

- Pod'un yapısı, amacı ve neden bir veya daha fazla konteyner içerdiği. Paylaşılan kaynaklar (ağ, depolama) ve konteynerler arası iletişim.

Pod Yaşam Döngüsü ve Durumları

- Pod'ların Pending, Running, Succeeded, Failed gibi farklı aşamaları ve bu durumların anlamları. Durum değişikliklerini izleme.

Modül 4: Kubernetes Nesnelerini YAML ile Tanımlama

Bildirimsel Yaklaşım ve YAML Formatı

- Kubernetes'te nesneleri tanımlamak için kullanılan YAML (YAML Ain't Markup Language) formatının sözdizimi ve yapısı. apiVersion, kind, metadata, spec, status alanlarının anlamları.

Temel Kaynaklar İçin YAML Dosyaları Yazma

- Basit Pod, Deployment ve Service tanımları için YAML manifest dosyaları oluşturma pratikleri. Temel alanların yapılandırılması.

Modül 5: Deployment Kaynağı ile Stateless Uygulama Dağıtımı

Deployment'ların Rolü ve Yapısı

- Stateless (durum bilgisi tutmayan) uygulamaların yüksek erişilebilirlik ve ölçeklenebilirlikle çalışmasını sağlayan Deployment kaynağının tanıtımı. Replika Set'lerle ilişkisi.

Deployment Oluşturma ve Güncelleme

- Bir Deployment manifesti yazarak uygulama dağıtma. kubectl apply ile güncellemeleri tetikleme. replicas ve selector alanlarının kullanımı.

Modül 6: Service Kaynağı ile Ağ Erişimi Temelleri

Pod Gruplarına Ağ Erişimi Sağlama

- Değişken Pod IP'lerine rağmen stabil bir ağ kimliği ve erişim noktası sağlayan Service kavramı. Service Discovery mekanizması (DNS).

Temel Service Tipleri

- Küme içi erişim için ClusterIP ve Node üzerinden dış erişim için NodePort Service tiplerinin tanıtımı ve temel yapılandırmaları.

Modül 7: Namespaces, Labels ve Selectors ile Organizasyon

Kaynak İzolasyonu: Namespaces

- Büyük kümelerde kaynakları farklı ortamlar, projeler veya takımlar için mantıksal olarak ayırmak amacıyla Namespaces kullanımı. Varsayılan (default) Namespace.

Nesneleri Gruplandırma: Labels ve Selectors

- Kubernetes nesnelerine key-value çiftleri şeklinde etiketler (Labels) ekleyerek organize etme. Belirli Label'lara sahip nesneleri hedeflemek için Selector mekanizması (örneğin, bir Service'in hangi Pod'ları hedefleyeceği).

Modül 8: Temel Log Yönetimi ve İzleme

Çalışan Konteyner Loglarına Erişim

- Bir Pod içindeki konteynerlerin standart çıktılarına ve hatalarına kubectl logs komutu ile erişim. Geçmiş logları görüntüleme.

Basit Log Analizi Teknikleri

- kubectl logs komutunun filtreleme ve takip (follow) seçenekleri. Basit sorun giderme senaryolarında logları kullanma.

Modül 9: Konfigürasyon Yönetimi: ConfigMap ve Secret'lar

Uygulama Konfigürasyonunu Ayırma: ConfigMap

- Uygulama kodundan bağımsız yapılandırma verilerini (veritabanı bağlantı dizeleri, API endpointleri) depolamak için ConfigMap kullanımı.

Hassas Verileri Yönetme: Secret

- Şifreler, API anahtarları, sertifikalar gibi hassas bilgileri güvenli bir şekilde depolamak için Secret kullanımı. Base64 kodlaması.

ConfigMap ve Secret'ları Pod'lara Bağlama

- Bu yapılandırma verilerini Pod'lara environment variable (ortam değişkeni) veya volume (dosya olarak) şeklinde bağlama yöntemleri.

EĞİTİM İÇERİĞİ

Modül 10: Kalıcı Depolama: Persistent Volume ve PVC

Verinin Kalıcılığı: Persistent Volume (PV)
- Konteyner yaşam döngüsünden bağımsız olarak veriyi saklamak için küme genelinde sağlanan depolama kaynakları olan PV kavramı. Farklı depolama tipleri (NFS, iSCSI, Cloud Provider diskleri).

Depolama Talepleri: Persistent Volume Claim (PVC)
- Uygulamaların (Pod'ların) belirli özelliklerde (boyut, erişim modu) kalıcı depolama talebini ifade eden PVC kavramı. PV ve PVC arasındaki binding (eşleşme) süreci.

PVC'yi Pod'a Bağlama
- Tanımlanan bir PVC'yi bir Pod'a volume olarak bağlayarak Pod'un kalıcı depolamaya erişimini sağlama.

Modül 11: Uygulama Sağlığı Kontrolleri: Liveness ve Readiness Probes

Uygulama Sağlığını İzleme
- Kubernetes'in Pod'ların içindeki uygulamaların sağlığını nasıl kontrol ettiği. Yeniden başlatma veya trafiği yönlendirmeme kararları.

Liveness Probe: Yeniden Başlatma Kararları
- Uygulamanın "canlı" olup olmadığını belirleyen Liveness Probe'lar. Başarısız olursa konteyneri yeniden başlatma. HTTP, TCP, Exec Probe tipleri.

Readiness Probe: Trafik Yönlendirme Kararları
- Uygulamanın dışarıdan gelen isteklere yanıt vermeye "hazır" olup olmadığını belirleyen Readiness Probe'lar. Başarısız olursa Service'in o Pod'a trafik göndermemesini sağlama. Kullanım senaryoları.

Modül 12: Batch İşleri: Job ve CronJob Kaynakları

Tek Seferlik Görevler: Job Kaynağı
- Belirli bir işi tamamlayıp sonra sonlanan (batch) uygulamalar için kullanılan Job kaynağı. Başarı koşulları ve yeniden deneme mekanizmaları.

Periyodik Görevler: CronJob Kaynağı
- Belirli bir takvime göre (cron formatında) periyodik olarak Job'ları çalıştıran CronJob kaynağı. Zamanlama, eşzamanlılık ve geçmiş Job yönetimi.

Modül 13: Uygulama Güncelleme Stratejileri: Rolling Update & Rollback

Kesintisiz Güncelleme: Rolling Update
- Yeni sürüm Pod'ların yavaş yavaş eski sürüm Pod'ların yerine geçirilmesiyle uygulama kesintisi olmadan güncelleme yapma stratejisi. Deployment'ın Rolling Update mekanizması.

Hata Durumunda Geri Dönüş: Rollback
- Hatalı bir güncelleme yapıldığında önceki stabil sürüme hızlı ve güvenli bir şekilde geri dönme mekanizması. kubectl rollout undo komutunun kullanımı.

Modül 14: Kaynak Talepleri ve Sınırları (Requests & Limits)

Kaynak Yönetiminin Önemi
- Küme istikrarını sağlamak, kaynakları verimli kullanmak ve uygulamaların öngörülebilir performans göstermesini sağlamak için kaynak yönetiminin rolü.

CPU ve Memory İçin Talepler (Requests)
- Bir konteynerin çalışmak için ihtiyaç duyduğu minimum CPU ve Memory miktarını belirtme. Scheduler'ın bu talepleri nasıl kullandığı.

CPU ve Memory İçin Sınırlar (Limits)
- Bir konteynerin kullanabileceği maksimum CPU ve Memory miktarını belirtme. Sınır aşımlarının etkileri (CPU throttling, Memory OOMKill).

Modül 15: Init Containers ve Kullanım Senaryoları

Uygulama Öncesi Görevler
- Ana uygulama konteyneri/konteynerleri başlamadan önce tamamlanması gereken kurulum, konfigürasyon çekme, veritabanı migrasyonu gibi görevleri yerine getiren Init Containers kavramı.

Init Containers Yaşam Döngüsü
- Init Containers'ın sırayla çalıştığı ve birinin başarısız olması durumunda Pod'un durumu. Tipik kullanım senaryoları ve avantajları.

Modül 16: Service Tipleri (Derinlemesine Analiz)

Farklı Ağ Senaryoları İçin Service Tipleri
- ClusterIP, NodePort, LoadBalancer ve ExternalName Service tiplerinin detaylı mimarileri ve çalışma prensipleri.

Her Tipin Kullanım Amacı ve Yapılandırması
- İç iletişim, geliştirme/test amaçlı dış erişim, bulut ortamında tam entegre dış erişim gibi farklı ihtiyaçlara yönelik doğru Service tipini seçme ve yapılandırma. Ingress Controller kavramına giriş (ileriye hazırlık).

Modül 17: StatefulSet Kaynağına Giriş

Durum Bilgisi Gerektiren Uygulamalar
- Veritabanları, mesaj kuyrukları gibi her replikanın kendi kimliği ve kalıcı depolaması olması gereken Stateful uygulamalar için Deployment yerine kullanılan StatefulSet kaynağına genel bakış.

StatefulSet Temel Özellikleri
- Stabil ağ kimlikleri (hostname) ve artan sırada replika isimleri sağlama. Pod yeniden planlandığında aynı PVC'ye bağlanma garantisi.

EĞİTİM İÇERİĞİ

Modül 18: Ağ Politikaları (Network Policies) Tasarımı

Podlar Arası Ağ Güvenliği
- Varsayılan olarak tüm Pod'ların birbiriyle iletişim kurabildiği Kubernetes ağında, belirli Pod grupları arasındaki ağ trafiğini kısıtlamak için Network Policies kullanımı.

Policy Kurallarının Yapılandırılması
- Ingress (gelen) ve Egress (giden) kuralları tanımlama. Pod ve Namespace Selector'lar ile politikaları uygulama. Ağ eklentilerinin (CNI) Network Policy desteği.

Modül 19: Çoklu Konteyner Tasarımları: Sidecar Pattern Uygulamaları

Sidecar Modelinin Amacı
- Ana uygulama konteyneriyle aynı Pod içinde çalışan, uygulamanın yan görevlerini (log toplama, metrik yayma, güvenlik proxy'si) yerine getiren yardımcı konteynerler (sidecar) kullanma deseni.

Sidecar Pattern Senaryoları
- Logging ajanları, metrik toplayıcılar, servis mesh proxy'leri, dosya senkronizasyonu gibi örnek kullanım durumları. Ortak volume ve ağ paylaşımı.

Modül 20: Uygulama Güvenlik Ayarları: Security Contexts

Pod ve Konteyner Güvenliği
- Bir Pod veya bireysel bir konteyner düzeyinde güvenlik ayarlarını (örneğin, hangi kullanıcı (UID) ve grup (GID) kimliğiyle çalışacağı) yapılandırma.

Privileged Mode, SELinux ve AppArmor
- Kısıtlı veya ayrıcalıklı (privileged) konteyner çalıştırma. İşletim sistemi güvenlik modülleri (SELinux, AppArmor) ile entegrasyon. En düşük ayrıcalık (least privilege) prensibinin Kubernetes'te uygulanması.

Modül 21: Gelişmiş kubectl Kullanımı (Filtering, Jsonpath)

Karmaşık Sorgular ve Filtreleme
- Büyük kümelerde belirli özelliklere sahip nesneleri bulmak için kubectl komutlarının gelişmiş filtreleme seçenekleri (-l, --field-selector).

Çıktıları Biçimlendirme ve Özel Raporlar
- kubectl çıktısını JSON, YAML veya go-template formatında alma. jsonpath sözdizimi ile çıktılardan belirli alanları çekerek özel raporlar oluşturma ve otomasyon için kullanma.

Modül 22: Kubernetes Uygulama Sorun Giderme (Troubleshooting)

Yaygın Sorunları Teşhis Etme
- Pod'ların başlamaması (ImagePullBackOff, CrashLoopBackOff), Service erişim sorunları, depolama bağlantı hataları gibi yaygın uygulama sorunlarını giderme teknikleri.

Etkili Sorun Giderme Araçları
- kubectl describe ile nesne durumunu ve olayları (Events) inceleme, kubectl logs ile konteyner çıktılarını kontrol etme, kubectl exec ile çalışan konteyner içinde komut çalıştırma.

Modül 23: CI/CD Ortamlarına Entegrasyon (Konsept)

Kubernetes ve Modern CI/CD Süreçleri
- Yazılım geliştirme, test, derleme ve dağıtım süreçlerinde (CI/CD pipeline) Kubernetes'in uygulama teslim platformu olarak rolü.

Bildirimsel Yapılandırmanın CI/CD ile Uyumu
- Kubernetes manifest dosyalarının versiyon kontrol sistemlerinde yönetilmesi ve CI/CD pipeline'ları tarafından otomatik olarak dağıtılması. Temel stratejiler (Blue/Green, Canary'ye giriş).

Modül 24: Stateful Uygulamalar ve Yönetimi (Derinlemesine)

StatefulSet Detaylı Yapılandırması
- Pod şablonları, volume claim şablonları, yükseltme stratejileri (Rolling Update), Pod silme politikaları.

Veritabanları ve Dağıtık Sistemleri Çalıştırma
- Kubernetes üzerinde veritabanı (PostgreSQL, MySQL vb.) veya durum bilgisi tutan dağıtık sistemlerin (Kafka, ZooKeeper) StatefulSet kullanarak nasıl çalıştırılacağı ve yönetileceği. Operator desenine giriş.

Modül 25: Kubernetes Events ve İleri Düzey Log Analizi

Küme Düzeyindeki Olayları İzleme
- Kubernetes nesneleriyle ilgili sistem olaylarını (Events) kubectl events veya kubectl describe ile izleme. Olayların anlamlandırılması ve sorun teşhisindeki rolü.

Merkezi Log Toplama ve Analizine Giriş
- Dağıtık uygulamalardan logların toplanması için temel yaklaşımlar (sidecar, node-level agent). Elasticsearch, Fluentd, Kibana (EFK) veya diğer merkezi loglama çözümlerine genel bakış.

Modül 26: Service Accounts ve Temel RBAC (Geliştirici Bakış Açısı)

Uygulama Kimlikleri: Service Accounts
- Kubernetes kümesi içinde çalışan Pod'ların (uygulamaların) API Server'a veya diğer küme kaynaklarına erişimi için kullanılan kimlikler olan Service Accounts.

Kaynak Erişim Kontrolü: Temel RBAC
- Rol Tabanlı Erişim Kontrolü (Role-Based Access Control - RBAC) kavramı. Service Account'lara belirli izinler (Role, ClusterRole) verilmesi ve bu izinlerin RoleBinding veya ClusterRoleBinding ile ilişkilendirilmesi. Geliştirici perspektifinden sık kullanılan izin türleri.

EĞİTİM SÜRESİ

- 3 Gün
- Ders Süresi: 50 dakika
- Eğitim Saati: 10:00 - 17:00

Eğitim formatında eğitimler 50 dakika + 10 dakika moladır. 12:00-13:00 saatleri arasında 1 saat yemek arasındaki verilir. Günde toplam 6 saat eğitim verilir. 3 günlük formatta 18 saat eğitim verilmektedir.

Eğitimler uzaktan eğitim formatında tasarlanmıştır. Her eğitim için teams linkleri gönderilir. Katılımcılar bu linklere girerek eğitimlere katılırlar. Ayrıca farklı remote çalışma araçları da eğitmen tarafından tüm katılımlara sunulur. Katılımcılar bu araçları kullanarak eğitimlere katılırlar.

Eğitim içeriğinde github ve codespace kullanılır. Katılımcılar bu platformlar üzerinden örnek projeler oluşturur ve eğitmenle birlikte eğitimlerde sorulan sorulara ve taleplere uygun iceriğe cevap verir. Katılımcılar bu araçlarla eğitimlerde sorulan sorulara ve taleplere uygun iceriğe cevap verir.
Eğitim yapay zeka destekli kendi kendine öğrenme formasyonu ile tasarlanmıştır. Katılımcılar eğitim boyunca kendi kendine öğrenme formasyonu ile eğitimlere katılırlar. Bu eğitim formatı sayesinde tüm katılımcılar gelecek tüm yaşamlarında kendilerini güncellemeye devam edebilecekler ve her türlü sorunun karşısında çözüm bulabilecekleri yeteneklere sahip olacaklardır.

EĞİTİM YÖNTEMİ

- **Linux & Komut Satırı Temelleri:** Linux ortamında rahatça çalışabilme ve temel terminal komutlarına hakimiyet (örn. ls, cd, cp, mv, rm).
- **Konteyner & Docker Bilgisi:** Konteyner kavramını, Docker'ın temel kullanımı (imaj oluşturma, çalıştırma) ve konteynerleştirme mantığını anlama.
- **YAML Okur Yazarlığı:** Kubernetes manifestlerini oluşturan YAML dosyalarını okuyabilme, anlayabilme ve basit düzenlemeler yapabilme.
- **Temel Uygulama Geliştirme Bilgisi:** Tercihen bir programlama dilinde (Python, Go, Java vb.) temel uygulama geliştirme deneyimi.

KATILIMCILARDAN BEKLENTİLERİMİZ

1. Temel Linux Bilgisi ve Komut Satırı Yetkinliği
Kubernetes'in Linux üzerinde çalışması sebebiyle, katılımcıların temel Linux komutlarına (ls, cd, cp, mv, rm, mkdir vb.) ve komut satırı (terminal) kullanımına hakim olmaları kritik önem taşır. Sınav tamamen terminal üzerinde ilerleyeceği için, Vim veya Nano gibi metin editörlerini etkin kullanabilmek de büyük bir avantaj sağlayacaktır.

2. Konteyner Temelleri ve Docker Bilgisi
Kubernetes'in konteyner orkestrasyonu aracı olması nedeniyle, katılımcıların Docker gibi bir konteyner platformunun temel kavramlarını (imajlar, konteynerler) anlamaları ve temel Docker komutlarını (docker run, docker ps) çalıştırabilmeleri beklenir. Bu bilgi, Kubernetes'teki Pod'lar ve Deployment'lar gibi yapıları kavramayı kolaylaştıracaktır.

3. Temel YAML Bilgisi
Kubernetes kaynak tanımları genellikle YAML formatında yazılır. Katılımcıların YAML'ın temel yapısını (girinti, anahtar-değer çiftleri, listeler) okuyup anlayabilmeleri ve basit YAML dosyalarını oluşturup düzenleyebilmeleri önemlidir.

4. Problem Çözme ve Analitik Düşünme Becerisi
CKA sınavının önemli bir kısmı sorun giderme (Troubleshooting) üzerine kuruludur. Katılımcıların karşılaştıkları problemleri analiz etme, mantıksal adımlar izleyerek kök nedeni bulma ve etkili çözümler üretme becerisine sahip olmaları beklenir. Resmi Kubernetes dokümantasyonunu (kubernetes.io/docs) etkin kullanabilme de bu süreçte hayati rol oynar.

5. Yoğun Pratik Motivasyonu
CKA sertifikasyonunda başarı, sadece dersleri takip etmekle değil, yoğun ve düzenli pratikle gelir. Katılımcılarımızın eğitimde verilen tüm uygulamalı laboratuvar çalışmalarını eksiksiz yapmaları, hatta kendi başlarına (Minikube, KIND gibi araçlarla) veya sınav simülatörleri (örn. Killer.sh) üzerinde bolca pratik yapmaya istekli olmaları en büyük beklentimizdir. Sınavın zaman kısıtlaması nedeniyle, hızlı ve doğru işlem yapma alışkanlığı kazanmak kritik öneme sahiptir.

HEDEF KİTLE

1. DevOps Mühendisleri
Uygulama dağıtımı, otomasyonu ve sürekli entegrasyon/sürekli teslimat (CI/CD) süreçlerinden sorumlu olan DevOps mühendisleri için CKA, Kubernetes ortamlarını daha etkin yönetme ve optimize etme yetkinliği kazandırır.

2. Sistem Yöneticileri (SysAdmins)
Sunucuların, ağ altyapısının ve uygulama platformlarının kurulumu, bakımı ve sorun gidermesinden sorumlu sistem yöneticileri, CKA ile konteynerize edilmiş iş yüklerini ve Kubernetes kümelerini yönetme becerilerini geliştirirler.

3. Bulut Mühendisleri
Genel bulut platformlarında (AWS, Azure, GCP) çalışan ve bulut tabanlı altyapıları tasarlayan, uygulayan ve yöneten mühendisler için CKA, bulut yerel (cloud-native) uygulamaları ve Kubernetes servislerini daha iyi yönetmelerini sağlar.

4. Yazılım Geliştiriciler (DevOps Rolüne Geçmek İsteyenler)
Konteyner ve Kubernetes dünyasına ilgi duyan, geliştirdikleri uygulamaları Kubernetes üzerinde dağıtmak ve yönetmek isteyen yazılım geliştiriciler için CKA, operasyonel süreçleri anlama ve DevOps rollerine geçiş yapma konusunda güçlü bir temel sunar.

5. IT Mimarları ve Çözüm Mimarları
Kurumsal IT altyapısını veya bulut çözümlerini tasarlayan mimarlar için CKA, Kubernetes'in mimari yapısını, güvenlik ve ölçeklenebilirlik prensiplerini derinlemesine anlayarak daha sağlam ve optimize edilmiş çözümler sunmalarına yardımcı olur.

6. Kubernetes Öğrenmek İsteyen Herkes
Yukarıdaki unvanlara sahip olmasa da, güçlü bir Linux temeli ve öğrenme motivasyonu olan herkes CKA eğitimine katılabilir. Özellikle kariyerini bulut bilişim ve modern altyapı yönetimi alanında şekillendirmek isteyen profesyoneller için CKA, sektörde aranan bir yetkinlik kapısıdır.

"

Kurumsal size özel eğitimler hazırlıyoruz. Her eğitim yeni bir heyecan.

Vebende A.Ş.

Kurumsal Terzi Usulü Butik Eğitimler.

Size özel hazırlanan seminer, danışmanlık, eğitim ve hizmetlerimizle yüksek verim elde edin. Paranızı boşa harcamayın. Zaman çok değerli.

Her Eğitim Yeni Bir Heyecan

2000 yılından günümüze devam eden eğitim heyecanı. Uluslararası tecrübe, proje geliştirme deneyimleri, danışmanlıklar ve arge mühendislik deneyimlerimizi ülkemize sunmak için yeni bir konsept tasarladık.
Sizi dinliyor, takımınıza uygun özel içerikler ile hazırlanmış eğitimler hazırlıyoruz. Her eğitim özel bir çalışma, içerik üretimi, uygulama örnekleri hazırlıkları, sunumlar hazırlamayı gerektiriyor. Aynı eğitimi yönetim kadrosuna farklı, teknik ve mimar ekiplerinize farklı içerikler ile hazırlıyoruz. Her eğitim yeni bir macera ve heyecan.

Bizimle İletişime Geçin

iletisim@vebende.com.tr

www.vebende.com.tr

İzmir - Türkiye

Kurumsal
Eğitimler

www.vebende.com.tr

İletişime Geçin

Whatsapp: 0542 5505704
Tel: 0532 512 7811

iletisim@vebende.com.tr

www.vebende.com.tr

Adalet Mah. Manas Bulvarı
No:39 İç Kapı N0: 3107 Bayraklı
İZMİR

[Sürekli Güncellenen Eğitimlerimizi Sitemizden Takip Edin.]

"

Kurumsal terzi usulu size özel eğitimler ve uluslararası deneyime sahip eğitmenler ile çalışmanın keyfi

"

Size Neler Sunuyoruz?

- Kitaptan okunan eğitimler sunmuyoruz. Sizin için özel hazırlanan içerikler ve uygulamalı eğitimler hazırlıyoruz.
- Her eğitim için sunumlar, materyaller, örnek senaryolar size özel hazırlıyoruz.
- Her eğitim için katılımcılara özel github repoları hazırlanıyor. Eğitimden sonra da katılımcılar hayat boyu kaynaklara ulaşmaya devam edebilsinler diye, **"katılımcılar ve eğitmenle birlikte yapılan tüm çalışmaları"** github repolarında saklıyoruz.

Misyonumuz

1. Ülkemizde dünyanın en güncel teknolojilerini kazandırmak. En güncel teknolojilerine hakim takımlarına katkı sağlamak.
2. Alışılmışın dışında en güncel teknolojileri deneyimli eğitmenler ve uygulamalar ile sunmak.
3. Siber güvenlik konusunda hassas çalışmalar sunarak, çağımızın büyük kabusu siber tehditler konusunda deneyimli bilgili takımların oluşmasına katkı sağlamak.