CLOUD, DEVOPS, AND CONTAINERS

Cloud-Native Development on Azure

Level: Practitioner • 2 days (expandable to 3) • Virtual, In-person

Overview

Cloud-native is one of the industry's most abused terms, but underneath the buzzword is a real discipline: building applications that assume the platform, expect failure, scale elastically, and lean on managed services instead of reinventing them. On Azure the pieces are all there; the hard part is knowing which pieces to choose and how they compose into a system.

This is a hands-on, practitioner course. Rather than survey every Azure service with "cloud" in the name, it goes deep on a working set: containers with Azure Container Registry and Azure Container Apps, messaging with queues and events, and the resilience and observability patterns that make distributed systems dependable. The gradient runs from cloud-native principles, through building and scaling containerized services, to inter-service communication, resilience, and operations. Every module includes a lab, and each module builds on the one before.

Who Should Attend

- Developers building new distributed applications on Azure
- Teams modernizing monoliths toward services and containers
- Architects who want hands-on grounding in Azure's cloud-native stack

Prerequisites

- Working proficiency in at least one programming language
- Basic Azure familiarity; true beginners should take *Microsoft Azure Fundamentals (AZ-900)* first
- Basic container familiarity helps; *Introduction to Docker and Containers* covers it

What You Will Learn

- Explain the cloud-native principles behind the buzzword: twelve-factor design, statelessness, and elasticity
- Build and ship containerized services with Azure Container Registry and Azure Container Apps
- Design services that scale out cleanly, with state and sessions handled deliberately
- Choose the right Azure messaging service: Storage Queues, Service Bus, or Event Grid
- Apply resilience patterns: retries, timeouts, circuit breakers, and idempotency
- Instrument and operate a distributed application with Application Insights

Course Outline

Day one: principles and running services

- Cloud-Native, Concretely
  - What cloud-native actually means: twelve-factor design and its limits
  - Monolith first, services when they pay: an honest starting point
  - The Azure cloud-native map: which service for which job

  - Lab: assess a sample application against cloud-native principles and plan its evolution
- Containers on Azure
  - Building images and pushing to Azure Container Registry
  - Azure Container Apps: serverless containers without running a cluster
  - Configuration, secrets, and managed identity for containers
  - Lab: containerize a service and run it on Azure Container Apps
- Scaling and Statelessness
  - Designing services to scale out: state, sessions, and caching
  - Autoscaling on demand and on queue depth
  - Lab: load-test the service, watch it scale, and fix what breaks

Day two: communication, resilience, and operations

- Services Talking to Services
  - Synchronous versus asynchronous communication, and why async wins so often
  - Storage Queues, Service Bus, and Event Grid: choosing well
  - Lab: decouple two services with a queue and prove the system survives a consumer outage
- Building In Resilience
  - Failure is normal: retries, timeouts, and circuit breakers
  - Idempotency, and thinking clearly about duplicate work
  - Lab: inject failures and verify the system recovers without losing work
- Observing and Operating
  - Distributed tracing across services with Application Insights
  - Health checks, alerts, and dashboards that mean something
  - Lab: trace a request across the whole system and diagnose an induced slowdown

Extended Version

The three-day version keeps the same gradient and adds:

- Where Kubernetes fits: when to step up to AKS, connecting to *Deploying to Azure Kubernetes Service (AKS)*
- Data in distributed systems: per-service data and Cosmos DB in brief
- CI/CD for cloud-native services, connecting to *CI/CD with GitHub Actions*
- A capstone: build and operate a small multi-service application end to end on Azure