# *AW260 Developing Serverless Solutions on AWS*

**Kurzbeschreibung:**

In diesem Kurs **AW260 Developing Serverless Solutions on AWS** erhalten Entwickler Einblicke in Best Practices für die Entwicklung serverloser Anwendungen mit AWS Lambda und anderen Services der serverlosen Plattform von AWS und wenden diese praktisch an.

Sie nutzen AWS-Frameworks, um im Rahmen von praktischen Übungen eine serverlose Anwendung bereitzustellen – von einfachen bis hin zu komplexeren Themen. Während des gesamten Kurses **AW260 Developing Serverless Solutions on AWS** verwenden Sie AWS-Dokumentation, um praxisgerechte Methoden zum eigenständigen Lernen und Problemlösen über das Klassenzimmer hinaus zu entwickeln.

Dieser Kurs **AW260 Developing Serverless Solutions on AWS** beinhaltet Präsentationen, praktische Übungen (Hands-on Labs), Demonstrationen, Videos, Wissensabfragen und Gruppenübungen.

**Zielgruppe:**

Dieser Kurs **AW260 Developing Serverless Solutions on AWS** richtet sich an:

- Entwickler mit ersten Kenntnissen zu serverlosen Architekturen und Erfahrung in der Entwicklung in der AWS Cloud

**Voraussetzungen:**

Um an dem Kurs **AW260 Developing Serverless Solutions on AWS** bei qSkills teilnehmen zu können, sollten Sie das folgende AWS-Training besucht haben:

- AW230 Developing on AWS

Darüber hinaus sollten Sie folgende Voraussetzungen erfüllen:

- Vertraut sind mit den Grundlagen der AWS Cloud-Architektur
- Über Kenntnisse in der Entwicklung von Anwendungen auf AWS verfügen, wie sie dem Kurs „Developing on AWS“ entsprechen
- Kenntnisse aus den folgenden digitalen Trainingsmodulen mitbringen:
    - AWS Lambda Foundations
    - Amazon API Gateway for Serverless Applications

**Sonstiges:**

**Dauer:** 3 Tage

**Preis:** 1995 Euro plus Mwst.

**Ziele:**

In diesem Kurs **AW260 Developing Serverless Solutions on AWS** lernen Sie:

- Eventbasierte Best Practices bei der Entwicklung serverloser Anwendungen mit geeigneten AWS-Services anzuwenden
- Herausforderungen und Abwägungen beim Übergang zur serverlosen Entwicklung zu identifizieren und Empfehlungen für Ihre Entwicklungsorganisation und -umgebung auszusprechen
- Serverlose Anwendungen mithilfe von Mustern zu entwickeln, die AWS-verwaltete Services miteinander verbinden, unter Berücksichtigung von Servicecharakteristika wie Kontingente, verfügbare Integrationen, Aufrufmodell, Fehlerbehandlung und Ereignisquellen-Payload
- Verfügbare Optionen zur Umsetzung von Infrastructure as Code zu vergleichen, darunter AWS CloudFormation, AWS Amplify, AWS Serverless Application Model (AWS SAM) und AWS Cloud Development Kit (AWS CDK)
- Best Practices beim Schreiben von Lambda-Funktionen anzuwenden – einschließlich Fehlerbehandlung, Logging, Wiederverwendung von Umgebungen, Nutzung von Layers, Zustandslosigkeit, Idempotenz sowie Konfiguration von Parallelität und Arbeitsspeicher
- Best Practices zur Observability und zum Monitoring in serverlosen Anwendungen umzusetzen
- Sicherheitsbest Practices für serverlose Anwendungen anzuwenden
- Zentrale Skalierungsaspekte serverloser Anwendungen zu identifizieren und diesen geeignete Methoden, Tools oder Best Practices zur Steuerung zuzuordnen
- AWS SAM, AWS CDK und AWS Developer Tools zur Konfiguration eines CI/CD-Workflows zu nutzen und die Bereitstellung serverloser Anwendungen zu automatisieren
- Eine Liste serverloser Ressourcen zu erstellen und aktiv zu pflegen, die Sie bei der Weiterentwicklung und Einbindung in die Serverless-Community unterstützen

**Inhalte/Agenda:**

- ♦ **Einführung**
  - ♦ ◊ Vorstellung der zu entwickelnden Anwendung
    ◊ Zugang zu Kursressourcen (Teilnehmerunterlagen, Lab-Anleitungen, Online-Kursergänzung)
- ♦ ◊
- ♦ **Thinking Serverless**
  - ♦ ◊ Best Practices zur Entwicklung moderner serverloser Anwendungen
    ◊ Eventgesteuertes Design
    ◊ AWS-Services für eventgesteuerte serverlose Anwendungen
- ♦ ◊
- ♦ **API-gesteuerte Entwicklung und synchrone Ereignisquellen**
  - ♦ ◊ Eigenschaften klassischer API-basierter Webanwendungen mit Request/Response-Modell
    ◊ Rolle von Amazon API Gateway in serverlosen Architekturen
    ◊ Try-it-out-Übung: HTTP API-Endpunkt mit Lambda-Funktion einrichten
    ◊ Überblick zu API-Typen (REST/HTTP, WebSocket, GraphQL)
- ♦ ◊
- ♦ **Einführung in Authentifizierung, Autorisierung und Zugriffskontrolle**
  - ♦ ◊ Authentifizierung vs. Autorisierung
    ◊ Optionen zur Authentifizierung bei APIs via API Gateway
    ◊ Amazon Cognito in serverlosen Anwendungen
    ◊ Cognito User Pools vs. föderierte Identitäten
- ♦ ◊
- ♦ **Frameworks für serverlose Bereitstellungen**
  - ♦ ◊ Imperative vs. deklarative Programmierung für Infrastructure as Code
    ◊ Vergleich: CloudFormation, AWS CDK, Amplify, AWS SAM
    ◊ Funktionen von AWS SAM und der AWS SAM CLI für lokale Emulation und Tests
- ♦ ◊
- ♦ **Komponenten mit Amazon EventBridge und Amazon SNS entkoppeln**
  - ♦ ◊ Entwicklungsaspekte bei der Nutzung asynchroner Ereignisquellen
    ◊ Funktionen und Anwendungsfälle von Amazon EventBridge
    ◊ Try-it-out-Übung: Benutzerdefinierten EventBridge-Bus und Regel erstellen
    ◊ Vergleich: Amazon SNS vs. EventBridge
    ◊ Try-it-out-Übung: Amazon SNS-Thema mit Filter konfigurieren
- ♦ ◊
- ♦ **Eventbasierte Entwicklung mit Queues und Streams**
  - ♦ ◊ Entwicklungsaspekte beim Einsatz von pollingbasierten Ereignisquellen für Lambda
    ◊ Unterschiede zwischen Queues und Streams als Ereignisquellen
    ◊ Konfigurationsempfehlungen für Amazon SQS oder Amazon Kinesis Data Streams als Lambda-Ereignisquelle
    ◊ Try-it-out-Übung: Amazon SQS-Warteschlange mit Dead-Letter Queue als Ereignisquelle einrichten
- ♦ ◊
- ♦ **Hands-On Labs**
  - ♦ ◊ Lab 1: Bereitstellung einer einfachen serverlosen Anwendung
    ◊ Lab 2: Message Fan-Out mit Amazon EventBridge
- ♦ ◊
- ♦ **Gute Lambda-Funktionen schreiben**
  - ♦ ◊ Einfluss des Lambda-Lebenszyklus auf den Code
    ◊ Best Practices für Lambda-Funktionen
    ◊ Funktionen konfigurieren
    ◊ Code, Versionen und Aliasse
    ◊ Try-it-out-Übung: Lambda-Funktion konfigurieren und testen
    ◊ Fehlerbehandlung in Lambda
    ◊ Umgang mit partiellen Fehlern bei Queues und Streams
- ♦ ◊
- ♦ **Orchestrierung mit Step Functions**
  - ♦ ◊ AWS Step Functions in serverlosen Architekturen
    ◊ Try-it-out-Übung: Step Functions States
    ◊ Callback-Muster
    ◊ Standard vs. Express Workflows
    ◊ Direkte Integrationen mit Step Functions
    ◊ Try-it-out-Übung: Fehlerbehebung bei Standard Step Functions Workflow
- ♦ ◊
- ♦ **Observability und Monitoring**
  - ♦ ◊

◊ Die drei Säulen der Observability
◊ Amazon CloudWatch Logs und Logs Insights
◊ Effektives Logging
◊ Try-it-out-Übung: Logs interpretieren
◊ AWS X-Ray für Observability
◊ Try-it-out-Übung: X-Ray aktivieren und Traces interpretieren
◊ CloudWatch-Metriken und Embedded Metrics Format
◊ Try-it-out-Übung: Metriken und Alarme
◊ Try-it-out-Übung: ServiceLens

- ♦ ◊
- ♦ **Hands-On Labs**
  - ♦ ◊ Lab 3: Workflow-Orchestrierung mit AWS Step Functions
    ◊ Lab 4: Observability und Monitoring
- ♦ ◊
- ♦ **Sicherheit serverloser Anwendungen**
  - ♦ ◊ Sicherheitsbest Practices für serverlose Anwendungen
    ◊ Sicherheit auf allen Ebenen
    ◊ API Gateway und Anwendungssicherheit
    ◊ Lambda und Anwendungssicherheit
    ◊ Schutz von Daten in serverlosen Datenspeichern
    ◊ Auditing und Nachvollziehbarkeit
- ♦ ◊
- ♦ **Skalierung in serverlosen Anwendungen**
  - ♦ ◊ Skalierungsaspekte in serverlosen Anwendungen
    ◊ Skalierungssteuerung über API Gateway
    ◊ Lambda-Concurrency-Management
    ◊ Skalierung unterschiedlicher Ereignisquellen mit Lambda
- ♦ ◊
- ♦ **Automatisierung der Bereitstellungspipeline**
  - ♦ ◊ Bedeutung von CI/CD für serverlose Anwendungen
    ◊ Tools in einer serverlosen Pipeline
    ◊ AWS SAM-Funktionen für serverlose Bereitstellung
    ◊ Best Practices für Automatisierung
    ◊ Kursabschluss
- ♦ ◊
- ♦ **Hands-On Labs**
  - ♦ ◊ Lab 5: Absicherung serverloser Anwendungen
    ◊ Lab 6: Serverloses CI/CD auf AWS
    ♦ ◊