Google Cloud
Professional Cloud Developer

Introduction

A Professional Cloud Developer builds scalable and highly available applications using Google-recommended practices and tools.

This individual has experience with cloud-native applications, developer tools, managed services, and next-generation databases. A Professional Cloud Developer also has proficiency with at least one general-purpose programming language and is skilled at producing meaningful metrics and logs to debug and trace code.

The Professional Cloud Developer exam assesses your ability to:

- Design highly scalable, available, and reliable cloud-native applications
- Build and test applications
- Deploy applications

Course Objective

- Design highly scalable, available, and reliable cloud-native applications
- Build and test applications
- Deploy applications
- Integrate Google Cloud Platform services
- Manage application performance monitoring

Course Outline

Section 1: Designing highly scalable, available, and reliable cloud-native applications

1.1 Designing high-performing applications and APIs. Considerations include:

- Microservices
- Scaling velocity characteristics/tradeoffs of IaaS (infrastructure as a service) vs. CaaS (container as a service) vs. PaaS (platform as a service)
- Geographic distribution of Google Cloud services (e.g., latency, regional services, zonal services)
- Defining a key structure for high-write applications using Cloud Storage, Cloud Bigtable, Cloud Spanner, or Cloud SQL
- User session management
- Caching solutions
- Deploying and securing API services
- Loosely coupled asynchronous applications (e.g., Apache Kafka, Pub/Sub)
- Graceful shutdown on platform termination
- Google-recommended practices and documentation

1.2 Designing secure applications. Considerations include:

- Implementing requirements that are relevant for applicable regulations (e.g., data wipeout)
- Security mechanisms that protect services and resources
- Security mechanisms that secure/scan application binaries and manifests
- Storing and rotating application secrets and keys (e.g., Cloud KMS, HashiCorp Vault)
- Authenticating to Google services (e.g., application default credentials, JSON Web Token (JWT), OAuth 2.0)
- IAM roles for users/groups/service accounts

- Securing service-to-service communications
- Running services with least privileged access (e.g., Workload Identity)
- Certificate-based authentication (e.g., SSL, mTLS)
- Google-recommended practices and documentation

1.3 Managing application data. Considerations include:

- Defining database schemas for Google-managed databases (e.g., Firestore, Cloud Spanner, Cloud Bigtable, Cloud SQL)
- Choosing data storage options based on use case considerations, such as:
- Time-limited access to objects
- Data retention requirements
- Structured vs. unstructured data
- Strong vs. eventual consistency
- Data volume
- Frequency of data access in Cloud Storage
- Google-recommended practices and documentation

1.4 Application modernization. Considerations include:

- Using managed services
- Refactoring a monolith to microservices
- Designing stateless, horizontally scalable services
- Google-recommended practices and documentation

Section 2: Building and testing applications

2.1 Setting up your local development environment. Considerations include:

- Emulating Google Cloud services for local application development
- Creating Google Cloud projects
- Using the command-line interface (CLI), Google Cloud Console, and Cloud Shell tools
- Using developer tooling (e.g., Cloud Code, Skaffold)

2.2 Writing efficient code. Considerations include:

- Algorithm design
- Modern application patterns
- Software development methodologies
- Debugging and profiling code

2.3 Testing. Considerations include:

- Unit testing
- Integration testing
- Performance testing
- Load testing

2.4 Building. Considerations include:

- Source control management
- Creating secure container images from code
- Developing a continuous integration pipeline using services (e.g., Cloud Build, Container Registry) that construct deployment artifacts
- Reviewing and improving continuous integration pipeline efficiency

Section 3: Deploying applications

3.1 Recommend appropriate deployment strategies using the appropriate tools (e.g., Cloud Build, Spinnaker, Tekton, Anthos Configuration Manager) for the target computing environment (e.g., Compute Engine, Google Kubernetes Engine). Considerations include:

- Blue/green deployments
- Traffic-splitting deployments
- Rolling deployments
- Canary deployments

3.2 Deploying applications and services on Compute Engine. Considerations include:

- Installing an application into a virtual machine (VM)
- Managing service accounts for VMs
- Bootstrapping applications
- Exporting application logs and metrics
- Managing Compute Engine VM images and binaries

3.3 Deploying applications and services to Google Kubernetes Engine (GKE). Considerations include:

- Deploying a containerized application to GKE
- Managing Kubernetes RBAC and Google Cloud IAM relationships
- Configuring Kubernetes namespaces
- Defining workload specifications (e.g., resource requirements)
- Building a container image using Cloud Build
- Configuring application accessibility to user traffic and other services
- Managing container lifecycle
- Define Kubernetes resources and configurations

3.4 Deploying a Cloud Function. Considerations include:

- Cloud Functions that are triggered via an event from Google Cloud services (e.g., Pub/Sub, Cloud Storage objects)
- Cloud Functions that are invoked via HTTP
- Securing Cloud Functions

3.5 Using service accounts. Considerations include:

- Creating a service account according to the principle of least privilege
- Downloading and using a service account private key file

Section 4: Integrating Google Cloud services

4.1 Integrating an application with data and storage services. Considerations include:

- Read/write data to/from various databases (e.g., SQL)
- Connecting to a data store (e.g., Cloud SQL, Cloud Spanner, Firestore, Cloud Bigtable)
- Writing an application that publishes/consumes data asynchronously (e.g., from Pub/Sub)
- Storing and retrieving objects from Cloud Storage

4.2 Integrating an application with computing services. Considerations include:

- Implementing service discovery in GKE and Compute Engine
- Reading instance metadata to obtain application configuration
- Authenticating users by using OAuth2.0 Web Flow and Identity-Aware Proxy
- Authenticating to Cloud APIs with Workload Identity
-

4.3 Integrating Cloud APIs with applications. Considerations include:

- Enabling a Cloud API
- Making API calls using supported options (e.g., Cloud Client Library, REST API or gRPC, APIs Explorer) taking into consideration:
- Batching requests
- Restricting return data
- Paginating results
- Caching results
- Error handling (e.g., exponential backoff)
- Using service accounts to make Cloud API calls

Section 5: Managing application performance monitoring

5.1 Managing Compute Engine VMs. Considerations include:

- Debugging a custom VM image using the serial port
- Diagnosing a failed Compute Engine VM startup
- Sending logs from a VM to Cloud Logging
- Viewing and analyzing logs
- Inspecting resource utilization over time

5.2 Managing Google Kubernetes Engine workloads. Considerations include:

- Configuring logging and monitoring
- Analyzing container lifecycle events (e.g., CrashLoopBackOff, ImagePullErr)
- Viewing and analyzing logs
- Writing and exporting custom metrics
- Using external metrics and corresponding alerts
- Configuring workload autoscaling

5.3 Troubleshooting application performance. Considerations include:

- Creating a monitoring dashboard
- Writing custom metrics and creating log-based metrics
- Using Cloud Debugger
- Reviewing stack traces for error analysis
- Exporting logs from Google Cloud
- Viewing logs in the Google Cloud Console
- Reviewing application performance (e.g., Cloud Trace, Prometheus, OpenTelemetry)
- Monitoring and profiling a running application
- Using documentation, forums, and Google Cloud support

Prerequisites

- General knowledge of IT architecture
- Software development experience

Recommended Experience

3+ years of industry experience including 1+ years designing and managing solutions using Google Cloud.

Target Audience

- Software developers who want to build applications on Google Cloud Platform
- People preparing for the Google Professional Cloud Developer exam
- Software Engineer
- Applications Developer
- Sr. Engineer
- Cloud Developer

Duration

3 days training course