Google Cloud

Professional Cloud Developer

認定試験ガイド

Professional Cloud Developer とは、Google が推奨するツールとベスト プラクティスを使用して、スケーラブルかつ安全で可用性の高いアプリケーションを構築し、デプロイするデベロッパーのことです。クラウドネイティブ アプリケーション、コンテナ化アプリケーション、API、デベロッパー ツール、オーケストレーション ツール、マネージド サービス、テスト戦略、サーバーレス プラットフォーム、次世代データベースに関する経験を有している必要があります。さらに、少なくとも 1 つの汎用プログラミング言語に精通しており、コードに計測機能を追加して指標、ログ、トレースを生成できることが求められます。

*注: この試験はコーディングのスキルを直接評価するものではありません。 Google Cloud のサービスや推奨方法を活用して、スケーラブルで可用性の高いアプリケーションの構築、テスト、デプロイ、管理を行う能力を評価することに重点を置いています。少なくとも 1 つの汎用コーディング言語に精通していれば、コード スニペットを使用して質問を理解できるはずです。

セクション 1: スケーラビリティ、可用性、信頼性に優れたクラウド ネイティブなアプリケーションの設計（試験内容の約 33%）

1.1 高パフォーマンスのアプリケーションと API を設計する。考慮事項:

- マイクロサービス アーキテクチャ
- ユースケースと要件に基づいた適切なプラットフォームの選択（IaaS [Infrastructure as a Service]、CaaS [Container as a Service]、PaaS [Platform as a Service]、FaaS [Function as a Service] など）
- アプリケーションのモダナイゼーション（コンテナ化など）
- Google Cloud サービスの地理的な分散に関する理解（レイテンシ、リージョン サービス、ゾーンサービスなど）
- ユーザー セッション管理
- キャッシュ ソリューション
- HTTP REST か gRPC（Google リモート プロシージャ コール）か
- API サービス（Apigee など）が提供する Service Control 機能の組み込み
- 疎結合の非同期アプリケーション（Apache Kafka、Pub/Sub、Eventarc など）
- コードへの計測機能の追加による指標、ログ、トレースの生成
- 費用とリソースの最適化
- エラー、障害、スケーリング イベントの適切な処理

1.2 安全なアプリケーションを設計する。考慮事項:

- 適用される規制要件に対応するためのデータ ライフサイクルとデータ所在地の実装
- 脆弱性を特定し、サービスとリソースを保護するセキュリティ メカニズム（Identity-Aware Proxy [IAP]、Web Security Scanner など）
- アプリケーションのバイナリ、依存関係、マニフェストを保護またはスキャンするセキュリティ メカニズム（Container Analysis など）
- アプリケーション シークレットと暗号化キーの保存、アクセス、ローテーション（例: Secret Manager、Cloud Key Management Service）
- Google Cloud サービスに対する認証（アプリケーションのデフォルト認証情報、JSON Web Token [JWT]、OAuth 2.0 など）
- Identity Platform を使用したエンドユーザー アカウント管理と認証
- ユーザー、グループ、サービス アカウントに対する Identity and Access Management（IAM）のロール
- サービス間の通信の保護（サービス メッシュ、Kubernetes ネットワーク ポリシー、Kubernetes Namespace など）
- キーレスの最小権限アクセスを使ったサービスの実行（Workload Identity、Workload Identity 連携など）
- 証明書ベースの認証（SSL、mTLS など）
- ソフトウェア アーティファクトのためのサプライ チェーンレベル（SLSA）

1.3 アプリケーション データのストレージ オプションを選択する。考慮事項:

- オブジェクトへの時間制限付きアクセス
- データの保持に関する要件
- 構造化データか非構造化データか（例: SQL か NoSQL か）
- 強整合性か結果整合性か
- データ容量
- データのアクセス パターン
- オンライン トランザクション処理（OLTP）かデータ ウェアハウジングか

セクション 2: アプリケーションの構築とテスト（試験内容の約 26%）

2.1 ローカル開発環境を設定する。考慮事項:

- ローカル アプリケーション開発を対象とした Google Cloud サービスのエミュレート
- Google Cloud コンソール、Google Cloud SDK、Cloud Shell、Cloud Workstations の使用

- デベロッパー ツールの使用（一般的な IDE、Cloud Code、Skaffold など）
- Google Cloud サービスに対する認証（Cloud SQL Auth Proxy、AlloyDB Auth プロキシなど）

2.2 構築する。考慮事項:

- ソース コントロール管理
- コードからの安全なコンテナ イメージの作成
- デプロイメント アーティファクトを構成するサービス（Cloud Build、Artifact Registry など）を使用した、継続的インテグレーション パイプラインの開発
- コードとテストビルドの最適化

2.3 テストする。考慮事項:

- 単体テスト
- 統合テスト（エミュレータの使用を含む）
- パフォーマンス テスト
- 負荷テスト
- 障害テストとカオス エンジニアリング

セクション 3: アプリケーションのデプロイ（試験内容の約 19%）

3.1 適切な機能ロールアウト戦略を適用する。考慮事項:

- A/B テスト
- フィーチャー トグル
- 下位互換性
- バージョニング API（Apigee など

3.2 アプリケーションをサーバーレス コンピューティング環境にデプロイする。考慮事項:

- ソースコードからのアプリケーションのデプロイ
- トリガーを使用した関数の呼び出し
- イベント レシーバの設定（Eventarc、Pub/Sub など）
- アプリケーション API の公開と保護（Apigee など）

3.3 Google Kubernetes Engine（GKE）にアプリケーションとサービスをデプロイする。考慮事項:

- コンテナ化したアプリケーションの GKE へのデプロイ
- Kubernetes のロールベース アクセス制御（RBAC）と IAM の統合
- ワークロードの仕様の定義（リソース要件など）
- Cloud Build を使用したコンテナ イメージの作成

セクション 4: アプリケーションと Google Cloud サービスの統合（試験内容の約 22%）

4.1 アプリケーションにデータサービスとストレージ サービスを統合する。考慮事項:

- データストアへの接続の管理（Cloud SQL、Firestore、Bigtable、Cloud Storage など）
- さまざまなデータストアのデータの読み取りと書き込み
- （Pub/Sub やストリーミング データソースなどから）非同期でデータを公開または使用するアプリケーションの作成
- Workflows、Eventarc、Cloud Tasks、Cloud Scheduler を使用したアプリケーション サービスのオーケストレーション

4.2 アプリケーションと Google Cloud API を統合する。考慮事項:

- Google Cloud のサービスの有効化
- 以下の点を考慮し、サポートされているオプション（Cloud クライアント ライブラリ、REST API または gRPC、API Explorer など）を使用した API の呼び出し
    - 一括処理リクエスト
    - 戻りデータの制限
    - 結果のページ分け
    - 結果のキャッシュ保存
    - エラー処理（指数バックオフなど）
- サービス アカウントを使用した Cloud API 呼び出し
- Google Cloud のオペレーション スイートとの統合