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、スケーラブルで安全なアプリケーションを構築し、構成するデベロッパーのことです。クラウドネイティブ アプリケーションの設計から高度な ML 機能の統合まで、開発ライフサイクル全体に精通しています。また、生成 AI API を利用してインテリジェントなユーザー エクスペリエンスを構築する責任も負っています。さらに、AI を活用した開発ツール（AI コーディング アシスタント、コンテキスト エンジニアリング、自動デバッグ エージェントなど）を利用して、デリバリーを加速し、コードの品質を向上させるのも Professional Cloud Developer の役目です。

セクション **1:** スケーラビリティ、安全性、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 **32%**）

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。以下のような点を考察します。

- 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（Compute Engine、Google Kubernetes Engine、Cloud Run など）
- アプリケーション コンテナの構築とリファクタリングおよび Cloud Run と GKE へのデプロイ
- Google Cloud サービスがどのように地理的に分散されるか（レイテンシ、リージョン サービス、ゾーンサービスなど）の理解
- ロードバランサのユースケースの理解
- セッション アフィニティを有効にしてコンテンツ配信のパフォーマンスを向上
- キャッシュ ソリューション（Memorystore など）の実装
- API の作成とデプロイ（HTTP REST、gRPC [リモート プロシージャ コール] など）
- アプリケーションのレート制限、認証、オブザーバビリティの使用（Apigee、Cloud API Gateway など）
- 非同期またはイベントドリブンなアプローチ（Eventarc、Pub/Sub など）を使用したアプリケーションの統合
- ワークロードのリソース要件の定義
- 費用とリソース使用量の最適化
- ゾーンとリージョンのフェイルオーバー モデルをサポートするデータ レプリケーションの理解
- Cloud Run または GKE 上の新しいサービスでのトラフィック分割戦略（段階的な展開、ロールバック、A/B テストなど）の使用
- 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

1.2 安全なアプリケーションを設計する。以下のような点を考察します。

- データの保持と組織のポリシーの実装（Cloud Storage オブジェクトのライフサイクル管理、Cloud Storage の保持ポリシーの使用とロックなど）
- 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）の使用
- Artifact Analysis や Security Command Center で特定されたものを含め、脆弱性への対応と解決
- アプリケーション シークレット、認証情報、暗号鍵の保存、アクセス、ローテーション（Secret Manager、Cloud Key Management Service、Workload Identity 連携など）
- 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、Cloud SQL Auth Proxy、AlloyDB Auth Proxy、Identity Platform、WIF など）
- サービス アカウントの Identity and Access Management（IAM）ロールを使用したクラウド リソースの保護
- 安全なサービス間通信（Cloud Service Mesh、Kubernetes ネットワーク ポリシー、ダイレクト VPC 外向き、プライベート サービスの接続など）の組み込み
- 最小権限アクセスでのサービスの実行
- Binary Authorization を使用したアプリケーション アーティファクトの保護

1.3 データを保存してアクセスする。以下のような点を考察します。

- データ量とパフォーマンス要件に基づいた適切なストレージ システムの選択
- 構造化データベース（AlloyDB、Spanner など）と非構造化データベース（Bigtable、Firestore など）に適したスキーマの設計
- AlloyDB、Bigtable、Cloud SQL、Spanner、Cloud Storage の結果整合性と強整合性のレプリケーションがもたらす影響の理解
- Cloud Storage オブジェクトへのアクセスを許可する署名付き URL の作成
- 分析と AI / ML ワークロードのための BigQuery へのデータの書き込み

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 **23%**）

2.1 開発環境を設定する。以下のような点を考察します。

- ローカル アプリケーション開発とローカル単体テストのための Google Cloud CLI を使用した Google Cloud サービスのエミュレート
- Google Cloud コンソール、Cloud SDK、Cloud Code、Gemini Cloud Assist、Cloud Shell、Cloud Workstations の使用

- 適切な統合（Cloud SDK、AI ツール [コーディング アシスタント、MCP サーバー] など）を使用した IDE の構成

2.2 構築する。以下のような点を考察します。

- Cloud Build と Artifact Registry を使用してソースコードからコンテナをビルドして保存
- Cloud Build での来歴の構成（例: Binary Authorization）

2.3 テストする。以下のような点を考察します。

- AI コーディング アシスタントを利用した単体テストの作成
- Cloud Build での自動統合テストの実行

セクション **3:** デプロイのためのクラウドネイティブ アプリケーションの構成（試験内容の約 **24%**）

3.1 アプリケーションを Cloud Run にデプロイする。以下のような点を考察します。

- 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（Eventarc、Pub/Sub など）を使用した Cloud Run サービスの呼び出し
- イベント レシーバの設定（Eventarc、Pub/Sub など）
- アプリケーションの API のバージョニング、公開、保護（Apigee など）

3.2 コンテナを GKE にデプロイする。以下のような点を考察します。

- コンテナ化されたアプリケーションのデプロイ
- Kubernetes ヘルスチェックを実装してアプリケーションの可用性を向上
- HorizontalPodAutoscaler 属性（スケーリング、指標）の組み込み

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 **21%**）

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。以下のような点を考察します。

- さまざまな Google Cloud データストア（Cloud SQL、Firestore、Cloud Storage など）への接続の管理
- さまざまな Google Cloud データソースとの間でのデータの読み書き

- メッセージング サービスを使用したデータをパブリッシュおよび利用するアプリケーションの作成

4.2 Google Cloud API を使用する。以下のような点を考察します。

- 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て Cloud API 呼び出しを行う

4.3 トラブルシューティングとオブザーバビリティ。以下のような点を考察します。

- Google Cloud Observability の指標、ログ、トレースを使用してトラブルシューティングを容易にするためのコードの計測
- Google Cloud Observability を使用した問題の特定と解決
- Error Reporting を使用したアプリケーションの問題の管理
- トレース ID を使用してサービス間でトレーススパンを関連付ける
- AI を活用したオブザーバビリティの使用