Google Cloud Run 実装マニュアル - 初心者向け

# 📋 目次

1. [前提条件](#前提条件)

2. [Google Cloud プロジェクトの準備](#google-cloud-プロジェクトの準備)

3. [必要なAPIの有効化](#必要なapiの有効化)

4. [サービスアカウントの作成と権限設定](#サービスアカウントの作成と権限設定)

5. [Secret Manager の設定](#secret-manager-の設定)

6. [Artifact Registry の準備](#artifact-registry-の準備)

7. [Cloud Build の設定](#cloud-build-の設定)

8. [Cloud Run のデプロイ](#cloud-run-のデプロイ)

9. [ドメインとSSL証明書の設定](#ドメインとssl証明書の設定)

10. [監視とログの設定](#監視とログの設定)

11. [トラブルシューティング](#トラブルシューティング)

# 🎯 前提条件

## 必要なもの

- Google Cloud アカウント（無料枠でも可）

- クレジットカード（認証用、課金は最小限）

- ローカルマシンにインストール済み：

- [Google Cloud CLI](https://cloud.google.com/sdk/docs/install)

- [Docker](https://docs.docker.com/get-docker/)

- [Git](https://git-scm.com/downloads)

## 推奨環境

- OS: Windows 10/11, macOS, Linux

- メモリ: 8GB以上

- ストレージ: 10GB以上の空き容量

# 🏗️ Google Cloud プロジェクトの準備

## 1. プロジェクトの作成

1. [Google Cloud Console](https://console.cloud.google.com/) にアクセス

2. 右上の「プロジェクトを選択」をクリック

3. 「新しいプロジェクト」をクリック

4. プロジェクト名を入力（例：`rag-system-2024`）

5. 「作成」をクリック

## 2. プロジェクトIDの確認

プロジェクトIDを確認（後で使用するためメモしておく）

gcloud config get-value project

## 3. 請求の有効化

1. [Google Cloud Console](https://console.cloud.google.com/billing) にアクセス

2. プロジェクトを選択

3. 「請求を有効にする」をクリック

4. 既存の請求アカウントを選択または新規作成

# 🔌 必要なAPIの有効化

以下のAPIを有効化する必要があります：

## 1. 一括有効化（推奨）

プロジェクトIDを設定（実際のプロジェクトIDに置き換えてください）

export PROJECT\_ID="your-project-id"

gcloud config set project $PROJECT\_ID

必要なAPIを一括有効化

gcloud services enable \

cloudbuild.googleapis.com \

run.googleapis.com \

artifactregistry.googleapis.com \

firestore.googleapis.com \

aiplatform.googleapis.com \

secretmanager.googleapis.com \

identitytoolkit.googleapis.com \

storage.googleapis.com \

cloudresourcemanager.googleapis.com \

iam.googleapis.com \

monitoring.googleapis.com \

logging.googleapis.com

## 2. 個別有効化（必要な場合）

各APIを個別に有効化する場合：

1. [Google Cloud Console - API ライブラリ](https://console.cloud.google.com/apis/library) にアクセス

2. 以下のAPIを検索して有効化：

- \*\*Cloud Build API\*\*

- \*\*Cloud Run API\*\*

- \*\*Artifact Registry API\*\*

- \*\*Cloud Firestore API\*\*

- \*\*Vertex AI API\*\*

- \*\*Secret Manager API\*\*

- \*\*Identity Platform API\*\*

- \*\*Cloud Storage API\*\*

- \*\*Cloud Resource Manager API\*\*

- \*\*IAM API\*\*

- \*\*Cloud Monitoring API\*\*

- \*\*Cloud Logging API\*\*

# 👤 サービスアカウントの作成と権限設定

## 1. サービスアカウントの作成

サービスアカウントを作成

gcloud iam service-accounts create rag-system-sa \

--display-name="RAG System Service Account" \

--description="Service account for RAG System"

## 2. 必要な権限の付与

基本的な権限

gcloud projects add-iam-policy-binding $PROJECT\_ID \

--member="serviceAccount:rag-system-sa@$PROJECT\_ID.iam.gserviceaccount.com" \

--role="roles/datastore.user"

Vertex AI 権限

gcloud projects add-iam-policy-binding $PROJECT\_ID \

--member="serviceAccount:rag-system-sa@$PROJECT\_ID.iam.gserviceaccount.com" \

--role="roles/aiplatform.user"

Cloud Storage 権限

gcloud projects add-iam-policy-binding $PROJECT\_ID \

--member="serviceAccount:rag-system-sa@$PROJECT\_ID.iam.gserviceaccount.com" \

--role="roles/storage.objectViewer"

Secret Manager 権限

gcloud projects add-iam-policy-binding $PROJECT\_ID \

--member="serviceAccount:rag-system-sa@$PROJECT\_ID.iam.gserviceaccount.com" \

--role="roles/secretmanager.secretAccessor"

Cloud Run 権限

gcloud projects add-iam-policy-binding $PROJECT\_ID \

--member="serviceAccount:rag-system-sa@$PROJECT\_ID.iam.gserviceaccount.com" \

--role="roles/run.invoker"

Artifact Registry 権限

gcloud projects add-iam-policy-binding $PROJECT\_ID \

--member="serviceAccount:rag-system-sa@$PROJECT\_ID.iam.gserviceaccount.com" \

--role="roles/artifactregistry.reader"

ログ書き込み権限

gcloud projects add-iam-policy-binding $PROJECT\_ID \

--member="serviceAccount:rag-system-sa@$PROJECT\_ID.iam.gserviceaccount.com" \

--role="roles/logging.logWriter"

監視権限

gcloud projects add-iam-policy-binding $PROJECT\_ID \

--member="serviceAccount:rag-system-sa@$PROJECT\_ID.iam.gserviceaccount.com" \

--role="roles/monitoring.metricWriter"

## 3. サービスアカウントキーの作成（オプション）

サービスアカウントキーを作成（ローカル開発用）

gcloud iam service-accounts keys create rag-system-sa-key.json \

--iam-account=rag-system-sa@$PROJECT\_ID.iam.gserviceaccount.com

# 🔐 Secret Manager の設定

## 1. APIキーの保存

OpenAI APIキーを保存

echo -n "your-openai-api-key" | gcloud secrets create openai-api-key --data-file=-

Anthropic APIキーを保存

echo -n "your-anthropic-api-key" | gcloud secrets create anthropic-api-key --data-file=-

Google Search APIキーを保存

echo -n "your-google-search-api-key" | gcloud secrets create google-search-api-key --data-file=-

## 2. シークレットの確認

シークレットの一覧を確認

gcloud secrets list

シークレットの詳細を確認（値は表示されません）

gcloud secrets describe openai-api-key

# 🏪 Artifact Registry の準備

## 1. リポジトリの作成

大阪リージョンにリポジトリを作成

gcloud artifacts repositories create rag-system \

--repository-format=docker \

--location=asia-northeast2 \

--description="RAG System Docker Repository"

## 2. 認証設定

Docker認証を設定

gcloud auth configure-docker asia-northeast2-docker.pkg.dev

# 🔨 Cloud Build の設定

## 1. Cloud Build サービスアカウントの権限設定

Cloud Build サービスアカウントに権限を付与

gcloud projects add-iam-policy-binding $PROJECT\_ID \

--member="serviceAccount:$PROJECT\_NUMBER@cloudbuild.gserviceaccount.com" \

--role="roles/run.admin"

gcloud projects add-iam-policy-binding $PROJECT\_ID \

--member="serviceAccount:$PROJECT\_NUMBER@cloudbuild.gserviceaccount.com" \

--role="roles/iam.serviceAccountUser"

gcloud projects add-iam-policy-binding $PROJECT\_ID \

--member="serviceAccount:$PROJECT\_NUMBER@cloudbuild.gserviceaccount.com" \

--role="roles/artifactregistry.writer"

## 2. cloudbuild.yaml の確認

プロジェクトの `deployment/cloudbuild.yaml` ファイルが正しく設定されていることを確認：

steps:

# 1. Dockerイメージビルド（マルチステージビルド対応）

- name: 'gcr.io/cloud-builders/docker'

args: [

'build',

'-t',

'asia-northeast2-docker.pkg.dev/$PROJECT\_ID/rag-system/app:$COMMIT\_SHA',

'-t',

'asia-northeast2-docker.pkg.dev/$PROJECT\_ID/rag-system/app:latest',

'-f',

'deployment/Dockerfile',

'--cache-from',

'asia-northeast2-docker.pkg.dev/$PROJECT\_ID/rag-system/app:latest',

'.'

]

# 2. Artifact Registryへプッシュ

- name: 'gcr.io/cloud-builders/docker'

args: [

'push',

'--all-tags',

'asia-northeast2-docker.pkg.dev/$PROJECT\_ID/rag-system/app'

]

# 3. Cloud Runへデプロイ（大阪リージョン）

- name: 'gcr.io/google.com/cloudsdktool/cloud-sdk'

entrypoint: gcloud

args:

- 'run'

- 'deploy'

- 'rag-system'

- '--image'

- 'asia-northeast2-docker.pkg.dev/$PROJECT\_ID/rag-system/app:$COMMIT\_SHA'

- '--region'

- 'asia-northeast2' # 大阪リージョン

- '--platform'

- 'managed'

- '--service-account'

- 'rag-system-sa@$PROJECT\_ID.iam.gserviceaccount.com'

- '--set-secrets'

- 'OPENAI\_API\_KEY=openai-api-key:latest'

- '--set-secrets'

- 'ANTHROPIC\_API\_KEY=anthropic-api-key:latest'

- '--set-secrets'

- 'GOOGLE\_SEARCH\_API\_KEY=google-search-api-key:latest'

- '--set-env-vars'

- 'GCP\_PROJECT\_ID=$PROJECT\_ID'

- '--set-env-vars'

- 'GCP\_REGION=asia-northeast2'

- '--set-env-vars'

- 'STREAMLIT\_SERVER\_HEADLESS=true'

- '--set-env-vars'

- 'APP\_ENV=production'

- '--min-instances'

- '1'

- '--max-instances'

- '10'

- '--memory'

- '2Gi'

- '--cpu'

- '2'

- '--timeout'

- '300'

- '--concurrency'

- '80'

- '--cpu-boost'

- '--no-cpu-throttling'

timeout: 1200s

options:

logging: CLOUD\_LOGGING\_ONLY

machineType: 'E2\_HIGHCPU\_8'

ビルド結果の保存

images:

- 'asia-northeast2-docker.pkg.dev/$PROJECT\_ID/rag-system/app:$COMMIT\_SHA'

- 'asia-northeast2-docker.pkg.dev/$PROJECT\_ID/rag-system/app:latest'

# 🚀 Cloud Run のデプロイ

## 1. 手動デプロイ（初回）

プロジェクトディレクトリに移動

cd /path/to/your/rag-system

Cloud Build を実行

gcloud builds submit --config deployment/cloudbuild.yaml

## 2. デプロイ状況の確認

デプロイ状況を確認

gcloud builds list --limit=5

Cloud Run サービスの確認

gcloud run services list --region=asia-northeast2

## 3. サービスの詳細確認

サービスの詳細を確認

gcloud run services describe rag-system --region=asia-northeast2

# 🌐 ドメインとSSL証明書の設定

## 1. カスタムドメインの設定（オプション）

カスタムドメインをマッピング

gcloud run domain-mappings create \

--service rag-system \

--domain your-domain.com \

--region asia-northeast2

## 2. SSL証明書の自動設定

Cloud Run は自動的にSSL証明書を提供します。追加設定は不要です。

# 📊 監視とログの設定

## 1. Cloud Monitoring の設定

1. [Cloud Monitoring](https://console.cloud.google.com/monitoring) にアクセス

2. 「ワークスペースの作成」をクリック

3. プロジェクトを選択して「作成」をクリック

## 2. アラートポリシーの作成

アラートポリシーを作成

gcloud alpha monitoring policies create \

--policy-from-file=deployment/monitoring-policy.yaml

## 3. ログの確認

Cloud Run のログを確認

gcloud logs read "resource.type=cloud\_run\_revision AND resource.labels.service\_name=rag-system" --limit=50

# 🔧 トラブルシューティング

## よくある問題と解決方法

### 1. 権限エラー

権限を確認

gcloud projects get-iam-policy $PROJECT\_ID

サービスアカウントの権限を再設定

gcloud projects add-iam-policy-binding $PROJECT\_ID \

--member="serviceAccount:rag-system-sa@$PROJECT\_ID.iam.gserviceaccount.com" \

--role="roles/run.admin"

### 2. メモリ不足エラー

Cloud Run のメモリを増やす

gcloud run services update rag-system \

--region asia-northeast2 \

--memory 4Gi

### 3. タイムアウトエラー

タイムアウト時間を延長

gcloud run services update rag-system \

--region asia-northeast2 \

--timeout 600

### 4. 外部API接続エラー

1. Secret Manager のシークレットが正しく設定されているか確認

2. APIキーが有効か確認

3. ネットワーク設定を確認

## ログの確認方法

リアルタイムログを確認

gcloud logs tail "resource.type=cloud\_run\_revision AND resource.labels.service\_name=rag-system"

エラーログのみを確認

gcloud logs read "resource.type=cloud\_run\_revision AND resource.labels.service\_name=rag-system AND severity>=ERROR" --limit=20

# 📚 参考リンク

## 公式ドキュメント

- [Cloud Run ドキュメント](https://cloud.google.com/run/docs)

- [Cloud Build ドキュメント](https://cloud.google.com/build/docs)

- [Artifact Registry ドキュメント](https://cloud.google.com/artifact-registry/docs)

- [Secret Manager ドキュメント](https://cloud.google.com/secret-manager/docs)

- [Vertex AI ドキュメント](https://cloud.google.com/vertex-ai/docs)

## チュートリアル

- [Cloud Run クイックスタート](https://cloud.google.com/run/docs/quickstarts/build-and-deploy)

- [Cloud Build チュートリアル](https://cloud.google.com/build/docs/quickstart-build)

- [Artifact Registry チュートリアル](https://cloud.google.com/artifact-registry/docs/quickstart)

## 料金計算

- [Cloud Run 料金](https://cloud.google.com/run/pricing)

- [Cloud Build 料金](https://cloud.google.com/build/pricing)

- [Vertex AI 料金](https://cloud.google.com/vertex-ai/pricing)

# 🎯 次のステップ

1. \*\*アプリケーションの動作確認\*\*

- デプロイされたURLにアクセス

- 各機能のテスト実行

2. \*\*パフォーマンスの最適化\*\*

- メモリとCPUの使用量監視

- レスポンス時間の測定

3. \*\*セキュリティの強化\*\*

- ファイアウォールルールの設定

- アクセス制御の確認

4. \*\*バックアップの設定\*\*

- Firestore のバックアップ設定

- データの定期バックアップ

---

\*\*注意\*\*: このマニュアルは2024年8月時点の情報に基づいています。最新の情報については、各公式ドキュメントを参照してください。