Cloud Functions

1- Connectez et étendez les services cloud

Cloud Functions fournit une couche connective de logique qui vous permet d'écrire du code pour connecter et étendre les services cloud. Écoutez et répondez à un téléchargement de fichier vers Cloud Storage, à une modification de journal ou à un message entrant sur un 'topic' Cloud Pub/Sub. Cloud Functions augmente les services cloud existants et vous permet de répondre à un nombre croissant de cas d'utilisation avec une logique de programmation arbitraire.
Les fonctions Cloud Functions ont accès aux informations d'identification du compte de service Google et sont donc authentifiées de manière transparente avec la majorité des services Google Cloud tels que Datastore, Cloud Spanner, Cloud Translation API, Cloud Vision API, ainsi que de nombreux autres. De plus, Cloud Functions est pris en charge par de nombreuses bibliothèques clientes Node.js , ce qui simplifie encore davantage ces intégrations.

Les fonctions Cloud Functions peuvent être écrites en Node.js, Python et Go, et sont également exécutées dans des environnements d'exécution spécifiques au langage.

2- Serverless

Cloud Functions supprime le travail de gestion des serveurs, de configuration des logiciels, de mise à jour des frameworks et de correction des systèmes d'exploitation. Le logiciel et l'infrastructure sont entièrement gérés par Google afin que vous ajoutiez simplement du code.

De plus, l'approvisionnement des ressources se produit automatiquement en réponse aux événements. Cela signifie qu'une fonction peut évoluer de quelques appels par jour à plusieurs millions d'appels sans aucun travail de votre part.

3- Cas d'utilisation

Les charges de travail asynchrones, par exemple les automatisations ETL légères ou cloud, comme le déclenchement de builds d'applications, n'ont plus besoin de leur propre serveur et d'un développeur pour les câbler. Vous déployez simplement une fonction cloud liée à l'événement que vous souhaitez et vous avez terminé.

La configurabilité et l'instantanéité de Cloud Functions en fait également un candidat idéal pour les API et les webhooks légers.

Dans la mesure où il existe un provisionnement automatique des endpoints HTTP lorsque vous déployez une fonction HTTP, aucune configuration compliquée n'est requise, contrairement à certains autres services.

Use Case	**Description**
Traitement des données / ETL	Écoutez et répondez aux événements Cloud Storage , par exemple lorsqu'un fichier est créé, modifié ou supprimé. Traitez les images, effectuez le transcodage vidéo, validez et transformez les données et invoquez n'importe quel service sur Internet à partir de votre fonction cloud.
Webhooks	Via un simple déclencheur HTTP , répondez aux événements provenant de systèmes tiers tels que GitHub, Slack, Stripe ou

	de n'importe quel endroit pouvant envoyer des requêtes HTTP.
Lightweight APIs	Composez des applications à partir de bits de logique légers et faiblement couplés, rapides à créer et évolutifs instantanément. Vos fonctions peuvent être pilotées par les événements ou appelées directement via HTTP/S.
Mobile Backend	Utilisez la plate-forme mobile de Google pour les développeurs d'applications, Firebase , et écrivez votre backend mobile dans Cloud Functions. Écoutez et répondez aux événements de Firebase Analytics, Realtime Database, Authentication et Storage.
IoT	Imaginez des dizaines ou des centaines de milliers d'appareils diffusant des données dans Cloud Pub/Sub, lançant ainsi Cloud Functions pour traiter, transformer et stocker des données. Cloud Functions vous permet de le faire d'une manière totalement sans serveur.

4- Créer une fonction

Tout d'abord, vous allez créer une fonction simple nommée helloWorld. Cette fonction écrit un message dans les journaux Cloud Functions. Il est déclenché par des événements de fonction cloud et accepte une fonction de rappel (callback) utilisée pour signaler l'achèvement de la fonction.

Pour créer une fonction cloud:

1. Dans la ligne de commande Cloud Shell, créez un répertoire pour le code de fonction: `mkdir gcf_hello_world`
2. Move to the **`gcf_hello_world`** directory:

   `cd gcf_hello_world`

3. Create and open **`index.js`** to edit.

   `nano index.js`

4. Copy the following into the **`index.js`** file

```
/**
* Background Cloud Function to be triggered by Pub/Sub.
* This function is exported by index.js, and executed when
* the trigger topic receives a message.
*
* @param {object} data The event payload.
* @param {object} context The event metadata.
*/
exports.helloWorld = (data, context) => {
const pubSubMessage = data;
const name = pubSubMessage.data
    ? Buffer.from(pubSubMessage.data, 'base64').toString() : "Hello
World";

console.log(`My Cloud Function: ${name}`);
};
```

   Exit nano (Ctrl+x) and save (Y) the file.

5- Créer un bucket de stockage cloud

Utilisez la commande suivante pour créer un nouveau compartiment de stockage cloud pour votre fonction: `gsutil mb -p [PROJECT_ID] gs://[BUCKET_NAME]`

- **BUCKET_NAME** est le nom que vous donnez au bucket. Ce doit être un nom unique au monde. Pour plus d'informations, consultez Consignes de dénomination de bucket .

```
gsutil mb -p widigital-ci gs://widigital-ci-cloud-functions
```

```
ahmedhosni_contact@cloudshell:~/gcf_hello_world (widigital-ci)$ gsutil mb -p widigital-ci gs://widigital-ci-cloud-functions
Creating gs://widigital-ci-cloud-functions/...
```

6- Déployez votre fonction

```
gcloud functions deploy helloWorld \
  --stage-bucket widigital-ci-cloud-functions \
  --trigger-topic hello_world \
  --runtime nodejs10
```

```
ahmedhosni_contact@cloudshell:~/gcf_hello_world (widigital-ci)$ gcloud functions deploy helloWorld   --stage-bucket widigital-ci-cloud-functions   --trigger-topic hello_worl
d   --runtime nodejs10
Deploying function (may take a while - up to 2 minutes)...done.
availableMemoryMb: 256
entryPoint: helloWorld
eventTrigger:
  eventType: google.pubsub.topic.publish
  failurePolicy: {}
  resource: projects/widigital-ci/topics/hello_world
  service: pubsub.googleapis.com
ingressSettings: ALLOW_ALL
labels:
  deployment-tool: cli-gcloud
name: projects/widigital-ci/locations/us-central1/functions/helloWorld
runtime: nodejs10
serviceAccountEmail: widigital-ci@appspot.gserviceaccount.com
sourceArchiveUrl: gs://widigital-ci-cloud-functions/us-central1-projects/widigital-ci/locations/us-central1/functions/helloWorld-guuleycpehkw.zip
status: ACTIVE
timeout: 60s
updateTime: '2020-08-20T20:40:30.514Z'
versionId: '2'
```

Vérifiez l'état de la fonction:

```
gcloud functions describe helloWorld
```

Un état ***ACTIVE*** indique que la fonction a été déployée.

```
ahmedhosni_contact@cloudshell:~/gcf_hello_world (widigital-ci)$ gcloud functions describe helloWorld
availableMemoryMb: 256
entryPoint: helloWorld
eventTrigger:
  eventType: google.pubsub.topic.publish
  failurePolicy: {}
  resource: projects/widigital-ci/topics/hello_world
  service: pubsub.googleapis.com
ingressSettings: ALLOW_ALL
labels:
  deployment-tool: cli-gcloud
name: projects/widigital-ci/locations/us-central1/functions/helloWorld
runtime: nodejs10
serviceAccountEmail: widigital-ci@appspot.gserviceaccount.com
sourceArchiveUrl: gs://widigital-ci-cloud-functions/us-central1-projects/widigital-ci/locations/us-central1/functions/helloWorld-guuleycpehkw.zip
status: ACTIVE
timeout: 60s
updateTime: '2020-08-20T20:40:30.514Z'
versionId: '2'
```

Chaque message publié dans la rubrique déclenche l'exécution de la fonction, le contenu du message est transmis en tant que données d'entrée.

7- Tester la fonction

Une fois que vous avez déployé la fonction et que vous savez qu'elle est active, vérifiez que la fonction écrit un message dans le journal cloud après avoir

détecté un événement. Entrez cette commande pour créer un test de message de la fonction.

```
DATA=$(printf 'Hello World!'|base64) && gcloud functions call helloWorld --data '{"data":"'$DATA'"}'
```

Celà renvoie l'ID d'exécution de la fonction, ce qui signifie qu'un message a été écrit dans le journal. Exemple de sortie:

```
ahmedhosni_contact@cloudshell:~/gcf_hello_world (widigital-ci)$ DATA=$(printf 'Hello World!'|base64) && gcloud functions call helloWorld --data '{"data":"'$DATA'"}'
executionId: nuzjupzvcqry
```

8- Examiner les connexions

Consultez les journaux pour voir vos messages dans l'historique des journaux.

```
gcloud functions logs read helloWorld --limit 4
```

Si la fonction s'est exécutée avec succès, les messages du journal s'affichent comme suit:

```
ahmedhosni_contact@cloudshell:~ (widigital-ci)$ gcloud functions logs read helloWorld --limit 6
LEVEL  NAME        EXECUTION_ID  TIME_UTC                 LOG
D      helloWorld  nuzjckwkc1tx  2020-08-20 20:55:51.969  Function execution started
       helloWorld  nuzjckwkc1tx  2020-08-20 20:55:51.979  My Cloud Function: Hello World!
D      helloWorld  nuzjckwkc1tx  2020-08-20 20:55:51.980  Function execution took 12 ms, finished with status: 'ok'
D      helloWorld  nuzjhbnobc4d  2020-08-20 20:59:25.640  Function execution started
       helloWorld  nuzjhbnobc4d  2020-08-20 20:59:25.651  My Cloud Function: ►e►j+►
D      helloWorld  nuzjhbnobc4d  2020-08-20 20:59:25.652  Function execution took 12 ms, finished with status: 'ok'
```

Vous pouvez aussi visualiser les journaux d’exécution depuis la console: