Rapport SACC - Couche de Donnée

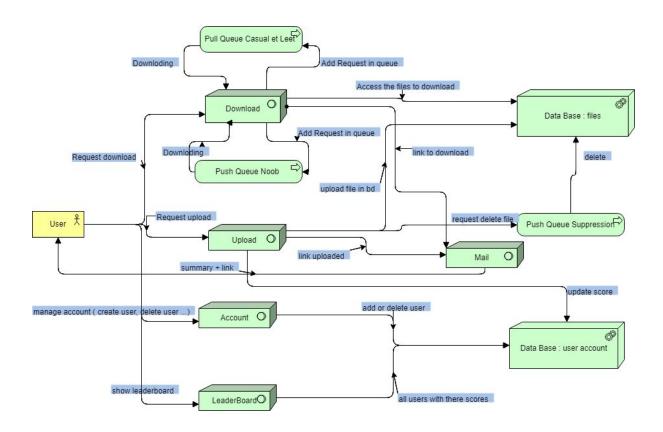
Mohamed Chennouf, Meersman Rudy, Duminy Gaétan et Ivan Picard Marchetto



Google Cloud

1. Introduction

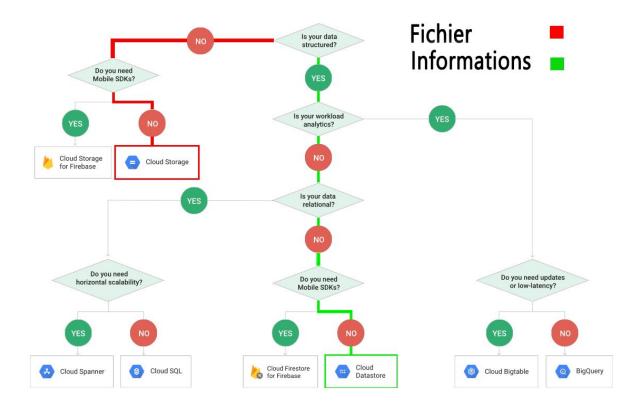
Pour pouvoir réaliser notre application cloud de stockage et partage de fichiers, nous avions défini une architecture pour répondre à ce besoin. Dans cette architecture, nous n'avions pas précisé la façon dont le stockage allait être géré.



Le but est de déterminer qu'elle type de donné nous allons choisir pour la base de données des fichiers et la base de données des utilisateurs.

Rappel sur les objectifs et contraintes

Un utilisateur peut stocker 1 à 4 fichiers par minute en fonction de son niveau (Noob, Casual, Leet). Il paraît donc normal d'avoir des comptes pour chaque utilisateur et donc une database pour les stocker. Cette database stockera l'email de l'utilisateur, qui sera utilisé comme clé unique, un rang et un nombre de points. Nous devons aussi en avoir une pour le stockage de fichier mais il faut faire attention à la taille de ces derniers. De plus, nous devons faire attention aux pannes et surtout à la scalabilité en fonction de la charge des requêtes sur la base pour que cela ne devienne pas un problème par la suite.



2. Stockage des données

Sachant que nous avons des données structurées pour les informations de fichier (Id, Poids, EmailUtilisateur, URL, etc.), nous pouvons partir sur une approche noSQL.

En effet, cette approche nous permet de scaler plus facilement la base de données en cas d'heure de pointe et pourra toujours donner une bonne performance sachant que l'on pourrait avoir énormément de requêtes sur celle-ci.

De plus, le noSQL ne demande pas un type précis de data ce qui nous donne une très bonne flexibilité dans leurs architectures. Le Datastore peut automatiquement gérer la réplication de données et se scale automatiquement pour les grosses données. Les transactions sont basées sur les propriétés ACID et donc garantissent les transactions de nos données.

Type de donnée:

Utilisateurs:

- Kind:
 - Utilisateur
- Properties:
 - Niveau (String : { Noob, Casual, Leet })
 - ➤ Point (int)

```
★ Key:
➤ Email (string)
Fichiers:
★ Kind:
➤ File
❖ Properties:
➤ EmailUtilisateur (String)
➤ Url (String)
➤ Poids (int)
➤ Statut (String: {Alive, ToDelete})
➤ DateCreation (Time)
➤ Type (String)
❖ Key:
```

3. Stockage des fichiers

> Id (int)

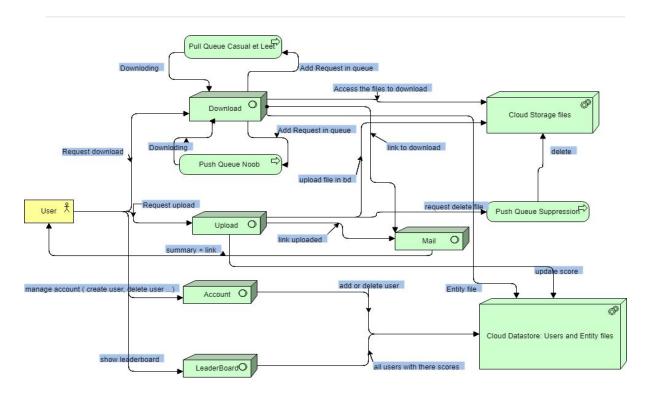
Nos fichiers sont non structuré (image, vidéo, texte...) et d'après la documentation, le Cloud Storage permet de stocker ce genre de fichier.

La disponibilité du service ainsi que la durabilité du système offre de bon atout pour le stockage de fichier (selon Google, nous avons une durabilité de 99,99999% sur 1 an et une disponibilité régional de 99.9%).

Les opérations de classe A nous permettent de faire du upload avec 5000 requêtes gratuites par mois. Puis on prend des packs de 10000 requêtes classe A payant pour compléter.

Les opérations de classe B nous permettent de faire du download avec 50 000 requêtes gratuites par mois , ce qui nous paraît être un nombre suffisant de requêtes.

4. Nouvelle Architecture



Nous nous sommes rendu compte que notre architecture était incorrecte et qu'il fallait relier notre module de download aux deux bases de données : Cloud Storage pour les fichiers et Cloud Datastore pour les entités relatives aux fichiers (URL, poids, id, etc.).

5. Coût de la plateforme de stockage

A	8	c	D	E	F	G	Н:	- 3	J.	К	L	М	N
	Personnes	ops leets	Me mount fishing	Temps garder min									
	1000	4	30	30		3600	Go						
	1000	-	500	30		60000	Go						
			300			60000	40			_		Téléchargement	
										Nb users	Type		Upload/jou
Cloud Storage Belgique	Cout Go entre	Cout Go entre 1 & 10 To	Cout Go > 10To	Cout 10000 Ops		Quantité	Offerte			0.0000000000000000000000000000000000000	Noob	1	
						5	Go			200	Casual	2	
						5000	Ops A				Leet	4	
Stockage de données	\$0,02	\$0,02	\$0.02			50000	Ops B						
Utilisation du réseau entre buckets Gratuit entre buckets eu	\$0,00	\$0,00	\$0,00				10000000				Jour	1500	1500
Réseau Upload	\$0,00	\$0,00	\$0.00						30	Jours :	Mensuel	45000	
Réseau Download	50,12	\$0,11	\$0.08							-	-	100000	-
Opérations classe A				\$0,05					Requete Download	Go Download			
Opérations classe B				\$0,00				Jour	1500	45			
								Mois	45000	1350			
Coût Stockage	\$71,90							11.01-0		10.10			
Coût ops classe A	\$0,20												
Coût ops classe B	\$ -												
Réseau Download	\$ 158,50												
Total	\$72,10												
Coût utilisateur / mois	\$0,08												
Remarques													
Coût de suppression anticipée et													
coût ops supérieur de Nearline & Coldline			<										

Le coût par mois pour le **cloud storage** peut s'évaluer de la manière suivante:

- Le coût de la capacité maximale de stockage à louer
- Le coût des opérations de type A mensuel
- Le coût des opérations de type B mensuel

La capacité de stockage maximale se calcule suivant ce scénario: Nos 1000 utilisateurs de l'application sont de rang "**leet**" et décide, de faire tous en même temps, 4 upload d'un fichier de 30 Mo en moyenne par minute durant 30 minutes soit le temps de vie d'un fichier sur notre plateforme. Seulement 5 Go sont offert par mois puis le prix est de \$0.02 par Go.

Les requêtes de type A sont limité à 5000 par mois gratuitement puis le coût est de \$0.05 pour le lot de 10000. On estime le nombre de requête d'upload à :

$$(700*1 + 200*2 + 100*4) * 30 = 45000$$
 soit $(45000 - 5000) / 10000 * $0.05 = 0.20 par mois

Les requêtes de type B sont limité à 50000 par mois gratuitement, dans notre scénario nous avons 45000 requêtes de download par mois. Les besoins sont donc couvert.

Au total, on en a pour \$72.10/mois soit \$0.08/mois par utilisateur pour le cloud storage.

On utilise un **Datastore** pour stocker les utilisateurs au nombre de 1000 dans notre scénario et les informations des fichiers uploadés sur la plateforme. Les avantages sont les quantités

d'opérations journalières offertes ainsi que le stockage nécessaire qui couvrent l'intégralité de nos besoins pour ce service.

Au final, si on cumule avec le coût des instances et des queues, cela nous donne \$0.08 + \$0.47 = \$0.55 par utilisateurs.

Références:

- 1. https://cloud.google.com/storage/pricing#network-egress
- 2. https://cloud.google.com/datastore/docs/concepts/overview
- 3. https://cloud.google.com/storage-options/
- 4. https://cloud.google.com/datastore/pricing