Rapport Final Projet SACC

Author : Liechtensteger Michael, Génovèse Matthieu, Chennouf Mohamed, Fezai Ahmed

Introduction

Le but de ce projet est de développer une application de conversion de vidéos déployée sur le cloud de Google. Durant les premières semaines, nous avons défini une architecture pour l'application, et une architecture justifiant les supports de stockages que nous allions utiliser. Dans ce rapport, nous expliquerons la façon dont marche notre application et les différentes méthodes mises en place lors du développement.

Réalisations

Utilisation de l'application

Pour utiliser l'application, l'utilisateur doit d'abord créer un compte. Pour cela, nous avons défini une servlet pour lui permettre de rentrer ses informations dans une requête POST. Les informations de l'utilisateur sont alors rentrées dans le Datastore, ce qui lui permet de convertir une vidéo. Le compte est obligatoire pour convertir une vidéo, cependant tous les utilisateurs peuvent déposer une vidéo sans avoir de compte. Via une requête, l'utilisateur peut demander une conversion de vidéo en entrant son nom, et le nom de la vidéo à convertir, cette vidéo est stocké avec un poids de 1Mb par seconde de vidéo. Une fois la demande effectuée, celle-ci part dans une queue (différente en fonction du grade de l'utilisateur) pour être traitée, et un mail est envoyé à l'utilisateur pour lui indiquer que sa vidéo est en cours de traitement. La vidéo arrive dans un convertisseur, qui simule un temps d'attente en fonction du temps de la vidéo, et crée un deuxième fichier d'un poids de 1MB par seconde de vidéo. Une fois la vidéo convertie, elle est placée dans le Cloud Storage pendant une durée limitée (5 ou 10 minutes selon le grade de l'utilisateur qui a demandé la conversion) et un mail est envoyé à l'utilisateur avec un lien lui permettant de télécharger la vidéo.

Pour gérer la suppression de vidéos dans le Cloud Storage, nous avons défini une cron vérifiant toutes les minutes les temps des vidéos stockées, et pouvant supprimer les vidéos si le temps dépasse 5 minutes pour les bronzes et silvers, ou 10 minutes pour les golds.
Si la taille de la vidéo est trop grande, nous avons décidé de scinder la vidéo en plusieurs parties, pour éviter de surcharger la machine virtuelle Java lors de la création du fichier.

Gestion des queues

Lors de la réalisation de ce projet, nous avons utilisé 4 queues pour gérer les conversions des vidéos fournies par les utilisateurs. Nous avons définies une push queue pour les bronzes et deux pull queues pour les silvers et les golds. Nous avons choisi de mettre deux queues pour les silvers et les golds pour pouvoir faire en sorte que les utilisateurs étant soit silvers soit golds ne se retrouvent pas bloqués si leur queue est déjà pleine. Si les silvers sont deux fois plus nombreux que les golds, il est possible que la demande d'un silver soit placée dans la queue des golds. Cependant, un gold aura toujours la priorité sur les autres, et peut également utiliser la queue des silvers si celle des golds est pleine. La dernière queue s'occupe de réceptionner les demande de conversion des silver / golds mais elles ne les traite pas, elle s'occupe simplement de les rediriger vers les queues adéquates.

Scénario

Un utilisateur Toto veut convertir la vidéo de ses vacances à la plage. Pour cela, il s'enregistre sur la base de données en appelant la servlet de création de comptes (en mettant en paramètre son nom, son email et le niveau de son compte). Une fois son compte créé, il appelle la servlet d'envoi de vidéos pour envoyer sa vidéo (en précisant le titre et la durée de la vidéo) sur le cloud Storage, puis appelle la servlet de conversion de vidéos pour lancer la conversion de sa vidéo (en précisant son nom et le titre de sa vidéo). Un mail est envoyé à Toto lui précisant que la conversion a démarrée. Une fois la conversion finie, un lien est envoyé à Toto pour lui permettre de télécharger sa vidéo convertie.

Implémentation détaillée

Gestion des requêtes HTTP

Nos requêtes HTTP sont traitées dans différentes Servlet :

-Servlet SubmitVideo (adresse /submit)
C'est cette servlet qui s'occupe de récupérer le nom et la durée d'une vidéo (requête POST application/json) afin de l'uploader sur le cloud.

-Servlet MainPage (adresse /)
Cette Servlet affiche à l'utilisateur une interface définie dans le fichier "mainpage.jsp" qui lui permet d'effectuer la création de compte, la soumission de vidéo et la conversion de video directement.
-Servlet ConverVideo (adresse /convert)

Cette Servlet lance la conversion d'une vidéo pour un utilisateur donnée et une vidéo donnée (requête POST application/json), c'est cette servlet qui va initier le remplissage des queues.

-Servlet CreateAccount (adresse /createaccount)
Cette Servlet créer un utilisateur, à l'aide de son nom, email, grade (requête POST application/json), il ne peut pas y avoir deux utilisateurs avec le même pseudo.

-Servlet VideoStatus (adresse /status?username=username)
C'est la Servlet qui permet à l'utilisateur d'accéder à ses conversions en cours et ses conversions terminées (requête GET). Les vidéos qui ont expirées ne sont plus affichées.

-Servlet CronStorage
C'est la servlet qui est utilisé par notre job cron, qui a pour tâche de vérifier les temps des vidéos, il les supprime si elles dépassent le temps de 5 min (silver / bronze) ou 10 min (gold).

Gestion du datastore

Nous utilisons le datastore pour stocker divers entités. Chaque utilisateur est représenté par l'entité User :

- -username (String unique)
- -email(String)
- -accountlevel(String old / silver / bronze)
- -currentVid(int)

Le paramètre "currentVid" compte le nombre de VideUser qui ont le statut "waiting" ou "processing".

Lorsqu'une vidéo est uploadée, une entité vidéo est créé :

- -videoname (String)
- -videolength (int)
- -nbPart (int)

Lorsqu'une demande de conversion est effectuée, une entité VideoUser est créée, c'est un extend de Video elle possède donc les même champs de base plus d'autres :

- -downloadLink (String)
- -status (String waiting / processing / done)
- -owner (String)
- -submitTime (DateTime)

Si la vidéo est découper en plusieurs parties, plusieurs lien de téléchargement sont générés vers les différents fichiers.
Lors d'un ajout d'une requête dans une des 2 pull queues (bronze / silver), l'entité unique QueueStatus est mises à jour, elle contient :

- -id (int)

-nbGold (int)
-nbSilver(int)

Gestion du storage

Dans notre storage nous stockons 2 types de données, les vidéos uploadées et les vidéos converties. les vidéos uploadées sont découpées si elles dépassent la taille de 70Mo, elle seront nommées de la sorte : “videonamepartX”, où X est le numéro de la partie.
Les vidéos converties sont associés à un utilisateur spécifique, elles sont nommées : “videonameusernamepartX” . C’est uniquement la vidéo converti qui est supprimée après un certain temps.

Gestion des queues

Nous avons 4 queues pour gérer les différentes requêtes :

-une push queue Bronze
les conversions demandées par les bronze sont mises à la chaîne à l’intérieur et traitée dans l’ordre où elles ont étées émises.

-une push queue dispatch
cette queue réceptionne les demande de conversion des silver / golds mais elles ne les traite pas, elle s’occupe simplement de les rediriger vers les queues adéquates.

-une pull queue silver
la queue réservée aux silver, elle permet de traiter les demandes de conversion 3 par 3 au maximum pour un utilisateur silver. Si cette queue est surchargée de l’ordre du double de la queue des golds, les prochains silver seront mis dans la queues gold.

-une pull queue gold
la queue réservé aux golds, elle traite leurs demandes 5 par 5 maximum, si la queue est surchargé par rapport aux silvers, les golds suivants seront traités dans la queue des silver.