

Rapport complémentaire TP1 TLC

Le 23 Février 2017

Rapport de :

PITON Stephen

Groupe de TP :

EL GHZIZAL Yousra
PARISSE Simon
PITON Stephen

URL de notre Google App Engine :

<http://tlctp1-156207.appspot.com/>

URL de notre dépôt Git :

<https://github.com/yousrael/TP1TLC>

Le travail ayant été réalisé à trois, il est difficile de savoir qui a travaillé sur quelle partie, comme nous avons tous travaillé sur tous les différents aspects du TP, je vais parler des parties sur lesquelles j'ai le plus travaillé. Principalement la fonctionnalité de suppression des annonces et les tests par envois de requêtes sur le serveur Google pour tester la réaction de la plateforme appengine.

Supprimer les annonces recherchées :

Pour la suppression des annonces, un bouton suppression a été ajouté, il nous permet de supprimer les différentes annonces affichées sur la page. Il redirige vers une page deleteServlet.jsp qui est responsable de la suppression.

Si aucune recherche n'est effectuée il permet de supprimer l'intégralité des résultats affichés (ou limiter au nombre maximum d'objets retournés par l'objectify).

Le critère de sélection d'une annonce est directement copié de la fonction de recherche et les mêmes vérifications sur le prix, mot dans le titre et date sont effectuées.

The screenshot shows a dark-themed web application interface. On the left, the word "HOME" is displayed in white. On the right, there is a search section with a "Search" input field and a "Sign in" button. Below the search field, there are two price range sliders: "MINIMUM PRICE:" with a value of "23" and "MAXIMUM PRICE:". At the bottom of the search section, there are two date input fields: "01-01-2000" and "01-01-2022", followed by a "Search" button. Below the search section, there is a teal-colored box containing the following information: "Price : 1.0E7 \$", "Date : Wed Jun 21 00:00:00 CEST 2017", and "Author : An anonymous person".

Remove All Visible Entry

Problème rencontré pour la suppression :

Il faut bien vérifier si des paramètres ont été utilisés pour la suppression, si c'est le cas on les utilise, si ce n'est pas le cas alors on les crée avec des valeurs par défaut permettant de sélectionner toutes les annonces.

Après la suppression on retourne à la page principale sans critère de recherche pour éviter de chercher des annonces qui ont été supprimées.

Test des requêtes :

Une classe de test a été codée pour exécuter plusieurs requêtes à la suite, tels que l'ajout de plusieurs annonces afin de mesurer la scalabilité de Google App Engine.

Classe : **Test.java**

Le fichier test comprend des fonctions d'insertion, de recherche et de suppression avec un paramètre sur le nombre d'actions à effectuer en un appel.

Pour les différents Servlets, nous avons dû définir les méthodes doGet avec le même code que doPost afin de pouvoir facilement les appeler avec curl par exemple.

Les différents tests ont été effectués à la suite lors d'une exécution du fichier Test.java, par insertion, recherche et suppression successives d'une centaine d'éléments à chaque fois. Le fichier de test a été exécuté une fois tout seul, puis une deuxième fois avec 5 instances en parallèle.

Si on regarde les temps moyens pour les différentes actions, on constate qu'il sont relativement similaires, ce qui indique que le service Google scale notre application de façon optimale.

Cependant, lors du test intensif la page principale s'est retrouvée inaccessible assez vite du fait des trop nombreuses requêtes.

insert min: 261ms	max: 4713ms	moyenne: 434ms	total: 43403ms
search min: 298ms	max: 9022ms	moyenne: 577ms	total: 57728ms
delete min: 245ms	max: 1644ms	moyenne: 428ms	total: 4283ms
insert min: 253ms	max: 908ms	moyenne: 361ms	total: 36188ms
delete min: 229ms	max: 1517ms	moyenne: 401ms	total: 4014ms
insert min: 267ms	max: 633ms	moyenne: 354ms	total: 35497ms
delete min: 260ms	max: 1531ms	moyenne: 424ms	total: 4243ms
insert min: 270ms	max: 724ms	moyenne: 345ms	total: 34508ms
delete min: 254ms	max: 1076ms	moyenne: 421ms	total: 4212ms
search min: 358ms	max: 1480ms	moyenne: 513ms	total: 51305ms
insert min: 253ms	max: 555ms	moyenne: 339ms	total: 33969ms
insert min: 256ms	max: 668ms	moyenne: 344ms	total: 34451ms
insert min: 252ms	max: 1202ms	moyenne: 362ms	total: 36274ms
search min: 370ms	max: 1948ms	moyenne: 676ms	total: 67631ms
search min: 219ms	max: 4540ms	moyenne: 618ms	total: 61889ms
insert min: 270ms	max: 3993ms	moyenne: 385ms	total: 38570ms
search min: 227ms	max: 4611ms	moyenne: 576ms	total: 57643ms
insert min: 256ms	max: 3725ms	moyenne: 388ms	total: 38824ms
search min: 214ms	max: 4558ms	moyenne: 574ms	total: 57451ms

delete min: 203ms max: 2065ms moyenne: 487ms total: 4870ms

Dans les images suivantes on peut voir que lors de l'exécution des 5 tests en parallèle vers notre application, le nombre de requêtes a augmenté et le nombre d'instances de notre serveur a aussi augmenté automatiquement.

