Aplicatie exemplu - Guestbook :

<https://cloud.google.com/appengine/docs/standard/java/gettingstarted/creating-guestbook>

Pentru a clona aplicatia de pe github, se va scrie in linia de comanda :

git clone <https://github.com/GoogleCloudPlatform/appengine-java-guestbook-multiphase.git>

**Helpful Links**

Datastore quickstart: <https://cloud.google.com/datastore/docs/quickstart>

Datastore Queries: <https://cloud.google.com/datastore/docs/concepts/queries>

Indexes: <https://cloud.google.com/datastore/docs/concepts/indexes>

Transactions: <https://cloud.google.com/datastore/docs/concepts/transactions>

Data Consistency: <https://cloud.google.com/datastore/docs/concepts/structuring_for_strong_consistency>

Utilizarea Datastore prin Objectify:

<https://cloud.google.com/appengine/docs/standard/java/gettingstarted/using-datastore-objectify>

**Tema:**

1. Tema de gandire: Ce inseamna “atomicitate” in contextul bazelor de date? Datastore ofera tranzactii atomice? Cum puteti argumenta acest lucru?

* Atomicicatatea tranzactiilor in bazele de date se pot executa un set de operatii unde toate au succes sau toate esueaza. Din punct de vedere tranzactional Datastore prin demararea unei tranzactii programabile, grupul de entitati furnizeaza ACID(atomicitate, consistenta, isolatie, si durata) caracteristicii in tranzactii.
* Relația grupului de entități este formată în mod static pe baza numirii de chei. Nu poate fi schimbată după crearea entității. Singura opțiune disponibilă pentru schimbarea relației este de a șterge entitățile dintr-un grup de entități și de a le recrea din nou. Această provocare ne împiedică să folosim grupuri de entități pentru a defini domeniile ad-hoc pentru consecvență sau tranzacționalitate dinamic. În schimb, domeniul de coerență și de operativitate sunt strâns legate de grupul de entități statice definit la momentul proiectării . De exemplu, luați în considerare un scenariu în care doriți să implementați un transfer bancar între două conturi bancare. Acest scenariu de afaceri necesită o consistență și o tranzacționalitate puternică. Cu toate acestea, cele două conturi nu pot fi grupate într-un singur grup de entități în ultimul minut sau nu se bazează pe un părinte global. Acest grup de entități ar crea o strangulare pentru întregul sistem care ar împiedica executarea altor cereri de transfer bancar. Astfel, grupurile de entități nu pot fi utilizate în acest mod.
* Cu toate acestea, există un mod alternativ pentru ca un transfer bancar să fie implementat într-un mod foarte scalabil și disponibil. O metodă pentru a satisface această cerință este să utilizați tranzacțiile Cross-group (XG) pentru tranzacționalitate și căutarea prin Cloud Datastore prin metoda cheie sau o interogare a strămoșilor pentru consistență. Tranzacțiile grupurilor transfrontaliere reprezintă o caracteristică Cloud Datastore care vă permite să aveți caracteristici ACID pentru până la douăzeci și cinci de entități sau entități dintr-o singură tranzacție. Prin utilizarea tranzacțiilor XG, puteți crea un scop tranzacțional dinamic cu cele două conturi bancare în momentul procesării cererii.
* Rețineți că tranzacțiile XG asigură numai tranzacționalitatea. Pentru a vă asigura o consistență puternică atunci când citiți cele două conturi bancare, ar trebui să utilizați căutarea după metoda cheie sau o interogare a strămoșilor. Veți primi o eroare dacă încercați să executați o interogare care nu este o interogare a strămoșilor în cadrul unei tranzacții.

1. In aplicatia exemplu (Guestbook) implementati urmatorul comportament: Atunci cand utilizatorul (logat) scrie o salutare, datele utilizatorului sunt stocate intr-o entitate noua “UserDto”: email, nickname, authdomain, data la care a scris ultimul mesaj (salutare), ID catre mesajul (salutarea) respectiv/a.

Tema de gandire: De ce nu putem utiliza clasa User din Users API (com.google.appengine.api.users) ca si entitate pentru a salva datele despre utilizator si este necesar o clasa DTO? Google DTO!

1. Extra credit pentru cei care doresc : Obligati utilizatorul sa se logheze in aplicatie (inainte sa i se incarce pagina). Si stergeti optiunea din partea de sus a paginii de logare / delogare.

Hint 0 : Cel mai usor ar fi utilizand metoda “Authorization code flow”, dar exista mai multe metode-vezi Hint3

Hint 1 : <https://cloud.google.com/appengine/docs/standard/java/config/webxml#Security_and_Authentication>

Hint 2 : <https://stackoverflow.com/questions/28127680/oauth-401-invalid-credentials-after-some-period-of-time>

Hint 3 : <https://developers.google.com/api-client-library/java/google-oauth-java-client/oauth2#authorization_code_flow>

"Feedback constructiv pentru (To stop, To start, To continue):

• mentor :Andrei Petraru - To stop - (nu a fost un lucru care sa ma derajeze pentru care sa se opreasca ) , to start - feedback la code review ar prinde bine un exemplul din cand in cand de cum ar vrea sa arate codul care il cere (exemple facute de el, asa trebuie sa arate nu stiu ce), to continue - sa fie la fel de amuzant si cu aceiasi stare de spirit de a bine dispune lumea

• program internship - To stop - , To start - , To continue - experienta buna care face legatura intre lucrurile facute la facultate cu munca din piata muncii sa spun asa, sa te confrunci cu realitatea in care trebuie sa lucrezi. E un lucru bun ca incurajeaza si studentii care inca mai au studii de facut sa participe la orele care le au (program flexibil).

• spatiul de lucru (birou, bucatarie) si echipamentul primit" - To stop - diferentierea echipamentului primit (unii au primit laptop altii unitati) neavand toti aceiasi mobilitate cand sa prezentam ceva, To start - o integrare a internilor cu cei care lucreaza, sa nu fie neaparat un spatiu separat cum a fost cazul nostru sa fim doar interni sus, To continue - sa ofere in continuare suportul pentru recreere - spatiul din bucatarie, fructe, cafea, alte jocuri pentru relaxare etc.