**DEADLINE TUGAS TANGGAL: 17 DESEMBER 2023 jam 23:59**

GLS CASE STUDY: Hackathon Team Management

Di tugas kali ini, anda akan dituntut untuk membaca dan mengimplementasi materi di luar perkuliahan. Anda diminta untuk menyelesaikan permasalahan dengan menggunakan beberapa design pattern. Untuk referensi silahkan melihat halaman berikut: [Refactoring and Design Patterns](https://refactoring.guru/). Tugas ini juga akan meminta anda untuk menggunakan File reader yang tidak diajarkan di kelas sehari-sehari.

Case yang perlu diselesaikan adalah:

Anda akan diminta untuk membuat sebuah aplikasi console dengan database relational semi **menggunakan file csv** 😊. Database dapat diakses pada file **database.zip**. Case yang diangkat adalah data lomba Hackathon SOCS 2023. Pada file database.zip akan berisikan 3 file, yaitu user.csv yang menyimpan data user dan team.csv yang menyimpan data team. Tugas utama adalah membuat menu dengan menu insert, show dan delete. Menu yang dibutuhkan adalah sebagai berikut:

1. Menu Utama
2. Insert Data
3. Show
4. Exit

Berikut penjelasan singkat untuk menu tersebut:

1. **Menu utama**. Jika dipilih, akan menampilkan menu utama awal program.
2. **Insert Data**. Jika dipilih, akan mengisi data kedalam file .csv. Agar sesuai, tanyakan data apa yang ingin diisi, apakah Team atau User? Jika Team maka masukkan nama teamnya. Jika User, maka masukkan kebutuhan User, serta nama Team. Jika nama team ada di file, maka masukkan user kedalam team tersebut. Jika tidak ada, maka buat team baru, dan masukkan user ke team tersebut. **Pastikan juga bahwa 1 team memiliki maksimal 3 user.** Contoh:

Which table to insert? 1. User, 2. Team.  
1  
add name: fikri  
add nim: 2440127716  
add team: team1

User created!

Contoh jika user dimasukkan ke dalam team yang sudah penuh (3 user):

Which table to insert? 1. User, 2. Team.  
1  
add name: fikri  
add nim: 2440127716  
add team: venti

Error: Team full.

1. **Show.** Jika dipilih, maka tanyakan table mana yang ingin ditampilkan oleh user. Kemudian, tanyakan apakah user ingin menambahkan kondisi? Jika iya, maka minta user untuk memberikan kondisi, pisahkan dengan titik koma dan enter. Contoh:

Which table to show? 1. User, 2. Team.  
1  
Want to filter by condition? 1. Yes, 2. No.   
1  
add condition, separate by semicolon.   
name;=;kevin

Jika tidak ada kondisi, maka alur untuk meminta input dari user:

Which table to show? 1. User, 2. Team.  
1  
Want to filter by condition? 1. Yes, 2. No.   
2

1. **Exit.** Keluar dari program.

**Berikut arahan untuk program**:

1. Buatlah kelas **Connection** yang akan dapat:
   1. membuka file .csv
   2. membaca file .csv
   3. menulis ke dalam file .csv.

**Jika anda menggunakan design pattern untuk koneksi ini seperti object pool atau singleton, maka akan ada nilai tambahan.**

1. Gunakan **Façade Pattern** untuk melakukan proses **”query”** dari dan ke file-file csv tersebut.
2. Buatlah class **Model** untuk setiap file .csv yang gunanya adalah untuk menyimpan data sementara di object object tersebut dari file .csv. Sehingga, akan ada 2 kelas yaitu **User** dan **Team.** Sehingga kelas **User** dan kelas **Team** menjadi subclass dari kelas **Model**.
3. Anda juga akan diminta untuk dapat melakukan **join table** **kecil kecilan** pada poin ke 5.
4. Buatlah interface Class **Repository** yang akan diimplementasi oleh kelas-kelas Model yang lain. Sehingga, akan ada 2 Repository Class yaitu **UserRepository** dan **TeamRepository**. Tugas kelas Repository ini adalah untuk memproses query ke file-file .csv yang ada. Untuk itu, implementasi setiap Repository dengan pendekatan **façade pattern** untuk mengabstraksi proses logic di dalamnya. Interface Class **Repository** memiliki method:
   1. **find** untuk mencari data dari file .csv yang dimodelkan oleh kelas tersebut. Parameter adalah:
      1. string berisikan kolom untuk difilter.
      2. array of string berisikan kondisi filter kolom.
      3. Boolean Join table. True jika ingin melakukan join. False jika tidak.
      4. String nama table join.
      5. Connection file object.

Method ini akan me-return ArrayList of Object file csv yang telah dibuat pada requirement no 3. Semisal jika dijalankan dengan object connection bernama conn:

userRepository.find(“name”, [“=”,“kevin”], true, “Team”,conn);

Maka program akan mencari data di file user.csv, mencari user dengan nama persis kevin. Konsepnya sama dengan menjalankan sql query:

”Select user.id, user.name, user.nim, team.name from user inner join team on user.teamId = team.id where user.name = kevin”

Beberapa hal yang perlu diperhatikan:

1. Jika hasil query data tidak ada, maka return **null**.
2. Jika ada (paling tidak 1 row data), maka return ArrayList of User.
3. Jika kondisi parameter pertama dan kedua diberikan null, maka cari data tanpa ada kondisi, atau return semua data dari file .csv. Contoh lain:

teamRepository.find(null, null, null, null, conn);

Akan sama dengan

Select \* from Team;

1. Tampilkan error dan return null jika (lihat yang di-bold):
   1. String kolom kondisi filter itu null tapi ada kolom filternya. Contoh:

userRepository.find(**“name”, null**, null, null,conn);

* 1. String kondisi filter tidak null tapi kolom filter null. Contoh:

userRepository.find(**null, [“=”,“kevin”]**, null, null,conn);

* 1. String table join null, tapi boolean kondisi nya true.

userRepository.find(“name”, [“=”,“kevin”], **true, null**,conn);

* 1. String table join ada, tapi boolean kondisinya null.

userRepository.find(“name”, [“=”,“kevin”], **null, “Team”**,conn);

* 1. Selain itu, tampilkan juga error dan return null jika nilai parameter yang diberikan selain dari null dan spesifikasi tipe data yang telah diberikan. (semisal parameter pertama atau kedua diisi array of string kosong [“”], atau nama table join diisi string kosong “”).
  2. **findOne.** Untuk fungsi ini, sama persis dengan method **find**. Bedanya adalah fungsi ini hanya akan mereturn satu object, bukan Arraylist of object.
  3. **insert** untuk mengisi data ke file .csv object tersebut (write to file). Parameter adalah array of string data yang ingin diisi dan connection file object. **Id dari data yang dimasukkan akan auto increment dari id yang ada didalam file .csv**. Jika berhasil maka return object dengan data baru tersebut. Jika tidak berhasil, tampilkan pesan eror dan return null.

teamRepository.insert([“Team1”], conn);

Jika mengisi data User, maka contoh kode untuk input data adalah:

userRepository.insert([”user1”,”2600150125”,”Team1”],conn);

1. Kelompokkan setiap kelas dan superclass nya atau abstract class/interface class nya dalam package masing-masing (semisal Package Models untuk kelas User dan Team, Package Repository untuk Repository, UserRepository dan TeamRepository).
2. Pengumpulan dengan mengumpilkan link repository github. Isi github adalah file2 CSV dan project java.

GOOD LUCK AND HAVE FUN GUYS