Latihan Unit Testing

1. Test case selectAllStudents bertujuan untuk melakukan test method selectAllStudents pada class StudentServiceDatabase.
   Pada implementasi nya method selectAllStudents akan mereturn list of object StudentModel. Pertama yang dilakukan adalah di create object studentModel dan disimpan dalam ArrayList studentModels.
   Kedua, di create object studentModel yang juga disimpan dalam ArrayList checks.
   Ketiga, dibentuk mocking object dengan method stub selectAllStudents. Tujuannya kita ingin menghilangkan dependency antara object studentMapper dengan object lain. Karena fokus kita adalah pengetesan di method selectAllStudents. Jadi dibuat return value dari method selectAllStudents bukan real object melainkan mock object.
   Lalu pada when dilakukan pengetesan method selectAllStudents.
   Pada then dilakukan pengecekan dengan menggunakan assertion yaitu
   1. Check jika list dari object StudentModel tidak Null
   2. isEmpty() mereturn boolean apakah suatu variabel memiliki value atau tidak. Assert jika list tidak empty
   3. Check if zsize dari list equals to 1
   4. Check jika object pada list check dan object pada list checks sama

   Hasil test nya Runs 2/2

2. Test case ini untuk melakukan test method addStudent. Pertama dibentuk sebuah object studentModel. Kemudian di bentuk object mock dan stub method yang akan mereturn boolean value true menandakan bahwa proses add student berhasil dilakukan.
   Pada when dilakukan pengetesan method addStudent yang hasilnya adalah boolean value.
   Pada then dicek apakah resut test sama dengan true

   Hasil testnya Run 3/3

Yang dipelajari di Tutorial ini

Yang dipelajari adalah bagaimana melakukan unit testing pada aplikasi. Unit test bertujuan untuk melakukan pengetesan terhadap masing-masing method. Satu test case akan melakukan fungsi test untuk satu method. Hal yang perlu diketahui adalah input dan output yang diharapkan dalam suatu method. Saat melakukan unit test perlu memperhatikan dependency method dengan method lain yang dipanggil dalam method yang di test karena fokus kita adalah satu method sehingga kita tidak perlu benar-benar menjalankan fungsi lain yang dipanggil di method tersebut. Sehingga perlu melakukan stubbing method.

Stubbing method artinya kita mensimulasikan method tersebut seolah-olah melakukan real fungsi yang dilakukannya. Kita dapat memodifikasi return value yang dihasilkan method tersebut. Pada tutorial ini menggunakan 2 dependency yaitu Junit dan Mockito (untuk mocking object).

Yang dipelajari adalah melakukan load testing, untuk mengukur performance dari aplikasi kita menggunakan Jmeter. Kita bisa mensimulasikan jumlah user yang mengakses aplikasi kita dan menghitung sample time untuk masing-masing request. Dari hasil load testing kita dapat melakukan optimasi pada aplikasi kita misalnya optimasi di sisi database structure, query dan lainnya.

Langkah-langkah Optimasi

Optimasi yang dilakukan adalah dengan menambah index pada primary key npm dan juga menambah attribute unique pada primary key npm. Hasil dari optimasi tersebut tidak begitu significant. Fungsi view all student sebelum optimasi memiliki average time 42295 ms dan setelah optimasi menjadi 38287. Select student berhasil sebelum optimasi memiliki average time 26 ms dan setelah optimasi 13 ms. Select student tidak berhasil sebelum optimasi memiliki average 24 ms dan setelah optimasi menjadi 11 ms. Indexing berpengaruh ketika query memiliki where clause.

Sebenarnya banyak faktor eksternal yang mempengaruhi load testing, terutama pada kapasitas laptop yang digunakan. Jika memiliki RAM, CPU, memory yang lebih besar maka load test lebih cepat.