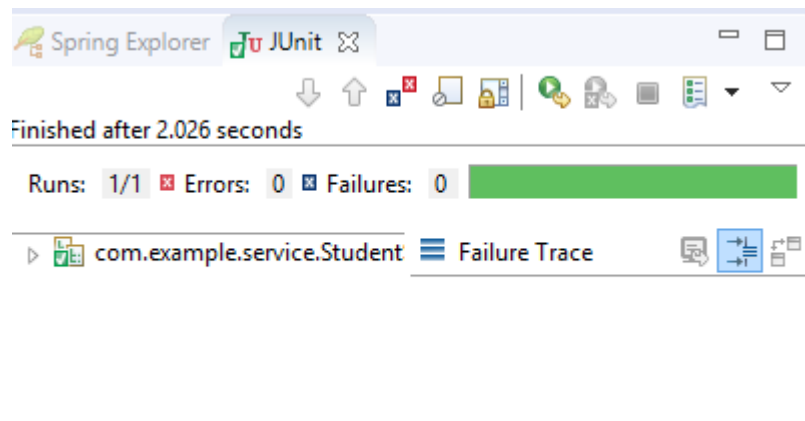


Nama : Olvi Lora S

NPM : 1606954943

Menjalankan Test pada Spring Boot



Latihan Unit Testing

1. Yang dilakukan pada method selectAllStudents adalah:

- Mendefinisikan variable studentModels yang bertipe list yang diisi dengan tipe StudentModel. Kemudian melakukan instansiasi objek StudentModel dengan parameter npm=1506737823, nama=Chanek dan gpa=3.5. Kemudian instansiasi tersebut ditambahkan ke list studentModels.

```
// Given
List<StudentModel> studentModels = new ArrayList<>();
StudentModel studentModel = new StudentModel("1506737823", "Chanek", 3.5);
studentModels.add(studentModel);
```

- Mendefinisikan variable checks yang bertipe list yang diisi dengan tipe StudentModel. Kemudian melakukan instansiasi objek check dengan parameter npm=1506737823, nama=Chanek dan gpa=3.5. Kemudian instansiasi tersebut ditambahkan ke list checks.

```
List<StudentModel> checks = new ArrayList<>();
StudentModel check = new StudentModel("1506737823", "Chanek", 3.5);
checks.add(check);
```

- Mengkonfigurasi Mockito untuk mengembalikan studentModels yang sudah ditambahkan sebelumnya menggunakan BDDMockito.

```
BDDMockito.given(studentMapper.selectAllStudents()).willReturn(studentModels);
```

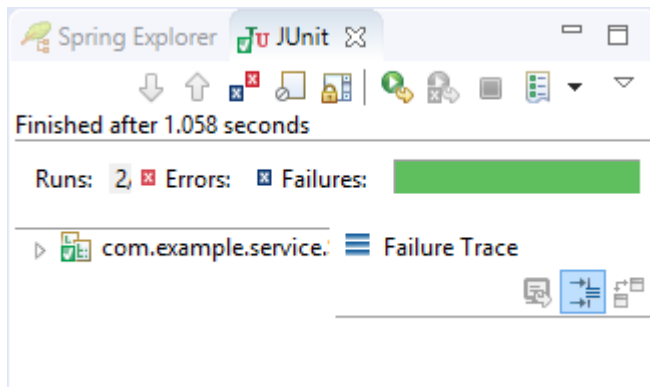
- Mendefinisikan variable test yang bertipe list yang berisi StudentModel. List test berisi List Student dengan npm=1506737832.

```
// When
List<StudentModel> test = studentService.selectAllStudents();
```

- Membandingkan hasil di List test tidak bernilai null, membandingkan jika hasil di List tidak kosong, membandingkan ukuran di List test sama dengan 1 dan membandingkan hasil di List test sama dengan List check yang sudah ditambahkan.

```
// Then
assertThat(test, notNullValue());
assertThat(test.isEmpty(), equalTo(false));
assertThat(test.size(), equalTo(1));
assertThat(test, equalTo(checks));
```

Hasil dari test method selectAllStudents adalah sebagai berikut.



2. Yang dilakukan pada method addStudent adalah:

- Menginstansiasi objek studentModel dan variabel check dengan parameter npm=1506737823, nama=Chanek dan gpa=3.5. Kemudian mengkonfigurasi Mockito untuk mengembalikan true ketika objek studentModel ditambahkan ke student menggunakan BDDMockito.

```
// Given
StudentModel studentModel = new StudentModel("1506737823", "Chanek", 3.5);
StudentModel check = new StudentModel("1506737823", "Chanek", 3.5);
BDDMockito.given(studentService.addStudent(studentModel)).willReturn(true);
```

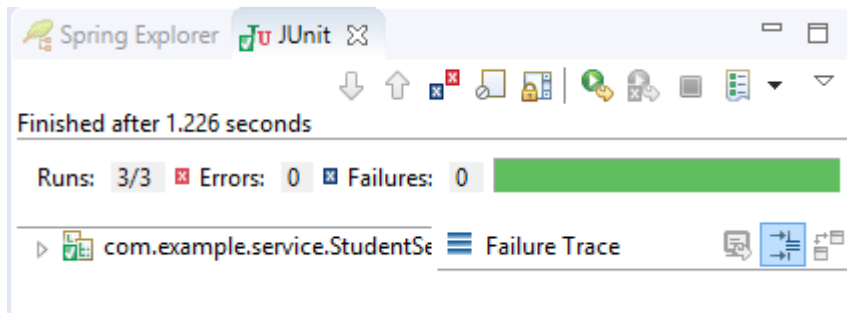
- Mendefinisikan Boolean test dengan nilai true jika studentModel berhasil ditambahkan dan menghasilkan false jika studentModel tidak berhasil ditambahkan.

```
// When
boolean test = studentService.addStudent(studentModel);
```

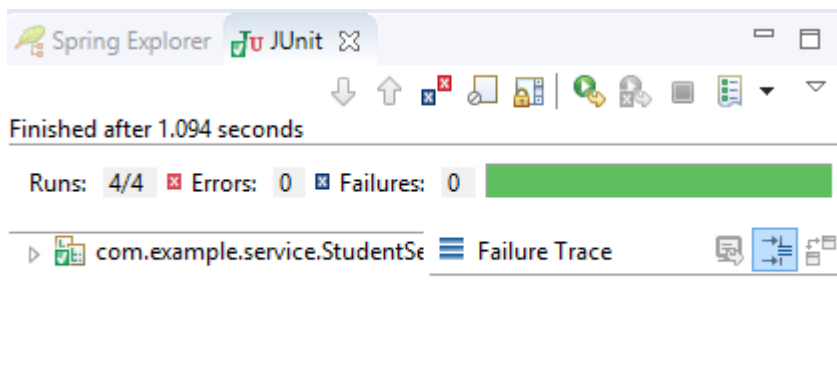
- Mockito mengkonfigurasi studentMapper seharusnya menambahkan student dengan npm=1506737823. Kemudian mengecek hasil method add student yang didefinisikan bernilai true.

```
// Then
BDDMockito.then(studentMapper).should().addStudent(check);
assertThat(test, equalTo(true));
```

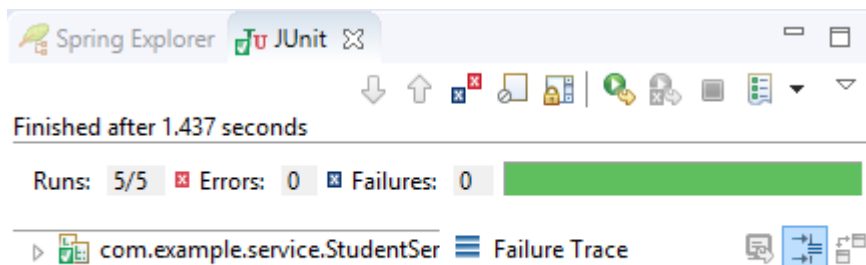
Hasil dari test method addStudent adalah:



3. Hasil testing method deleteStudent



4. Hasil testing method updateStudent



Latihan Load Testing

Sebelum Dioptimasi

1. Load testing selectStudent berhasil

Number of Threads (users) yang diset untuk testing ini adalah 100. Path yang dipanggil untuk testing ini adalah <http://localhost:8080/student/view/1506737823>. Hasil testing apabila melakukan select student dengan npm 1506737823 dengan banyaknya user yang mengkases 100 orang adalah seperti berikut.

Time yang dihabiskan : 11

Average : 63

Sample #	Start Time	Thread Name	Label	Sample Time(m...	Status	Bytes	Sent Bytes	Latency	Connect Time...
1	12:58:52.276	Thread Group 1...	HTTP Request	201	✓	398	141	200	37
2	12:58:52.375	Thread Group 1...	HTTP Request	102	✓	398	141	101	3
3	12:58:52.550	Thread Group 1...	HTTP Request	37	✓	398	141	37	3
4	12:58:52.671	Thread Group 1...	HTTP Request	43	✓	398	141	43	3
5	12:58:52.772	Thread Group 1...	HTTP Request	46	✓	398	141	46	3
6	12:58:52.871	Thread Group 1...	HTTP Request	38	✓	398	141	38	3
7	12:58:52.939	Thread Group 1...	HTTP Request	38	✓	398	141	38	2
8	12:58:53.039	Thread Group 1...	HTTP Request	42	✓	398	141	41	3
9	12:58:53.138	Thread Group 1...	HTTP Request	34	✓	398	141	33	3
10	12:58:53.239	Thread Group 1...	HTTP Request	44	✓	398	141	44	3
11	12:58:53.339	Thread Group 1...	HTTP Request	33	✓	398	141	33	2
12	12:58:53.473	Thread Group 1...	HTTP Request	68	✓	398	141	66	3
13	12:58:53.540	Thread Group 1...	HTTP Request	43	✓	398	141	43	3
14	12:58:53.674	Thread Group 1...	HTTP Request	47	✓	398	141	47	3
15	12:58:53.774	Thread Group 1...	HTTP Request	43	✓	398	141	43	3
16	12:58:53.938	Thread Group 1...	HTTP Request	30	✓	398	141	30	2
17	12:58:53.938	Thread Group 1...	HTTP Request	32	✓	398	141	32	2
18	12:58:54.038	Thread Group 1...	HTTP Request	36	✓	398	141	36	3
19	12:58:54.137	Thread Group 1...	HTTP Request	30	✓	398	141	30	2
20	12:58:54.273	Thread Group 1...	HTTP Request	38	✓	398	141	37	3
21	12:58:54.337	Thread Group 1...	HTTP Request	35	✓	398	141	35	3
22	12:58:54.437	Thread Group 1...	HTTP Request	42	✓	398	141	42	2
23	12:58:54.573	Thread Group 1...	HTTP Request	40	✓	398	141	40	3
24	12:58:54.671	Thread Group 1...	HTTP Request	33	✓	398	141	33	3
25	12:58:54.737	Thread Group 1...	HTTP Request	40	✓	398	141	40	4
26	12:58:54.836	Thread Group 1...	HTTP Request	44	✓	398	141	44	4
27	12:58:54.937	Thread Group 1...	HTTP Request	39	✓	398	141	37	3
28	12:58:55.034	Thread Group 1...	HTTP Request	40	✓	398	141	40	2
29	12:58:55.138	Thread Group 1...	HTTP Request	35	✓	398	141	35	2
30	12:58:55.239	Thread Group 1...	HTTP Request	36	✓	398	141	36	2

Summary: No of Samples 100, Latest Sample 40, Average 80, Deviation 79

2. Load testing selectStudent gagal

Number of Threads (users) yang diset untuk testing ini adalah 100. Path yang dipanggil untuk testing ini adalah `http://localhost:8080/student/view/1506777`. Hasil testing apabila melakukan select student dengan npm 1506777 (npm ini tidak ada di database) dengan banyaknya user yang mengkses 100 orang adalah seperti berikut.

Time yang dihabiskan : 10

Average : 35

Sample #	Start Time	Thread Name	Label	Sample Time(m...	Status	Bytes	Sent Bytes	Latency	Connect Time...
1	13:00:22.817	Thread Group 1...	HTTP Request	84	✓	363	136	84	4
2	13:00:23.000	Thread Group 1...	HTTP Request	43	✓	363	136	43	3
3	13:00:23.020	Thread Group 1...	HTTP Request	34	✓	363	136	34	2
4	13:00:23.129	Thread Group 1...	HTTP Request	30	✓	363	136	30	3
5	13:00:23.218	Thread Group 1...	HTTP Request	27	✓	363	136	27	2
6	13:00:23.323	Thread Group 1...	HTTP Request	27	✓	363	136	26	2
7	13:00:23.424	Thread Group 1...	HTTP Request	27	✓	363	136	26	2
8	13:00:23.547	Thread Group 1...	HTTP Request	26	✓	363	136	26	2
9	13:00:23.625	Thread Group 1...	HTTP Request	27	✓	363	136	26	2
10	13:00:23.751	Thread Group 1...	HTTP Request	41	✓	363	136	41	3
11	13:00:23.827	Thread Group 1...	HTTP Request	27	✓	363	136	27	2
12	13:00:23.920	Thread Group 1...	HTTP Request	26	✓	363	136	26	2
13	13:00:24.028	Thread Group 1...	HTTP Request	32	✓	363	136	32	2
14	13:00:24.135	Thread Group 1...	HTTP Request	27	✓	363	136	27	3
15	13:00:24.232	Thread Group 1...	HTTP Request	30	✓	363	136	30	2
16	13:00:24.332	Thread Group 1...	HTTP Request	39	✓	363	136	39	4
17	13:00:24.432	Thread Group 1...	HTTP Request	29	✓	363	136	28	2
18	13:00:24.534	Thread Group 1...	HTTP Request	29	✓	363	136	29	5
19	13:00:24.632	Thread Group 1...	HTTP Request	38	✓	363	136	38	3
20	13:00:24.733	Thread Group 1...	HTTP Request	32	✓	363	136	32	2
21	13:00:24.851	Thread Group 1...	HTTP Request	33	✓	363	136	32	2
22	13:00:24.937	Thread Group 1...	HTTP Request	97	✓	363	136	97	2
23	13:00:25.037	Thread Group 1...	HTTP Request	26	✓	363	136	26	3
24	13:00:25.137	Thread Group 1...	HTTP Request	25	✓	363	136	25	2
25	13:00:25.291	Thread Group 1...	HTTP Request	40	✓	363	136	40	3
26	13:00:25.339	Thread Group 1...	HTTP Request	35	✓	363	136	35	2
27	13:00:25.440	Thread Group 1...	HTTP Request	29	✓	363	136	28	2
28	13:00:25.596	Thread Group 1...	HTTP Request	36	✓	363	136	35	2
29	13:00:25.653	Thread Group 1...	HTTP Request	34	✓	363	136	34	2
30	13:00:25.751	Thread Group 1...	HTTP Request	34	✓	363	136	34	2

Summary: No of Samples 100, Latest Sample 40, Average 35, Deviation 11

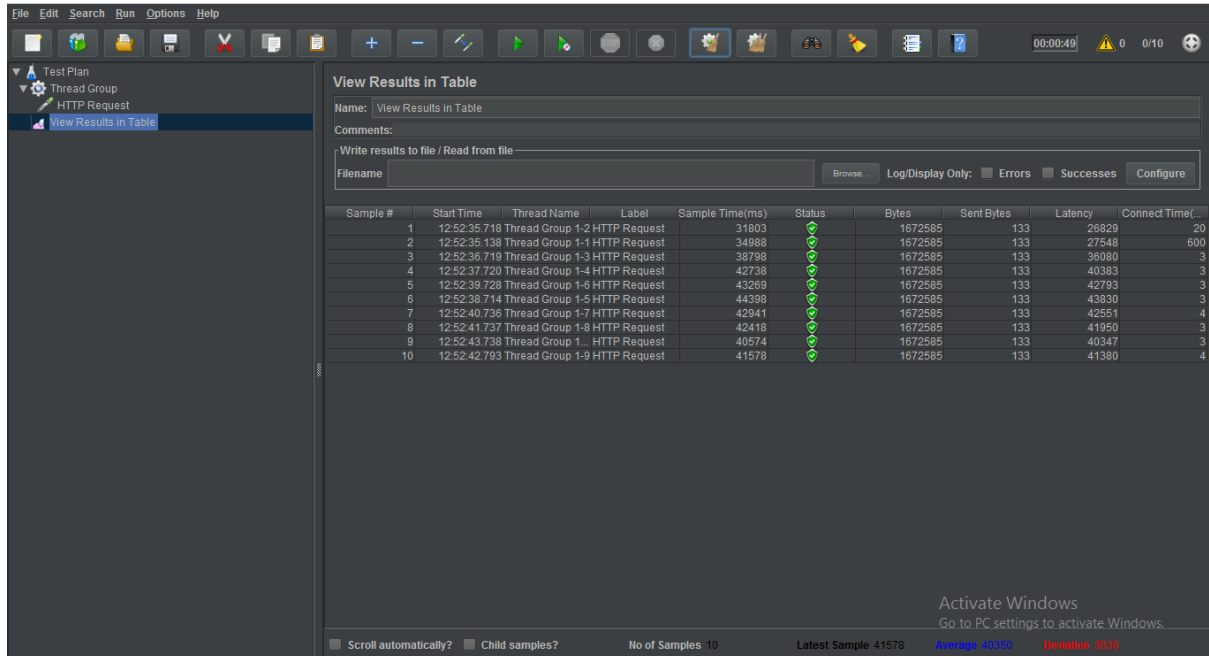
3. Load testing selectAllStudent

Number of Threads (users) yang diset untuk testing ini adalah 10. Diset 10 karena laptop saya tidak bisa compatible jika users nya di atas 10 dan menghasilkan error java space.

Path yang dipanggil untuk testing ini adalah <http://localhost:8080/student/viewall>. Hasil testing apabila melakukan select semua student dengan banyaknya user yang mengkases 10 orang adalah seperti berikut.

Time yang dihabiskan : 49

Average : 40350



The screenshot shows the JMeter 'View Results in Table' window. The table displays 10 samples of HTTP requests. The 'Sample Time(ms)' column shows values ranging from 31803 to 41578. The 'Status' column shows green checkmarks for all samples. The 'Bytes' column shows 1672585 for all samples. The 'Sent Bytes' column shows 133 for all samples. The 'Latency' column shows values ranging from 26829 to 41380. The 'Connect Time' column shows values ranging from 20 to 600.

Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time
1	12:52:35.718	Thread Group 1-2	HTTP Request	31803	✓	1672585	133	26829	20
2	12:52:35.138	Thread Group 1-1	HTTP Request	34988	✓	1672585	133	27548	600
3	12:52:36.719	Thread Group 1-3	HTTP Request	38798	✓	1672585	133	36080	3
4	12:52:37.720	Thread Group 1-4	HTTP Request	42738	✓	1672585	133	40383	3
5	12:52:39.728	Thread Group 1-6	HTTP Request	43269	✓	1672585	133	42793	3
6	12:52:38.714	Thread Group 1-5	HTTP Request	44398	✓	1672585	133	43830	3
7	12:52:40.736	Thread Group 1-7	HTTP Request	42941	✓	1672585	133	42551	4
8	12:52:41.737	Thread Group 1-8	HTTP Request	42418	✓	1672585	133	41950	3
9	12:52:43.738	Thread Group 1-9	HTTP Request	40574	✓	1672585	133	40347	3
10	12:52:42.793	Thread Group 1-9	HTTP Request	41578	✓	1672585	133	41380	4

At the bottom of the window, the summary statistics are displayed: Average: 40350, Deviation: 3238. The 'No of Samples' is 10, and the 'Latest Sample' is 41578.

Kemudian dilakukan optimasi agar response time yang dihasilkan bisa lebih cepat. Optimasi yang dilakukan dengan melakukan indexing. Indexing adalah sebuah data struktur yang menyimpan nilai spesifik sebuah kolom pada suatu table. Dengan melakukan indexing dapat membantu mempercepat proses eksekusi sebuah query. Hasil response time setelah melakukan optimasi lebih cepat dibandingkan sebelum dilakukan indexing.

Setelah Dioptimasi

1. Load testing selectStudent berhasil

Number of Threads (users) yang diset untuk testing ini adalah 100. Path yang dipanggil untuk testing ini adalah <http://localhost:8080/student/view/1506737823>. Hasil testing apabila melakukan select student dengan npm 1506737823 dengan banyaknya user yang mengkases 100 orang adalah seperti berikut.

Time yang dihabiskan : 10

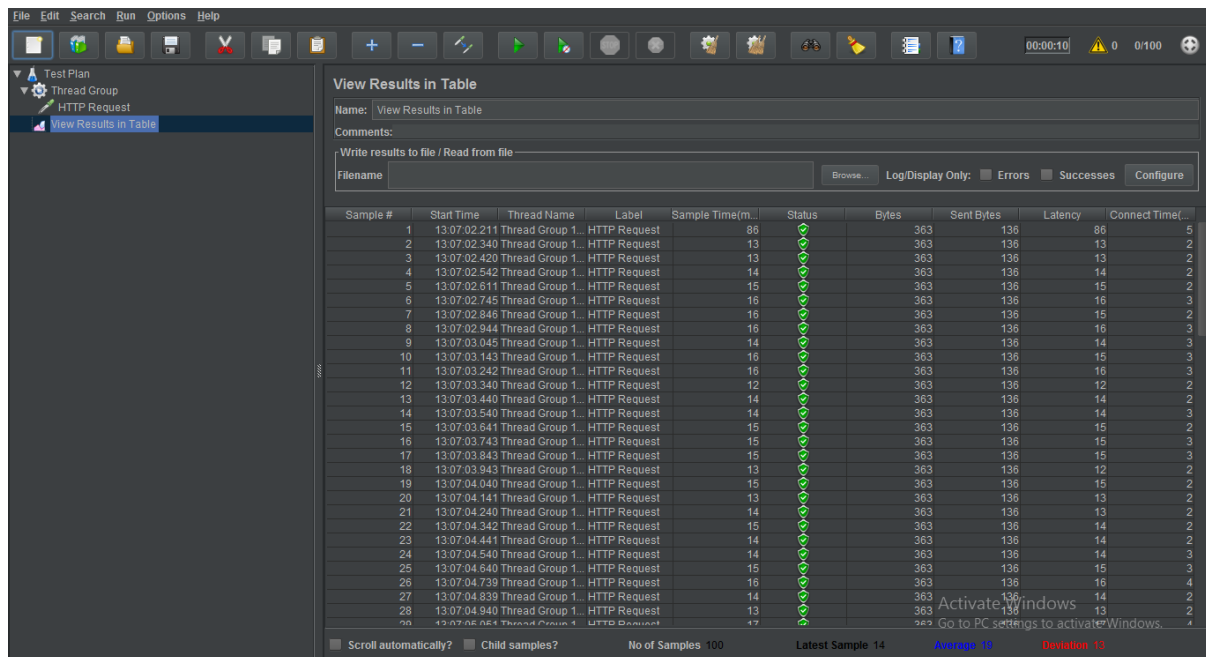
Average : 18

2. Load testing selectStudent gagal

Number of Threads (users) yang diset untuk testing ini adalah 100. Path yang dipanggil untuk testing ini adalah `http://localhost:8080/student/view/1506777`. Hasil testing apabila melakukan select student dengan npm 1506777 (npm ini tidak ada di database) dengan banyaknya user yang mengkasres 100 orang adalah seperti berikut.

Time yang dihabiskan : 10

Average : 19



Sample #	Start Time	Thread Name	Label	Sample Time(m)	Status	Bytes	Sent Bytes	Latency	Connect Time...
1	13.07.02.211	Thread Group 1...	HTTP Request	86	✓	363	136	86	5
2	13.07.02.240	Thread Group 1...	HTTP Request	13	✓	363	136	13	2
3	13.07.02.420	Thread Group 1...	HTTP Request	13	✓	363	136	13	2
4	13.07.02.542	Thread Group 1...	HTTP Request	14	✓	363	136	14	2
5	13.07.02.611	Thread Group 1...	HTTP Request	15	✓	363	136	15	2
6	13.07.02.745	Thread Group 1...	HTTP Request	16	✓	363	136	16	3
7	13.07.02.846	Thread Group 1...	HTTP Request	16	✓	363	136	15	2
8	13.07.02.944	Thread Group 1...	HTTP Request	16	✓	363	136	16	3
9	13.07.03.045	Thread Group 1...	HTTP Request	14	✓	363	136	14	3
10	13.07.03.143	Thread Group 1...	HTTP Request	16	✓	363	136	15	3
11	13.07.03.242	Thread Group 1...	HTTP Request	16	✓	363	136	16	3
12	13.07.03.340	Thread Group 1...	HTTP Request	12	✓	363	136	12	2
13	13.07.03.440	Thread Group 1...	HTTP Request	14	✓	363	136	14	2
14	13.07.03.540	Thread Group 1...	HTTP Request	14	✓	363	136	14	3
15	13.07.03.641	Thread Group 1...	HTTP Request	15	✓	363	136	15	2
16	13.07.03.743	Thread Group 1...	HTTP Request	15	✓	363	136	15	3
17	13.07.03.843	Thread Group 1...	HTTP Request	15	✓	363	136	15	3
18	13.07.03.943	Thread Group 1...	HTTP Request	13	✓	363	136	12	2
19	13.07.04.040	Thread Group 1...	HTTP Request	15	✓	363	136	15	2
20	13.07.04.141	Thread Group 1...	HTTP Request	13	✓	363	136	13	2
21	13.07.04.240	Thread Group 1...	HTTP Request	14	✓	363	136	13	2
22	13.07.04.342	Thread Group 1...	HTTP Request	15	✓	363	136	14	2
23	13.07.04.441	Thread Group 1...	HTTP Request	14	✓	363	136	14	2
24	13.07.04.540	Thread Group 1...	HTTP Request	14	✓	363	136	14	3
25	13.07.04.640	Thread Group 1...	HTTP Request	15	✓	363	136	15	3
26	13.07.04.739	Thread Group 1...	HTTP Request	16	✓	363	136	16	4
27	13.07.04.839	Thread Group 1...	HTTP Request	14	✓	363	136	14	2
28	13.07.04.940	Thread Group 1...	HTTP Request	13	✓	363	136	13	2
29	13.07.05.064	Thread Group 1...	HTTP Request	17	✓	363	136	17	4

Summary: No of Samples 100, Latest Sample 14, Average 19, Deviation 13

3. Load testing selectAllStudent

Number of Threads (users) yang diset untuk testing ini adalah 10. Diset 10 karena laptop saya tidak bisa compatible jika users nya di atas 10 dan menghasilkan error java space.

Path yang dipanggil untuk testing ini adalah `http://localhost:8080/student/viewall`. Hasil testing apabila melakukan select semua student dengan banyaknya user yang mengkasres 10 orang adalah seperti berikut.

Time yang dihabiskan : 33

Average : 22620

View Results in Table

Name: View Results in Table

Comments:

Write results to file / Read from file

Filename: Browse...

Log/Display Only: ☐ Errors ☐ Successes

Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time...
1	13:11:11.195	Thread Group 1-1	HTTP Request	6969	✓	1672585	133	5961	4
2	13:11:12.198	Thread Group 1-2	HTTP Request	15369	✓	1672585	133	11571	2
3	13:11:13.199	Thread Group 1-3	HTTP Request	22488	✓	1672585	133	20457	2
4	13:11:14.199	Thread Group 1-4	HTTP Request	25836	✓	1672585	133	23994	2
5	13:11:15.199	Thread Group 1-5	HTTP Request	26952	✓	1672585	133	25010	3
6	13:11:16.200	Thread Group 1-6	HTTP Request	27152	✓	1672585	133	26596	2
7	13:11:17.200	Thread Group 1-7	HTTP Request	26369	✓	1672585	133	26079	2
8	13:11:18.200	Thread Group 1-8	HTTP Request	25817	✓	1672585	133	25398	3
9	13:11:19.201	Thread Group 1-9	HTTP Request	25120	✓	1672585	133	24868	2
10	13:11:20.201	Thread Group 1-10	HTTP Request	24209	✓	1672585	133	24011	2

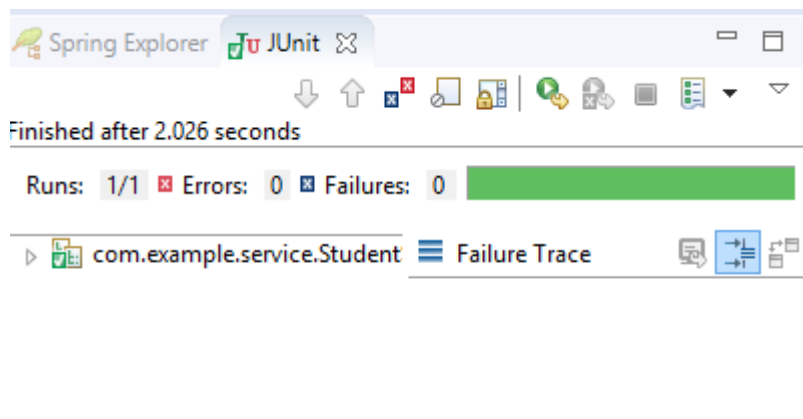
Activate Windows
Go to PC settings to activate Windows.

☐ Scroll automatically? ☐ Child samples? No of Samples: 10 Latest Sample: 24209 Average: 22020 Deviation: 6165

Rangkuman

Pada tutorial 6 ini saya belajar tentang Unit testing. Unit testing yaitu melakukan pengujian terhadap suatu method. Pengujian dilakukan menggunakan Junit dan Mockito. Junit merupakan sebuah testing framework yang terfokus pada pengecekan class dan method. Mockito merupakan dependencies untuk menciptakan dummy dalam pengecekan sehingga programmer dapat memanipulasi objek untuk dijadikan test case.

Kemudian saya belajar membuat method yang akan ditesting, yaitu dengan menambahkan class `StudentDataServiceDatabaseTest`. Dalam class tersebut terdapat anotasi `@Mock` yang berguna untuk membuat dummy pada `StudentMapper` agar hasil dari `StudentMapper` dapat dimanipulasi sesuai dengan test yang ingin dilakukan. Kemudian terdapat anotasi `@Before` untuk melakukan persiapan sebelum testing. Persiapannya dengan mengaktifkan semua anotasi `Mock` yang terdapat class ini. Untuk menguji method yang telah dibuat dapat dilakukan dengan klik kanan class test, run as Junit test. Kemudian akan ditampilkan gambar seperti dibawah yang menggambarkan test yang berjalan 1 dan tidak ada error maupun failure yang terjadi.



Pada setiap method terdapat 3 segmen yaitu:

1. Given : untuk menginisialisasi objek yang akan dites dan juga objek yang akan jadi parameter pengecekannya
2. When : mengetes method akan berjalan dengan memasukkannya pada objek baru.
3. Then : pengecekan terhadap objek baru apakah null atau sesuai dengan objek check yang telah disiapkan sebelumnya.

Pada tutorial ini juga saya belajar menggunakan JMeter untuk mengukur performansi aplikasi yang dibuat. Dan belajar melakukan optimasi di aplikasi supaya lebih cepat. Optimasi yang dilakukan dengan melakukan indexing di table student.