

Alex F. Manihuruk
1706106614

Unit Testing dan Load Testing Web App Sederhana

1. Unit Testing

Pada *testing* kali ini menggunakan Mockito dan Junit. Ada 3 segmen penting untuk setiap testnya yaitu Given, When, dan Then.

Given

Given bisa dibilang sebagai tahap persiapan karena di sinilah objek dibuat, baik objek yang akan dites maupun objek yang akan jadi parameter pengecekannya.

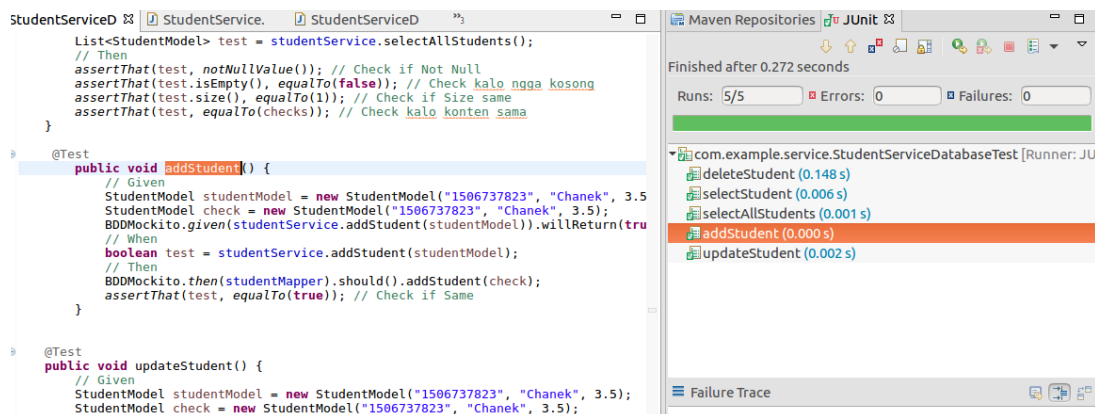
When

When bisa dibilang sebagai tahap eksekusi, karena pada dasarnya di sini hanya memanggil *method* yang diuji.

Then

Then sebagai tahap verifikasi, karena di bagian ini akan dicek hasil apa pun dari langkah eksekusi yang dilaksanakan sebelumnya. Bisa berupa pengecekan *null* atau dengan hasil lain yang telah *mocked*

Berikut hasil dari *testing* dari 5 *method* yaitu *deleteStudent()*, *selectStudent()*, *selectAllStudent*, *addStudent()*, dan *updateStudent()*.



The screenshot displays an IDE with two main panels. The left panel shows Java code for unit tests in the `StudentServiceD` class. The code includes tests for `selectAllStudents()`, `addStudent()`, and `updateStudent()`, utilizing Mockito for mocking and JUnit for assertions. The right panel shows the JUnit runner output, indicating that the tests finished after 0.272 seconds with 5 runs, 0 errors, and 0 failures. The output lists the execution time for each test method: `deleteStudent` (0.148 s), `selectStudent` (0.006 s), `selectAllStudents` (0.001 s), `addStudent` (0.000 s), and `updateStudent` (0.002 s).

```
StudentServiceD
List<StudentModel> test = studentService.selectAllStudents();
// Then
assertThat(test, notNullValue()); // Check if Not Null
assertThat(test.isEmpty(), equalTo(false)); // Check kalo ngga kosong
assertThat(test.size(), equalTo(1)); // Check if Size same
assertThat(test, equalTo(checks)); // Check kalo konten sama
}

@Test
public void addStudent() {
    // Given
    StudentModel studentModel = new StudentModel("1506737823", "Chanek", 3.5);
    StudentModel check = new StudentModel("1506737823", "Chanek", 3.5);
    BDDMockito.given(studentService.addStudent(studentModel)).willReturn(true);
    // When
    boolean test = studentService.addStudent(studentModel);
    // Then
    BDDMockito.then(studentMapper).should().addStudent(check);
    assertThat(test, equalTo(true)); // Check if Same
}

@Test
public void updateStudent() {
    // Given
    StudentModel studentModel = new StudentModel("1506737823", "Chanek", 3.5);
    StudentModel check = new StudentModel("1506737823", "Chanek", 3.5);
}
```

JUnit Output:

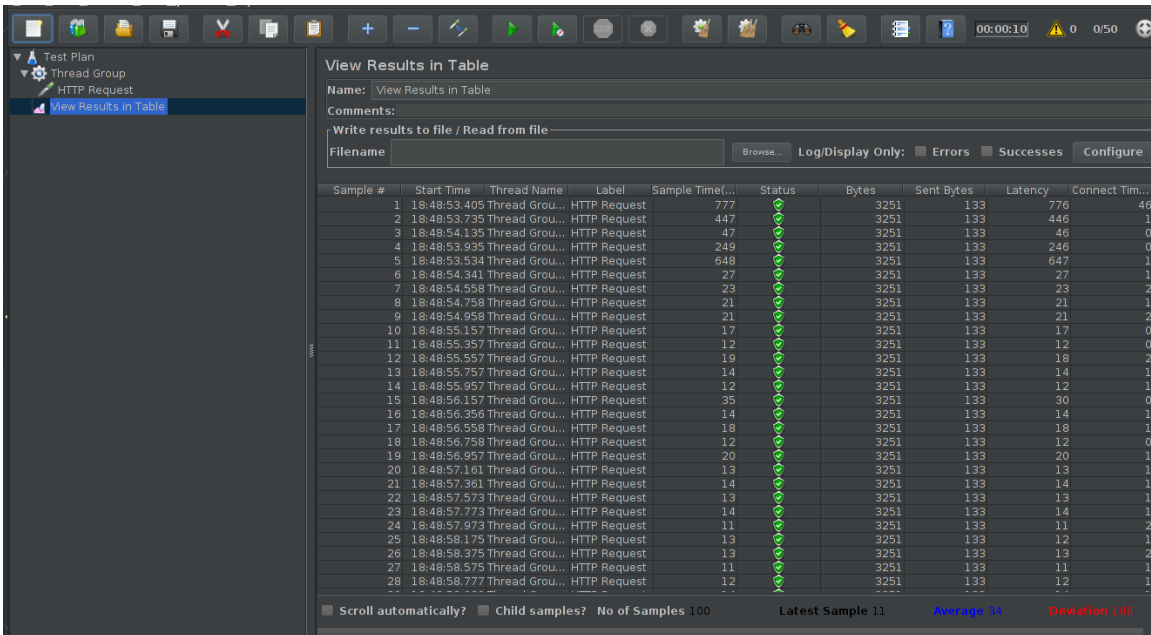
```
Finished after 0.272 seconds
Runs: 5/5 Errors: 0 Failures: 0
com.example.service.StudentServiceDatabaseTest [Runner: JUnit]
  deleteStudent (0.148 s)
  selectStudent (0.006 s)
  selectAllStudents (0.001 s)
  addStudent (0.000 s)
  updateStudent (0.002 s)
```

2. Load testing

Load testing bertujuan untuk melakukan pengujian respon dari sebuah sistem atau aplikasi dengan memberikan request tertentu. Load testing dilakukan untuk mengetahui performa dan respon dari suatu sistem atau aplikasi baik dalam kondisi respon normal maupun dalam kondisi respon yang ekstrim. Load testing dapat membantu kita untuk menentukan kapasitas maksimum dari sistem atau aplikasi yang kita buat.

Pada *load testing* ini akan menggunakan Apache Jmeter. Berikut tampilan-tampilan *load testing* dari beberapa method.

selectAllStudent

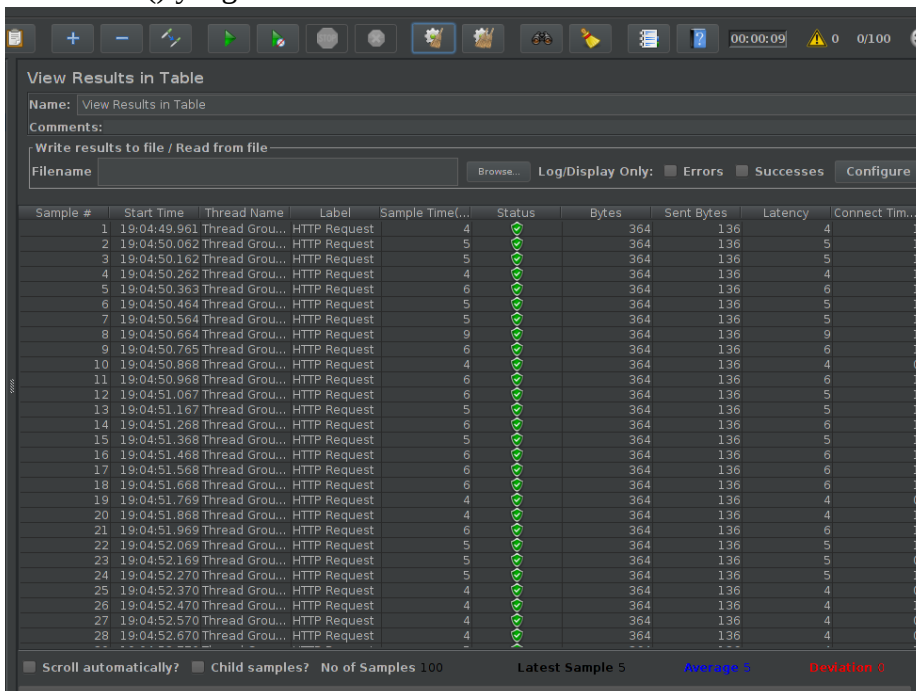


The screenshot shows the Apache JMeter interface with the 'View Results in Table' window open. The table displays 28 samples of HTTP requests. The status column shows green checkmarks for all samples, indicating successful requests. The latency and connect time columns show values for each sample.

Sample #	Start Time	Thread Name	Label	Sample Time...	Status	Bytes	Sent Bytes	Latency	Connect Tim...
1	18:48:53.405	Thread Grou...	HTTP Request	777	✓	3251	133	776	46
2	18:48:53.735	Thread Grou...	HTTP Request	447	✓	3251	133	446	1
3	18:48:54.135	Thread Grou...	HTTP Request	47	✓	3251	133	46	0
4	18:48:53.935	Thread Grou...	HTTP Request	249	✓	3251	133	246	0
5	18:48:53.534	Thread Grou...	HTTP Request	648	✓	3251	133	647	1
6	18:48:54.341	Thread Grou...	HTTP Request	27	✓	3251	133	27	1
7	18:48:54.558	Thread Grou...	HTTP Request	23	✓	3251	133	23	2
8	18:48:54.758	Thread Grou...	HTTP Request	21	✓	3251	133	21	1
9	18:48:54.958	Thread Grou...	HTTP Request	21	✓	3251	133	21	2
10	18:48:55.157	Thread Grou...	HTTP Request	17	✓	3251	133	17	0
11	18:48:55.357	Thread Grou...	HTTP Request	12	✓	3251	133	12	0
12	18:48:55.557	Thread Grou...	HTTP Request	19	✓	3251	133	18	2
13	18:48:55.757	Thread Grou...	HTTP Request	14	✓	3251	133	14	1
14	18:48:55.957	Thread Grou...	HTTP Request	12	✓	3251	133	12	1
15	18:48:56.157	Thread Grou...	HTTP Request	35	✓	3251	133	30	0
16	18:48:56.356	Thread Grou...	HTTP Request	14	✓	3251	133	14	1
17	18:48:56.558	Thread Grou...	HTTP Request	18	✓	3251	133	18	1
18	18:48:56.758	Thread Grou...	HTTP Request	12	✓	3251	133	12	0
19	18:48:56.957	Thread Grou...	HTTP Request	20	✓	3251	133	20	1
20	18:48:57.161	Thread Grou...	HTTP Request	13	✓	3251	133	13	1
21	18:48:57.361	Thread Grou...	HTTP Request	14	✓	3251	133	14	1
22	18:48:57.573	Thread Grou...	HTTP Request	13	✓	3251	133	13	1
23	18:48:57.773	Thread Grou...	HTTP Request	14	✓	3251	133	14	1
24	18:48:57.973	Thread Grou...	HTTP Request	11	✓	3251	133	11	2
25	18:48:58.175	Thread Grou...	HTTP Request	13	✓	3251	133	12	1
26	18:48:58.375	Thread Grou...	HTTP Request	13	✓	3251	133	13	2
27	18:48:58.575	Thread Grou...	HTTP Request	11	✓	3251	133	11	1
28	18:48:58.777	Thread Grou...	HTTP Request	12	✓	3251	133	12	1

Summary statistics at the bottom: Latest Sample 11, Average 34, Deviation 139.

selectStudent() yang berhasil



The screenshot shows the Apache JMeter interface with the 'View Results in Table' window open. The table displays 28 samples of HTTP requests. The status column shows green checkmarks for all samples, indicating successful requests. The latency and connect time columns show values for each sample.

Sample #	Start Time	Thread Name	Label	Sample Time...	Status	Bytes	Sent Bytes	Latency	Connect Tim...
1	19:04:49.961	Thread Grou...	HTTP Request	4	✓	364	136	4	1
2	19:04:50.062	Thread Grou...	HTTP Request	5	✓	364	136	5	1
3	19:04:50.162	Thread Grou...	HTTP Request	5	✓	364	136	5	1
4	19:04:50.262	Thread Grou...	HTTP Request	4	✓	364	136	4	1
5	19:04:50.363	Thread Grou...	HTTP Request	6	✓	364	136	6	1
6	19:04:50.464	Thread Grou...	HTTP Request	5	✓	364	136	5	1
7	19:04:50.564	Thread Grou...	HTTP Request	5	✓	364	136	5	1
8	19:04:50.664	Thread Grou...	HTTP Request	9	✓	364	136	9	1
9	19:04:50.765	Thread Grou...	HTTP Request	6	✓	364	136	6	1
10	19:04:50.868	Thread Grou...	HTTP Request	4	✓	364	136	4	0
11	19:04:50.968	Thread Grou...	HTTP Request	6	✓	364	136	6	1
12	19:04:51.067	Thread Grou...	HTTP Request	6	✓	364	136	6	1
13	19:04:51.167	Thread Grou...	HTTP Request	5	✓	364	136	5	1
14	19:04:51.268	Thread Grou...	HTTP Request	6	✓	364	136	6	1
15	19:04:51.368	Thread Grou...	HTTP Request	5	✓	364	136	5	1
16	19:04:51.468	Thread Grou...	HTTP Request	6	✓	364	136	6	1
17	19:04:51.568	Thread Grou...	HTTP Request	6	✓	364	136	6	1
18	19:04:51.668	Thread Grou...	HTTP Request	6	✓	364	136	6	1
19	19:04:51.769	Thread Grou...	HTTP Request	4	✓	364	136	4	0
20	19:04:51.868	Thread Grou...	HTTP Request	4	✓	364	136	4	1
21	19:04:51.969	Thread Grou...	HTTP Request	6	✓	364	136	6	1
22	19:04:52.069	Thread Grou...	HTTP Request	5	✓	364	136	5	1
23	19:04:52.169	Thread Grou...	HTTP Request	5	✓	364	136	5	0
24	19:04:52.270	Thread Grou...	HTTP Request	5	✓	364	136	5	1
25	19:04:52.370	Thread Grou...	HTTP Request	4	✓	364	136	4	0
26	19:04:52.470	Thread Grou...	HTTP Request	4	✓	364	136	4	1
27	19:04:52.570	Thread Grou...	HTTP Request	4	✓	364	136	4	0
28	19:04:52.670	Thread Grou...	HTTP Request	4	✓	364	136	4	0

Summary statistics at the bottom: Latest Sample 5, Average 5, Deviation 0.

selectSudent() yang gagal

View Results in Table.jmx (/home/blackw/ALEX/UI/KULIAH/2/apap/apache-jmeter-4.0/bin/View Results in Table.jmx) - Apache JMeter (4.0 r182341) 19:04

File Edit Search Run Options Help

Test Plan
Thread Group
HTTP Request
View Results in Table

View Results in Table

Name: View Results in Table

Comments:

Write results to file / Read from file

Filename: Browse... Log/Display Only: ☐ Errors ☐ Successes ☐ Configure

Sample #	Start Time	Thread Name	Label	Sample Time(...)	Status	Bytes	Sent Bytes	Latency	Connect Tim...
1	19:03:51.572	Thread Grou...	HTTP Request	4	✓	340	139	4	1
2	19:03:51.676	Thread Grou...	HTTP Request	4	✓	340	139	4	1
3	19:03:51.777	Thread Grou...	HTTP Request	5	✓	340	139	5	1
4	19:03:51.881	Thread Grou...	HTTP Request	4	✓	340	139	3	0
5	19:03:51.991	Thread Grou...	HTTP Request	5	✓	340	139	5	1
6	19:03:52.090	Thread Grou...	HTTP Request	5	✓	340	139	5	1
7	19:03:52.191	Thread Grou...	HTTP Request	6	✓	340	139	5	1
8	19:03:52.291	Thread Grou...	HTTP Request	4	✓	340	139	4	0
9	19:03:52.391	Thread Grou...	HTTP Request	7	✓	340	139	7	1
10	19:03:52.491	Thread Grou...	HTTP Request	5	✓	340	139	5	0
11	19:03:52.591	Thread Grou...	HTTP Request	4	✓	340	139	4	1
12	19:03:52.693	Thread Grou...	HTTP Request	6	✓	340	139	5	1
13	19:03:52.796	Thread Grou...	HTTP Request	3	✓	340	139	3	0
14	19:03:52.896	Thread Grou...	HTTP Request	5	✓	340	139	5	1
15	19:03:52.997	Thread Grou...	HTTP Request	5	✓	340	139	4	1
16	19:03:53.096	Thread Grou...	HTTP Request	3	✓	340	139	3	1
17	19:03:53.196	Thread Grou...	HTTP Request	3	✓	340	139	3	1
18	19:03:53.297	Thread Grou...	HTTP Request	5	✓	340	139	4	1
19	19:03:53.396	Thread Grou...	HTTP Request	5	✓	340	139	5	1
20	19:03:53.496	Thread Grou...	HTTP Request	5	✓	340	139	5	1
21	19:03:53.597	Thread Grou...	HTTP Request	5	✓	340	139	5	1
22	19:03:53.698	Thread Grou...	HTTP Request	4	✓	340	139	4	1
23	19:03:53.797	Thread Grou...	HTTP Request	4	✓	340	139	4	1
24	19:03:53.898	Thread Grou...	HTTP Request	5	✓	340	139	5	1
25	19:03:53.997	Thread Grou...	HTTP Request	5	✓	340	139	5	1
26	19:03:54.098	Thread Grou...	HTTP Request	4	✓	340	139	4	1
27	19:03:54.199	Thread Grou...	HTTP Request	4	✓	340	139	4	0
28	19:03:54.302	Thread Grou...	HTTP Request	5	✓	340	139	5	1
--	--	--	--	--	--	--	--	--	--

☐ Scroll automatically? ☐ Child samples? No of Samples 100 Latest Sample 5 Average 4 Deviation 0