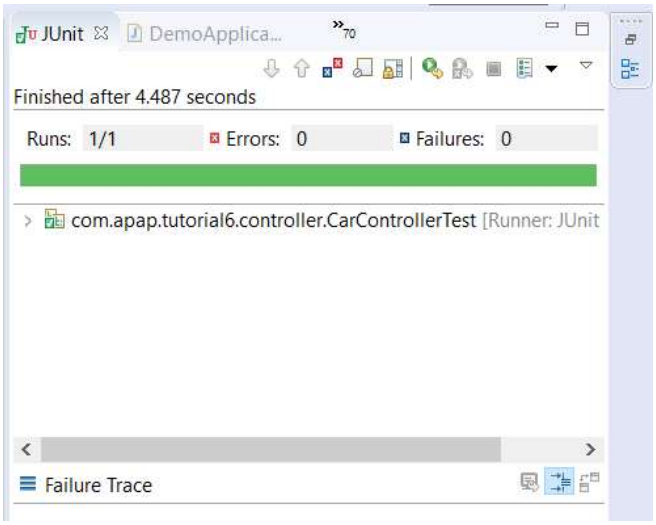
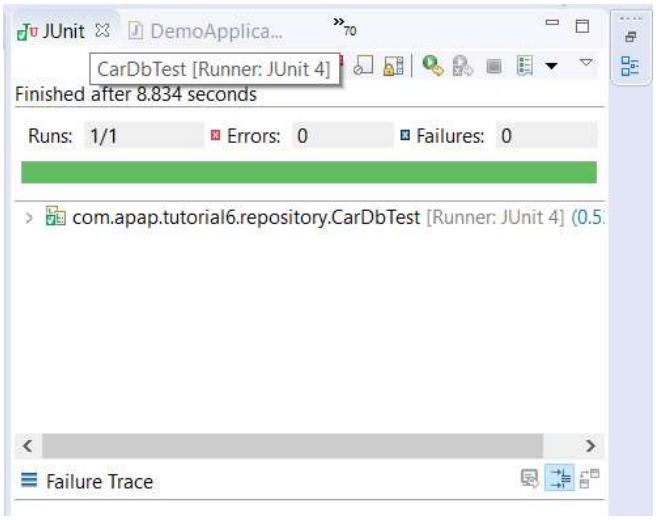
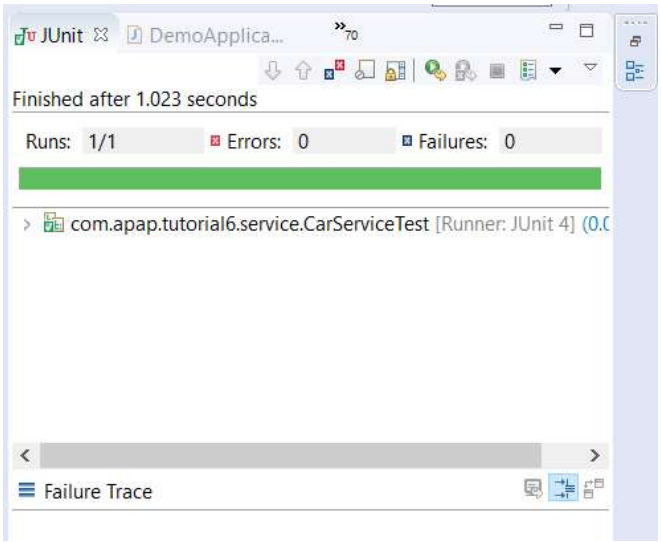
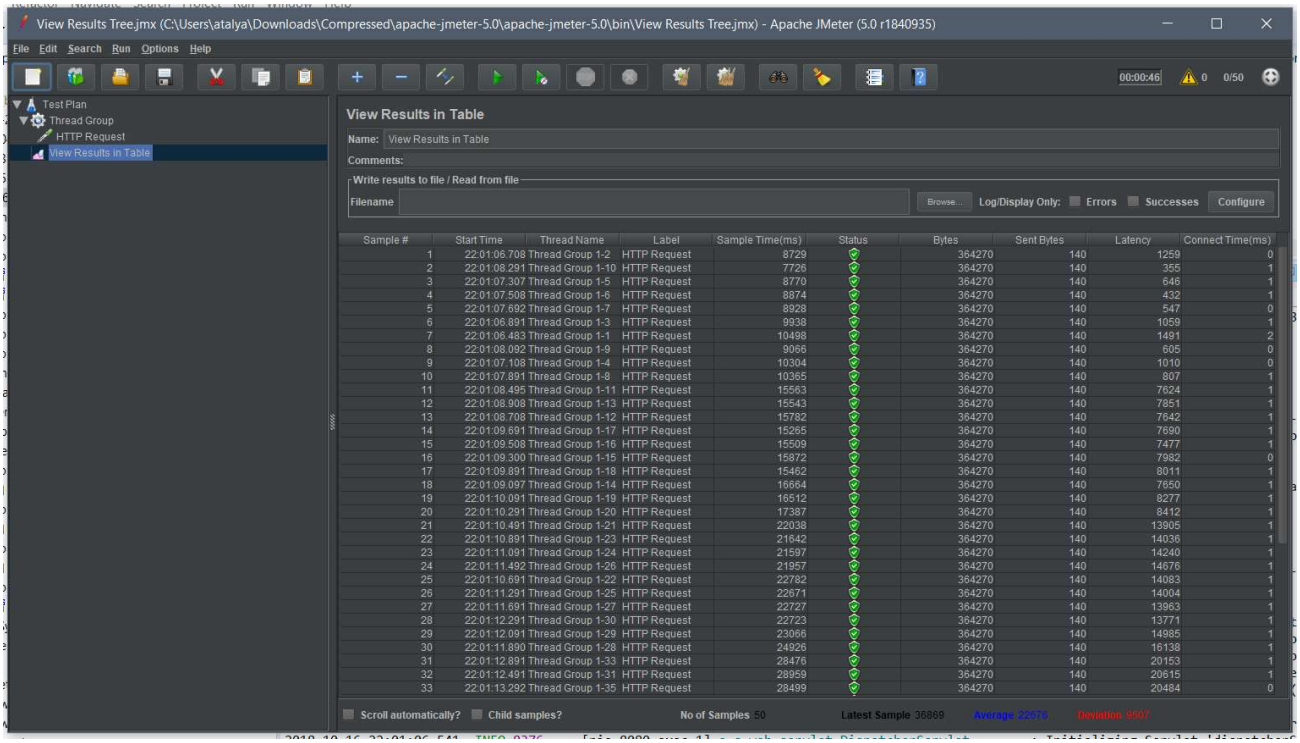

JUnit
DemoApplica...
Finished after 4.487 seconds
Runs: 1/1
Errors: 0
Failures: 0
com.apap.tutorial6.controller.CarControllerTest [Runner: JUnit
Failure Trace


JUnit
DemoApplica...
CarDbTest [Runner: JUnit 4]
Finished after 8.834 seconds
Runs: 1/1
Errors: 0
Failures: 0
com.apap.tutorial6.repository.CarDbTest [Runner: JUnit 4]
Failure Trace


JUnit
DemoApplica...
Finished after 1.023 seconds
Runs: 1/1
Errors: 0
Failures: 0
com.apap.tutorial6.service.CarServiceTest [Runner: JUnit 4]
Failure Trace


View Results Tree.jmx (C:\Users\atalya\Downloads\Compressed\apache-jmeter-5.0\apache-jmeter-5.0\bin\View Results Tree.jmx) - Apache JMeter (5.0 r1840935)
File Edit Search Run Options Help
00:00:46
0
0/50
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(ms) Status Bytes Sent Bytes Latency Connect Time(ms)
1 22:01:06.708 Thread Group 1-2 HTTP Request 8729 364270 140 1259 0
2 22:01:08.291 Thread Group 1-10 HTTP Request 7726 364270 140 355 1
3 22:01:07.307 Thread Group 1-5 HTTP Request 8770 364270 140 646 1
4 22:01:07.508 Thread Group 1-6 HTTP Request 8874 364270 140 432 1
5 22:01:07.692 Thread Group 1-7 HTTP Request 8928 364270 140 547 0
6 22:01:06.891 Thread Group 1-3 HTTP Request 9938 364270 140 1059 1
7 22:01:06.483 Thread Group 1-1 HTTP Request 10498 364270 140 1491 2
8 22:01:08.092 Thread Group 1-9 HTTP Request 9066 364270 140 605 0
9 22:01:07.108 Thread Group 1-4 HTTP Request 10304 364270 140 1010 0
10 22:01:07.891 Thread Group 1-8 HTTP Request 10365 364270 140 807 1
11 22:01:08.495 Thread Group 1-11 HTTP Request 15563 364270 140 7624 1
12 22:01:08.908 Thread Group 1-13 HTTP Request 15543 364270 140 7851 1
13 22:01:08.708 Thread Group 1-12 HTTP Request 15782 364270 140 7642 1
14 22:01:09.691 Thread Group 1-17 HTTP Request 15265 364270 140 7690 1
15 22:01:09.508 Thread Group 1-16 HTTP Request 15509 364270 140 7477 1
16 22:01:09.300 Thread Group 1-15 HTTP Request 15872 364270 140 7982 0
17 22:01:09.891 Thread Group 1-18 HTTP Request 15462 364270 140 8011 1
18 22:01:09.097 Thread Group 1-14 HTTP Request 16664 364270 140 7650 1
19 22:01:10.091 Thread Group 1-19 HTTP Request 16512 364270 140 8277 1
20 22:01:10.291 Thread Group 1-20 HTTP Request 17387 364270 140 8412 1
21 22:01:10.491 Thread Group 1-21 HTTP Request 22038 364270 140 13905 1
22 22:01:10.891 Thread Group 1-23 HTTP Request 21642 364270 140 14036 1
23 22:01:11.091 Thread Group 1-24 HTTP Request 21597 364270 140 14240 1
24 22:01:11.492 Thread Group 1-26 HTTP Request 21957 364270 140 14676 1
25 22:01:10.691 Thread Group 1-22 HTTP Request 22782 364270 140 14083 1
26 22:01:11.291 Thread Group 1-25 HTTP Request 22671 364270 140 14004 1
27 22:01:11.691 Thread Group 1-27 HTTP Request 22727 364270 140 13963 1
28 22:01:12.291 Thread Group 1-30 HTTP Request 22723 364270 140 13771 1
29 22:01:12.091 Thread Group 1-29 HTTP Request 23066 364270 140 14985 1
30 22:01:11.890 Thread Group 1-28 HTTP Request 24926 364270 140 16138 1
31 22:01:12.891 Thread Group 1-33 HTTP Request 28476 364270 140 20153 1
32 22:01:12.491 Thread Group 1-31 HTTP Request 28959 364270 140 20615 1
33 22:01:13.292 Thread Group 1-35 HTTP Request 28499 364270 140 20484 0
Scroll automatically? Child samples? No of Samples 50 Latest Sample 36869 Average 22676 Deviation 9507

1. Mengapa perlu menginisiasi object DealerModel, sedangkan yang di test hanya CarModel?
   ➔ Karena untuk mencari suatu objek car harus diketahui objek dealernya. Atau dengan kata lain, terdapat attribute pada CarModel yang me-reference ke attribute di class DealerModel.

2. Jelaskan apa yang akan terjadi jika object DealerModel dihapus dan tidak dilakukan setDealer pada CarModel?
   → compile error. Karena pada given terdapat baris kode `carModel.setDealer(dealerModel);` Yang memerlukan inisiasi objek DealerModel.

3. Jelaskan apa yang dilakukan oleh code
   ```
   Mockito.when(carDb.findByType(carModel.get().getType())).thenReturn(carModel);
   ```
   ➔ Mockito digunakan untuk membuat dan mengonfigurasikan mock objects. when digunakan untuk memanggil method yang akan di-test, yaitu method findByType(String type) di CarDb. Lalu, `thenReturn(carModel)` digunakan untuk melakukan pengecekan pada objek baru.

4. Jelaskan apa yang dilakukan oleh code
   ```
   Mockito.when(carService.getCarDetailByType(carModel.get().getType())).thenReturn(carModel);
   ```
   ➔ Mockito digunakan untuk membuat dan mengonfigurasikan mock objects. when digunakan untuk memanggil method yang akan di-test, yaitu method getCarDetailByType(String type) di CarService. Lalu, `thenReturn(carModel)` digunakan untuk melakukan pengecekan pada objek baru.

5. Jelaskan apa yang ditest oleh code
   ```
   .andExpect(MockMvcResultMatchers.status().isOk())
   ```
   → code akan melakukan pengecekan terhadap hasil dari pemanggilan route tersebut. Code ini mengecek kode status respons adalah HttpStatus.OK (200).

6. Jelaskan apa yang ditest oleh code
   ```
   .andExpect(MockMvcResultMatchers.jsonPath("$.type", Matchers.is(car.getType())));
   ```
   → code akan melakukan pengecekan terhadap hasil dari pemanggilan route tersebut. Code ini melakukan verifikasi bahwa pernyataan “$.type” itu benar dan cocok dengan tipe mobil (car.getType()).

7. Jelaskan anotasi @ResponseBody yang ada pada route “car/view”
   ```
   @RequestMapping(value = "/car/view", method = RequestMethod.GET)
   private @ResponseBody CarModel view(@RequestParam(value = "type") String type, Model model) {
       CarModel archive = carService.getCarDetailByType(type).get();
       return archive;
   }
   ```
   ➔ Anotasi @RespondBody dapat diletakkan pada metode dan menunjukkan bahwa pengembalian jenis harus ditulis langsung ke tubuh respons HTTP.