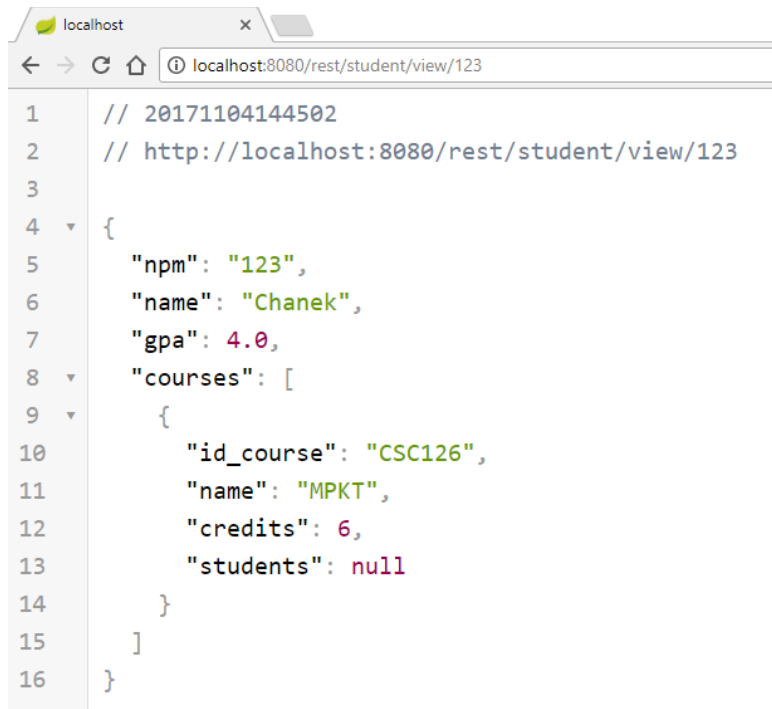


Write up tutorial 7



The screenshot shows a web browser window with a single tab labeled 'localhost'. The address bar displays 'localhost:8080/rest/student/view/123'. Below the address bar, a REST client interface shows a JSON response for the GET request. The response is a JSON object with the following structure:

```
1 // 20171104144502
2 // http://localhost:8080/rest/student/view/123
3
4 {
5   "npm": "123",
6   "name": "ChaneK",
7   "gpa": 4.0,
8   "courses": [
9     {
10      "id_course": "CSC126",
11      "name": "MPKT",
12      "credits": 6,
13      "students": null
14    }
15  ]
16 }
```

Latihan 1: Buatlah service untuk mengembalikan seluruh student yang ada di basis data. Service ini mirip seperti method viewAll di Web Controller. Service tersebut di-mapping ke “/rest/student/viewall”.



```
localhost:8080/rest/stude x
localhost:8080/rest/student/viewall

1 // 20171104144754
2 // http://localhost:8080/rest/student/viewall
3
4 [
5   {
6     "npm": "123",
7     "name": "Chanek",
8     "gpa": 4.0,
9     "courses": [
10      {
11        "id_course": "CSC126",
12        "name": "MPKT",
13        "credits": 6,
14        "students": null
15      }
16    ]
17  },
18  {
19    "npm": "124",
20    "name": "Chanek Jr.",
21    "gpa": 3.0,
22    "courses": [
23      {
24        "id_course": "CSC123",
25        "name": "PSP",
26        "credits": 4,
27        "students": null
28      },
29      {
30        "id_course": "CSC124",
31        "name": "SDA",
32        "credits": 3,
33        "students": null
34      }
35    ]
36  },
37  {
38    "npm": "125",
39    "name": "Chanek Jr plus",
40    "gpa": 3.2,
41    "courses": []
42  },
43  {
44    "npm": "1406557535",
45    "name": "Luthfi Abdurrahim401",
46    "gpa": 4.01,
47    "courses": [
48      {
49        "id_course": "CSC123",
50        "name": "PSP",
51        "credits": 4,
52        "students": null
53      }
54    ]
55  },
56  {
57    "npm": "140655753599",
58    "name": "Luthfi Abdurrahim3",
59    "gpa": 3.9901,
60    "courses": [
61      {
62        "id_course": "CSC123",
63        "name": "PSP",
64        "credits": 4,
65        "students": null
66      }
67    ]
68  },
69  {
70    "npm": "1406557536",
71    "name": "Luthfi Abdurrahima",
72    "gpa": 3.999,
73    "courses": []
74  }
75 ]
```

```
[{"npm":"123","name":"Chanek","gpa":4.0,"courses":
[{"id_course":"CSC126","name":"MPKT","credits":6,"students":null}]},
{"npm":"124","name":"Chanek Jr.", "gpa":3.0,"courses":
[{"id_course":"CSC123","name":"PSP","credits":4,"students":null},
{"id_course":"CSC124","name":"SDA","credits":3,"students":null}]},
{"npm":"125","name":"chanek Jr plus","gpa":3.2,"courses":[]},
{"npm":"1406557535","name":"Luthfi Abdurrahim401","gpa":4.01,"courses":
[{"id_course":"CSC123","name":"PSP","credits":4,"students":null}]},
{"npm":"140655753599","name":"Luthfi Abdurrahim3","gpa":3.9901,"courses":
[{"id_course":"CSC123","name":"PSP","credits":4,"students":null}]},
{"npm":"1406557536","name":"Luthfi Abdurrahima","gpa":3.999,"courses":[]}]
```

Pada viewall student, hanya menambahkan method pada StudentRestController.java dengan code seperti ini:

```
@RequestMapping("/student/viewall")
public List<StudentModel> viewAll() {
    List<StudentModel> student =
        studentService.selectAllStudents();
    return student;
}
```

Dengan memanggil code seperti pada tutorial 5 view all, lalu mengubah return menjadi object, sehingga menghasilkan web service yang diinginkan.

Hanya menerapkan query yang ada pada mapper, lalu pada controller ini memanggilnya dengan studentservice.selectAllStudents yang di assign kepada variable bertipe List studentModel dengan nama object: student.

Latihan 2: Buatlah service untuk class Course. Buatlah controller baru yang terdapat service untuk melihat suatu course dengan masukan ID Course (view by ID) dan service untuk melihat semua course (view all).

```
//latihan 2
@RequestMapping("/course/view/{id_course}")
public CourseModel view(
    @PathVariable(value = "id_course")
    String id_course) {
    CourseModel course =
        studentService.selectCourse(id_course);
    return course;
}

//latihan 2
@RequestMapping("/course/viewall")
public List<CourseModel> viewAll() {
    List<CourseModel> course =
        studentService.selectAllCourses();
    return course;
}
```

Pada latihan 2 ini, penulis membuat controller baru yang bernama CourseRestController.java untuk menerapkan 2 method di atas. Method pertama bernama view() yaitu memanggil course berdasarkan id. Di dalam method tersebut, menerapkan pemanggilan courseModel dan melakukan selectCourse berdasarkan id yang ada pada studentservice dan mapper nya. Dengan mengembalikan / return object course, maka didapat json / web service yang diinginkan.

Method kedua yaitu viewAll, di dalamnya menerapkan pemanggilan coursemodel dengan list, lalu pemanggilan selectAllCourses pada studentservice yang akan menampilkan seluruh isi course yang ada pada basis data. Dengan return object course, maka akan didapatkan hasil json / web service yang diinginkan, yaitu menampilkan semua isi course yang ada pada basis data.

Berikut ini outputnya view course by id:



```

1 // 20171104145912
2 // http://localhost:8080/rest/course/view/CSC123
3
4 {
5   "id_course": "CSC123",
6   "name": "PSP",
7   "credits": 4,
8   "students": [
9     {
10      "npm": "124",
11      "name": "Chanek Jr.",
12      "gpa": 3.0,
13      "courses": null
14    },
15    {
16      "npm": "1406557535",
17      "name": "Luthfi Abdurrahim401",
18      "gpa": 4.01,
19      "courses": null
20    },
21    {
22      "npm": "140655753599",
23      "name": "Luthfi Abdurrahim3",
24      "gpa": 3.9901,
25      "courses": null
26    }
27  ]
28 }

```

Berikut ini output dari viewall course:

```
localhost:8080/rest/cours X
localhost:8080/rest/course/viewall

1 // 20171104150255
2 // http://localhost:8080/rest/course/viewall
3
4 [
5   {
6     "id_course": "CSC123",
7     "name": "PSP",
8     "credits": 4,
9     "students": [
10    {
11      "npm": "124",
12      "name": "Chanek Jr.",
13      "gpa": 3.0,
14      "courses": null
15    },
16    {
17      "npm": "1406557535",
18      "name": "Luthfi Abdurrahim401",
19      "gpa": 4.01,
20      "courses": null
21    },
22    {
23      "npm": "140655753599",
24      "name": "Luthfi Abdurrahim3",
25      "gpa": 3.9901,
26      "courses": null
27    }
28  ],
29 },
30 {
31   "id_course": "CSC124",
32   "name": "SDA",
33   "credits": 3,
34   "students": [
35     {
36       "npm": "124",
37       "name": "Chanek Jr.",
38       "gpa": 3.0,
39       "courses": null
40     }
41   ]
42 }
```

```
localhost:8080/rest/cours X
localhost:8080/rest/course/viewall

[{"id_course":"CSC123","name":"PSP","credits":4,"students":[{"npm":"124","name":"Chanek Jr.,"gpa":3.0,"courses":null}, {"npm":"1406557535","name":"Luthfi Abdurrahim401","gpa":4.01,"courses":null}, {"npm":"140655753599","name":"Luthfi Abdurrahim3","gpa":3.9901,"courses":null}]}], [{"id_course":"CSC124","name":"SDA","credits":3,"students":[{"npm":"124","name":"Chanek Jr.,"gpa":3.0,"courses":null}]}], [{"id_course":"CSC125","name":"DDP 1","credits":4,"students":[]}, {"id_course":"CSC126","name":"MPKT","credits":6,"students":[{"npm":"123","name":"Chanek","gpa":4.0,"courses":null}]}]]
```

Tutorial07Consumer.java berhasil berjalan:

The image shows two side-by-side web browser windows. The left window, titled 'View Student by NPM', displays the output of a REST client at the URL `localhost:9090/student/view/...`. It shows the following data:

- NPM = 123**
- Name = Chanek**
- GPA = 4.0**

The right window, titled 'localhost:8080/rest/stud...', shows the raw JSON response from a REST client at the URL `http://localhost:8080/rest/student/view/123`. The response is a JSON object with the following structure:

```
1 // 20171104162204
2 //
3 http://localhost:8080/rest/student/view/123
4 {
5   "npm": "123",
6   "name": "Chanek",
7   "gpa": 4.0,
8   "courses": [
9     {
10      "id_course": "CSC126",
11      "name": "MPKT",
12      "credits": 6,
13      "students": null
14    }
15  ]
16 }
```

Latihan 3: Implementasikan service consumer untuk view all Students dengan melengkapi method `selectAllStudents` yang ada di kelas `StudentServiceRest`.

Penjelasan:

Pertama, pada model dan method `viewall` `studentController.java` tidak ada perubahan, code tetap sesuai dengan tutorial6.

Lalu, pada class `StudentDAOImpl.java` yang ada pada package `service`, ditambahkan method ini:

```
@SuppressWarnings("unchecked")
@Override
public List<StudentModel> selectAllStudents() {
    List<StudentModel> students =
        restTemplate.getForObject
            ("http://localhost:8080/rest/student/viewall",
            List.class);
    return students;
}
```

Method tersebut akan mengembalikan object student berupa tipe `list studentmodel` yang didapatkan dari web service `/rest/student/viewall` (tipe json)

Lalu, pada class `StudentServiceRest.java` menerapkan method berikut:

```
@Override
public List<StudentModel> selectAllStudents ()
{
    log.info ("REST - select all students");
    return studentDAO.selectAllStudents();
}
```

Dengan begitu akan mengembalikan `selectAllStudents` berdasarkan sumber dari rest api yang sudah dibuat.

```
@Slf4j
@Service
@Primary
public class StudentServiceRest {
    @Autowired
```

Dengan menambahkan anotasi `@primary` pada kelas `studentservicerest`, meskipun ada `studentmapper` dan `studentservicedatabase.java` maka, yang digunakan akan `studentservicerest.java`

Screenshot Output:

Kiri: <http://localhost:9090/student/viewall/>

Kanan: <http://localhost:8080/rest/student/viewall>

**Daftar Mahasiswa**

Tutorial 6

## All Students

Show 10 entries

Search:

No	NPM	Name	GPA	Cum laude	Delete
1	123	Chanek	4.0	Cum Laude!	Delete
2	124	Chanek Jr.	3.0	Sangat Memuaskan	Delete
3	125	chanek Jr plus	3.2	Sangat Memuaskan	Delete
4	1406557535	Luthfi Abdurrahim401	4.01	Cum Laude!	Delete
5	140655753599	Luthfi Abdurrahim3	3.9901	Cum Laude!	Delete
6	1406557536	Luthfi Abdurrahima	3.999	Cum Laude!	Delete

Showing 1 to 6 of 6 entries

Previous **1** Next

[Delete Data](#)  
[Update Data](#)

**No. false**  
**NPM - 123**

```
1 // 20171104162737
2 // http://localhost:8080/rest/student/viewall
3
4 [
5   {
6     "npm": "123",
7     "name": "Chanek",
8     "gpa": 4.0,
9     "courses": [
10      {
11        "id_course": "CSC126",
12        "name": "MPKT",
13        "credits": 6,
14        "students": null
15      }
16    ]
17   },
18   {
19     "npm": "124",
20     "name": "Chanek Jr.",
21     "gpa": 3.0,
22     "courses": [
23      {
24        "id_course": "CSC123",
25        "name": "PSP",
26        "credits": 4,
27        "students": null
28      }
29    ],
30     {
31       "id_course": "CSC124",
32       "name": "SDA",
33       "credits": 3,
34       "students": null
35     }
36   },
37   {
38     "npm": "125",
39     "name": "chanek Jr plus"
```



Latihan 4: Implementasikan service consumer untuk class CourseModel dengan membuat class-class DAO dan service baru.

Penjelasan:

Penulis hanya perlu menambahkan CourseDAOImpl.java pada package service dan interface CourseDAO pada package dao. Dengan menerapkan kedua hal tersebut, dapat diakses data-nya melalui controller dan view layer.

Berikut ini source code dari CourseDAOImpl.java

```
@Service
public class CourseDAOImpl implements CourseDAO{
    @Autowired
    private RestTemplate restTemplate;

    @Override
    public CourseModel selectCourse(String id_course) {
        CourseModel course = restTemplate.getForObject("http://localhost:8080/rest/course/view/"+id_course,
            CourseModel.class);
        return course;
    }

    @SuppressWarnings("unchecked")
    @Override
    public List<CourseModel> selectAllCourses() {
        List<CourseModel> courses = restTemplate.getForObject("http://localhost:8080/rest/course/viewall",
            List.class);
        return courses;
    }
}
```

Dan Berikut ini source code dari CourseDAO.java

```
public interface CourseDAO {
    CourseModel selectCourse (String id_course);
    List<CourseModel> selectAllCourses();
}
```

Screenshot output:

1. selectAllCourse

Atas: <http://localhost:9090/course/viewall/>

Bawah : <http://localhost:8080/rest/course/viewall>

The screenshot shows a web browser window with the title 'Daftar Mata Kuliah'. The address bar shows 'localhost:9090/course/view...'. The page has a purple header with 'Tutorial 6' and a hamburger menu icon. Below the header, the title 'All Mata Kuliah' is displayed. There is a 'Show 10 entries' dropdown and a 'Search:' input field. A table with 4 columns (No, ID Course, Name, Credits) lists 4 courses. Below the table, it says 'Showing 1 to 4 of 4 entries' and has 'Previous', '1', and 'Next' navigation buttons.

No	ID Course	Name	Credits
1	CSC123	PSP	4
2	CSC124	SDA	3
3	CSC125	DDP 1	4
4	CSC126	MPKT	6

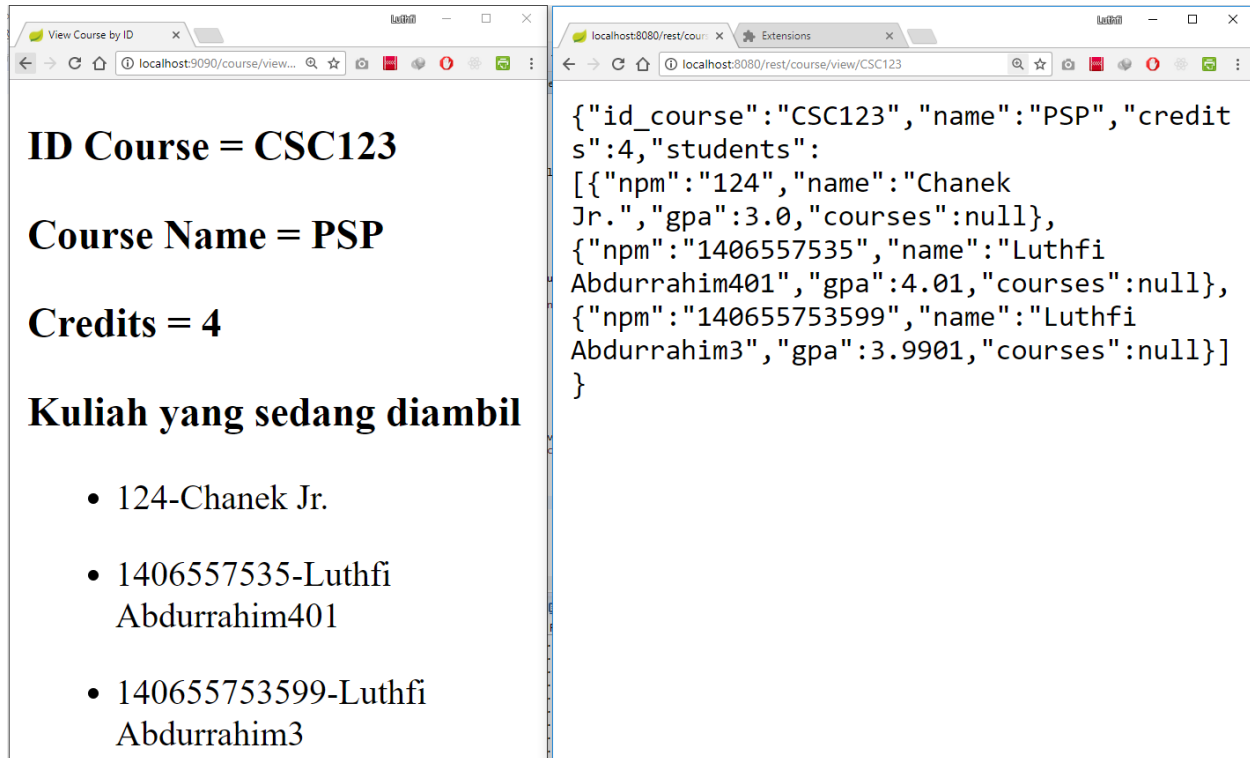
The screenshot shows a web browser window with the title 'localhost:8080/rest/cour...'. The address bar shows 'localhost:8080/rest/course/viewall'. The page displays a JSON response from a REST API. The response is a list of course objects, each containing 'id\_course', 'name', 'credits', and 'students'. The 'students' field is an array of student objects with 'npm', 'name', and 'gpa' fields.

```
[{"id_course": "CSC123", "name": "PSP", "credits": 4, "students": [{"npm": "124", "name": "Chanek Jr.", "gpa": 3.0, "courses": null}, {"npm": "1406557535", "name": "Luthfi Abdurrahim401", "gpa": 4.01, "courses": null}, {"npm": "140655753599", "name": "Luthfi Abdurrahim3", "gpa": 3.9901, "courses": null}]}, {"id_course": "CSC124", "name": "SDA", "credits": 3, "students": [{"npm": "124", "name": "Chanek Jr.", "gpa": 3.0, "courses": null}]}, {"id_course": "CSC125", "name": "DDP 1", "credits": 4, "students": []}, {"id_course": "CSC126", "name": "MPKT", "credits": 6, "students": [{"npm": "123", "name": "Chanek", "gpa": 4.0, "courses": null}]}]
```

2. Select Course by ID Course

Kiri: <http://localhost:9090/course/view/CSC123>

Kanan: <http://localhost:8080/rest/course/view/CSC123>



**View Course by ID**

**ID Course = CSC123**

**Course Name = PSP**

**Credits = 4**

**Kuliah yang sedang diambil**

- 124-Chanek Jr.
- 1406557535-Luthfi Abdurrahim401
- 140655753599-Luthfi Abdurrahim3

```
{
  "id_course": "CSC123",
  "name": "PSP",
  "credits": 4,
  "students": [
    {
      "npm": "124",
      "name": "Chanek Jr.",
      "gpa": 3.0,
      "courses": null
    },
    {
      "npm": "1406557535",
      "name": "Luthfi Abdurrahim401",
      "gpa": 4.01,
      "courses": null
    },
    {
      "npm": "140655753599",
      "name": "Luthfi Abdurrahim3",
      "gpa": 3.9901,
      "courses": null
    }
  ]
}
```

## Write-up

Hal yang penulis pelajari pada tutorial 7 kali ini yaitu:

Penulis mengetahui cara membuat webservice dengan adanya format json dan client yang mengambil objek json tersebut, menggunakan java springboot.

Lalu, hal terpentingnya yaitu, pada client side yang ingin mengambil rest service tersebut, tidak bisa autoconfiguration restTemplate jika menggunakan springboot  $\geq 1.4$ , sehingga penulis selalu mendapatkan error. Hingga menemukan solusinya yaitu dengan menerapkan method :

```
@Bean
public RestTemplate restTemplate() {
    return new RestTemplate();
}
```

Pada class Tutorial7ConsumerApplication.java

Sehingga, tidak akan terjadi error berikut:

Consider defining a bean of type 'org.springframework.web.client.RestTemplate' in your configuration.

Penulis menjadi mengetahui mengenai penerapan rest service yaitu pada client harus menambahkan service baru berupa StudentServiceRest.java, StudentDAOImpl.java dan StudentDAO.java serta penerapan @primary pada rest service. Sehingga dapat digunakan pada client untuk mengakses rest service tersebut.

Dengan adanya rest service ini, penerapan back end dan front end dapat semakin terpisah layernya, sehingga tidak terlalu dependensi, jika back end berubah maka front end tidak perlu diubah.