

# Tutorial 07

## Membuat Service Producer

Latihan 1: Mengembalikan data seluruh *students*

Method :

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

Penjelasan :

Mengambil data dari seluruh *student* yang terdapat pada *database* dan dikembalikan dalam bentuk list.

Screenshot :

The screenshot displays a JSON viewer with two panels. The left panel shows the full JSON structure, and the right panel shows a detailed view of a specific student's data.

**Left Panel (JSON View):**

- 0: {  
 npm: "1059",  
 name: "ananda",  
 gpa: 3.94,  
 courses: []  
}
- 1: {  
 npm: "1111",  
 name: "chanek aryanto",  
 gpa: 3.48,  
 courses: []  
}
- 2: {  
 npm: "123",  
 name: "anin",  
 gpa: 3.93,  
 courses: [  
 0: {  
 id\_course: "CSC126",  
 name: "MPKT",  
 credits: 6,  
 students: []  
 }  
 ]  
}
- 3: {  
 npm: "1234",  
 name: "chanek",  
 gpa: 3.45,  
 courses: []  
}

**Right Panel (Detailed View of Student 4):**

- 4: {  
 npm: "124",  
 name: "cherry",  
 gpa: 4,  
 courses: [  
 0: {  
 id\_course: "CSC123",  
 name: "PSP",  
 credits: 4,  
 students: []  
 },  
 1: {  
 id\_course: "CSC124",  
 name: "SDA",  
 credits: 3,  
 students: []  
 }  
 ]  
}
- 5: {  
 npm: "126",  
 name: "faas",  
 gpa: 3.74,  
 courses: []  
}
- 6: {  
 npm: "1289",  
 name: "fanas",  
 gpa: 3.79,  
 courses: []  
}

## Latihan 2: Mengembalikan data *course* berdasarkan ID

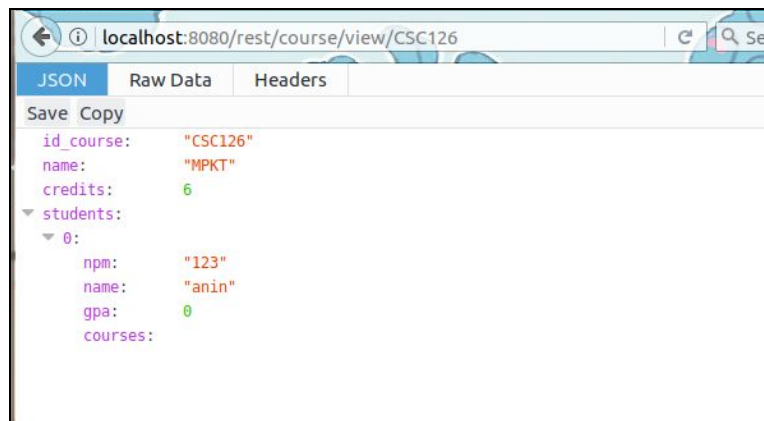
### Method :

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

### Penjelasan :

Mengambil data dari course dengan ID tertentu.

### Screenshot :



## Membuat Service Consumer

Latihan 3: Menampilkan data seluruh siswa

Method :

### I. **StudentServiceRest**

@Override

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

### II. **StudentDAOImpl**


@Override

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

Penjelasan :

- I. Memanggil *method* selectAllStudents dari studentDAO
- II. Mengambil list dari seluruh students dari hasil kembalian rest student viewall di producer.

Screenshot :



No	NPM	Nama	GPA	Cum laude	Delete	Update
1	1059	ananda	3.94	Cum laude	Delete	Update
2	1111	chanek aryanto	3.48	Cum laude	Delete	Update
3	123	anin	3.93	Cum laude	Delete	Update
4	1234	chanek	3.45	Cum laude	Delete	Update
5	124	cherry	4.0	Cum laude	Delete	Update
6	126	faas	3.74	Cum laude	Delete	Update
7	1289	fanass	3.79	Cum laude	Delete	Update

#### Latihan 4: Menampilkan data course berdasarkan ID

Method :

##### II. **StudentServiceRest**

@Override

```
public CourseModel selectCourse(String id) {  
    log.info("REST - select course with id {}", id);  
    return studentDAO.selectCourse(id);  
}
```

##### II. **StudentDAOImpl**

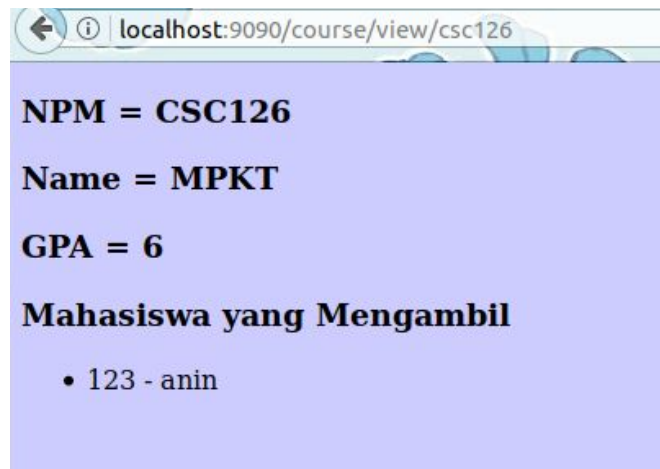
@Override

```
public CourseModel selectCourse(String id) {  
    CourseModel kurs =  
restTemplate.getForObject("http://localhost:8080/rest/course/view/"+i  
d, CourseModel.class);  
    return kurs;  
}
```

Penjelasan :

- I. Memanggil *method* selectCourse dari studentDAO
- II. Mengambil data dari rest course berdasarkan id sebagai object untuk ditampilkan.

Screenshot :



## Hal yang dipelajari

Tutorial untuk minggu ini hal yang saya pelajari ialah:

- Saya jadi tahu cara menjalankan dua project sekaligus dalam satu waktu
- Mengetahui cara mengatur port server dari project yang akan dijalankan
- Menggunakan rest controller
- Mempelajari anotasi @primary
- Menghubungkan dua project yang terpisah dengan rest controller