Kevin Noel Anugrah
1506689370


# Membuat Web Service Menggunakan Spring Boot Framework

Pada tutorial kali ini diajarkan bagaimana membuat service producer yang menghasilkan REST object dan service consumer yang dapat menerima dan menampilkannya.

## I. Service Producer

Controller untuk REST web service menggunakan notasi @RestController dan method-methodnya mengembalikan object yang jika dikirim dengan format JSON.

**Jawaban Latihan 1:** Untuk mengembalikan semua student, buat agar method mengembalikan list StudentModel dari method yang sudah terimplementasi di tutorial 5.

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

**Jawaban Latihan 2:** Untuk method view course by id maupun viewall sama seperti sebelumnya, hanya saja ganti mappingnya dan ubah menjadi CourseModel.

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

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

# II. Service Consumer

Untuk consumer, portnya diganti menjadi 9090 agar berbeda dengan producer. Lalu dibuat interface DAO dan implementasinya yang berisi method untuk meminta objek dari service producer. Lalu dibuat implementasi StudentService baru yang dapat mengambil data dari DAO tersebut.

**Jawaban Latihan 3:** Implementasi service consumer untuk view all students sama tahapnya dengan implentasi view student hanya saja diganti dengan list, yang pertama adalah menambahkan implementasi pada StudentDAOImpl:

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

Lalu, implementasikan method pada StudentServiceRest yang mengambil objek dari DAO.

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

**Jawaban Latihan 4:** Untuk implementasi view course dan view all course sama saja seperti sebelumnya tetapi ubah StudentModel dengan CourseModel. Untuk view course dan view all course, tambahkan pada StudentDAOImpl:

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


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

Lalu, pada StudentServiceRest:

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

@Override
public List<CourseModel> selectAllCourses() {
        log.info("REST - select all courses");
        return studentDAO.selectAllCourses();
}
```