

Tutorial 7 Writeup

Arsitektur dan Pemrograman Aplikasi Perusahaan

Nama : Komang Adelia Nala Ratih

NPM : 1406557541

Kelas : B

Buat sebuah file write-up. Jelaskan apa saja hal yang Anda pelajari dari tutorial ini juga penjelasan Anda terhadap setiap latihan yang ada di tutorial!

Saya mempelajari bagaimana mengimplementasikan *web service* pada Spring. Penggunaan *web service* pada Spring dibagi menjadi dua layer, yaitu *consumer* dan *producer*. Pemisahan layer tersebut ditujukan untuk memisahkan *backend* dan *frontend*, dimana *consumer* merupakan bagian *frontend*, sedangkan *producer* bagian *backend*. Dengan adanya *web service*, data dapat diakses oleh orang lain tanpa perlu mengakses *database*.

Membuat Service Producer

Latihan 1.

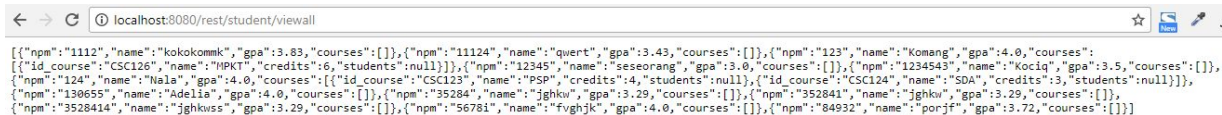
Buatlah service untuk mengembalikan seluruh student yang ada di basis data. Service ini mirip seperti method `viewAll` di Web Controller. Service tersebut di-mapping ke `"/rest/student/viewall"`.

Cantumkan dan Jelaskan method Anda.

Method

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

Tampilan



```
localhost:8080/rest/student/viewall
[{"npm": "1112", "name": "kokokommk", "gpa": 3.83, "courses": []}, {"npm": "11124", "name": "quert", "gpa": 3.43, "courses": []}, {"npm": "123", "name": "Komang", "gpa": 4.0, "courses": [{"id_course": "CSC126", "name": "MPKT", "credits": 6, "students": null}, {"npm": "12345", "name": "seseorang", "gpa": 3.0, "courses": []}, {"npm": "1234543", "name": "Kociq", "gpa": 3.5, "courses": []}, {"npm": "124", "name": "Nala", "gpa": 4.0, "courses": [{"id_course": "CSC123", "name": "PSP", "credits": 4, "students": null}, {"id_course": "CSC124", "name": "SDA", "credits": 3, "students": null}], {"npm": "130655", "name": "Adelia", "gpa": 4.0, "courses": []}, {"npm": "35284", "name": "jghkw", "gpa": 3.29, "courses": []}, {"npm": "352841", "name": "jghkw", "gpa": 3.29, "courses": []}, {"npm": "3528414", "name": "jghkwss", "gpa": 3.29, "courses": []}, {"npm": "56781", "name": "fvghjk", "gpa": 4.0, "courses": []}, {"npm": "84932", "name": "porjff", "gpa": 3.72, "courses": []}]
```

Penjelasan

Method **viewAll** berfungsi untuk mengembalikan daftar dari objek student pada database dengan memanggil method `selectAllStudents` yang ada pada `StudentService`. Method ini mengembalikan student dalam bentuk list dan format JSON.

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).

Cantumkan dan Jelaskan method Anda.

Method

view

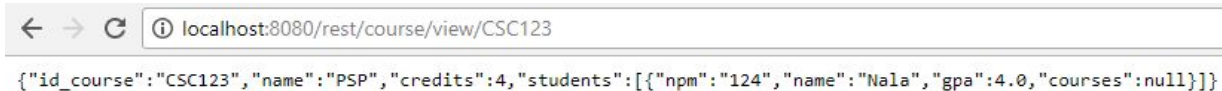
```
@RequestMapping("/course/view/{id_course}")
public CourseModel view(@PathVariable(value = "id_course") String id_course) {
    CourseModel course = courseService.selectCourse(id_course);
    return course;
}
```

viewAll

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

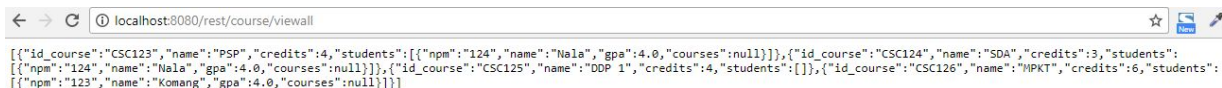
Tampilan

view



```
{ "id_course": "CSC123", "name": "PSP", "credits": 4, "students": [ { "npm": "124", "name": "Nala", "gpa": 4.0, "courses": null } ] }
```

viewAll



```
[ { "id_course": "CSC123", "name": "PSP", "credits": 4, "students": [ { "npm": "124", "name": "Nala", "gpa": 4.0, "courses": null } ] }, { "id_course": "CSC124", "name": "SDA", "credits": 3, "students": [ { "npm": "124", "name": "Nala", "gpa": 4.0, "courses": null } ] }, { "id_course": "CSC125", "name": "DDP 1", "credits": 4, "students": [ ] }, { "id_course": "CSC126", "name": "MPKT", "credits": 6, "students": [ { "npm": "123", "name": "Komang", "gpa": 4.0, "courses": null } ] } ]
```

Penjelasan

view

Method view berfungsi untuk mengembalikan objek course dengan id tertentu dari database dengan memanggil method selectCourse yang ada pada CourseService. Method ini mengembalikan objek course dan dalam format JSON.

viewAll

Method viewAll berfungsi untuk mengembalikan daftar dari objek course pada database dengan memanggil method selectAllCourses yang ada pada CourseService. Method ini mengembalikan course dalam bentuk list dan format JSON.

Membuat Service Consumer

Latihan 3.

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

Cantumkan dan jelaskan method Anda.

Method selectAllStudents pada interface StudentDAO

```
List<StudentModel> selectAllStudents();
```

Menginisiasi method selectAllStudents untuk kemudian diimplementasi pada StudentDAOImpl.

Method selectAllStudents pada kelas StudentDAOImpl

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

Method selectAllStudents menunjukkan RestTemplate untuk menerima parameter yang akan dikirimkan oleh url. Rest API tersebut akan diubah menjadi List dari Student sebagai objek yang akan dikembalikan dari method selectAllStudents.

Method selectAllStudents pada StudentServiceRest

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

Method selectAllStudents mengembalikan objek berupa List of Student yang didapatkan dari method selectAllStudents pada StudentDAO.

Latihan 4.

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

Cantumkan dan jelaskan method Anda.

Method selectCourse & selectAllCourses pada interface CourseDAO

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

Menginisiasi method selectCourse dan selectAllCourses untuk kemudian diimplementasi pada CourseDAOImpl.

Method selectCourse & selectAllCourses pada kelas CourseDAOImpl

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

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

Mengimplementasikan method selectCourse dan selectAllCourses pada kelas CourseDAOImpl.

Method **selectCourse** akan meng-*consume* objek REST dengan menerima parameter berupa url *producer web service*. Method ini akan menerima tipe objek berupa CourseModel.

Method **selectAllCourses** akan meng-*consume* objek REST dengan menerima parameter berupa url *producer web service*. Method ini akan menerima tipe objek berupa List of Course.

Method selectCourse dan selectAllCourses pada CourseServiceRest

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

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

Membuat CourseServiceRest dengan mengimplementais CourseService untuk mengambil data dari *web service*.

Method **selectCourse** mengembalikan objek berupa CourseModel dari method selectCourse pada CourseDAO.

Method **selectAllCourses** mengembalikan objek berupa List of CourseModel dari method selectAllCourses pada CourseDAO.