

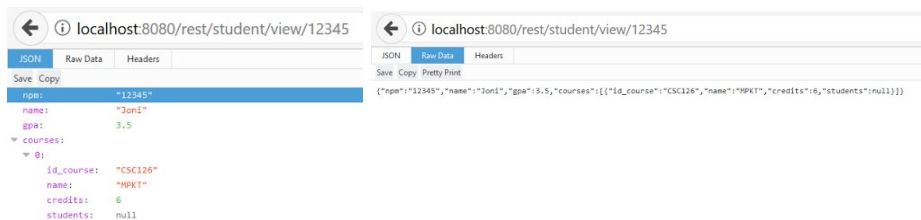
Write Up Tutorial 7

Analisis dan Pemrograman Aplikasi Perusahaan

Nama : Lintang Matahari Hasani NPM : 1506689231

I. Membuat Service Producer

Tampilan JSON ketika mengakses path <http://localhost:8080/rest/student/view/12345>



Latihan

1. **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”.

Tampilan ketika memanggil /rest/student/viewall



Tahap Pengerjaan :

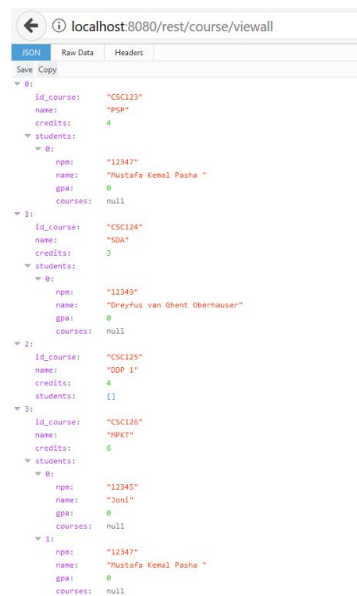
1. Membuat method viewall() yang mengembalikan List<StudentModel> pada Class StudentRestController

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

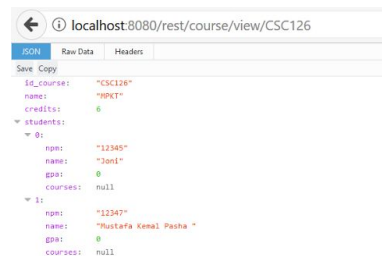
Method ini memanggil method selectAllStudents() yang mengembalikan objek List yang berisi semua student.

2. **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)

Tampilan ketika mengakses /rest/course/viewall



Tampilan ketika mengakses /rest/course/view/CSC126



Tahap Pengerjaan :

1. Membuat Interface Course Service yang memuat method selectCourse dengan parameter id course dan method selectAllCourses.

```
1 package com.example.service;
2
3 import java.util.List;
4
5
6
7
8 public interface CourseService
9 {
10     CourseModel selectCourse (String id);
11
12     List<CourseModel> selectAllCourses ();
13
14 }
```

2. Membuat Class CourseServiceDatabase yang mengimplementasikan Interface CourseService

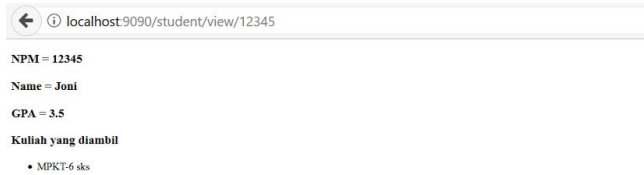
```
1 package com.example.service;
2
3 import java.util.List;
4
5
6
7
8
9
10
11
12
13
14
15 @Slf4j
16 @Service
17 public class CourseServiceDatabase implements CourseService
18 {
19     @Autowired
20     private StudentMapper student;
21
22
23     @Override
24     public CourseModel selectCourse (String idCourse)
25     {
26         log.info ("select course with ID {}", idCourse);
27         return student.selectCourse (idCourse);
28     }
29
30
31     @Override
32     public List<CourseModel> selectAllCourses ()
33     {
34         log.info ("select all courses");
35         return student.selectAllCourses();
36     }
37 }
```

Interface ini mengimplementasikan method selectCourse() yang menerima masukan id course yang dipilih dan method selectAllCourses() yang mengembalikan List berisi semua student.

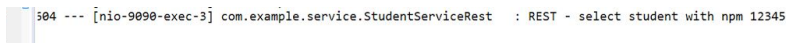
3. Membuat method selectCourse yang menerima masukan id course dan method selectAllCourses pada Class StudentRestController

II. Membuat Service Consumer

Tampilan ketika mengakses <http://localhost:9090/student/view/12345>



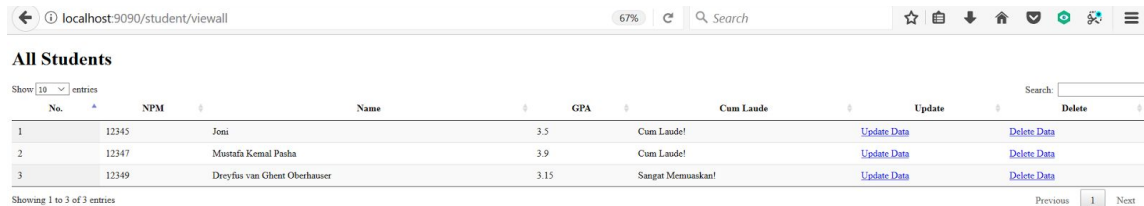
Tampilan pada console Tutorial07Consumer



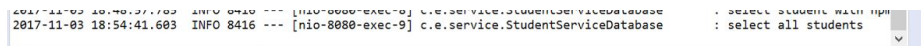
Latihan

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

Tampilan ketika mengakses <http://localhost:9090/student/viewall>



Tampilan pada console Tutorial07Consumer



Langkah pengerjaan :

- Mengimplementasikan method `selectAllStudents()` pada Class `StudentDAOImpl.java`

```
28 @Override
29 public List <StudentModel> selectAllStudents ()
30 {
31     List<StudentModel> students =
32         restTemplate.getForObject (
33             "http://localhost:8080/rest/student/viewall",
34             List.class );
35     return students;
36 }
37 }
38 }
```

Method ini memanggil method `getForObject()` yang berisi parameter URL producer (<http://localhost:8080/rest/student/viewall>) dan objek List. Object List yang diperoleh memuat semua object `StudentModel`.

2. Memanggil method selectAllStudents() di Class StudentServiceRest

```
30 @Override
31 public List < StudentModel > selectAllStudents ()
32 {
33     Log.info ( "REST - select all students" );
34     return studentDAO.selectAllStudents();
35 }
36
37
```

Method selectAllStudents() pada class StudentServiceRest mengembalikan object List<StudentModel> yang diperoleh dari memanggil method selectAllStudents() dari objek StudentDAO.

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

Tampilan ketika mengakses <http://localhost:9090/course/view/CSC125>



localhost:9090/course/view/CSC125

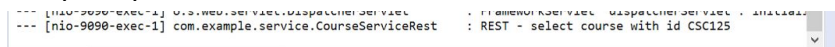
ID = CSC125

Nama = DDP 1

SKS = 4

Mahasiswa yang mengambil

Tampilan pada console Tutorial07Consumer



```
--- [nio-9090-exec-1] o.s.web.servlet.DispatcherServlet : REST - select course with id CSC125
```

Langkah Pengerjaan:

1. Membuat Interface CourseService

```
1 package com.example.service;
2
3 import java.util.List;
4
5 public interface CourseService
6 {
7     CourseModel selectCourse (String id);
8     List<CourseModel> selectAllCourses ();
9 }
10
11
12
13
14
```

Pada Interface ini terdapat method selectCourse() yang menerima masukan id course yang dipilih dan method selectAllCourses() yang mengembalikan objek List berisi semua course

2. Membuat Class CourseServiceRest

```
1 package com.example.service;
2
3 import java.util.List;
4
5 @Primary
6 @Slf4j
7 @Service
8 public class CourseServiceRest implements CourseService
9 {
10     @Autowired
11     private CourseDAO courseDAO;
12
13     @Override
14     public CourseModel selectCourse (String id)
15     {
16         Log.info ( "REST - select course with id {}" , id );
17         return courseDAO.selectCourse(id);
18     }
19
20     @Override
21     public List < CourseModel > selectAllCourses ()
22     {
23         Log.info ( "REST - select all courses" );
24         return courseDAO.selectAllCourses();
25     }
26 }
27
```

Class ini mengimplementasikan Interface CourseService

3. Membuat Interface CourseDAO

```
1 package com.example.dao;
2
3 import java.util.List;
4
5
6
7 public interface CourseDAO {
8     CourseModel selectCourse (String id);
9     List <CourseModel> selectAllCourses();
10 }
```

4. Membuat Class CourseDAOImpl

```
1 package com.example.dao;
2
3
4 import java.util.List;
5
6
7
8
9
10
11
12 @Service
13 public class CourseDAOImpl implements CourseDAO
14 {
15     @Autowired
16     private RestTemplate restTemplate;
17
18     @Override
19     public CourseModel selectCourse(String id)
20     {
21         CourseModel course =
22             restTemplate.getForObject (
23                 "http://localhost:8080/rest/course/view/" + id,
24                 CourseModel.class );
25         return course;
26     }
27
28     @Override
29     public List <CourseModel> selectAllCourses ()
30     {
31         List<CourseModel> courses =
32             restTemplate.getForObject (
33                 "http://localhost:8080/rest/course/viewall",
34                 List.class );
35         return courses;
36     }
37 }
```

Method selectCourse memanggil method getForObject() yang berisi parameter URL producer (<http://localhost:8080/rest/course/view/{id}>) dan objek CourseModel. Method ini menerima parameter id course yang dipilih.

Method selectAllCourses memanggil method getForObject() yang berisi parameter URL producer (<http://localhost:8080/rest/course/viewall>) dan objek List yang berisi semua CourseModel.

5. Membuat Class CourseController

III. Lessons Learned

Pada tutorial ini, saya belajar mengenai implementasi web service menggunakan Framework Spring Boot. Pada tutorial ini, saya mempelajari cara membuat aplikasi Service Producer. Aplikasi service producer adalah aplikasi yang menyediakan data untuk aplikasi service consumer. Data yang disediakan oleh aplikasi producer memiliki format JSON. Pada tutorial ini, saya juga belajar cara membuat aplikasi Service Consumer. Aplikasi Service Consumer adalah aplikasi yang menerima dan menggunakan data yang disediakan aplikasi Service Producer. Pada tutorial ini, Service Consumer menjadi frontend sistem.