

TUTORIAL 07

Membuat Web Service Menggunakan Spring Boot Framework

Membuat *Service producer*

1. Pada *package* **com.example.rest**, buat *class* **StudentRestController.java**

```
package com.example.rest;

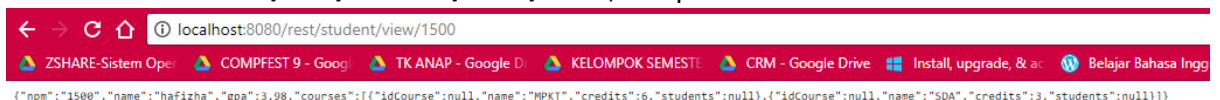
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.example.model.StudentModel;
import com.example.service.StudentService;

@RestController
@RequestMapping("/rest")
public class StudentRestController {
    @Autowired
    StudentService studentService;

    @RequestMapping("/student/view/{npm}")
    public StudentModel view (@PathVariable(value = "npm") String npm) {
        StudentModel student = studentService.selectStudent(npm);
        return student;
    }
}
```

2. Buka **localhost:8080/rest/student/view/1500**, tampilan dalam format JSON.



```
{
  "npm": "1500",
  "name": "hafizha",
  "gpa": 3.98,
  "courses": [
    {
      "idCourse": null,
      "name": "MPKT",
      "credits": 6,
      "students": null
    },
    {
      "idCourse": null,
      "name": "SDA",
      "credits": 3,
      "students": null
    }
  ]
}
```

LATIHAN 1

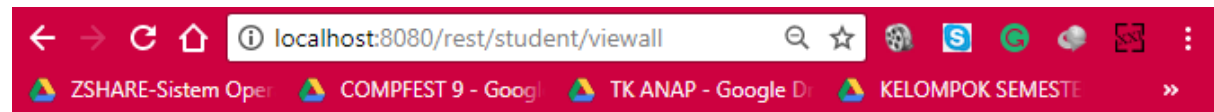
Buatlah *service* untuk mengembalikan seluruh student yang ada di basis data. *Service* ini mirip seperti method `viewAll` di Web Controller. *Service* tersebut di-mapping ke **`"/rest/student/viewall"`**.

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

Method **`viewall`** di atas akan mengembalikan *list* yang berisi seluruh student di *database* dengan memanggil *method* `selectAllStudents()` pada `studentService`. *List* yang dikembalikan

berupa daftar student beserta course yang diambil. Pemanggilannya adalah “rest/student/viewall”

Keluaran pemanggilan **localhost:8080/rest/student/viewall** adalah sebagai berikut.



```
[{"npm": "1500", "name": "hafizha", "gpa": 3.98, "courses": [{"idCourse": null, "name": "MPKT", "credits": 6, "students": null}, {"idCourse": null, "name": "SDA", "credits": 3, "students": null}]}, {"npm": "1501", "name": "chanek", "gpa": 3.45, "courses": [{"idCourse": null, "name": "PSP", "credits": 4, "students": null}]}, {"npm": "1502", "name": "hera", "gpa": 3.4, "courses": [{"idCourse": null, "name": "SDA", "credits": 3, "students": null}]}, {"npm": "1503", "name": "Ano", "gpa": 3.73, "courses": []}]
```

```
{
  "npm": "1500",
  "name": "hafizha",
  "gpa": 3.98,
  "courses": [
    {
      "idCourse": null,
      "name": "MPKT",
      "credits": 6,
      "students": null
    },
    {
      "idCourse": null,
      "name": "SDA",
      "credits": 3,
      "students": null
    }
  ]
},
{
  "npm": "1501",
  "name": "chanek",
  "gpa": 3.45,
  "courses": [
    {
      "idCourse": null,
      "name": "PSP",
      "credits": 4,
      "students": null
    }
  ]
},
{
  "npm": "1502",
  "name": "hera",
  "gpa": 3.4,
  "courses": [
    {
      "idCourse": null,
      "name": "SDA",
      "credits": 3,
      "students": null
    }
  ]
},
{
  "npm": "1503",
  "name": "Ano",
  "gpa": 3.73,
  "courses": []
}
]
```

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*).

Buat CourseRestController yang berisi *mapping* view dan viewall untuk course. Isi *class* sebagai berikut.

```
package com.example.rest;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.example.model.CourseModel;
import com.example.service.CourseService;

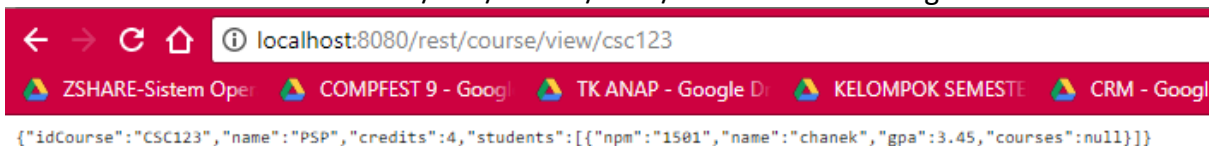
@RestController
@RequestMapping("/rest")
public class CourseRestController {
    @Autowired
    CourseService courseService;

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

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

Method view akan mengambil nilai id_course yang dimasukkan pada PathVariable dan mengembalikan nilai *attribute* dari course tersebut beserta student yang mengambil course tersebut. *Method* viewall akan mengembalikan semua course yang ada beserta student yang bersangkutan.

Contoh keluaran **localhost:8080/rest/course/view/csc123** adalah sebagai berikut.

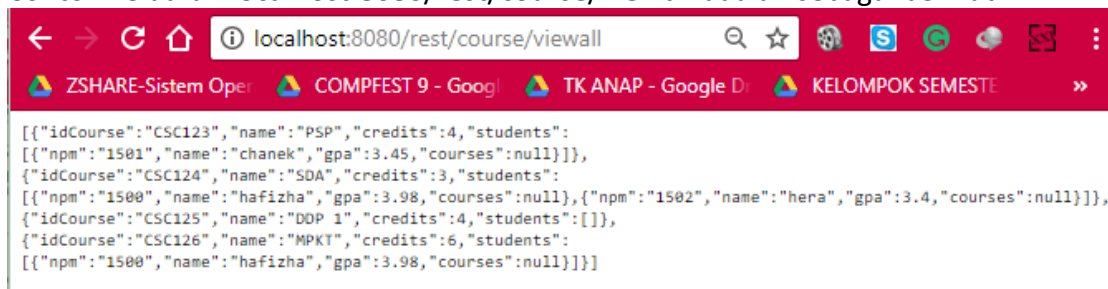


```

{
  "idCourse": "CSC123",
  "name": "PSP",
  "credits": 4,
  "students": [
    {
      "npm": "1501",
      "name": "chanek",
      "gpa": 3.45,
      "courses": null
    }
  ]
}

```

Contoh keluaran **localhost:8080/rest/course/viewall** adalah sebagai berikut.



```

{
  {
    "idCourse": "CSC123",
    "name": "PSP",
    "credits": 4,
    "students": [
      {
        "npm": "1501",
        "name": "chanek",
        "gpa": 3.45,
        "courses": null
      }
    ]
  },
  {
    "idCourse": "CSC124",
    "name": "SDA",
    "credits": 3,
    "students": [
      {
        "npm": "1500",
        "name": "hafizha",
        "gpa": 3.98,
        "courses": null
      },
      {
        "npm": "1502",
        "name": "hera",
        "gpa": 3.4,
        "courses": null
      }
    ]
  }
],
  {
    "idCourse": "CSC125",
    "name": "DDP 1",
    "credits": 4,
    "students": [
    ]
  },
  {
    "idCourse": "CSC126",
    "name": "MPKT",
    "credits": 6,
    "students": [
      {
        "npm": "1500",
        "name": "hafizha",
        "gpa": 3.98,
        "courses": null
      }
    ]
  }
]
}

```

Membuat Service Consumer

1. Ubah **application.properties** dengan menambahkan baris sebagai berikut.
`server.port=9090`

2. Isi *Interface* StudentDAO dengan kode sebagai berikut.

```
public interface StudentDAO {  
    StudentModel selectStudent (String npm);  
    List<StudentModel> selectAllStudents();  
}
```

3. Implementasi kelas StudentDAO dengan nama kelas StudentDAOImpl.java dengan isi sebagai berikut.

```
@Service  
public class StudentDAOImpl implements StudentDAO{  
    @Autowired  
    private RestTemplate restTemplate;  
  
    @Override  
    public StudentModel selectStudent (String npm) {  
        StudentModel student =  
            restTemplate.getForObject(  
                "http://localhost:8080/rest/student/view/"+npm,  
                StudentModel.class);  
        return student;  
    }  
  
    @Override  
    public List<StudentModel> selectAllStudents() {  
        return null;  
    }  
}
```

4. Tambahkan pada *file* APAPTutorial07ConsumerApplication.java sebagai berikut untuk men-define RestTemplate.

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

5. Buat *class* StudentServiceRest dengan isi sebagai berikut.

```

@Slf4j
@Service
@Primary
public class StudentServiceRest implements StudentService{
    @Autowired
    private StudentDAO studentDAO;

    @Override
    public StudentModel selectStudent(String npm) {
        log.info("REST - select student with npm {}", npm);
        return studentDAO.selectStudent(npm);
    }

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

    @Override
    public void addStudent (StudentModel student) {}

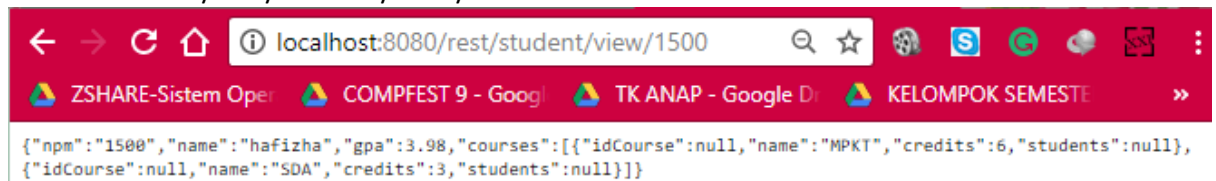
    @Override
    public void deleteStudent(String npm) {}

    @Override
    public void updateStudent(StudentModel student) {}
}

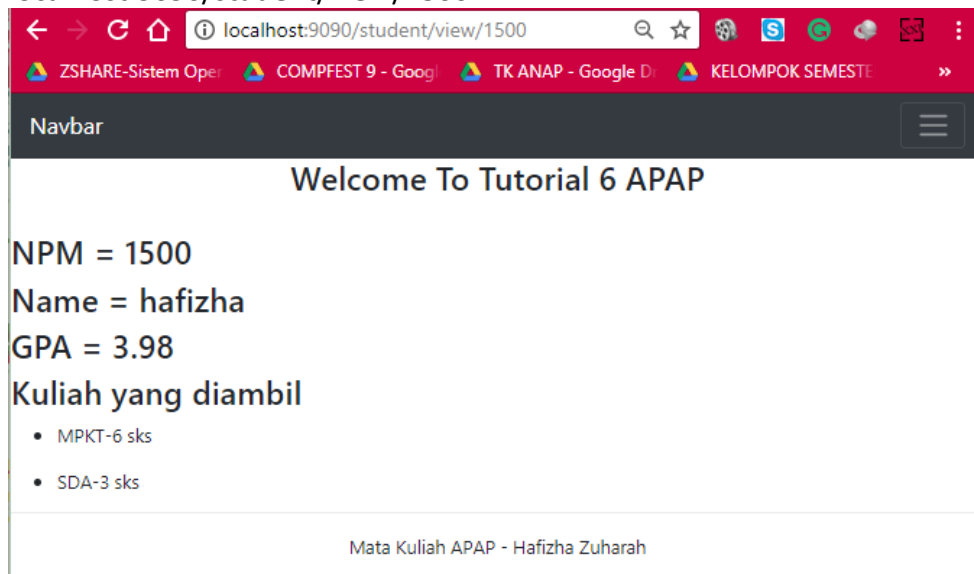
```

6. Jalankan kedua *project*.

localhost:8080/rest/student/view/1500



localhost:9090/student/view/1500



LATIHAN 3

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

Implementasikan `selectAllStudents()` pada `StudentServiceRest`.

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

Method tersebut akan menjalankan *method* `select all student` pada *database*.

Pada `StudentDAOImpl.java`, implementasikan *method* `selectAllStudents()`.

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

Contoh keluaran `localhost:9090/studet/viewall`

localhost:9090/student/viewall

ZSHARE-Sistem OperCOMPFEST 9 - GooglTK ANAP - Google DiKELOMPOK SEMESTER

Navbar

Welcome To Tutorial 6 APAP

All Students

No.	NPM	Name	GPA	Cum laude	Delete	Update
1	1500	hafizha	3.98	Cum Laude!	Delete Data	Update Data
2	1501	chanek	3.45	Sangat Memuaskan!	Delete Data	Update Data
3	1502	hera	3.4	Sangat Memuaskan!	Delete Data	Update Data
4	1503	Ano	3.73	Cum Laude!	Delete Data	Update Data

Mata Kuliah APAP - Hafizha Zuharah

LATIHAN 4

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

Buat *class* CourseDAO

```
package com.example.dao;

import java.util.List;

import com.example.model.CourseModel;

public interface CourseDAO {
    CourseModel selectCourse(String idCourse);

    List<CourseModel> selectAllCourses();
}
```

Implementasikan *class* StudentDAO

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

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

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

Buat *class* CourseServiceRest


```

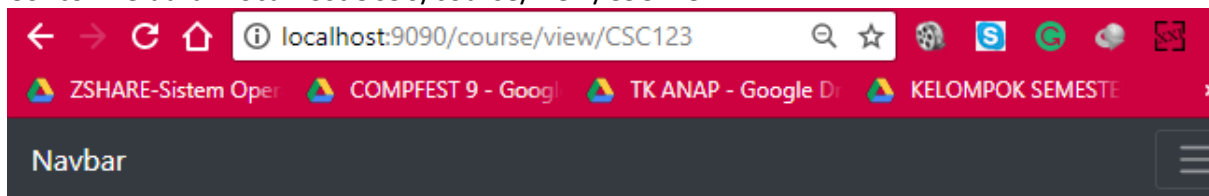
@Slf4j
@Service
@Primary
public class CourseServiceRest implements CourseService {
    @Autowired
    private CourseDAO courseDAO;

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

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

```

Contoh keluaran localhost:9090/course/view/CSC123



Welcome To Tutorial 6 APAP

ID = CSC123

Name = PSP

GPA = 4

Mahasiswa yang mengambil

- 1501 - chanek

Mata Kuliah APAP - Hafizha Zuharah

LESSON LEARNED

Pada tutorial 7 kali ini, saya mempelajari bagaimana cara mengimplementasikan *layer backend* dan *layer frontend* yang dibuat secara terpisah lalu dihubungkan saat menjalankannya. Pemisahan *responsibility* ini membuat sebuah aplikasi *service consumer* dapat fokus untuk menyediakan dan mengolah data saja ke pengguna tanpa perlu berurusan ke *database*. Sementara *service producer* dapat fokus untuk menyediakan data dari *database* dan biasanya tidak memiliki *view* yang dapat dilihat pengguna karena biasanya dalam bentuk

format JSON. Rest Template berguna sebagai penghubung *layer backend* dengan *layer frontend*.