

TUGAS TUTORIAL 07
ARSITEKTUR DAN PEMROGRAMAN APLIKASI
ENTERPRISE



Disusun oleh :

ANDESTA PUTRA
NPM. 1706106633

PROGRAM SARJANA 1 EKSTENSI SISTEM INFORMASI
FAKULTAS ILMU KOMPUTER
UNIVERSITAS INDONESIA
2018

A. Ringkasan Materi

Tutorial ini berfokus pada pemanfaatan Web Service. Web Service sendiri merupakan sebuah URL yang akan mengembalikan data berupa JSON atau XML yang telah disepakati sebagai Bahasa yang general dengan tujuan agar sistem dengan Bahasa yang berbeda dapat berkomunikasi satu sama lain.

B. Latihan

1. Membuat Service Producer

Untuk membuat service producer kita dapat mengimport data dari Tutorial 4.

Kemudian kita perlu membuat package baru yaitu com.example.rest dan membuat controller StudentRestController.java. berikut merupakan script dari controller tersebut.

```
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.StudentModel;
```

```
import com.example.service.StudentService;
```

```
@RestController
```

```
@RequestMapping("/rest")
```

```
public class StudentRestController
```

```
{
```

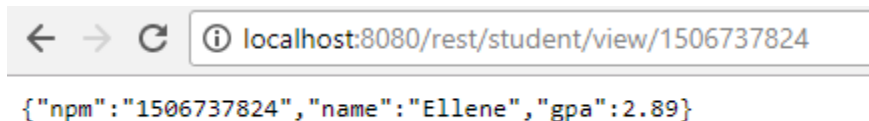
```
    @Autowired
```

```
StudentService studentService;
```

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

Dalam pembuatan controller ini digunakan anotasi `@RestController` sebagai syarat pembuatan controller service. Pada controller service sendiri nilai yang akan di return bukanlah view melainkan data dari objek yang telah kita buat pada function tersebut. Kemudian secara otomatis Spring Boot akan mengembalikan output tersebut dalam bentuk JSON pada tampilan Web.

Pada contoh kali ini kita akan memanggil path yang telah dibuat yaitu `/rest/student/view/npm`.



```
< > ↻ ⓘ localhost:8080/rest/student/view/1506737824  
  
{ "npm": "1506737824", "name": "Ellene", "gpa": 2.89 }
```

2. Membuat Service Consumer

Setelah selesai dengan pembuatan Web Service maka kita akan membuat program yang bertugas untuk mengambil data dari web service tersebut yaitu service Consumer.

Kita dapat mengimport data dari project tutorial 5 dan kemudian menambahkan `service.port:9090` pada `application.properties` karena program ini akan dijalankan pada `localhost:9090`.

Kemudian kita akan membuat interface `StudentDAO` pada package `dao`.

```
package com.example.dao;  
  
import java.util.List;  
  
import com.example.model.StudentModel;  
  
public interface StudentDAO {  
    StudentModel selectStudent (String npm);  
    List<StudentModel> selectAllStudents ();  
}
```

Kemudian kita buat implementasi dari interface tersebut pada kasus ini adalah `StudentDAOImpl.java` pada package yang sama. Berikut merupakan script dari class tersebut.

```

package com.example.dao;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.web.client.RestTemplate;

import com.example.model.StudentModel;

@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;
    }
}

```

Nantinya kelas ini akan digunakan untuk mengakses web service dan mengambil data yang dibutuhkan oleh sistem.

Selanjutnya kita akan membuat service pada kasus ini adalah StudentServiceRest.java dengan script sebagai berikut.

```

package com.example.service;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Primary;
import org.springframework.stereotype.Service;

import com.example.dao.StudentDAO;
import com.example.model.StudentModel;

import lombok.extern.slf4j.Slf4j;

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

```

```

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

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

@Override
public void addStudent (StudentModel student){}

@Override
public void deleteStudent (String npm){}

@Override
public void updateStudent (StudentModel student){}

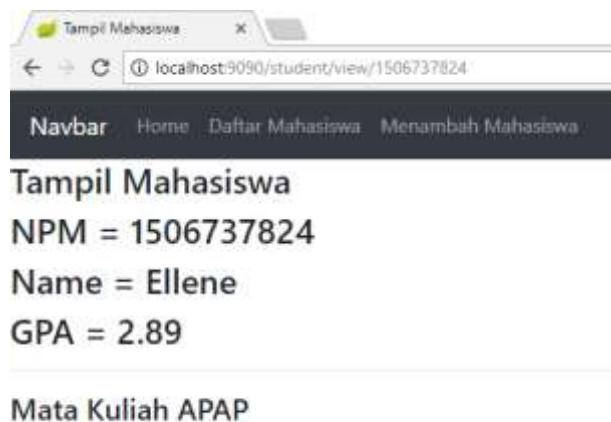
}

```

Pada service ini akan mengakses studentDAO yang telah kita buat sebelumnya sehingga dengan cara ini kita akan dapat mengakses web service yang telah dibuat. Adapun notasi @Primary digunakan untuk menandakan bahwa StudentServiceRest Menjadi service utama atau default ketika kita mengakses service pada project ini.

Selanjutnya kita bisa menjalankan project consumer dan producer secara bersamaan sehingga kedua sistem dapat saling berkomunikasi.

Berikut ini merupakan tampilan ketika kita mengakses service consumer.



Pada status running sendiri akan kita dapat bahwa service produser mendapat request untuk mengeksekusi path yang telah diinputkan oleh service consumer.

```
2018-04-14 22:33:33.065 INFO 8856 --- [nio-8080-exec-2] c.e.service.StudentServiceDatabase : select all students
```

Dengan begini kita telah berhasil membuat dua sistem yang saling berkomunikasi untuk dan menjalankan fungsinya masing-masing.

C. Pertanyaan

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"`.

Untuk membuat service tersebut kita dapat membuat method pada StudentRestController pada kasus ini akan dibuat method viewall dengan path `rest/student/viewall` sebagai berikut.

```
@RequestMapping("/student/viewall")
public List<StudentModel> viewall () {
    // StudentModel student = studentService.selectStudent (npm);

    List<StudentModel> students = studentService.selectAllStudents ();
    return students;
}
```

Pada method tersebut akan membuat list objek students dengan memanfaatkan `studentService.selectAllStudent()` yang telah dibuat pada tutorial 4 sebelumnya, yaitu berisikan query untuk mengambil data semua mahasiswa. Kemudian list objek tersebut akan di jadikan nilai return dari method ini yang mana akan otomatis di ubah ke dalam bentuk JSON. Berikut adalah tampilan dari eksekusi method ini.



```
[{"npm": "1506737823", "name": "Kernie", "gpa": 3.32}, {"npm": "1506737824", "name": "Ellene", "gpa": 2.89}, {"npm": "1506737825", "name": "Delmer", "gpa": 2.58}, {"npm": "1506737826", "name": "Kristian", "gpa": 2.33}, {"npm": "1506737827", "name": "Marilyn", "gpa": 5.46}, {"npm": "1506737828", "name": "Katti", "gpa": 3.73}, {"npm": "1506737829", "name": "Titus", "gpa": 3.11}, {"npm": "1506737830", "name": "Glenden", "gpa": 5.36}, {"npm": "1506737831", "name": "Tilt", "gpa": 3.39}, {"npm": "1506737832", "name": "Henri", "gpa": 2.31}, {"npm": "1506737833", "name": "Buffy", "gpa": 3.35}, {"npm": "1506737834", "name": "Roda", "gpa": 2.2}, {"npm": "1506737835", "name": "Petronilla", "gpa": 2.61}, {"npm": "1506737836", "name": "Kelley", "gpa": 3.27}, {"npm": "1506737837", "name": "Morey", "gpa": 3.05}, {"npm": "1506737838", "name": "Ronda", "gpa": 2.23}, {"npm": "1506737839", "name": "Corbin", "gpa": 3.93}, {"npm": "1506737840", "name": "Jeanie", "gpa": 2.5}, {"npm": "1506737841", "name": "Serrv", "gpa": 3.8}, {"npm": "1506737842", "name": "Lesley", "gpa": 3.0}, {"npm": "1506737843", "name": "Fliss", "gpa": 3.72}, {"npm": "1506737844", "name": "Harrell", "gpa": 3.15}, {"npm": "1506737845", "name": "Ibbie", "gpa": 3.45}, {"npm": "1506737846", "name": "Tony", "gpa": 2.69}, {"npm": "1506737847", "name": "Klarika", "gpa": 2.83}, {"npm": "1506737848", "name": "Brewster", "gpa": 2.62}, {"npm": "1506737849", "name": "Zacharie", "gpa": 3.12}, {"npm": "1506737850", "name": "Betsey", "gpa": 2.24}, {"npm": "1506737851", "name": "Avery", "gpa": 2.92}, {"npm": "1506737852", "name": "Gill", "gpa": 3.12}, {"npm": "1506737853", "name": "Stephan", "gpa": 3.62}, {"npm": "1506737854", "name": "Viola", "gpa": 3.91}, {"npm": "1506737855", "name": "Caryl", "gpa": 3.83}, {"npm": "1506737856", "name": "Nettie", "gpa": 2.52}, {"npm": "1506737857", "name": "Eylee", "gpa": 3.69}, {"npm": "1506737858", "name": "Bird", "gpa": 3.15}, {"npm": "1506737859", "name": "Geno", "gpa": 2.19}, {"npm": "1506737860", "name": "Saxon", "gpa": 2.91}, {"npm": "1506737861", "name": "Jolianne", "gpa": 2.41}, {"npm": "1506737862", "name": "Harris", "gpa": 2.14}, {"npm": "1506737863", "name": "Sheba", "gpa": 2.41}, {"npm": "1506737864", "name": "Harcellina", "gpa": 3.67}, {"npm": "1506737865", "name": "Aubrey", "gpa": 2.22}, {"npm": "1506737866", "name": "Bliss", "gpa": 2.7}, {"npm": "1506737867", "name": "Henry", "gpa": 2.32}, {"npm": "1506737868", "name": "Portia", "gpa": 2.56}, {"npm": "1506737869", "name": "Jewel", "gpa": 2.35}, {"npm": "1506737870", "name": "Andris", "gpa": 3.38}, {"npm": "1506737871", "name": "Nadiya", "gpa": 3.86}, {"npm": "1506737872", "name": "Aggie", "gpa": 2.92}, {"npm": "1506737873", "name": "Ramsey", "gpa": 3.6}, {"npm": "1506737874", "name": "Saundra", "gpa": 3.26}, {"npm": "1506737875", "name": "Sylas", "gpa": 2.43}, {"npm": "1506737876", "name": "Clyde", "gpa": 3.04}, {"npm": "1506737877", "name": "Rodestine", "gpa": 2.83}, {"npm": "1506737878", "name": "Max", "gpa": 2.53}, {"npm": "1506737879", "name": "Chester", "gpa": 2.77}, {"npm": "1506737880", "name": "Carl", "gpa": 3.01}, {"npm": "1506737881", "name": "Hilton", "gpa": 3.07}, {"npm": "1506737882", "name": "Eugene", "gpa": 3.74}, {"npm": "1506737883", "name": "Osmond", "gpa": 2.83}, {"npm": "1506737884", "name": "Cele", "gpa": 2.01}, {"npm": "1506737885", "name": "Kandy", "gpa": 3.3}, {"npm": "1506737886", "name": "Zachary", "gpa": 2.71}, {"npm": "1506737887", "name": "Onaida", "gpa": 3.12}, {"npm": "1506737888", "name": "Dietrich", "gpa": 2.72}, {"npm": "1506737889", "name": "Brucie", "gpa": 2.1}, {"npm": "1506737890", "name": "Lucha", "gpa": 3.27}, {"npm": "1506737891", "name": "Giorgia", "gpa": 2.05}, {"npm": "1506737892", "name": "Taber", "gpa": 3.61}, {"npm": "1506737893", "name": "Carlen", "gpa": 2.81}, {"npm": "1506737894", "name": "Bell", "gpa": 2.50}, {"npm": "1506737895", "name": "Nataniele", "gpa": 2.21}, {"npm": "1506737896", "name": "Dannie", "gpa": 3.43}, {"npm": "1506737897", "name": "Cassandry", "gpa": 2.74}
```

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

untuk membuatnya kita dapat menambahkan script berikut pada StudentServiceRest dimana nilai yang direturn adalah studentDAO dengan method selectAllStudents();

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

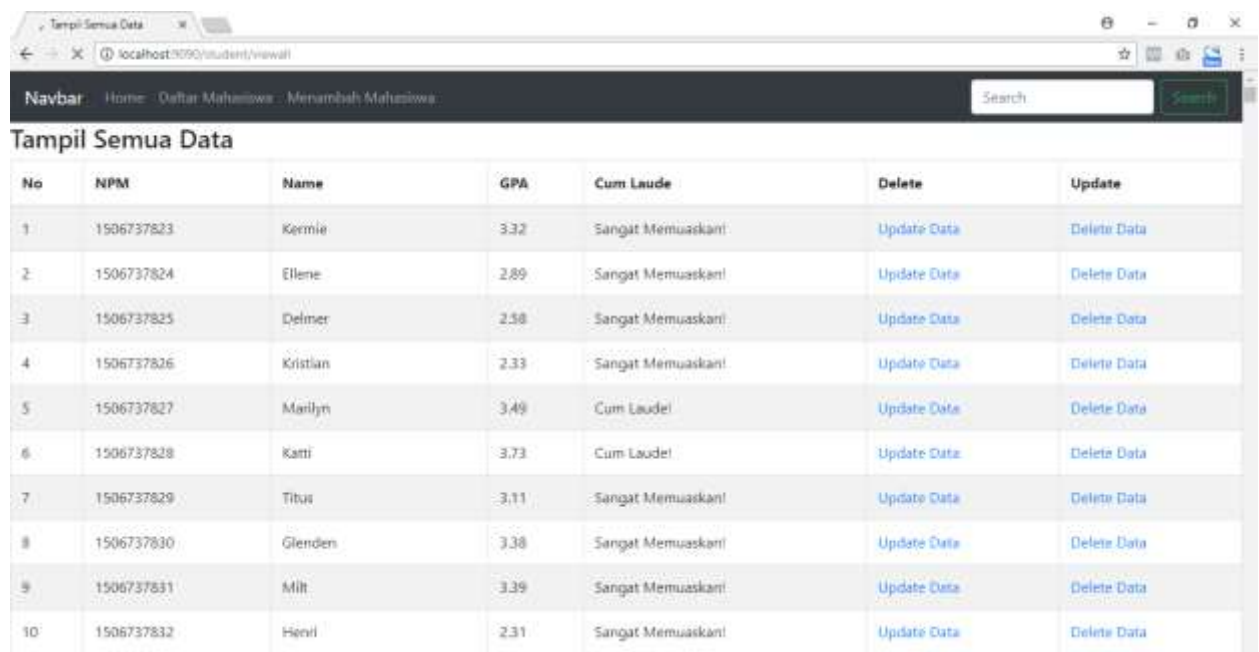
Selanjutnya kita akan membuat method selectAllStudents() pada StudentDAOImpl.

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

    return students;
}
```

Sama seperti method selectStudent, disini kita akan membuat object tapi kali ini dalam bentuk list dan juga kita akan memanfaatkan url yang telah kita buat untuk memanggil seluruh data mahasiswa pada web service yang telah kita buat yaitu /rest/student/viewall.

Tampilan dari eksekusi url tersebut adalah sebagai berikut.



The screenshot shows a web browser window with the URL `localhost:3090/student/viewall`. The page has a dark navbar with links for Home, Daftar Mahasiswa, and Menambah Mahasiswa, along with a search bar. The main content area is titled "Tampil Semua Data" and contains a table with the following data:

No	NPM	Name	GPA	Cum Laude	Delete	Update
1	1506737823	Kernie	3.32	Sangat Memuaskan!	Update Data	Delete Data
2	1506737824	Ellene	2.89	Sangat Memuaskan!	Update Data	Delete Data
3	1506737825	Delmer	2.58	Sangat Memuaskan!	Update Data	Delete Data
4	1506737826	Kristian	2.33	Sangat Memuaskan!	Update Data	Delete Data
5	1506737827	Marilyn	3.49	Cum Laude!	Update Data	Delete Data
6	1506737828	Katti	3.73	Cum Laude!	Update Data	Delete Data
7	1506737829	Titus	3.11	Sangat Memuaskan!	Update Data	Delete Data
8	1506737830	Glenden	3.38	Sangat Memuaskan!	Update Data	Delete Data
9	1506737831	Milt	3.39	Sangat Memuaskan!	Update Data	Delete Data
10	1506737832	Henri	2.31	Sangat Memuaskan!	Update Data	Delete Data