

## Web Service

Web service dalam sebuah aplikasi atau system berfungsi untuk menghubungkan aplikasi-aplikasi agar saling terhubung. Misalnya sebuah system terdiri dari aplikasi – aplikasi yang terdapat di beberapa server berbeda maka perlu membuat service untuk dapat menghubungkan aplikasi – aplikasi tersebut. Contohnya adalah layer backend (service producer) dan frontend (service consumer). Service consumer merupakan aplikasi yang berinteraksi dengan pengguna. Sedangkan service producer adalah aplikasi yang memberikan data kepada service consumer sesuai dengan data yang diminta. Sehingga kedua aplikasi dapat focus akan fungsi masing-masing.

Web service merupakan sebuah URL yang akan mengembalikan data dalam representasi yang telah disepakati misalnya JSON atau XML.

## Membuat Service Producer

Buatlah class StudentRestController.java seperti pada gambar di bawah ini.

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

Untuk membuat controller menggunakan REST web service pada class header perlu diberikan anotasi @RestController, bukan @Controller seperti controller biasa. @RequestMapping("/rest") pada class header tersebut artinya semua request mapping di dalam kelas tersebut harus selalu diawali dengan "/rest". Contohnya pemanggilan service view Student dengan npm 123 diakses melalui "/rest/student/view/123".

- Berbeda dengan controller Web pada biasanya yang mengembalikan String yang merepresentasikan view yang akan digunakan. Method-method di REST Controller mengembalikan objek yang ingin dikembalikan pada service tersebut.
- Spring Boot secara otomatis akan mengembalikan output berupa objek dengan format JSON pada tampilan Web.
- Beberapa object yang dapat dikembalikan misalkan model Class dan Map (key value Map)

Hasilnya adalah seperti pada gambar di bawah ini:



```
{ "npm": "1506737823", "name": "Kermie", "gpa": 3.32 }
```

### 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".

Pada StudentRestController perlu ditambahkan sebuah method dengan mapping /student/viewall yang berisi pemanggilan method selectAllStudent yang ada pada kelas StudentService. Dengan

demikian method ini akan memanggil method `selectAllStudent` dan mengembalikan semua data student dalam bentuk objek yaitu format JSON. Methodnya dapat dilihat pada gambar di bawah ini.

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

Contoh tampilan keluarannya yang berupa format JSON dapat dilihat pada gambar di bawah 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": "Kristia", "gpa": 3.53}, {"npm": "1506737827", "name": "Marilyn", "gpa": 3.49}, {"npm": "1506737828", "name": "Tatti", "gpa": 3.73}, {"npm": "1506737829", "name": "Titus", "gpa": 3.11}, {"npm": "1506737830", "name": "Glenden", "gpa": 3.38}, {"npm": "1506737831", "name": "Milt", "gpa": 3.39}, {"npm": "1506737832", "name": "Henri", "gpa": 2.13}, {"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": 3.95}, {"npm": "1506737838", "name": "Ronda", "gpa": 2.23}, {"npm": "1506737839", "name": "Corbin", "gpa": 3.93}, {"npm": "1506737840", "name": "Jannie", "gpa": 2.5}, {"npm": "1506737841", "name": "Berril", "gpa": 3.9}, {"npm": "1506737842", "name": "Lesley", "gpa": 3.0}, {"npm": "1506737843", "name": "Flss", "gpa": 3.72}, {"npm": "1506737844", "name": "Merrell", "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": "Zacherie", "gpa": 3.12}, {"npm": "1506737850", "name": "Betsey", "gpa": 2.24}, {"npm": "1506737851", "name": "Avery", "gpa": 3.38}, {"npm": "1506737852", "name": "Gill", "gpa": 3.12}, {"npm": "1506737853", "name": "Stephan", "gpa": 3.62}, {"npm": "1506737854", "name": "Viole", "gpa": 3.91}, {"npm": "1506737855", "name": "Caryl", "gpa": 3.83}, {"npm": "1506737856", "name": "Nettie", "gpa": 2.52}, {"npm": "1506737857", "name": "Emyle", "gpa": 3.69}, {"npm": "1506737858", "name": "Bird", "gpa": 3.15}, {"npm": "1506737859", "name": "Saxon", "gpa": 2.91}, {"npm": "1506737860", "name": "Lianne", "gpa": 2.41}, {"npm": "1506737862", "name": "Harris", "gpa": 2.12}, {"npm": "1506737863", "name": "Sheba", "gpa": 2.43}, {"npm": "1506737864", "name": "Marcellin", "gpa": 3.67}, {"npm": "1506737865", "name": "Aubrey", "gpa": 2.22}, {"npm": "1506737866", "name": "Bliss", "gpa": 2.77}, {"npm": "1506737867", "name": "Merry", "gpa": 2.32}, {"npm": "1506737868", "name": "Portia", "gpa": 2.56}, {"npm": "1506737869", "name": "Jewel", "gpa": 3.25}, {"npm": "1506737870", "name": "Andris", "gpa": 3.38}, {"npm": "1506737871", "name": "Nadiya", "gpa": 3.86}, {"npm": "1506737872", "name": "Aggie", "gpa": 2.92}, {"npm": "1506737873", "name": "Ramsay", "gpa": 3.6}, {"npm": "1506737874", "name": "Saundra", "gpa": 3.28}, {"npm": "1506737875", "name": "Sylys", "gpa": 2.43}, {"npm": "1506737876", "name": "Clyde", "gpa": 3.04}, {"npm": "1506737877", "name": "Modestine", "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": "Milton", "gpa": 3.07}, {"npm": "1506737882", "name": "Eugine", "gpa": 3.74}, {"npm": "1506737883", "name": "Osmond", "gpa": 2.03}, {"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": "Bruce", "gpa": 2.1}, {"npm": "1506737890", "name": "Lucho", "gpa": 3.27}, {"npm": "1506737891", "name": "Giorgia", "gpa": 2.05}, {"npm": "1506737892", "name": "Gail", "gpa": 2.61}, {"npm": "1506737893", "name": "Doris", "gpa": 2.81}, {"npm": "1506737894", "name": "Gail", "gpa": 2.5}, {"npm": "1506737895", "name": "Natanie", "gpa": 2.21}, {"npm": "1506737896", "name": "Dannis", "gpa": 3.63}, {"npm": "1506737897", "name": "Cassidy", "gpa": 2.74}, {"npm": "1506737898", "name": "Madeline", "gpa": 3.72}, {"npm": "1506737899", "name": "Linoel", "gpa": 3.34}, {"npm": "150673900", "name": "Cori", "gpa": 2.39}, {"npm": "1506737901", "name": "Baillie", "gpa": 2.79}, {"npm": "1506737902", "name": "Suzy", "gpa": 3.85}, {"npm": "1506737903", "name": "Mervin", "gpa": 3.06}, {"npm": "1506737904", "name": "Gannie", "gpa": 2.93}, {"npm": "1506737905", "name": "Alla", "gpa": 3.95}, {"npm": "1506737906", "name": "Rutter", "gpa": 2.43}, {"npm": "1506737907", "name": "Mela", "gpa": 3.4}, {"npm": "1506737908", "name": "Conrado", "gpa": 3.38}, {"npm": "1506737909", "name": "Isabella", "gpa": 2.77}, {"npm": "1506737910", "name": "Lynett", "gpa": 2.36}, {"npm": "1506737911", "name": "Wesley", "gpa": 3.33}, {"npm": "1506737912", "name": "Darrrel", "gpa": 2.57}, {"npm": "1506737913", "name": "Corrina", "gpa": 3.38}, {"npm": "1506737914", "name": "Lorry", "gpa": 2.8}, {"npm": "1506737915", "name": "Merrill", "gpa": 3.12}, {"npm": "1506737916", "name": "Claudie", "gpa": 2.14}, {"npm": "1506737917", "name": "Dorris", "gpa": 2.56}, {"npm": "1506737918", "name": "Silvia", "gpa": 3.37}, {"npm": "1506737919", "name": "Arluene", "gpa": 2.03}, {"npm": "1506737920", "name": "Elissaa", "gpa": 3.64}, {"npm": "1506737921", "name": "Shirley", "gpa": 3.23}, {"npm": "1506737922", "name": "Vonnle", "gpa": 3.75}, {"npm": "1506737923", "name": "Beverlee", "gpa": 3.39}, {"npm": "1506737924", "name": "Julissa", "gpa": 3.82}, {"npm": "1506737925", "name": "Lorena", "gpa": 2.65}, {"npm": "1506737926", "name": "Conner", "gpa": 3.63}, {"npm": "1506737927", "name": "Ludiana", "gpa": 2.61}, {"npm": "1506737928", "name": "Whitake", "gpa": 2.11}, {"npm": "1506737929", "name": "Coreen", "gpa": 2.81}, {"npm": "1506737930", "name": "Averell", "gpa": 2.8}, {"npm": "1506737931", "name": "Duffie", "gpa": 3.35}, {"npm": "1506737932", "name": "Tasia", "gpa": 2.63}, {"npm": "1506737933", "name": "Lorelle", "gpa": 3.84}, {"npm": "1506737934", "name": "Marlo", "gpa": 3.28}, {"npm": "1506737935", "name": "Imoigne", "gpa": 2.98}, {"npm": "1506737936", "name": "Phyllys", "gpa": 3.17}, {"npm": "1506737937", "name": "Xenia", "gpa": 3.47}, {"npm": "1506737938", "name": "Claybourne", "gpa": 2.91}, {"npm": "1506737939", "name": "Nicholas", "gpa": 3.45}

```

Dengan menggunakan link <http://json.parser.online.fr> hasilnya seperti pada gambar di bawah ini:

```

    {
      "npm": "1506737825",
      "name": "Delmer",
      "gpa": 2.58
    },
    {
      "npm": "1506737826",
      "name": "Kristian",
      "gpa": 2.33
    },
    {
      "npm": "1506737827",
      "name": "Marilyn",
      "gpa": 3.49
    },
    {
      "npm": "1506737828",
      "name": "Katti",
      "gpa": 3.73
    },
    {
      "npm": "1506737829",
      "name": "Titus",
      "gpa": 3.11
    }
  ]
}

```

## Membuat Service Consumer

Setelah membuat producer, maka langkah berikutnya adalah membuat service consumer yaitu aplikasi yang akan menggunakan data yang dihasilkan oleh produser. Pertama perlu menambahkan interface dengan nama StudentDAO seperti pada gambar berikut.

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

Selanjutnya akan ada kelas yang mengimplementasikan interface tersebut dengan nama StudentDAOImpl yang isinya seperti gambar berikut.

```
package com.example.dao;

import java.util.List;
@Service
public class StudentDAOImpl implements StudentDAO {
    @Autowired
    @Lazy
    private RestTemplate restTemplate;

    @Override
    public StudentModel selectStudent(String npm)
    {
        StudentModel student = restTemplate.getForObject("http://localhost:8090/rest/student/view/"+npm, StudentModel.class);
        return student;
    }

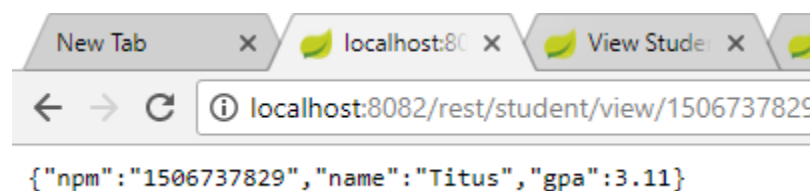
    @Override
    public List<StudentModel> selectAllStudents()
    {
        return null;
        // List<StudentModel> students = restTemplate.getForObject("http://localhost:8080/rest/student/viewall", StudentModel.class);
    }

    @Bean
    public RestTemplate restTemplate(RestTemplateBuilder builder) {
        return builder.build();
    }
}
```

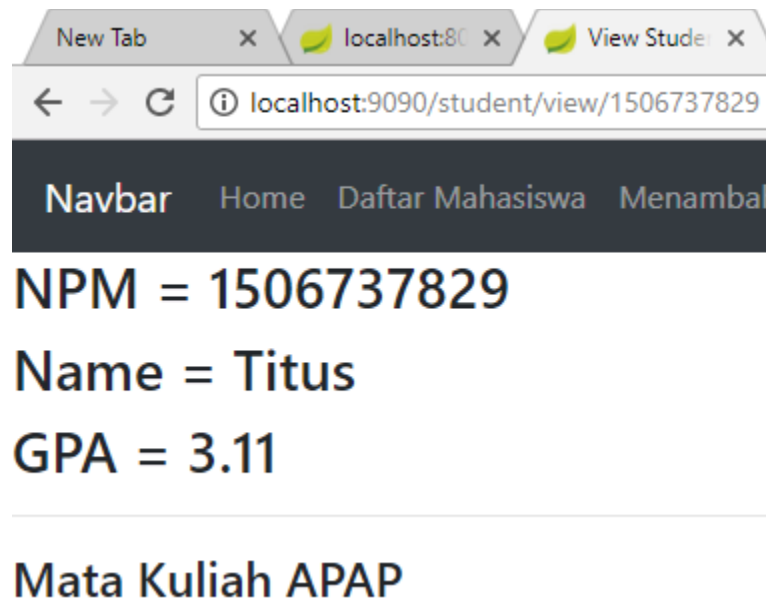
Penjelasan: Kita menggunakan kelas RestTemplate untuk meng-consume object REST web service. Method yang digunakan adalah getForObject yang menerima parameter berupa URL producer web service dan tipe class dari object yang didapatkan.

Selanjutnya, kita ingin mengubah agar StudentService mengambil data dari web service bukan dari database. Kita tidak perlu menghapus class StudentServiceDatabase. Karena scalable system, kita cukup menambahkan class baru yaitu StudentServiceRest yang mengimplement StudentService di package service.

Pertama jalankan service producer dengan menjalankan localhost:8099/rest/student/view/1506737829



Kemudian untuk menguji service consumer buka localhost:8090/student/view/1506737829



LATIHAN 1.

### Latihan 2

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

Caranya pertama tambahkan file berikut pada StudentServiceRest untuk memanggil studentDAO yang memiliki fungsi selectAllStudents.

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

Pada StudentDAO tambahkan deklarasi method selectAllStudents yang akan diimplementasi oleh kelas yang meng implements sehingga dapat memanggil service yang disediakan oleh producer.

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

Kemudian pada StudentDAOimpl tambahkan pemanggilan restTemplate yang memanggil url yang dimiliki oleh producer sehingga data yang dihasilkan nanti merupakan hasil pemanggilan producer seperti alamat

url yang diakses. Student model akan ditampung dalam arraylist untuk menampung data students hasil pemanggilan url service dan dikembalikan dalam list StudentModel seperti pada gambar di bawah ini.

```
@Override
public List<StudentModel> selectAllStudents()
{
    StudentModel[] students = restTemplate.getForObject("http://localhost:8082/rest/student/viewall/", StudentModel[].class);
    List<StudentModel> results = Arrays.asList(students);
    return results;
}
```

Berikut ini hasil pemanggilan service Producer.

The screenshot shows a web browser window with a REST client interface. The address bar shows the URL `localhost:8082/rest/student/viewall`. The response is a JSON array of student objects, each containing fields like `npm`, `name`, and `gpa`. The data is as follows:

npm	name	gpa
1506737823	Kermie	3.32
1506737824	Ellene	2.89
1506737825	Delmer	2.58
1506737826	Kristian	2.33
1506737827	Marilyn	3.49
1506737828	Katti	3.73
1506737829	Titus	3.11
1506737830	Glenden	3.38
1506737831	Milt	3.39
1506737832	Henri	2.31
1506737833	Buffy	3.35
1506737834	Roda	2.2
1506737835	Petronille	2.61
1506737836	Kelley	3.27
1506737837	Morey	3.95
1506737838	Ronda	2.23
1506737839	Corbin	3.93
1506737840	Jeanie	2.5
1506737841	Berri	3.9
1506737842	Lesley	3.0
1506737843	Flls	3.72
1506737844	Merell	3.15
1506737845	Ibbie	3.45
1506737846	Tony	2.69
1506737847	Klarika	2.83
1506737848	Brewster	2.62
1506737849	Zacherie	3.12
1506737850	Betsey	2.24
1506737851	Avery	3.38
1506737852	Gill	3.12
1506737853	Stephan	3.62
1506737854	Viole	3.91
1506737855	Caryl	3.83
1506737856	Nettie	2.52
1506737857	Emyle	3.69
1506737858	Bird	3.15
1506737859	Geno	2.19
1506737860	Saxon	2.91
1506737861	Juliana	2.41
1506737862	Marri	2.14
1506737863	Sheba	2.41
1506737864	Marcelline	3.67
1506737865	Aubrey	2.22
1506737866	Bliss	2.7
1506737867	Merry	2.32
1506737868	Portia	2.56
1506737869	Jewel	2.35
1506737870	Andris	3.38
1506737871	Nadiya	3.86
1506737872	Aggie	2.92
1506737873	Ramsay	3.6
1506737874	Saundra	3.28
1506737875	Sylas	2.43
1506737876	Clyde	3.04
1506737877	Modestine	2.83
1506737878	Max	2.53
1506737879	Chester	3.77
1506737880	Carl	3.01
1506737881	Milton	3.07
1506737882	Eugene	3.74
1506737883	Osmond	2.03
1506737884	Cele	2.01
1506737885	Kandy	3.3
1506737886	Zachary	2.71
1506737887	Oneida	3.12
1506737888	Dietrich	2.72
1506737889	Brucie	2.1
1506737890	Lucho	3.27
1506737891	Giorgia	2.05
1506737892	Taber	3.61
1506737893	Carlen	2.81
1506737894	Bell	2.58
1506737895	Nataniel	2.21
1506737896	Dannie	3.43
1506737897	Cassandry	2.74
1506737898	Madeline	3.72
1506737899	Lincol	3.34
1506737900	Cori	2.39
1506737901	Bailie	2.79
1506737902	Suzy	3.85
1506737903	Mervin	3.06

Kemudian consumer akan memanggil `student/viewall` yang mengambil data dari hasil producer seperti pada gambar di bawah ini.

The screenshot shows a web application interface with a table titled "All Students". The table has columns for NO, NPM, Name, GPA, Status, and actions (Delete Data, Update Data). The data is as follows:

NO	NPM	Name	GPA	Status	Delete Data	Update Data
1	1506737823	Kermie	3.32	Sangat Memuaskan	<a href="#">Delete Data</a>	<a href="#">Update Data</a>
2	1506737824	Ellene	2.89	Sangat Memuaskan	<a href="#">Delete Data</a>	<a href="#">Update Data</a>
3	1506737825	Delmer	2.58	Sangat Memuaskan	<a href="#">Delete Data</a>	<a href="#">Update Data</a>
4	1506737826	Kristian	2.33	Sangat Memuaskan	<a href="#">Delete Data</a>	<a href="#">Update Data</a>
5	1506737827	Marilyn	3.49	Cum Laude	<a href="#">Delete Data</a>	<a href="#">Update Data</a>
6	1506737828	Katti	3.73	Cum Laude	<a href="#">Delete Data</a>	<a href="#">Update Data</a>
7	1506737829	Titus	3.11	Sangat Memuaskan	<a href="#">Delete Data</a>	<a href="#">Update Data</a>
8	1506737830	Glenden	3.38	Sangat Memuaskan	<a href="#">Delete Data</a>	<a href="#">Update Data</a>
9	1506737831	Milt	3.39	Sangat Memuaskan	<a href="#">Delete Data</a>	<a href="#">Update Data</a>
10	1506737832	Henri	2.31	Sangat Memuaskan	<a href="#">Delete Data</a>	<a href="#">Update Data</a>
11	1506737833	Buffy	3.35	Sangat Memuaskan	<a href="#">Delete Data</a>	<a href="#">Update Data</a>
12	1506737834	Roda	2.2	Sangat Memuaskan	<a href="#">Delete Data</a>	<a href="#">Update Data</a>
13	1506737835	Petronille	2.61	Sangat Memuaskan	<a href="#">Delete Data</a>	<a href="#">Update Data</a>
14	1506737836	Kelley	3.27	Sangat Memuaskan	<a href="#">Delete Data</a>	<a href="#">Update Data</a>
15	1506737837	Morey	3.95	Cum Laude	<a href="#">Delete Data</a>	<a href="#">Update Data</a>