Tutorial 7

Membuat Web Service Menggunakan Spring Boot Framework

Pada tutorial ini mempelajari mengenai web service. Untuk mendukung turorial 7, diperlukan 2 project, yaitu Service Consumer dan Service Producer. Service consumer merupakan aplikasi yang berinteraksi dengan pengguna. Sementara service producer merupakan aplikasi yang memberikan data kepada service consumer berdasarkan apa yang service consumer minta. Dengan pemisahan tanggung jawab ini, aplikasi service consumer dapat fokus untuk menyediakan dan mengolah data saja ke pengguna tanpa perlu memiliki database. Sementara service producer hanya bertanggung jawab untuk menyediakan data dari database dan biasanya tidak memiliki view yang dapat dilihat pengguna. Untuk dapat berkomunikasi dengan service consumer, service producer menyediakan web service yang dapat dikonsumsi oleh service consumer. Web service merupakan sebuah URL yang akan mengembalikan data dalam representasi yang telah disepakati seperti JSON atau XML. Dalam tutorial ini Anda akan menggunakan JSON sebagai representasi data tersebut. Pada tutorial ini Anda akan membuat web service untuk mengakses data-data Student yang sudah digunakan pada tutorial 4 untuk Service Producer dan menggunakan tutorial 5 untuk Service Consumer.

Pada permbuatan service producer, menggunakan dengan tutorial 4, yaitu dengan tahap-tahap berikut ini:

1. Meng-Import berkas-berkas yang ada di tutorial 4 untuk Service producer ini.
2. Membuat package baru com.example.rest, atau sesuaikan dengan package Anda masing-masing
3. Membuat class baru yaitu StudentRestController.java pada package com.example.rest
   Class baru tersebut merupakan controller, namun menggunakan RestController, bukan 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)

Sedangkan, pada pembuatan Service consumer menggunakan data-data pada tutorial5, tahapan-tahapannya sebagai berikut:

1. Meng-Import berkas-berkas yang ada di tutorial 5 untuk Service Consumer ini
2. Mengubah server port menjadi 9090. Hal ini dikarenakan Service Producer dan Service Consumer harus dijalankan secara bersamaan, ubah berkas application.properties yang ada di folder resources dengan menambahkan baris
3. Menambahkan class interface StudentDAO
4. Membuat implementasi kelas StudentDAO tersebut dengan nama kelas StudentDAOImpl.java.
5. Selanjutnya, menambahkan class baru, yaitu StudentServiceRest yang mengimplementasi StudentService di package service. Hal ini digunakan untuk mengubah agar StudentService

mengambil data dari web service bukan dari database. Kita tidak perlu menghapus class StudentServiceDatabase.

Untuk meng-handle class StudentController kita untuk mengambil data dari web service, menggunakan anotasi @Primary digunakan. Autowired akan secara otomatis menginstansiasi StudentService menggunakan StudentServiceRest karena dia primary.

6. Jalankan kedua project Spring Boot untuk Service Producer dan Service Consumer tersebut
   - ✓ Menjalankan service producer dahulu, dengan menjalankan localhost:8080/rest/student/view/123.
   - ✓ Menjalankan program untuk menguji service consumer dengan membuka localhost:9090/student/view/123

   Untuk mengatasi error yang terjadi, maka dapat dihandle dengan:
   - Menambahkan kode pada package com.example pada Service Consumer (dapat dilihat pada workspace)
   - Menambahkan kode pada file StudentDAOImpl.java (dapat dilihat pada workspace)

LATIHAN

Latihan 1

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

Method tersebut digunakan untuk menampilkan seluruh data mahasiswa. Method menggunakan function list dan mengembalikan data seluruh mahasiswa.

Latihan 2

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

Method tersebut digunakan untuk menampilkan seluruh data mahasiswa. Method menggunakan list dan mengembalikan data seluruh mahasiswa dengan memanggil function selectAllAStudents() pada class studentDAO.