

Write-Up Tutorial 7: Membuat Web Service Menggunakan Spring Boot Framework

Ringkasan

Di dalam tutorial ini saya belajar menggunakan web service menggunakan Spring Boot Framework. Di dalam tutorial ini saya mempelajari bagaimana cara membuat service producer yang bertindak sebagai backend dari sistem dan service consumer yang bertindak sebagai front-end dari sistem.

Penggunaan web service bertujuan memudahkan sistem dengan skala besar sehingga bagian back-end dari sistem dapat fokus ke back-end sepenuhnya dan bagian front-end dari sistem dapat berfokus sepenuhnya.

Latihan 1

Untuk pengembangan method yang menampilkan keseluruhan student, saya menambahkan method berikut pada class StudentRestController.

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

Pada dasarnya cara kerja method ini mirip dengan method view all students yang terdapat pada class StudentController, hanya saja kali ini implementasinya lebih simpel. Method ini mengembalikan list berisi student yang diambil dari StudentService.

Latihan 2

Pada latihan 2, karena permintaan tutorial untuk menggunakan Rest Controller baru, maka saya melakukan beberapa adjustment, yaitu dengan memisahkan query MySQL yang berkaitan dengan Course ke interface baru, yaitu CourseMapper pada package com.example.dao. Selain itu, dilakukan juga pemisahan antara StudentService dan CourseService dengan membuat interface CourseService yang menyimpan method yang berhubungan dengan course pada package com.example.service. Adanya interface ini juga telah menyebabkan adanya class CourseServiceDatabase yang mengimplementasi CourseService pada package com.example.service.

Setelah itu, saya membuat class pada package com.example.rest dengan nama CourseRestController. Isi dari kelas tersebut adalah sebagai berikut.



```

1 package com.example.rest;
2
3 import java.util.List;
4
5 import org.springframework.beans.factory.annotation.Autowired;
6 import org.springframework.web.bind.annotation.PathVariable;
7 import org.springframework.web.bind.annotation.RequestMapping;
8 import org.springframework.web.bind.annotation.RestController;
9
10 import com.example.model.CourseModel;
11 import com.example.service.CourseService;
12
13 @RestController
14 @RequestMapping("/rest")
15 public class CourseRestController {
16     @Autowired
17     CourseService courseService;
18
19     @RequestMapping("/course/view/{id}")
20     public CourseModel viewById(@PathVariable(value = "id") String id) {
21         CourseModel course = courseService.selectCourse(id);
22         return course;
23     }
24
25     @RequestMapping("/course/viewall")
26     public List<CourseModel> viewAllCourses() {
27         List<CourseModel> courses = courseService.selectAllCourses();
28         return courses;
29     }
30 }
31

```

Pada gambar di atas terdapat dua method yang perlu diimplementasikan pada class CourseRestController, yaitu method mengambil course berdasarkan id dari course dan mengambil seluruh course. Cara kerja method ini kurang lebih sama dengan method yang ada di dalam StudentRestController, hanya saja method-method ini menggunakan CourseModel ketimbang StudentModel.

Latihan 3

Pada class StudentDAOImpl, implementasi dari method selectAllStudents adalah sebagai berikut.

```

@Override
public List<StudentModel> selectAllStudents() {
    ResponseEntity<StudentModel[]> response =
        restTemplate.getForEntity(
            "http://localhost:8080/rest/student/viewall", StudentModel[].class);
    StudentModel[] students = response.getBody();
    return Arrays.asList(students);
}

```

ResponseEntity digunakan pada method ini untuk memanggil method rest yang ada pada package Tutorial07Producer dengan tipe panggilan berupa array dari StudentModel. Setelah itu, response entity yang didapat digunakan untuk mengambil body dari response tersebut, yang tentunya masih dalam bentuk array. Pada baris return, variabel StudentModel yang masih dalam bentuk array dikonversi menjadi tipe data List<StudentModel>.

Latihan 4

Sukrian Syahnel Putra

1306402066

ADPAP - B

Karena permintaan instruksi tutorial, maka ada kelas-kelas baru yang terbentuk yang didedikasikan untuk Course. Selain adanya CourseDAO, CourseDAOImpl yang mengimplementasi CourseDAO, dan CourseServiceRest, terdapat kelas-kelas baru seperti CourseService dan CourseController. CourseController berisi method-method yang dikhususkan untuk service yang berkaitan dengan course. Sementara itu, CourseService berbentuk interface yang berisi method-method yang berkaitan dengan course. Interface inilah yang diimplementasi oleh CourseServiceRest.

Isi dari CourseDAOImpl adalah sebagai berikut.

```
CourseDAOImpl.java CourseServiceRest.java
1 package com.example.dao;
2
3 import java.util.Arrays;
4 import java.util.List;
5
6 import org.springframework.beans.factory.annotation.Autowired;
7 import org.springframework.http.ResponseEntity;
8 import org.springframework.stereotype.Service;
9 import org.springframework.web.client.RestTemplate;
10
11 import com.example.model.CourseModel;
12
13 @Service
14 public class CourseDAOImpl implements CourseDAO {
15
16     @Autowired
17     private RestTemplate restTemplate;
18
19     @Override
20     public CourseModel selectCourse(String id) {
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         ResponseEntity<CourseModel[]> response =
31             restTemplate.getForEntity(
32                 "http://localhost:8080/rest/course/viewall", CourseModel[].class);
33         CourseModel[] courses = response.getBody();
34         return Arrays.asList(courses);
35     }
36
37 }
```

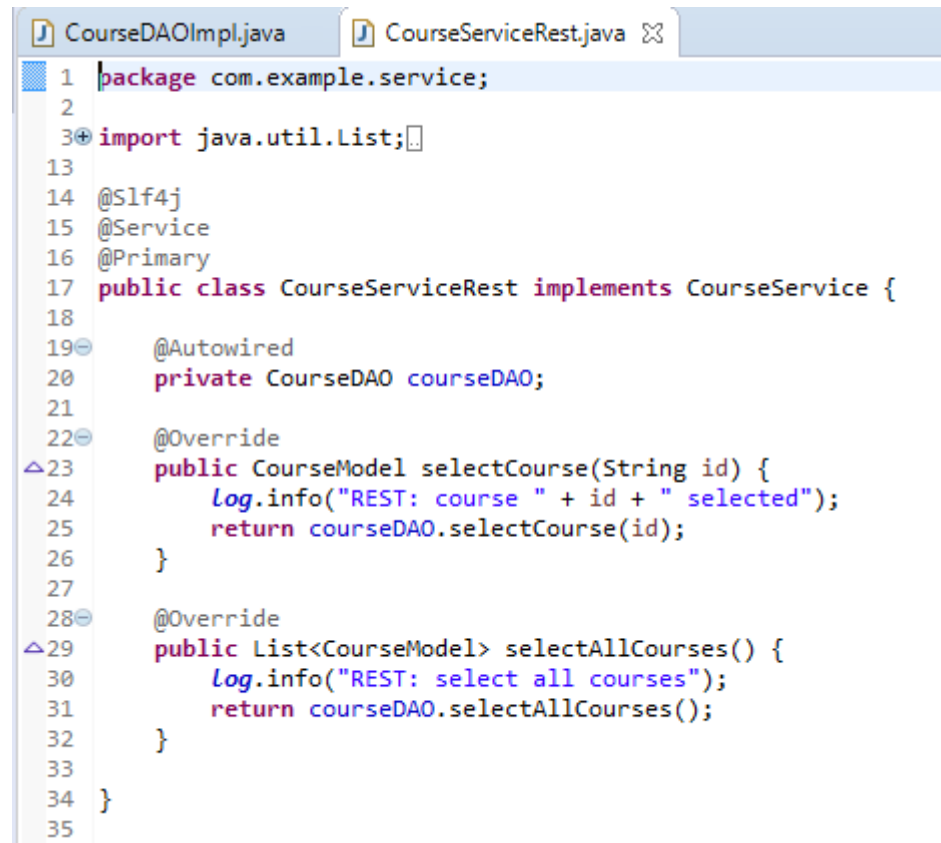
Terdapat dua method yang diimplementasi pada kelas ini, yaitu selectCourse dan selectAllCourses. Prosesnya kurang lebih sama dengan yang ada pada StudentDAOImpl, hanya saja berkaitan dengan CourseModel ketimbang StudentModel.

Sementara itu, berikut isi dari kelas CourseServiceRest

Sukrian Syahnel Putra

1306402066

ADPAP - B



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


Isi dari kelas ini juga kurang lebih mengikuti apa yang ada di kelas StudentServiceRest dengan perbedaan kelas ini mengimplemetasi DAO dari Course ketimbang Student.

Untuk pengujian, pada CourseController diimplementasi method-method sebagai berikut.

Sukrian Syahnel Putra

1306402066

ADPAP - B



```
1 package com.example.controller;
2
3 import java.util.List;
4
5 @Controller
6 public class CourseController {
7     @Autowired
8     CourseService courseDAO;
9
10    @RequestMapping("/course/view/{id}")
11    public String viewCourse (Model model, @PathVariable(value = "id") String id) {
12        CourseModel course = courseDAO.selectCourse(id);
13        if (course != null) {
14            model.addAttribute("course", course);
15            return "view-course";
16        }
17        model.addAttribute("id", id);
18        return "not-found-course";
19    }
20
21    @RequestMapping("/course/viewall")
22    public String viewAllCourses (Model model) {
23        List<CourseModel> courses = courseDAO.selectAllCourses();
24        model.addAttribute("courses", courses);
25        return "viewall-courses";
26    }
27 }
```

Method-method ini menirukan counterpart mereka pada StudentController, hanya saja mereka menggunakan CourseService ketimbang Student Service.

Terdapat template baru pada resources/templates yang digunakan untuk mengujikan method view all pada course, yaitu viewall-courses. View ini menampilkan seluruh data dari course berikut NPM dan nama student yang mengambil mata kuliah tersebut.