Tutorial 7

Membuat Web Service Menggunakan Spring Boot Framework

Write-up:

Pada tutorial ke-7 kali ini, saya mempelajari mengenai bagaimana pengimplementasian *web service* pada Spring. Implementasi yang dilakukan yaitu dengan memisahkan tampilan dari dua *layer*, yaitu antara *layer backend* dan *layer frontend*. Saya mempelajari bahwa tidak seperti *layer backend*, *layer frontend* tidak perlu mengakses ke database untuk menampilkan data yang ada. Cara lain yang dilakukan yaitu hanya dengan mengambil objek-objek yang diinginkan dari *layer backend*. Sehingga, frekuensi akses ke database untuk melakukan pengambilan data dapat menjadi lebih sedikit dari sebelumnya.

LATIHAN 1

1. Method viewAll pada kelas StudentRestController.java

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

Method ini berfungsi untuk mengembalikan daftar dari objek *student* yang ada di database, yang didapat dengan memanggil method selectAllStudents, yang mana dikembalikan dalam bentuk objek List. Namun, pada tampilan *view* nantinya, objek-objek tersebut dikembalikan dengan format JSON.

LATIHAN 2

1. Method view pada kelas CourseRestController.java

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

Method ini berfungsi untuk mengembalikan objek *course* dengan id course tertentu yang berada di database. Objek tersebut dicari dengan memanggil method selectCourse dan dikembalikan dalam bentuk objek CourseModel dan pada halaman web ditampilkan dengan format JSON.

2. Method viewAll pada kelas CourseRestController.java

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

Method ini berfungsi untuk mengembalikan daftar dari objek *course* yang ada di database, yang didapat dengan memanggil method selectAllCourses, yang mana dikembalikan dalam bentuk objek List. Namun, pada tampilan *view* nantinya, objek-objek tersebut dikembalikan dengan format JSON.

LATIHAN 3

1. Method selectAllStudents pada kelas StudentServiceRest.java

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

Method ini berfungsi untuk mengembalikan daftar dari objek-objek *student* yang akan diambil dari kelas StudentDAO, yang dilakukan dengan memanggil method selectAllStudents.

2. Method selectAllStudents pada kelas StudentDAOImpl.java

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

Method ini berfungsi untuk mengambil daftar objek-objek *student* yang ada pada halaman *producer* yaitu pada halaman “http://localhost:8080/rest/student/viewall” dan nantinya objek-objek tersebut akan ditampilkan pada halaman *consumer*.

LATIHAN 4

1. Method selectCourse pada kelas CourseServiceRest.java

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

Method ini berfungsi untuk mengembalikan objek *course* dengan id course tertentu yang akan diambil dari kelas CourseDAO, yang dilakukan dengan memanggil method selectCourse.

2. Method selectAllCourses pada kelas CourseServiceRest.java

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

Method ini berfungsi untuk mengembalikan daftar dari objek-objek *course* yang akan diambil dari kelas CourseDAO, yang dilakukan dengan memanggil method selectAllCourses.

3. Method selectCourse pada kelas CourseDAOImpl.java

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

Method ini berfungsi untuk mengambil objek *course* yang ada pada halaman *producer* yaotu pada halaman “http://localhost:8080/rest/course/view/{id_course}” dan nantinya objek tersebut akan ditampilkan pada halaman *consumer*.

4. Method selectAllCourses pada kelas CourseDAOImpl.java

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

Method ini berfungsi untuk mengambil daftar objek-objek *course* yang ada pada halaman *producer* yaitu pada halaman “http://localhost:8080/rest/course/viewall” dan nantinya objek-objek tersebut akan ditampilkan pada halaman *consumer*.