

Tutorial 3

Arsitektur dan Pemrograman Aplikasi Perusahaan

Menggunakan Model dan Service pada Project Spring Boot

A. Ringkasan Materi

Di tutorial 3 ini saya telah mempelajari mengenai penggunaan interface dalam MVC, class Model pada MVC, looping pada view, bagaimana data pada Springboot disimpan, yaitu data didapat melalui path variable dan request parameter, dan juga telah mempraktekkan penggunaan java util List.

B. Jawaban Poin Tutorial

Membuat Service

1. Klik kanan pada Project > New > Package. Buatlah package com.example.tutorial3.service
2. Buatlah interface StudentService.java pada package tersebut dengan isi sebagai berikut:

```
package com.example.tutorial3.service;

import java.util.List;

import com.example.tutorial3.model.StudentModel;

public interface StudentService {
    StudentModel selectStudent(String npm);

    List<StudentModel> selectAllStudents();

    void addStudent(StudentModel student);
}
```

3. Pada package yang sama, buat class InMemoryStudentService yang meng-implements StudentService dengan isi sebagai berikut:

Ratri Ayu
1506689276
C

```
package com.example.tutorial3.service;

import java.util.ArrayList;
import java.util.List;
import com.example.tutorial3.model.StudentModel;

public class InMemoryStudentService implements StudentService {
    private static List<StudentModel> studentList = new ArrayList<StudentModel>();

    @Override
    public StudentModel selectStudent(String npm) {

        // Implement
        return null;
    }

    @Override
    public List<StudentModel> selectAllStudents() {
        return studentList;
    }

    @Override
    public void addStudent(StudentModel student) {
        studentList.add(student);
    }
}
```

4. Implementasikan method selectStudent! Method ini menerima NPM mahasiswa dan mengembalikan object Student dengan NPM tersebut. Return null jika tidak ditemukan

Implementasi method selectStudent:

```
@Override
public StudentModel selectStudent(String npm) {
    for(int i=0; i<studentList.size(); i++) {
        if(studentList.get(i).getNpm().equalsIgnoreCase(npm)) {
            return studentList.get(i);
        }
    }
    return null;
}
```

Method tersebut mengiterasikan arraylist studentList untuk mencari studentModel dengan npm yang sama dengan parameter npm. Jika ada npm yang sama, maka akan direturn dari mahasiswa yang didapat, jika tidak berarti tidak ada mahasiswa tersebut (null).

Membuat Controller dan Fungsi Add

1. Klik kanan pada Project > New > Package. Buatlah package com.example.tutorial3.controller
2. Tambahkan method add yang menerima parameter name, npm, dan gpa dengan menggunakan request method GET. Buatlah class StudentController dengan isi sebagai berikut:

Ratri Ayu
1506689276
C

```
package com.example.tutorial3.controller;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import com.example.tutorial3.service.InMemoryStudentService;
import com.example.tutorial3.service.StudentService;
import com.example.tutorial3.model.StudentModel;

@Controller
public class StudentController {
    private final StudentService studentService;

    public StudentController() {
        studentService = new InMemoryStudentService();
    }

    @RequestMapping("/student/add")
    public String add(@RequestParam(value = "npm", required = true) String npm,
                     @RequestParam(value = "name", required = true) String name,
                     @RequestParam(value = "gpa", required = true) double gpa) {
        StudentModel student = new StudentModel(npm, name, gpa);
        studentService.addStudent(student);
        return "add";
    }
}
```

3. Selanjutnya pada directory resources/templates tambahkan add.html dengan isi sebagai berikut:

```
<html>
<head>
<title>Add</title>
</head>
<body>
    <h2>Data berhasil ditambahkan</h2>
</body>
</html>
```

4. Jalankan program dan buka

localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43

Pertanyaan 1: apakah hasilnya? Jika error, tuliskan penjelasan Anda.

Hasilnya adalah Data berhasil ditambahkan

← → ↻ ⓘ localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43

Data berhasil ditambahkan

localhost:8080/student/add?npm=12345&name=chanek

Pertanyaan 2: apakah hasilnya? Jika error, tuliskan penjelasan Anda.

← → ↻ ⓘ localhost:8080/student/add?npm=12345&name=chanek

Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Fri Sep 22 16:53:28 ICT 2017

There was an unexpected error (type=Bad Request, status=400).

Required double parameter 'gpa' is not present

Muncul Whitelabel Error Page (type=Bad Request, status=400), karena tidak ada parameter gpa yang seharusnya dibutuhkan.

Method View by NPM

1. Pada class StudentController tambahkan method berikut:

```
@RequestMapping("/student/view")
public String view(Model model, @RequestParam(value = "npm", required = true)
String npm) {
    StudentModel student = studentService.selectStudent(npm);
    model.addAttribute("student", student);
    return "view";
}
```

2. Selanjutnya pada directory resources/templates tambahkan view.html dengan isi sebagai berikut:

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<title>View Student by NPM</title>
</head>
<body>
<h3 th:text="'NPM = ' + ${student.npm}">Student NPM</h3>
<h3 th:text="'Name = ' + ${student.name}">Student Name</h3>
<h3 th:text="'GPA = ' + ${student.gpa}">Student GPA</h3>
</body>
</html>
```

3. Jalankan program dan buka

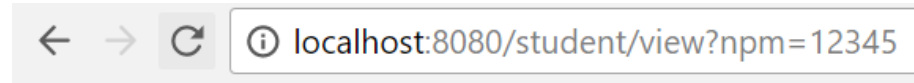
localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43

← → ↻ ⓘ localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43

Data berhasil ditambahkan

lalu buka localhost:8080/student/view?npm=12345

Ratri Ayu
1506689276
C



NPM = 12345

Name = chanek

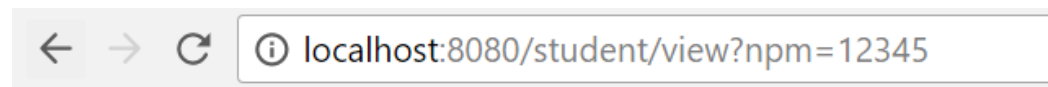
GPA = 3.43

Pertanyaan 3: apakah data Student tersebut muncul? Jika tidak, mengapa?

Ya, data Student tersebut muncul.

4. Coba matikan program dan jalankan kembali serta buka
localhost:8080/student/view?npm=12345

Pertanyaan 4: apakah data Student tersebut muncul? Jika tidak, mengapa?



Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

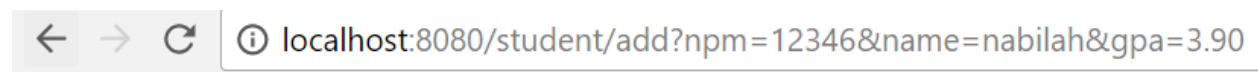
Fri Sep 22 17:44:56 ICT 2017

There was an unexpected error (type=Internal Server Error, status=500).

Exception evaluating SpringEL expression: "student.npm" (view:7)

Tidak Muncul. Karena data disimpan pada setiap *session*, sedangkan session akan diperbaharui setiap kali *springboot* dijalankan, sehingga data yang sebelumnya tidak akan disave pada *session* baru.

5. Coba tambahkan data Student lainnya dengan NPM yang berbeda.



Data berhasil ditambahkan

Method View All

1. Pada class StudentController tambahkan method berikut

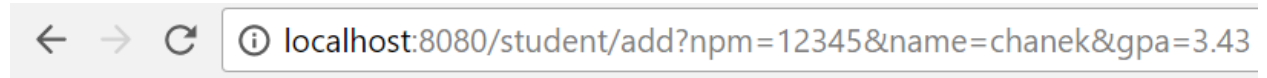
```
@RequestMapping("/student/viewall")
public String viewAll(Model model) {
    List<StudentModel> students = studentService.selectAllStudents();
    model.addAttribute("students", students);
    return "viewall";
}
```

2. Selanjutnya pada directory resources/templates tambahkan viewall.html dengan isi sebagai berikut:

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<title>View All Students</title>
</head>
<body>
<div th:each="student, iterationStatus: ${students}">
<h3 th:text="'No. ' + ${iterationStatus.count}">No. 1</h3>
<h3 th:text="'NPM = ' + ${student.npm}">Student NPM</h3>
<h3 th:text="'Name = ' + ${student.name}">Student Name</h3>
<h3 th:text="'GPA = ' + ${student.gpa}">Student GPA</h3>
<hr/>
</div>
</body>
</html>
```

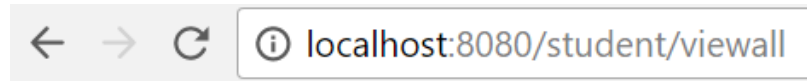
3. Jalankan program dan buka

localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43



Data berhasil ditambahkan

lalu buka localhost:8080/student/viewall



No. 1

NPM = 12345

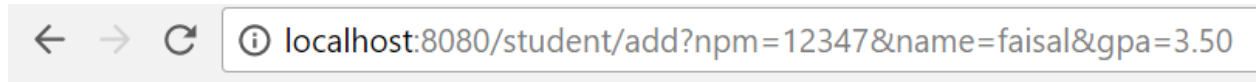
Name = chanek

GPA = 3.43

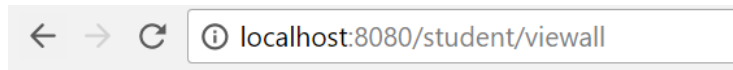
Pertanyaan 5: apakah data Student tersebut muncul?

Ya, data Student muncul.

4. Coba tambahkan data Student lainnya dengan NPM yang berbeda, lalu buka localhost:8080/student/viewall,



Data berhasil ditambahkan



No. 1

NPM = 12345

Name = chanek

GPA = 3.43

No. 2

NPM = 12345

Name = chanek

GPA = 3.43

No. 3

NPM = 12347

Name = faisal

GPA = 3.5

Pertanyaan 6: Apakah semua data Student muncul?
Ya, semua data Student muncul.

C. Implementasi Method selectStudent

```
@Override
public StudentModel selectStudent(String npm) {
    for(int i=0; i<studentList.size(); i++) {
        if(studentList.get(i).getNpm().equalsIgnoreCase(npm)) {
            return studentList.get(i);
        }
    }
    return null;
}
```

D. Latihan

1. Pada StudentController tambahkan sebuah method view Student dengan menggunakan Path Variable.

Tahap-tahap:

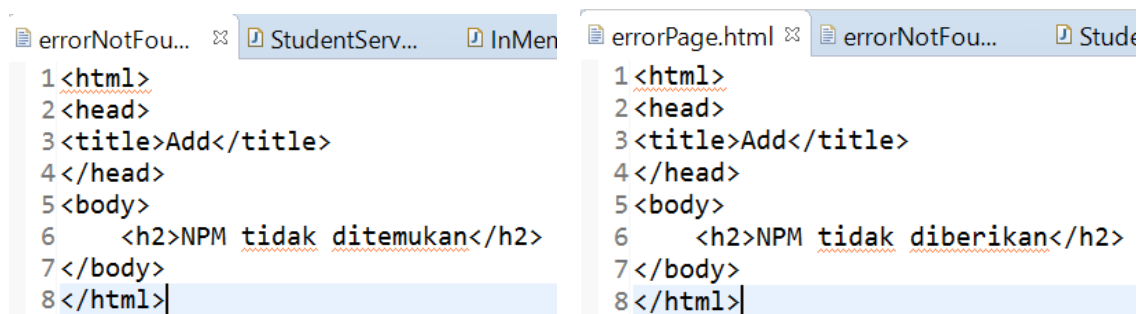
Menggunakan request mapping /student/view dan /student/view/{npm} untuk membuat method pada controller

```
@RequestMapping(value = {"/student/view", "/student/view/{npm}"})
public String studentPath(@PathVariable Optional<String> npm, Model model) {
    if (npm.isPresent()) {
        StudentModel student = studentService.selectStudent(npm.get());
        if(student == null) {
            return "errorNotFound";
        }
        else {
            model.addAttribute("name", student.getName());
            model.addAttribute("npm", student.getNpm());
            model.addAttribute("gpa", student.getGpa());
            return "view";
        }
    } else {
        return "errorPage";
    }
}
```

Kemudian, membuat file html baru untuk page error. Page error akan menampilkan: “NPM tidak ditemukan” jika tidak ada mahasiswa yang memiliki NPM yang sama dengan NPM yang dicari (dalam parameter)

“NPM tidak diberikan” jika tidak terdapat parameter NPM

Terdapat dua file html page error yaitu errorNotFound untuk NPM tidak ditemukan dan errorPage untuk NPM tidak diberikan.



```
1<html>
2<head>
3<title>Add</title>
4</head>
5<body>
6    <h2>NPM tidak ditemukan</h2>
7</body>
8</html>

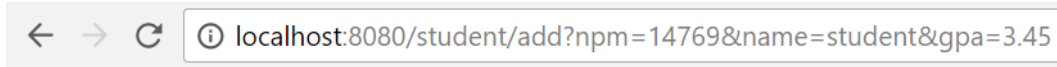
1<html>
2<head>
3<title>Add</title>
4</head>
5<body>
6    <h2>NPM tidak diberikan</h2>
7</body>
8</html>
```

Lalu mengimplementasikan untuk kondisi error dan kondisi sukses (NPM ditemukan) pada method controller


```
@RequestMapping(value = {"/student/view", "/student/view/{npm}"})  
public String studentPath(@PathVariable Optional<String> npm, Model model) {  
    if (npm.isPresent()) {  
        StudentModel student = studentService.selectStudent(npm.get());  
        if(student == null) {  
            return "errorNotFound";  
        }  
        else {  
            model.addAttribute("name", student.getName());  
            model.addAttribute("npm", student.getNpm());  
            model.addAttribute("gpa", student.getGpa());  
            return "view";  
        }  
    } else {  
        return "errorPage";  
    }  
}
```

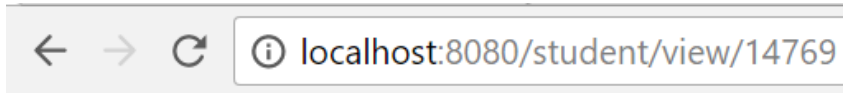
Misalnya, Anda ingin memasukkan data seorang Student yang memiliki NPM 14769, untuk melihat data yang baru dimasukkan tersebut dapat mengakses halaman localhost:8080/student/view/14769. Jika nomor NPM tidak diberikan atau tidak ditemukan kembalikan halaman error yang berisi informasi bahwa nomor NPM kosong atau tidak ditemukan.

Penambahan mahasiswa dengan NPM 14769



Data berhasil ditambahkan

Melihat data yang baru dimasukkan dengan mengakses halaman localhost:8080/student/view/14769

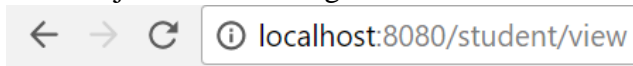


NPM = 14769

Name = student

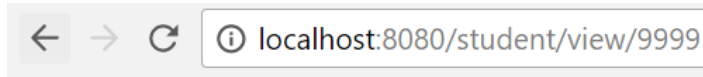
GPA = 3.45

Kondisi jika NPM kosong



NPM tidak diberikan

Kondisi jika NPM tidak ditemukan



2. Tambahkan fitur untuk melakukan delete Student berdasarkan NPM. Tampilkan sebuah halaman yang memberikan informasi bahwa data tersebut telah berhasil dihapus.

Tahap-tahap:

Membuat fitur delete dengan method Path Variable yang serupa

```
@RequestMapping(value = {"/student/delete", "/student/delete/{npm}"})
public String studentDelete(@PathVariable Optional<String> npm, Model model) {
    if (npm.isPresent()) {
        StudentModel student = studentService.selectStudent(npm.get());
        if (student == null) {
            return "errorNotFound";
        }
        else {
            StudentModel deletedStudent = studentService.deleteStudent(npm.get());
            model.addAttribute("name", deletedStudent.getName());
            model.addAttribute("npm", deletedStudent.getNpm());
            model.addAttribute("gpa", deletedStudent.getGpa());
            return "deleteView";
        }
    } else {
        return "errorPage";
    }
}
```

Lalu, buat view untuk delete

```
deleteView.html  errorPage.html  errorNotFou...  StudentSer
1<!DOCTYPE html>
2<html xmlns:th="http://www.thymeleaf.org">
3<head>
4<title>View Student by NPM</title>
5</head>
6<body>
7    <h3 th:text="'NPM = ' + ${npm}">Student NPM</h3>
8    <h3>Mahasiswa Berhasil Dihapus</h3>
9</body>
10</html>
```

Kemudian, membuat method deleteStudent(String npm) pada interface, yaitu:

InMemoryStudentService

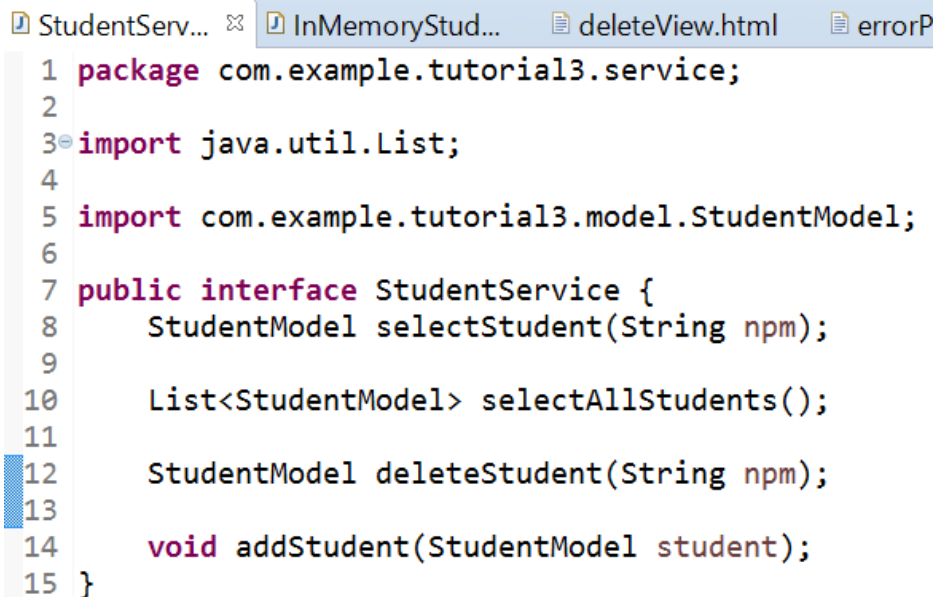
Pada method, akan menghapus student pada studentList sesuai dengan npm student yang ingin dihapus. Jika npm ditemukan, akan dihapus dari list, sedangkan jika tidak ditemukan tidak dilakukan apa-apa.

Ratri Ayu
1506689276
C

```
public StudentModel deleteStudent(String npm) {  
    for(int i=0; i<studentList.size(); i++) {  
        if(studentList.get(i).getNpm().equalsIgnoreCase(npm)) {  
            StudentModel temporary = studentList.get(i);  
            studentList.remove(i);  
            return temporary;  
        }  
    }  
    return null;  
}
```

StudentService

Dibuat parameter npm



```
StudentServ... InMemoryStud... deleteView.html errorP  
1 package com.example.tutorial3.service;  
2  
3 import java.util.List;  
4  
5 import com.example.tutorial3.model.StudentModel;  
6  
7 public interface StudentService {  
8     StudentModel selectStudent(String npm);  
9  
10    List<StudentModel> selectAllStudents();  
11  
12    StudentModel deleteStudent(String npm);  
13  
14    void addStudent(StudentModel student);  
15 }
```

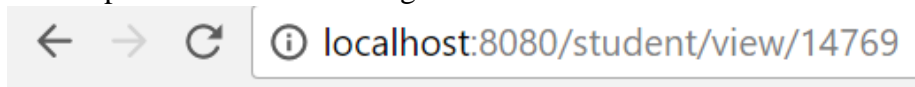
Setelah itu, method pada controller disesuaikan dengan kondisi dari delete terbaru. Jika sukses, method deleteStudent() akan dijalankan dan menggunakan data dari student yang telah didelete.

Ratri Ayu
1506689276
C

```
@RequestMapping(value = {"/student/delete", "/student/delete/{npm}"})  
public String studentDelete(@PathVariable Optional<String> npm, Model model) {  
    if (npm.isPresent()) {  
        StudentModel student = studentService.selectStudent(npm.get());  
        if(student == null) {  
            return "errorNotFound";  
        }  
        else {  
            StudentModel deletedStudent = studentService.deleteStudent(npm.get());  
            model.addAttribute("name", deletedStudent.getName());  
            model.addAttribute("npm", deletedStudent.getNpm());  
            model.addAttribute("gpa", deletedStudent.getGpa());  
            return "deleteView";  
        }  
    } else {  
        return "errorPage";  
    }  
}
```

Misalnya, setelah melakukan add Student pada soal nomor 1, cobalah untuk melakukan delete data tersebut dengan mengakses halaman localhost:8080/student/delete/14769.

Menampilkan data student dengan NPM 14769

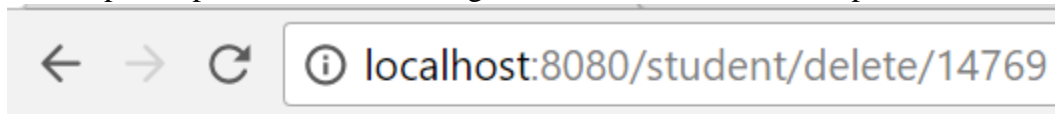


NPM = 14769

Name = student

GPA = 3.45

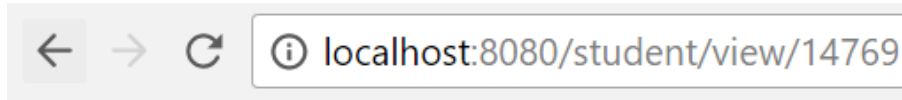
Menampilkan pesan mahasiswa dengan NPM 14769 berhasil dihapus



NPM = 14769

Mahasiswa Berhasil Dihapus

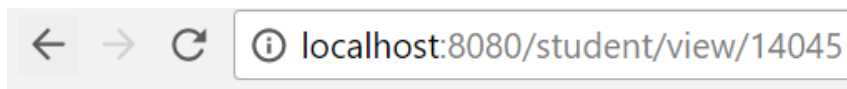
Melakukan pencarian NPM setelah didelete



NPM tidak ditemukan

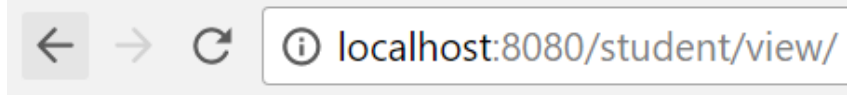
Jika nomor NPM tidak diberikan atau tidak ditemukan kembalikan halaman error yang berisi informasi bahwa nomor NPM kosong atau tidak ditemukan dan proses delete dibatalkan.

Kondisi NPM tidak ditemukan



NPM tidak ditemukan

Kondisi NPM kosong



NPM tidak diberikan