

TUTORIAL

Method selectStudent():

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

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

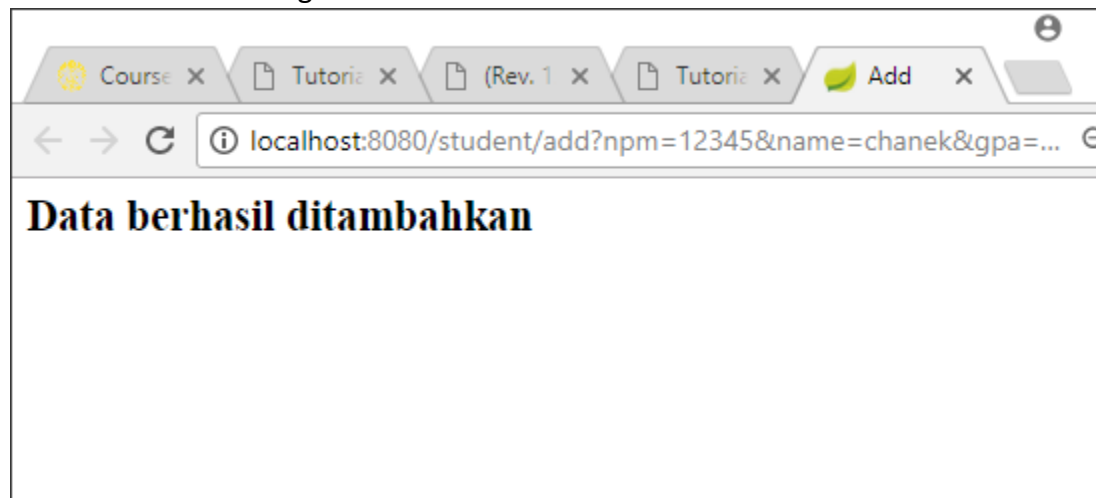
1. Jalankan program dan buka:

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

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

Jawab:

Muncul halaman sebagai berikut

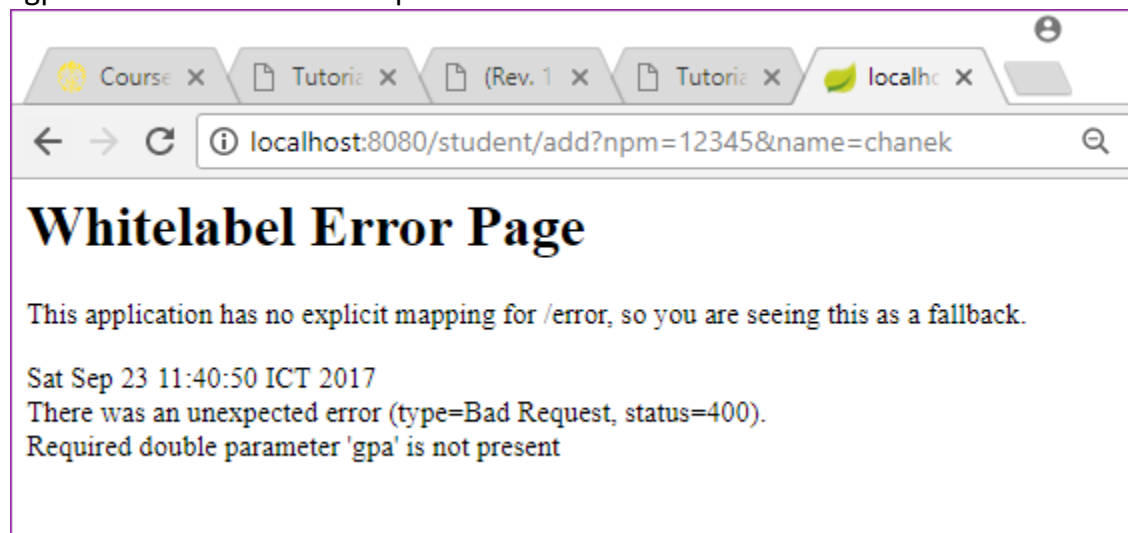


2. localhost:8080/student/add?npm=12345&name=chanek

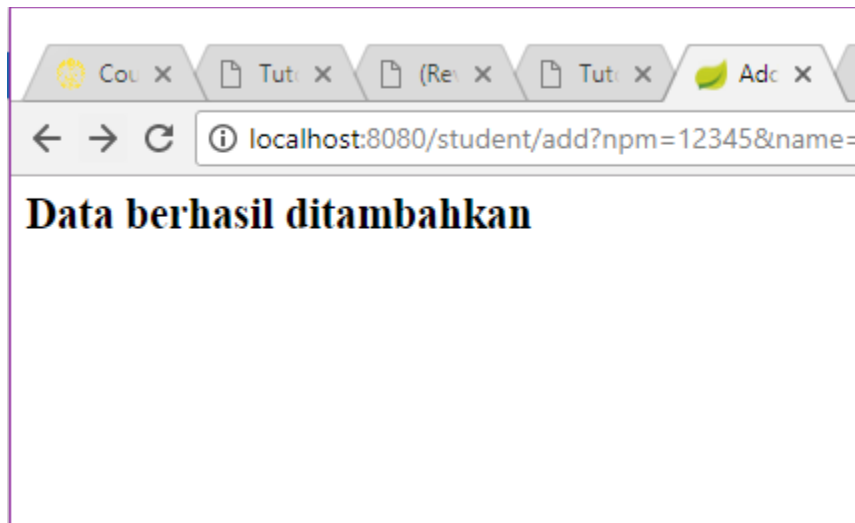
Pertanyaan 2: apakah hasilnya? Jika error, tuliskan penjelasan Anda.

Jawab:

Muncul *whitelabel error page* seperti berikut, karena dibutuhkan parameter "gpa" untuk melakukan RequestParam GET



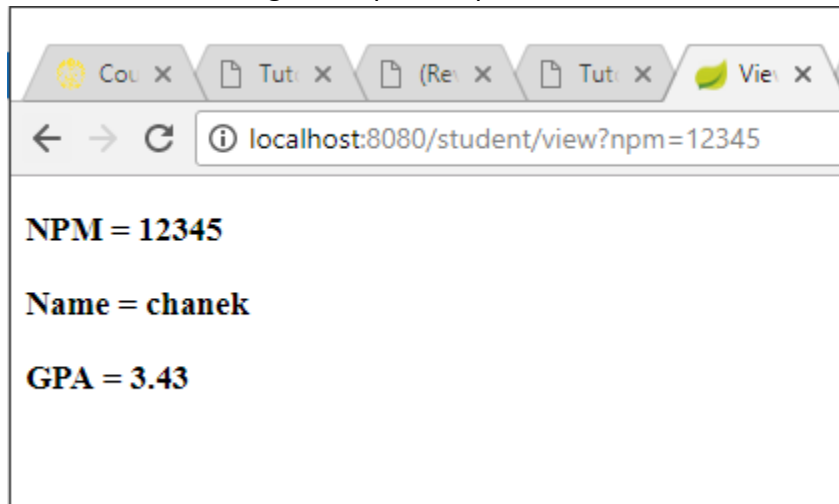
3. Jalankan program dan buka
localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43



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

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

Jawab: muncul, dengan tampilan seperti berikut

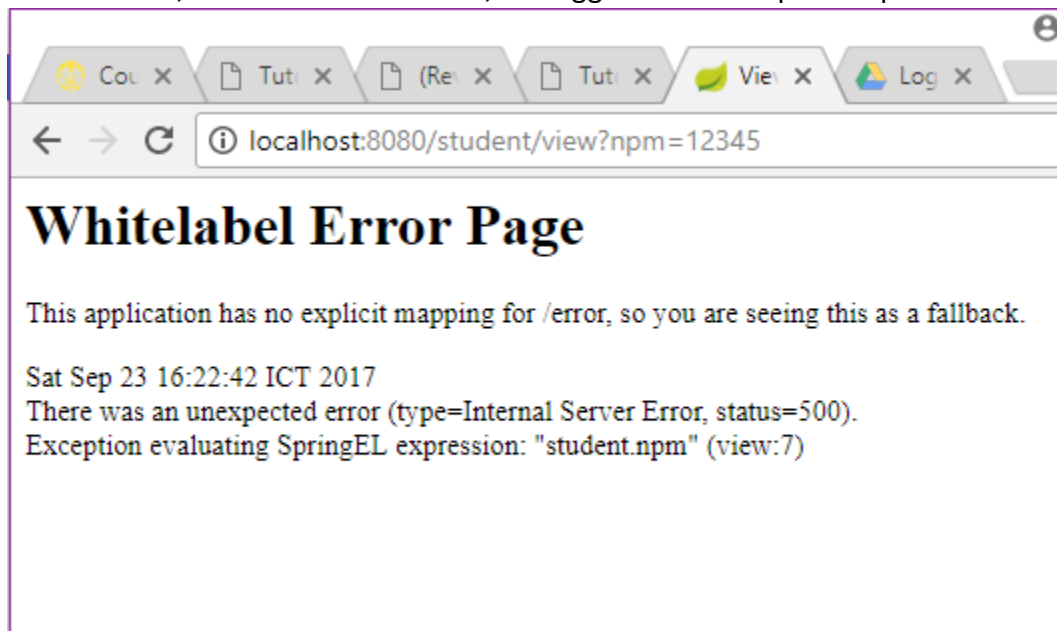


4. Coba matikan program dan jalankan kembali serta buka `localhost:8080/student/view?npm=12345`

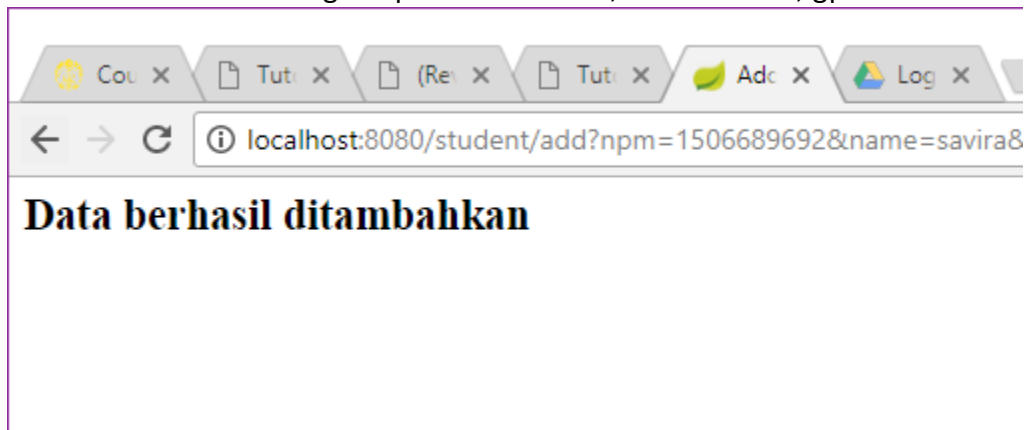
Pertanyaan 4: apakah data Student tersebut muncul? Jika tidak, mengapa?

Jawab: tidak, karena *session* diperbaharui setiap *springboot* di *re-run*, sehingga apabila

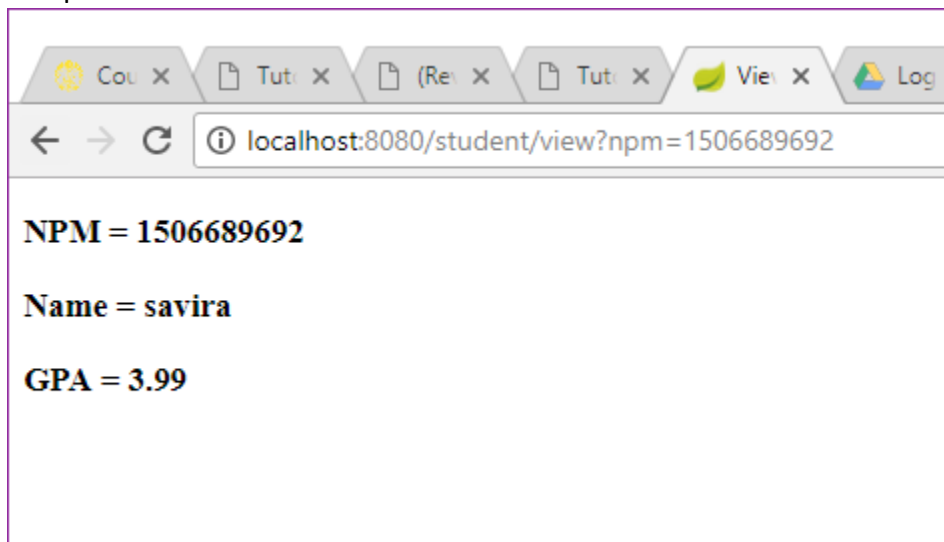
belum di-*add*, data tidak bisa di-*view*, sehingga muncul tampilan seperti berikut



Coba tambahkan data Student lainnya dengan NPM yang berbeda.
menambahkan data dengan npm=1506689595, nama=Savira, gpa=3.99



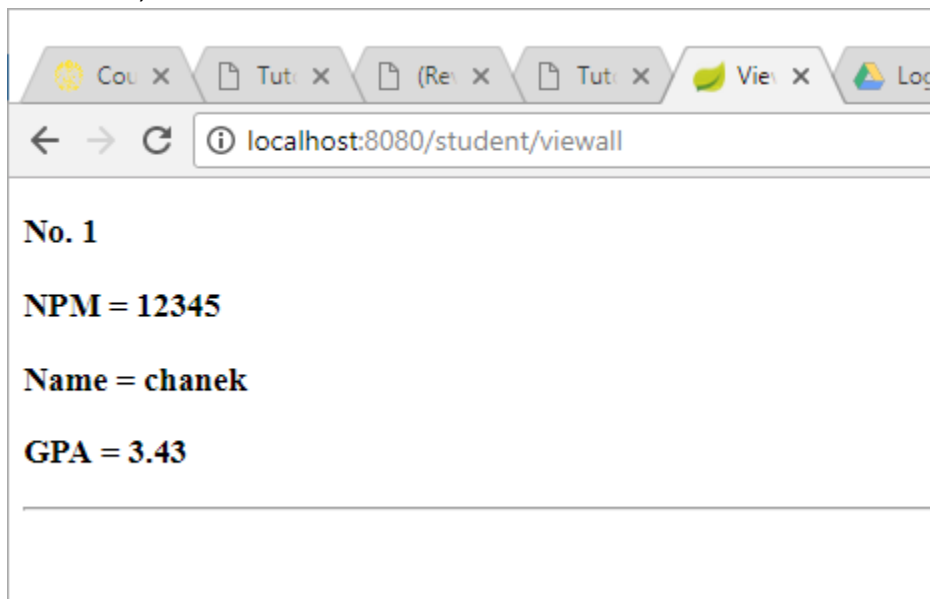
Tampilan ketika di-view



5. localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43 lalu buka localhost:8080/student/viewall,

Pertanyaan 5: apakah data Student tersebut muncul?

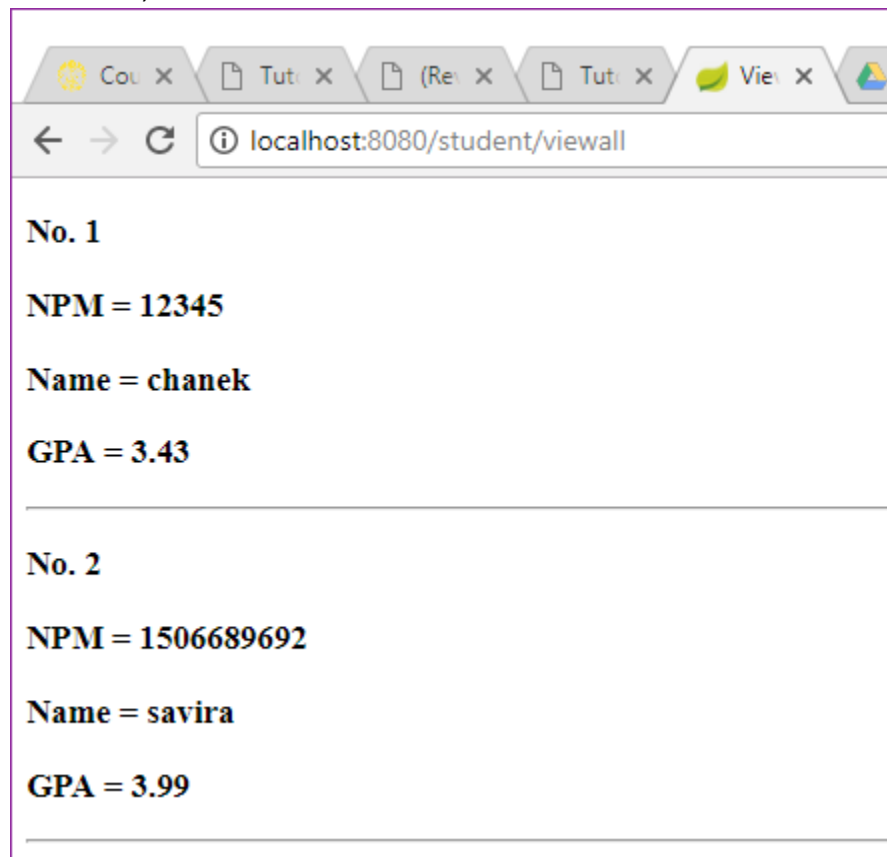
Jawab: Ya, muncul



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

Pertanyaan 6: Apakah semua data Student muncul?

Jawab: Ya, muncul



LATIHAN

1. Membuat method view menggunakan Path Varibale.

Langkah-langkah:

- a. Dalam `StudentController.java` buat method View menggunakan `PathVariable` sebagai berikut

```
@RequestMapping(value = {"/student/view", "student/view/{npm}}")
public String viewPath(@PathVariable Optional<String> npm, Model model) {

    if(npm.isPresent()) {
        StudentModel theStudents = studentService.selectStudent(npm.get());

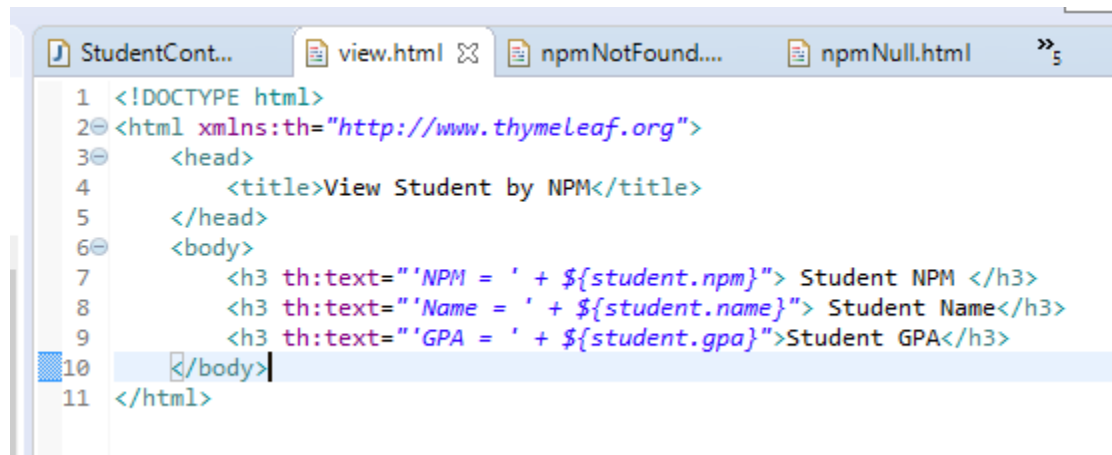
        if(theStudents != null) {
            model.addAttribute("student", theStudents);
            return "view";
        } else {
            return "npmNotFound";
        }

    } else {
        return "npmNull";
    }
}
```

Dalam kode tersebut dapat dilihat bahwa apabila NPM tidak tertera dalam URL akan mengembalikan npmNull.html, sedangkan apabila NPM tertera, akan dicek terlebih dahulu apakah ada atau tidak, apabila ada akan mengembalikan view.html, sedangkan kalau tidak ada akan mengembalikan npmNotFound.html

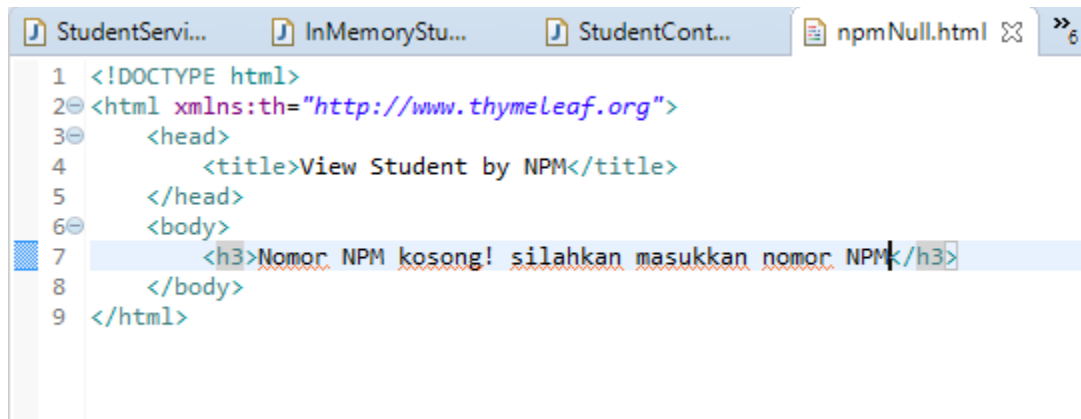
b. Membuat html untuk masing-masing hasil pengembalian

- view.html



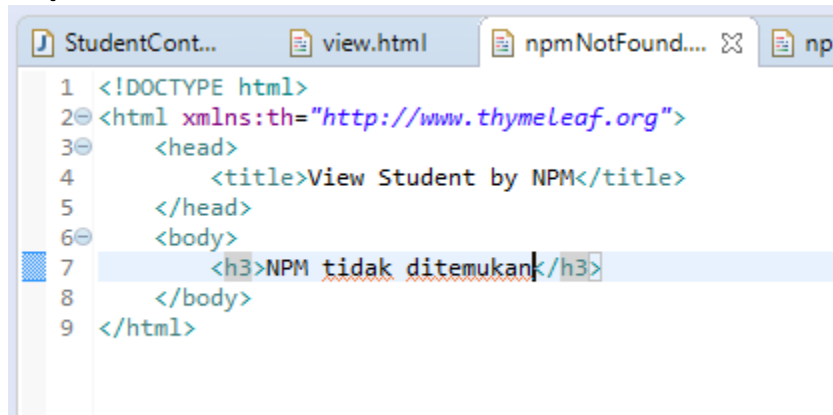
```
1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymeleaf.org">
3   <head>
4     <title>View Student by NPM</title>
5   </head>
6   <body>
7     <h3 th:text="'NPM = ' + ${student.npm}"> Student NPM </h3>
8     <h3 th:text="'Name = ' + ${student.name}"> Student Name</h3>
9     <h3 th:text="'GPA = ' + ${student.gpa}"> Student GPA</h3>
10  </body>
11 </html>
```

- npmNull.html



```
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>Nomor NPM kosong! silahkan masukkan nomor NPM</h3>
8   </body>
9 </html>
```

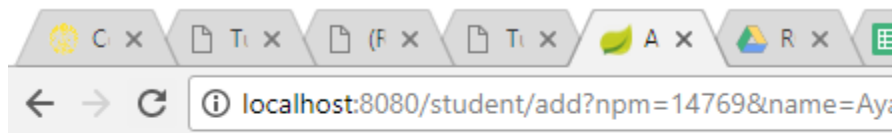
- npmNotFound.html



```
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>NPM tidak ditemukan</h3>
8   </body>
9 </html>
```

c. Setelah melakukan *method add* dengan data npm=14769, name=Ayam, gpa=3.99

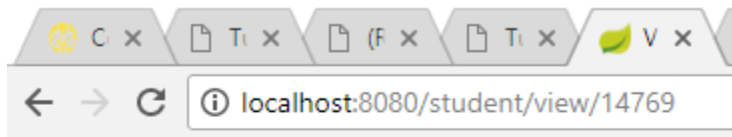
<http://localhost:8080/student/add?npm=14769&name=ayam&gpa=3.99>



Data berhasil ditambahkan

- **Melihat data yang baru dimasukkan menggunakan *Path Variable* dengan memasukkan url sebagai berikut**

<http://localhost:8080/student/view/14769>



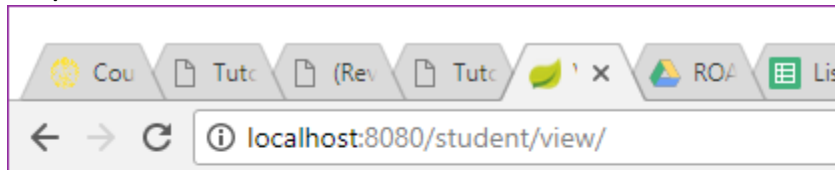
NPM = 14769

Name = Ayam

GPA = 3.99

- **Mencoba melihat data namun tidak menuliskan NPM**

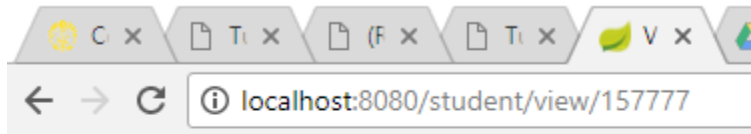
<http://localhost:8080/student/view/>



Nomor NPM kosong! silahkan masukkan nomor NPM

- **Mencoba melihat data dengan NPM yang belum dimasukkan**

<http://localhost:8080/student/view/157777>



NPM tidak ditemukan

2. Membuat fitur untuk *delete student* berdasarkan NPM

Langkah-langkah:

a. Membuat method `deleteStudent(String npm)` dalam `InMemoryStudentService.java` dengan isi sebagai berikut:

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

b. Lalu mencantumkan method `deleteStudent(String npm)` tersebut dalam interface `StudentService` seperti berikut ini

```
4  
5 public interface StudentService {  
6     StudentModel selectStudent(String npm);  
7     List<StudentModel> selectAllStudents();  
8     void addStudent(StudentModel student);  
9     StudentModel deleteStudent(String npm);  
10 }  
11
```

c. Lalu dalam `StudentController.java` tambahkan method `studentDelete`

dengan menggunakan PathVariable, seperti berikut:

```
@RequestMapping(value = {"/student/delete", "/student/delete/{npm}}")
public String studentDelete(@PathVariable Optional<String> npm, Model model) {

    if(npm.isPresent()) {
        StudentModel theStudents = studentService.selectStudent(npm.get());

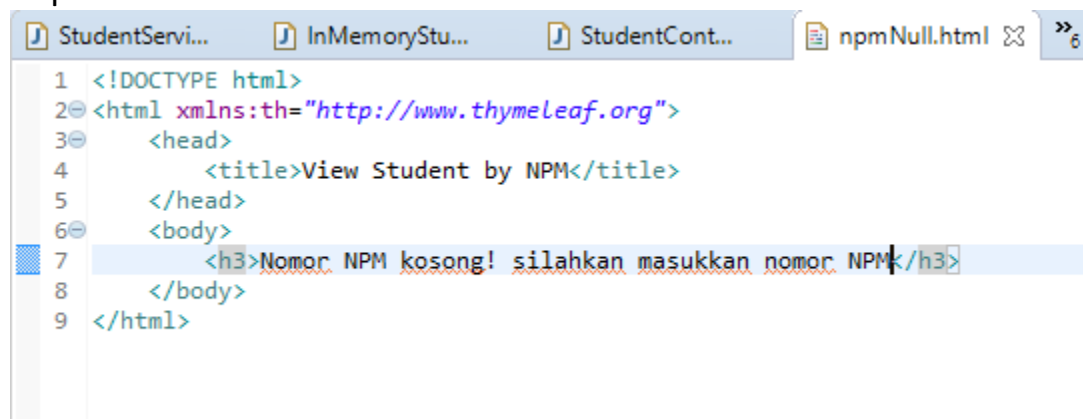
        if(theStudents != null) {
            model.addAttribute("student", theStudents);
            return "delete";
        } else {
            return "npmNotFound";
        }
    }

    } else {
        return "npmNull";
    }
}
```

Untuk masing-masing *condition* diberikan pengembalian masing-masing. Apabila NPM tidak tertera dalam url, maka dikembalikan npmNull.html. Apabila NPM tertera di url, akan dicek apakah ada di dalam List, jika ada akan dikembalikan delete.html, sedangkan jika tidak ada akan dikembalikan npmNotFound.html

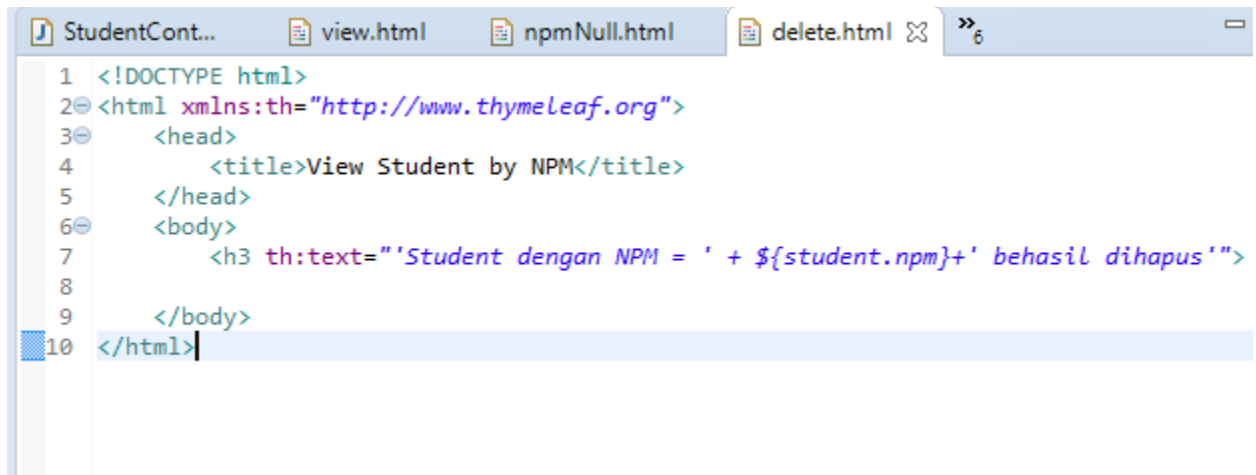
d. Membuat .html untuk masing-masing hasil pengembalian

- npmNull.html



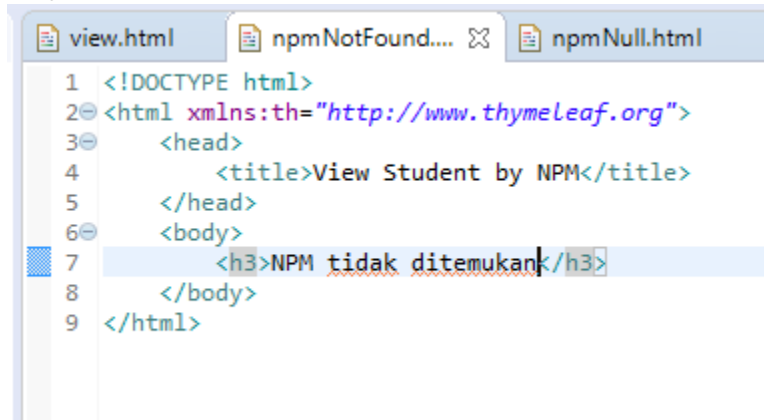
```
1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymeleaf.org">
3   <head>
4     <title>View Student by NPM</title>
5   </head>
6   <body>
7     <h3>Nomor NPM kosong! silahkan masukkan nomor NPM</h3>
8   </body>
9 </html>
```

- delete.html



```
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="'Student dengan NPM = ' + ${student.npm}+' berhasil dihapus'">
8
9   </body>
10 </html>
```

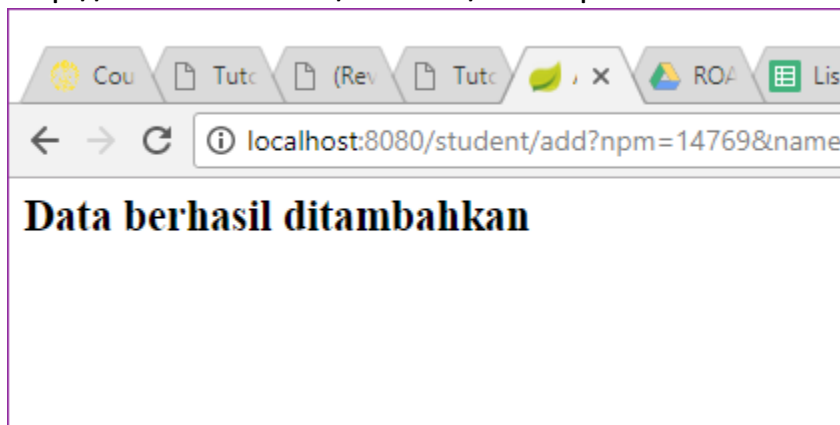
- npmNotFound.html



```
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>NPM tidak ditemukan</h3>
8   </body>
9 </html>
```

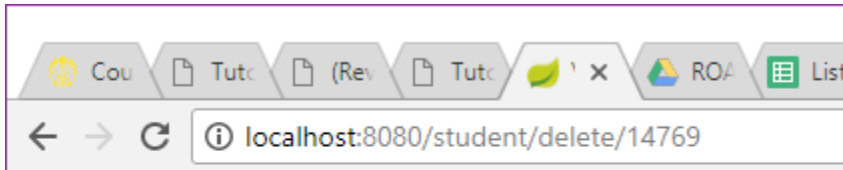
e. Setelah melakukan *method add* dengan data npm=14769, name=Ayam, gpa=3.99

<http://localhost:8080/student/add?npm=14769&name=ayam&gpa=3.99>



- Menghapus data student dengan NPM=14769

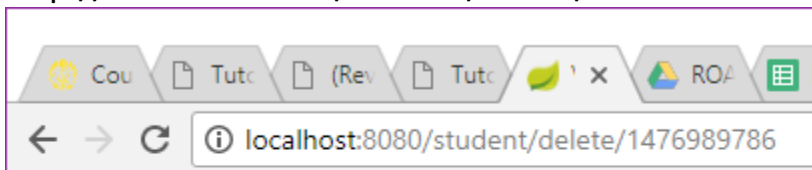
<http://localhost:8080/student/delete/14769>



Student dengan NPM = 14769 berhasil dihapus

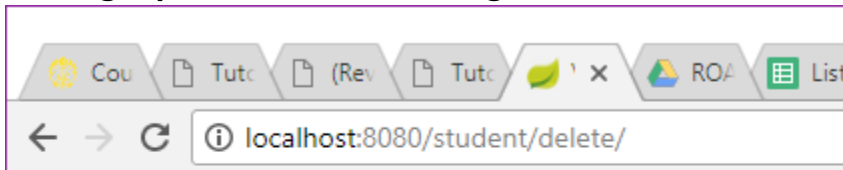
- Menghapus data student dengan NPM yang belum dimasukkan

<http://localhost:8080/student/delete/1476989786>



NPM tidak ditemukan

- Menghapus data student dengan tidak menuliskan NPM



Nomor NPM kosong! silahkan masukkan nomor NPM

Tutorial 3 APAP
Savira Nurul Ahila
1506689692
Kelas B

RINGKASAN

Dalam tutorial 3 ini saya mempelajari bagaimana cara membuat model dengan konsep MVC dalam project Spring Boot dengan cara yang baik dan benar. Saya juga mempelajari cara membuat service dengan fungsi create dan read data melalui konsep MVC itu sendiri. Lalu dengan tutorial ini saya menjadi paham hubungan dan interaksi antara setiap komponen MVC di project Spring Boot ini.