WRITE UP TUTORIAL 3

Ari Tri Wibowo Yudasubrata

NPM : 1506735175

APAP C

a. RINGKASAN MATERI

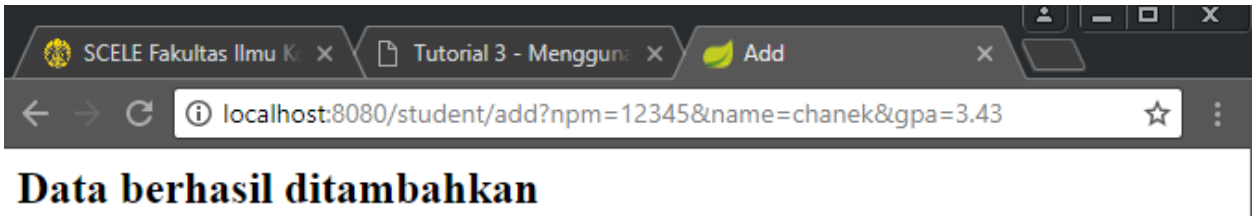
Materi pada tutorial ini adalah menggunakan **model** dan **service** pada Project Spring Boot. Model adalah sebuah objek yang menggambarkan informasi dari suatu hal. Dalam tutorial ini, yang disebut model adalah **Student** dengan atribut nama, npm, dan gpa. Sementara itu service adalah penghubung antara controller dengan database yang mendefinisikan method-method yang dapat dilakukan dalam sebuah class. Sederhananya, dalam tutorial kali ini, dengan adanya service, maka kita bisa melakukan fungsi select, selectAll, add, dan delete pada sebuah struktur data yang menyimpan banyak objek Student.

b. HASIL JAWABAN TUTORIAL

Membuat Controller dan Fungsi Add

Menjalankan program dan membuka
localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43

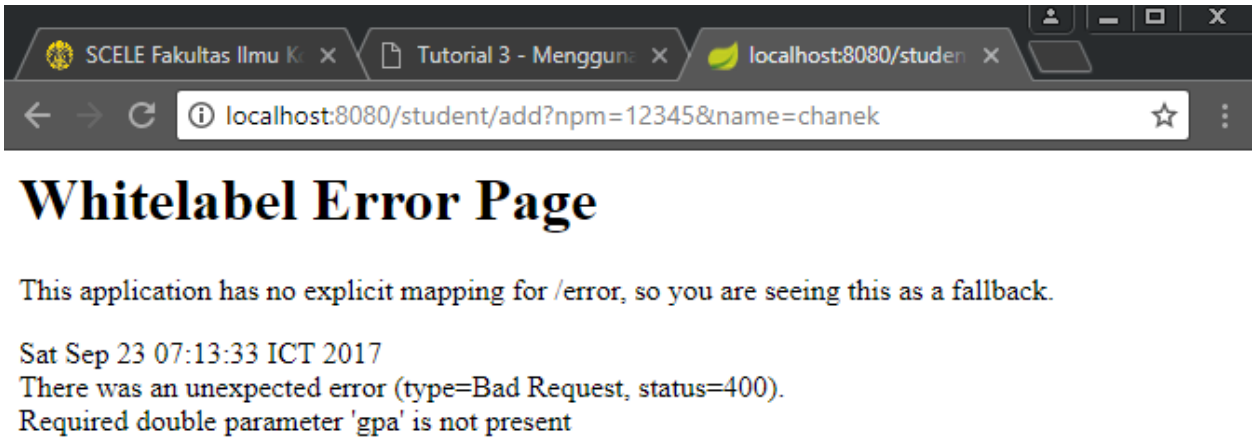
Tidak terjadi error, data berhasil ditambahkan



Menjalankan program dan membuka

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

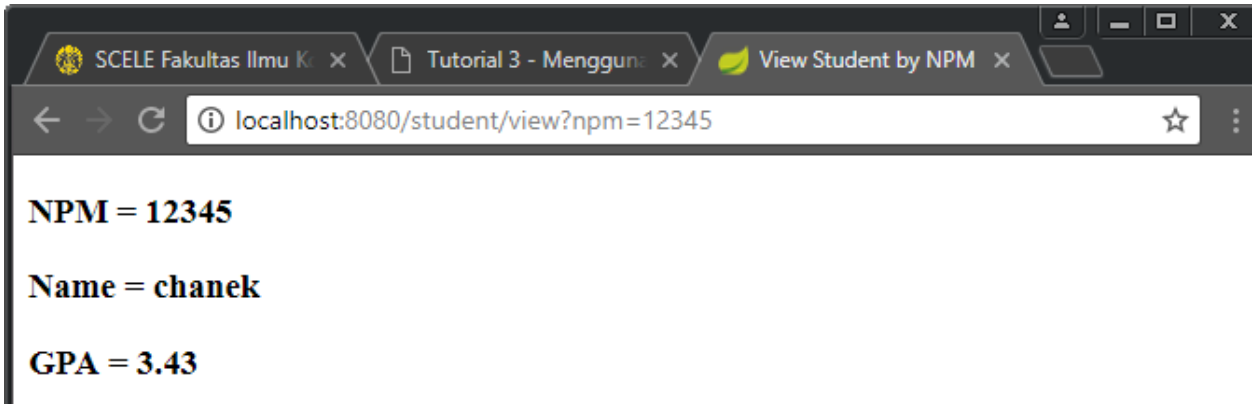
Terjadi error Bad Request, sebab tidak ada parameter gpa yang dimasukkan



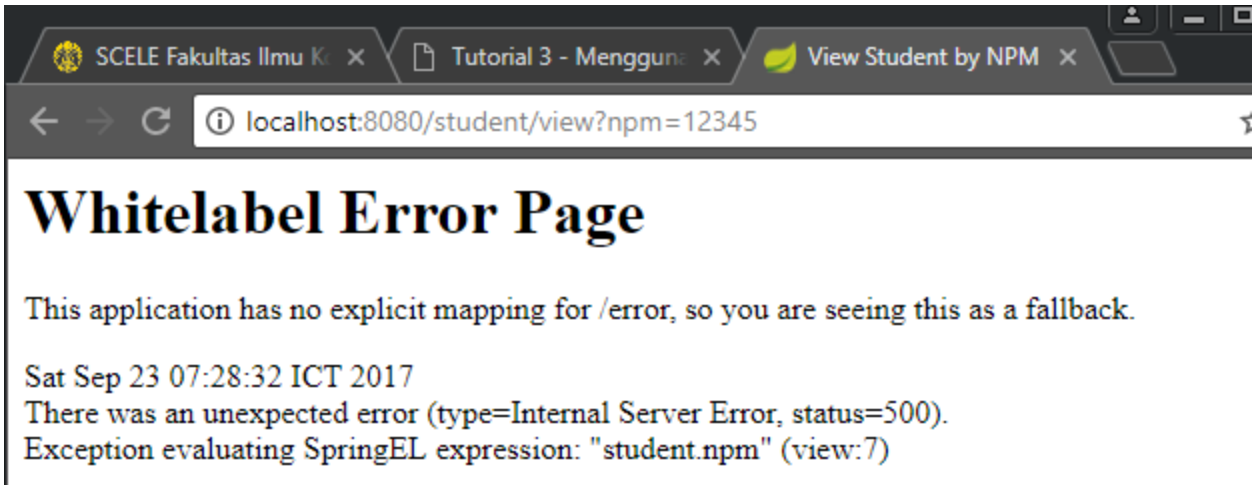
Method View By NPM

Membuka localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43 lalu membuka localhost:8080/student/view?npm=12345

Data student telah berhasil ditambahkan dan muncul



Mematikan program dan membuka kembali localhost:8080/student/view?npm=12345

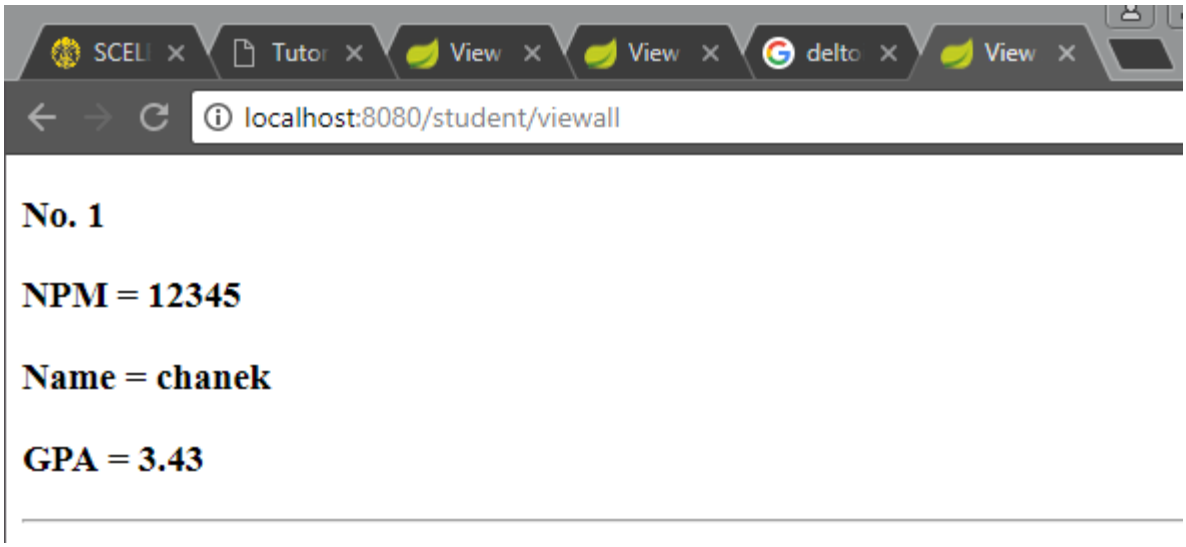


Tidak muncul, terjadi error 500 Server error. Hal ini karena ketika program diberhentikan lalu dijalankan kembali, maka semua data student yang tersimpan akan terhapus sebab ketika program dijalankan kembali, (dalam kelas InMemoryStudent Service) akan dibuat struktur data list ***studentList*** yang baru lagi.

Jika menambahkan data student baru terlebih dahulu, lalu membuka dengan npm data student yang baru itu, maka program akan berhasil menampilkannya

Method View All

Menjalankan program dan membuka localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43 lalu membuka localhost:8080/student/viewall



Menambahkan data student lainnya dengan NPM yang berbeda lalu membuka viewall

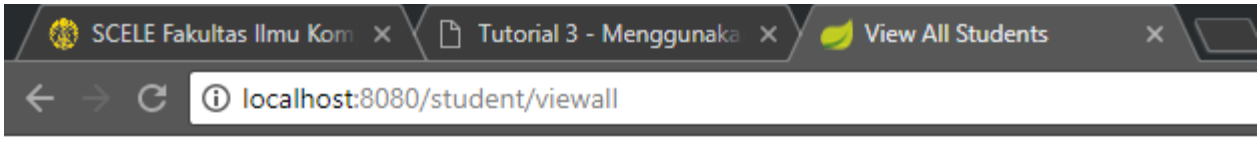
Menambahkan data student baru :

localhost:8080/student/add?npm=12346&name=deltoid&gpa=3.99

localhost:8080/student/add?npm=12347&name=ella&gpa=3.98

localhost:8080/student/add?npm=12348&name=franchoze&gpa=3.97

Menjalankan viewall



No. 1

NPM = 12345

Name = chanek

GPA = 3.43

No. 2

NPM = 12346

Name = deltoid

GPA = 3.99

No. 3

NPM = 12347

Name = ella

GPA = 3.98

No. 4

NPM = 12348

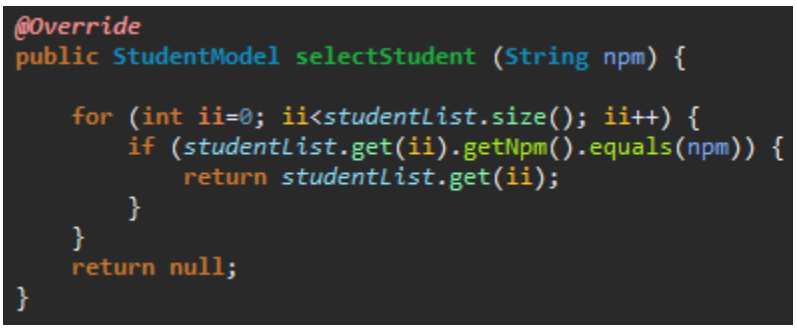
Name = franchoze

GPA = 3.97

Semua data student muncul

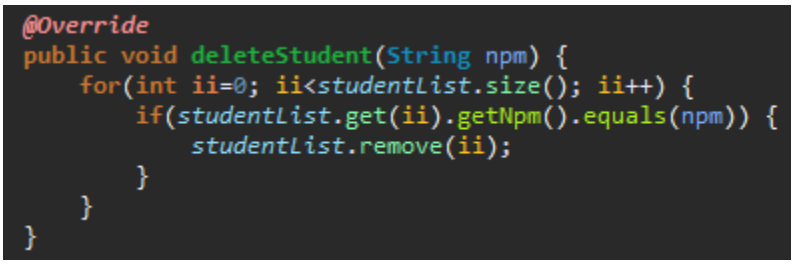
c. Method selectStudent yang diimplementasikan

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



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

d. Penjelasan fitur delete



```
@Override
public void deleteStudent(String npm) {
    for(int ii=0; ii<studentList.size(); ii++) {
        if(studentList.get(ii).getNpm().equals(npm)) {
            studentList.remove(ii);
        }
    }
}
```

Method deleteStudent dibuat pada class InMemoryStudentService yang meng-implement interface StudentService. Methode deleteStudent ini akan menerima parameter berupa npm dalam bentuk string, kemudian akan mengiterasi struktur data berupa ArrayList yang berisikan daftar student. Jika pada indeks tertentu

berhasil menemukan isi list yang npmnya sama dengan input yang diberikan, maka akan dihapus isi arraylist pada indeks tersebut.

```
@RequestMapping("/student/delete/{npm}")
public String delete(Model model, @PathVariable String npm) {
    StudentModel student = studentService.selectStudent(npm);
    model.addAttribute("npmrequest", npm);
    if(student !=null) {
        studentService.deleteStudent(npm);
        return "delete";
    }
        return "notfound";
}
```

Dalam studentController, terdapat method delete yang menerima input dari pathvariable berdasarkan npm. Jika berhasil menemukan npm yang akan di-delete, maka akan mengembalikan halaman yang menandakan proses delete telah sukses, jika tidak ditemukan akan mengembalikan halaman notfound.html yang menginformasikan bahwa npm tidak ditemukan.

Tahapan yang diselesaikan dalam fitur latihan

Membuat method view baru yang menggunakan path variable.

```
@RequestMapping("/student/view/{npm1}")
public String view1(Model model, @PathVariable String npm1) {
    StudentModel student = studentService.selectStudent(npm1);
    model.addAttribute("npmrequest", npm1);
    if(student != null) {
        model.addAttribute("student", student);
        return "view";
    }
        return "notfound";
}
```

Path variable berisikan npm. Dalam method tersebut, akan dicari mahasiswa berdasarkan npm yang sesuai dengan npm pada path variable. Jika berhasil menemukan, maka akan mengembalikan halaman view.html jika gagal maka akan mengembalikan halaman notfound.html (**membuat halaman notfound.html** yang menyatakan bahwa npm kosong atau tidak ditemukan)

```html
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
    <head>
        <title> Error Page </title>
    </head>
    <body>
        <h3 th:text = "'NPM ' + ${npmrequest} + ' kosong atau tidak dapat ditemukan'"></h3>
    </body>
</html>
```

Kemudian untuk melakukan delete Student berdasarkan npm, terlebih dahulu membuat method delete pada class InMemoryStudentService

```java
@Override
public void deleteStudent(String npm) {
    for(int ii=0; ii<studentList.size(); ii++) {
        if(studentList.get(ii).getNpm().equals(npm)) {
            studentList.remove(ii);
        }
    }
}
```

Kemudian dalam interface, tambahkan deleteStudent

```java
public interface StudentService {

    StudentModel selectStudent(String npm);
    List <StudentModel> selectAllStudents();
    void addStudent(StudentModel student);
    void deleteStudent(String npm);

}
```

Lalu dalam StudentController.java, dibuat method delete yang memproses perintah delete dengan pathvariable npm. Apabila berhasil menemukan npm yang akan di-delete, maka akan mengembalikan halaman yang menandakan delete telah berhasil sementara itu jika tidak ditemukan, akan mengembalikan halaman notfound.html

```java
@RequestMapping("/student/delete/{npm}")
public String delete(Model model, @PathVariable String npm) {
    StudentModel student = studentService.selectStudent(npm);
    model.addAttribute("npmrequest", npm);
    if(student !=null) {
        studentService.deleteStudent(npm);
        return "delete";
    }
        return "notfound";
}
```