Menggunakan Model dan Service pada Project Spring Boot

Ringkasan

Model merupakan sebuah objek yang merepresentasikan dan menyimpan informasi terhadap suatu hal. Contoh dari model adalah objek mahasiswa. Objek mahasiswa ini memiliki nama, alamat, tanggal lahir, nomor pokok mahasiswa, dsbnya. Model digunakan untuk merepresentasikan hal-hal tersebut dalam atribut yang dimiliki sebuah model.

Service adalah suatu layer yang menjadi mediator antara controller dan database. Pada service layer, disimpan business logic yang digunakan untuk mengolah data yang terdapat dalam database. Pengolahan ini meliputi kalkulasi data yang diambil dari database, manipulasi user input kedalam database, dsbnya. Contohnya adalah melakukan kalkulasi IPK sebuah model mahasiswa.

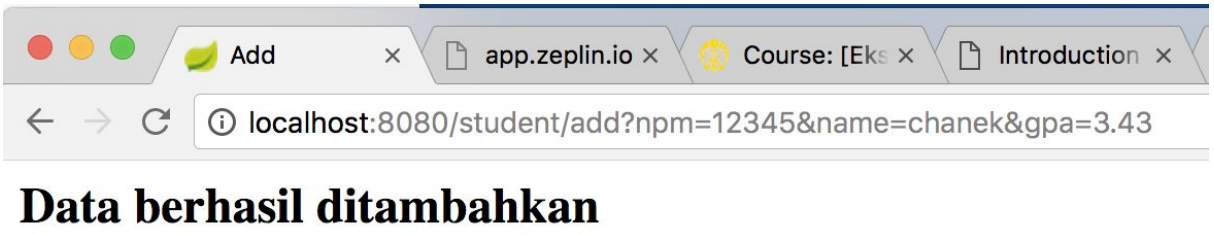
Tutorial

Berikut merupakan implementasi method “selectStudent” pada service studentService:

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

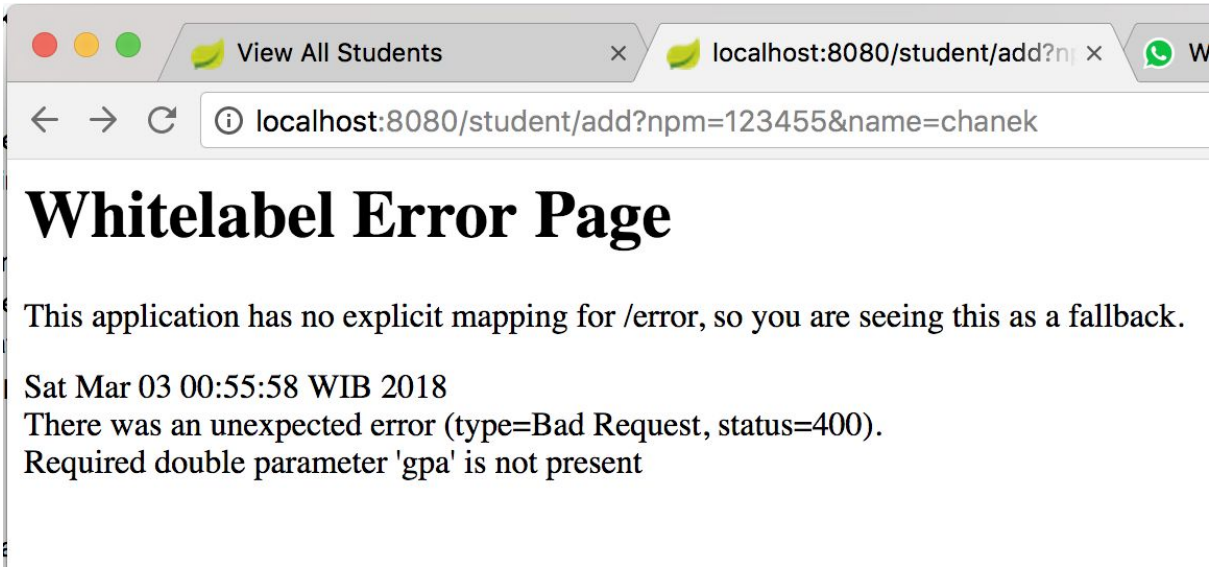
Method diatas menerima sebuah parameter yaitu npm untuk digunakan untuk menemukan object student yang dicari dalam list studentList. Kembalian dari method ini berupa object student berdasarkan npm yang dipakai sebagai parameter tersebut.

Membuat Controller dan Fungsi Add
Pertanyaan 1
Setelah menambahkan method add pada studentController dan menambahkan view untuk method add, maka method add dapat diakses dengan menggunakan URL localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43. Tampilannya adalah sebagai berikut. Parameter npm, name dan gpa harus diikutsertakan dalam mengakses method ini, karena parameter tersebut bersifat **required**.



Pertanyaan 2

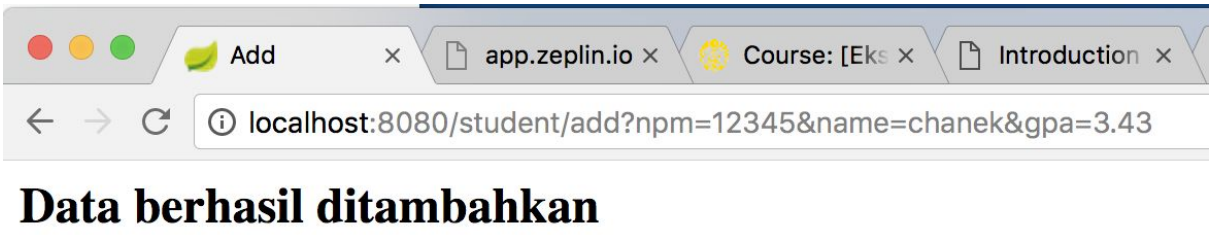
Mengakses method add dengan menggunakan URL localhost:8080/student/add?npm=12345&name=chanek akan menampilkan error. Karena parameter gpa dihapus, sedangkan parameter tersebut require dalam pemanggilan method add. Berikut tampilannya.

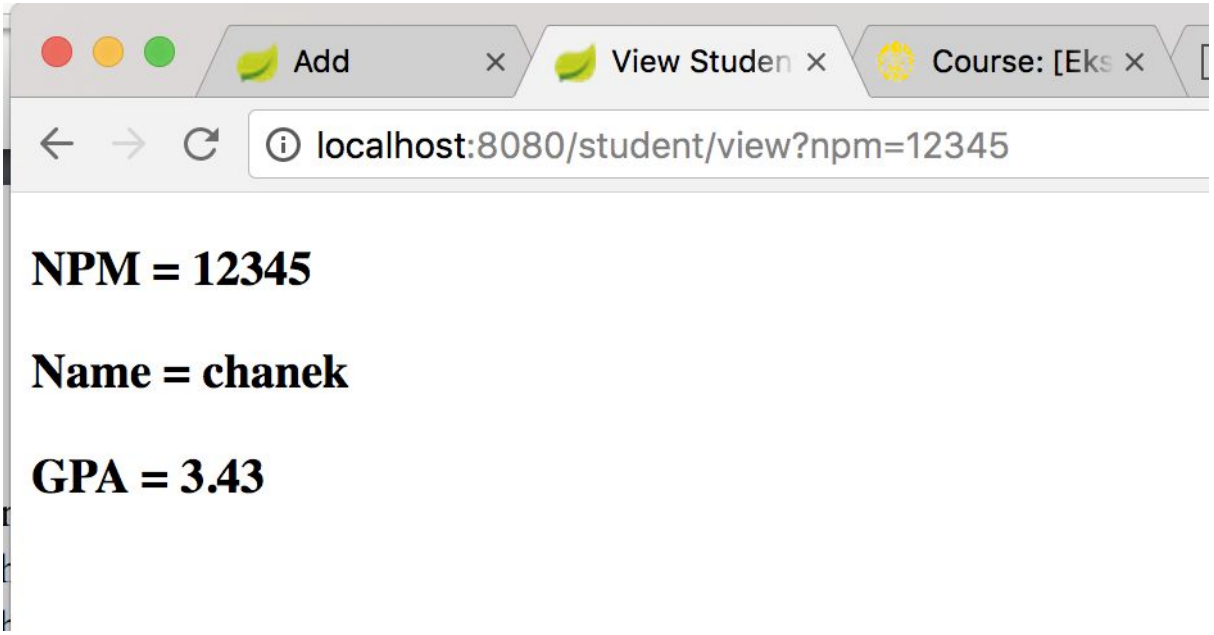


Method View by NPM

Pertanyaan 3

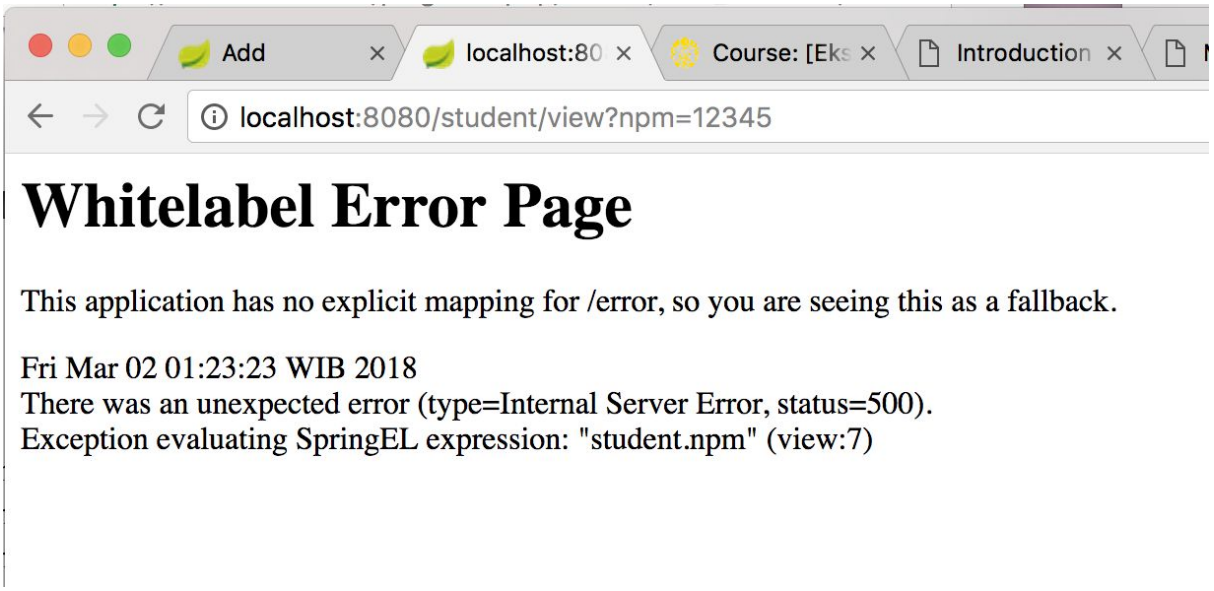
Setelah menambahkan method view by npm pada studentControll dan menambahkan view untuk method view by npm, maka method view dapat diakses menggunakan URL localhost:8080/student/view?npm=12345 setelah sebelumnya menambakan student dengan method add (localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43). Tampilannya adalah sebagai berikut.



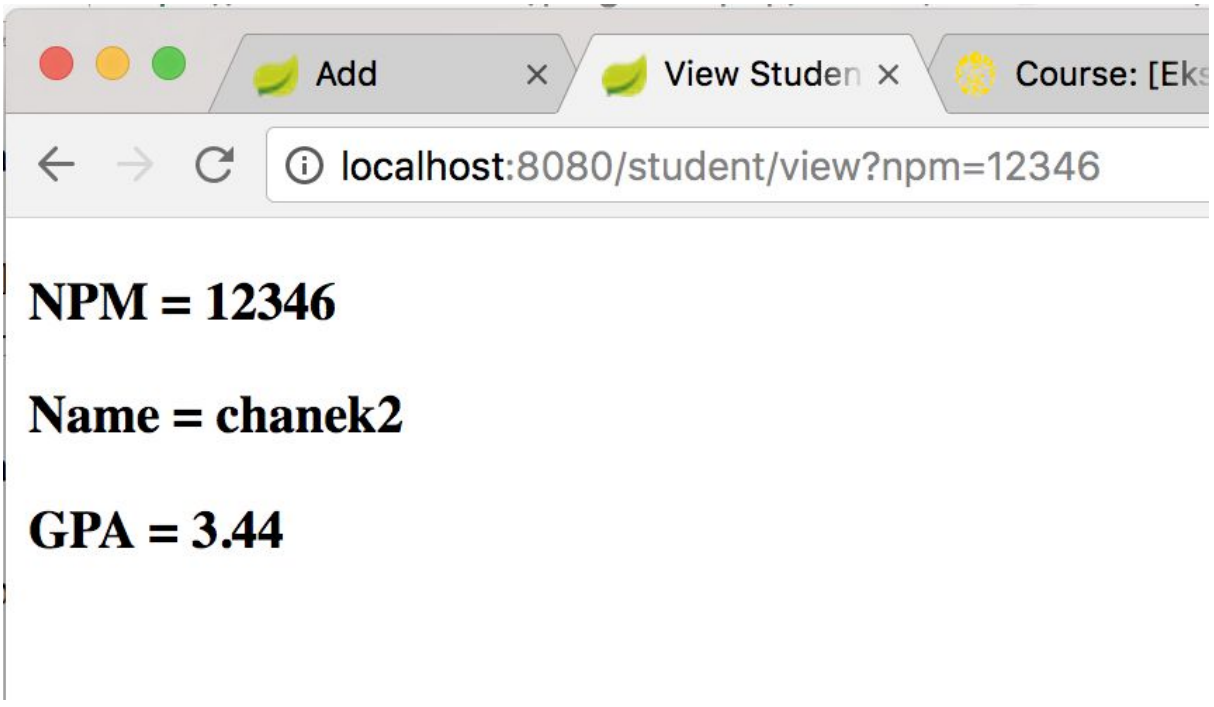


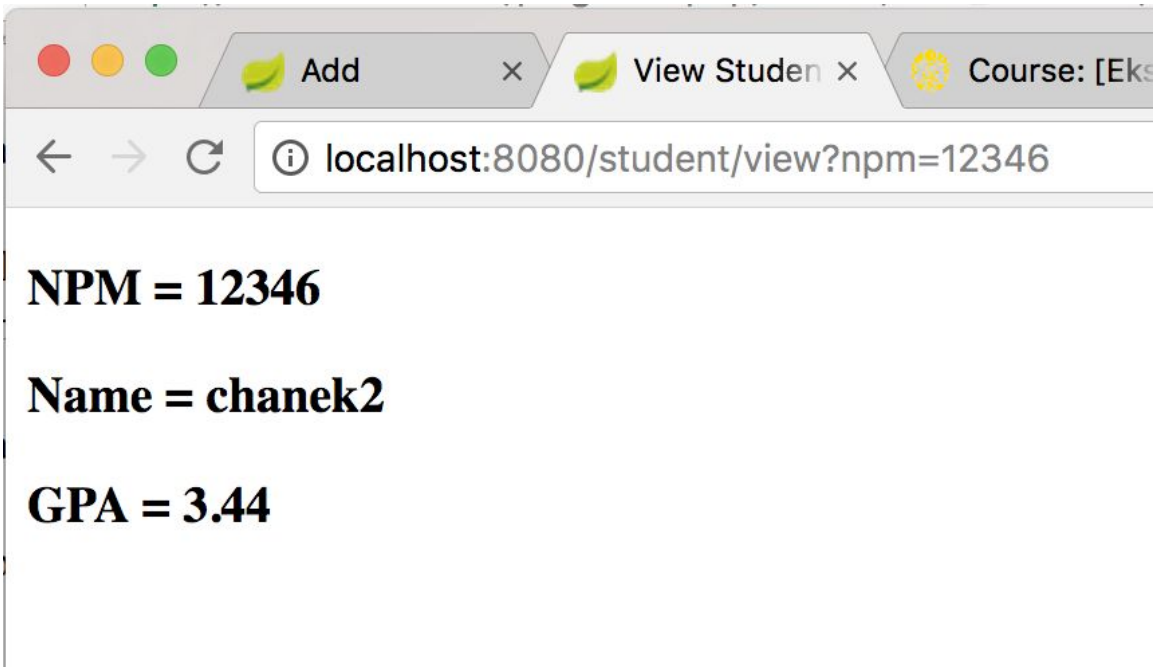
Pertanyaan 4

Ketika program dimatikan, maka data yang ditambahkan sebelumnya akan hilang karna data tersebut tidak disimpan permanen (tidak persistent). Sehingga, ketika mengakses method view by npm (localhost:8080/student/view?npm=12345) tanpa menambahkan student dengan method add (localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43), maka akan ditampilkan error. Karena object student tersebut belum ditambahkan, atau list student belum berisi object apapun. Berikut tampilannya.



5. Menambahkan data student lainnya dengan npm yang berbeda.

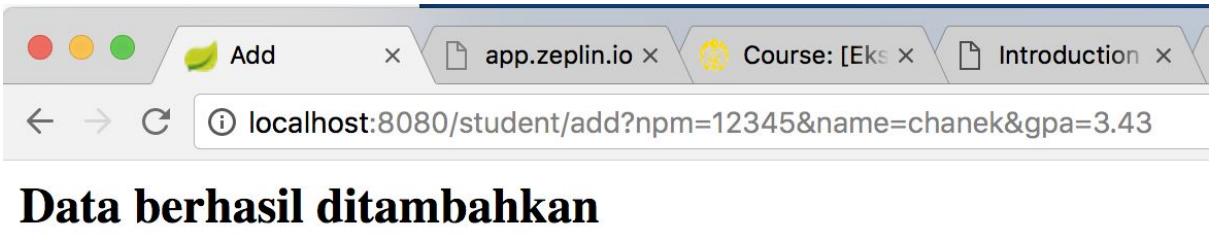


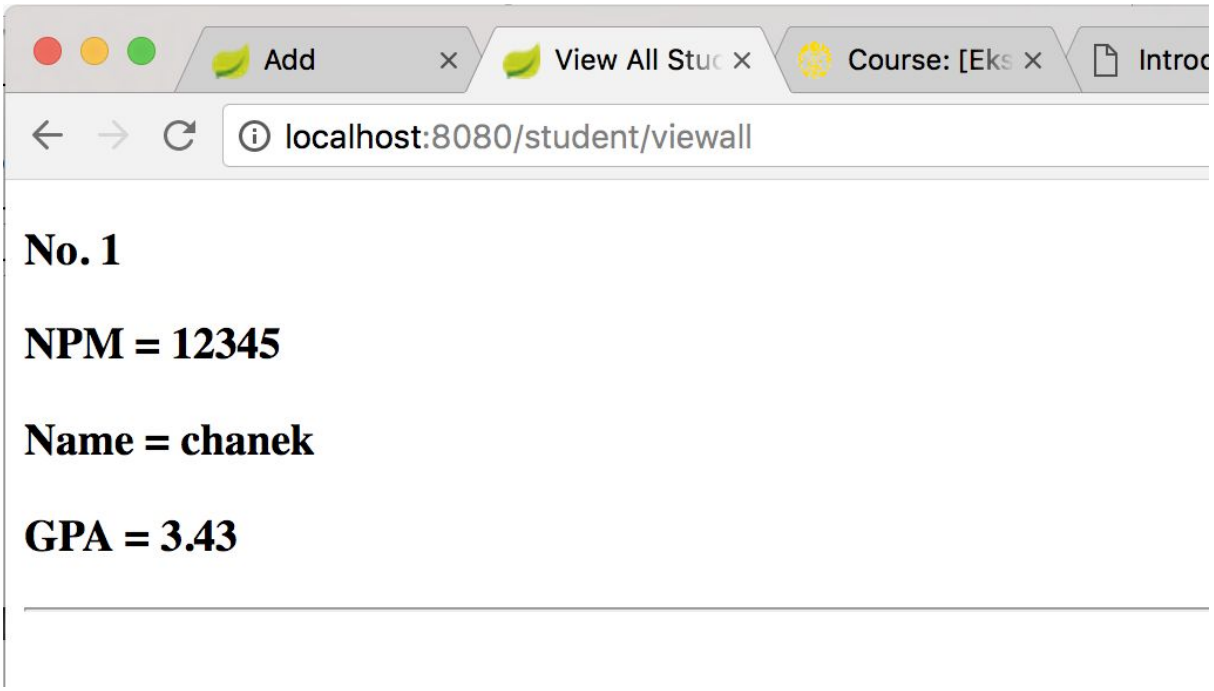


Method View All

Pertanyaan 5

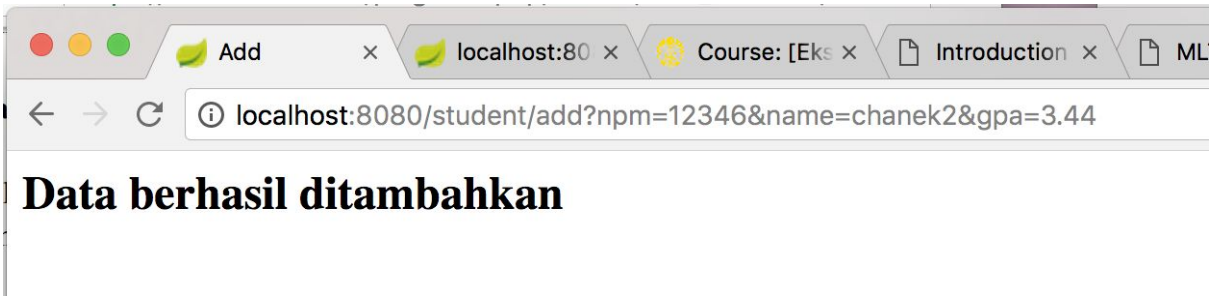
Setelah menambahkan method viewall pada studentControll dan menambahkan file view untuk method viewall, maka method viewall dapat diakses menggunakan URL localhost:8080/student/viewall setelah sebelumnya menambakan student dengan method add (localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43). Tampilannya adalah sebagai berikut.

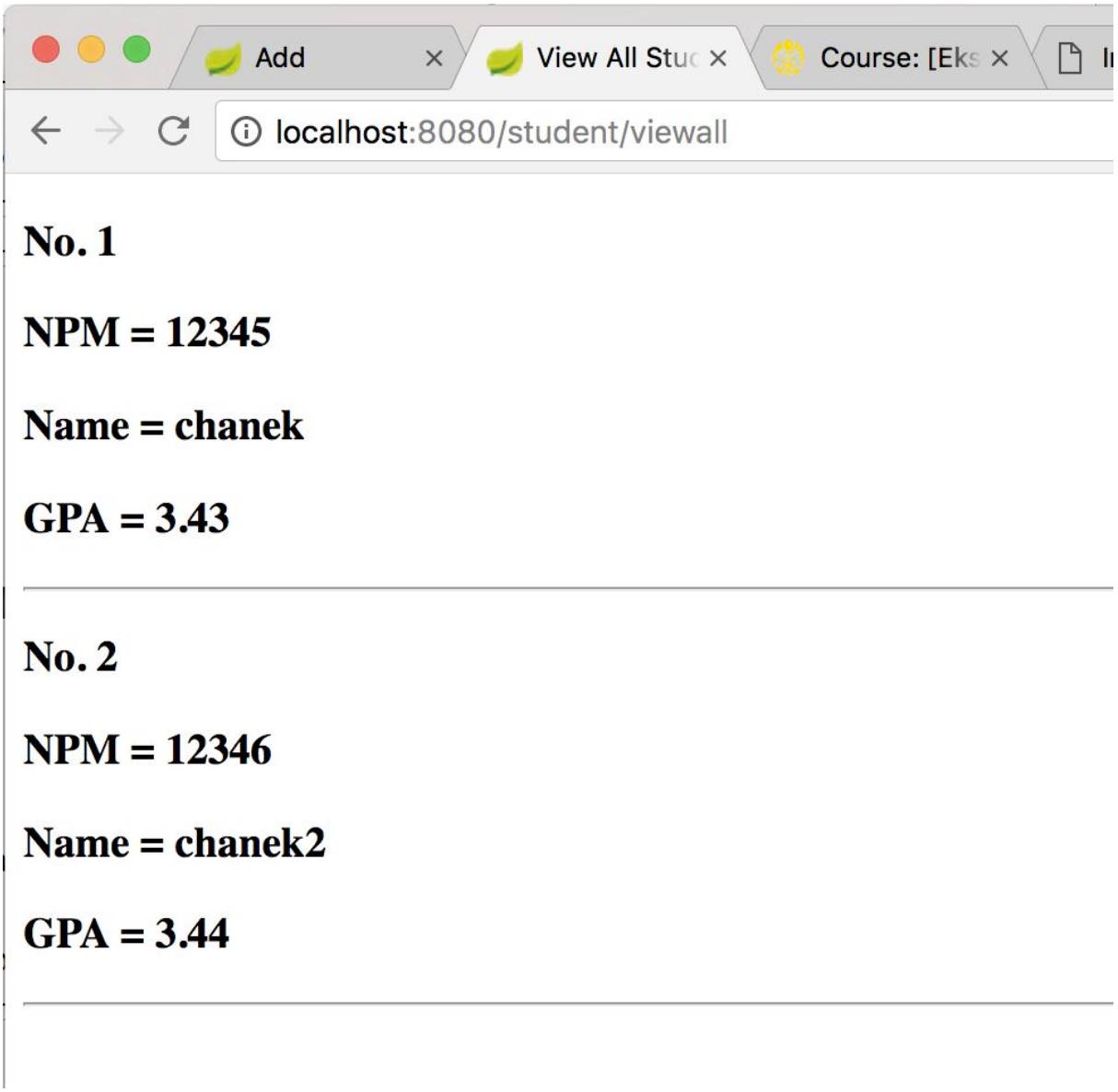




Pertanyaan 6

Menambahkan data student lainnya dengan npm yang berbeda, maka semua data student akan ditampilkan.





Latihan

1. Menambahkan method view by npm

Berikut method view student dengan menggunakan path variable. Pada method ini ditambahkan kondisi jika npm yang digunakan sebagai parameter tidak ditemukan dalam list student maka akan diredirect ke sebuah halaman web dengan pesan error “Student not found”

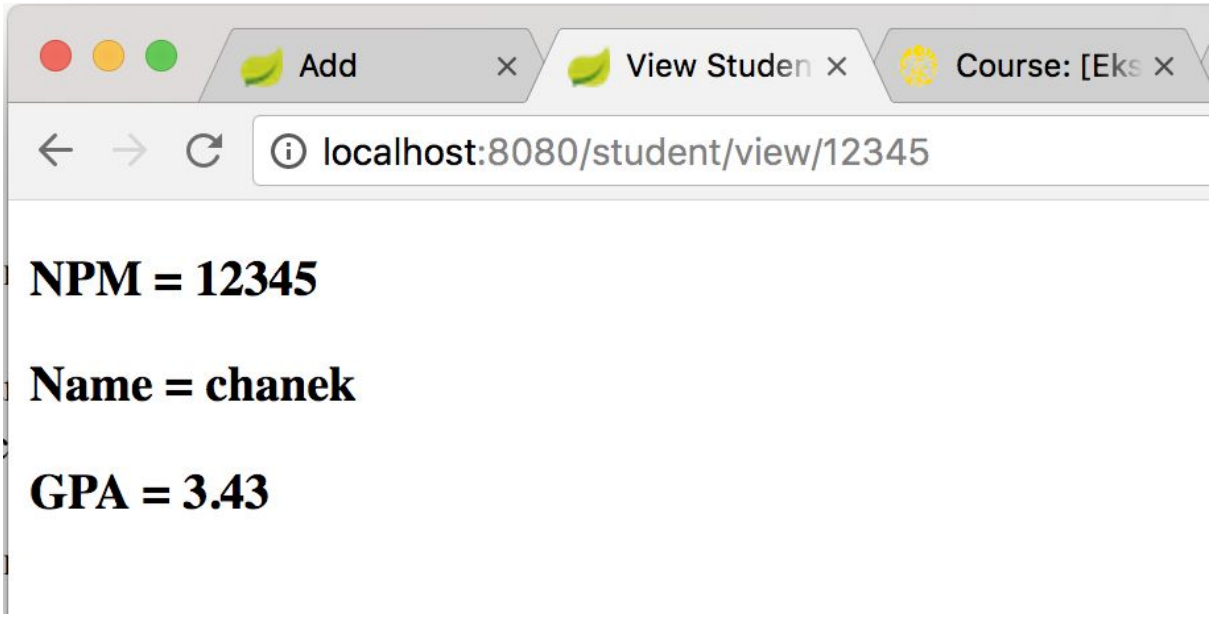
```
@RequestMapping(value = {"/student/view/", "/student/view/{npm}"})
    public String viewPath (Model model, @PathVariable Optional<String> npm){
        if(npm.isPresent()) {
            StudentModel student = studentService.selectStudent(npm.get());
            if (student != null) {
```

```
                        model.addAttribute("student", student);
                        return "view";
                }
                else{
                        model.addAttribute("npm", npm);
                        return "not-found";
                }
        }

        else {
                model.addAttribute("npm", npm);
                return "not-found";
        }

  }
```

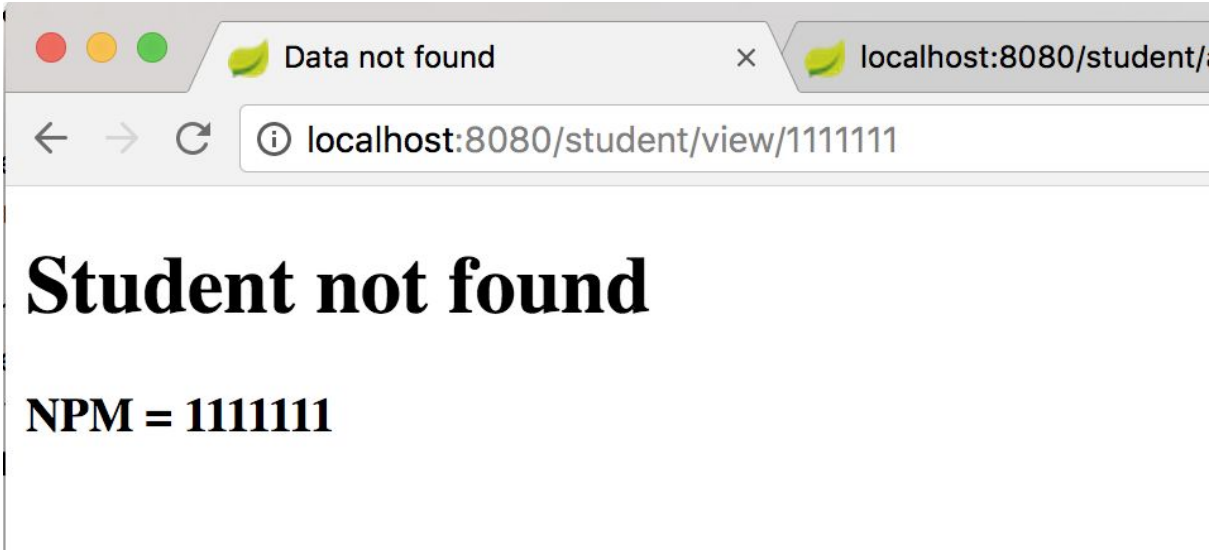
Berikut view dari method view by npm

```
<!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>
```

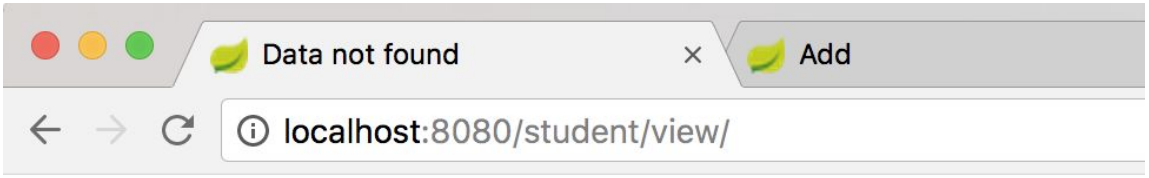
Berikut tampilan view student dengan path variable (Sebelumnya, tambahkan student dengan menggunakan method add student).



Jika npm yang digunakan untuk mengakses method view tidak ditemukan dalam list student, maka akan ditampilkan pesan error seperti gambar dibawah ini.



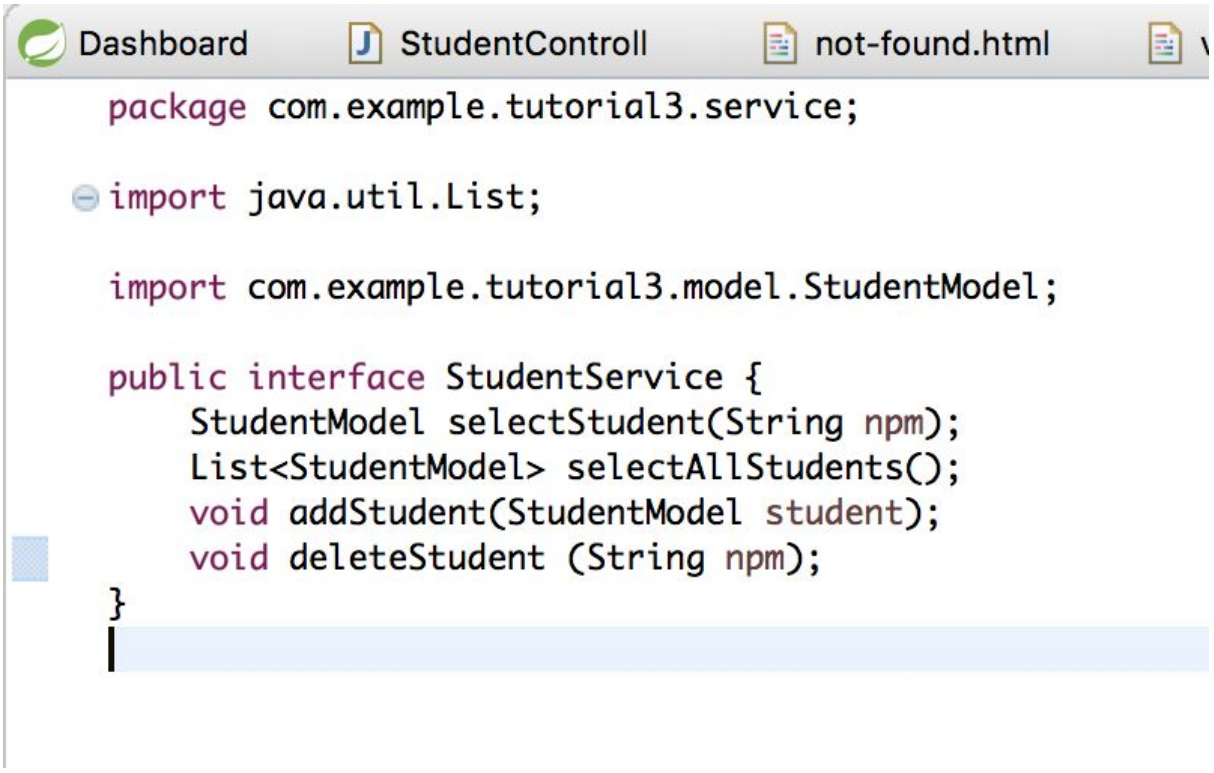
Jika method diakses tanpa menggunakan parameter (http://localhost:8080/student/view/) , maka akan ditampilkan pesan error seperti gambar dibawah.



Student not found

NPM = Optional.empty

2. Delete student by npm
Pertama, tambahkan method pada interface studentService



Kemudian implementasikan method tersebut pada class InmemoryStudentService

```
    }

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

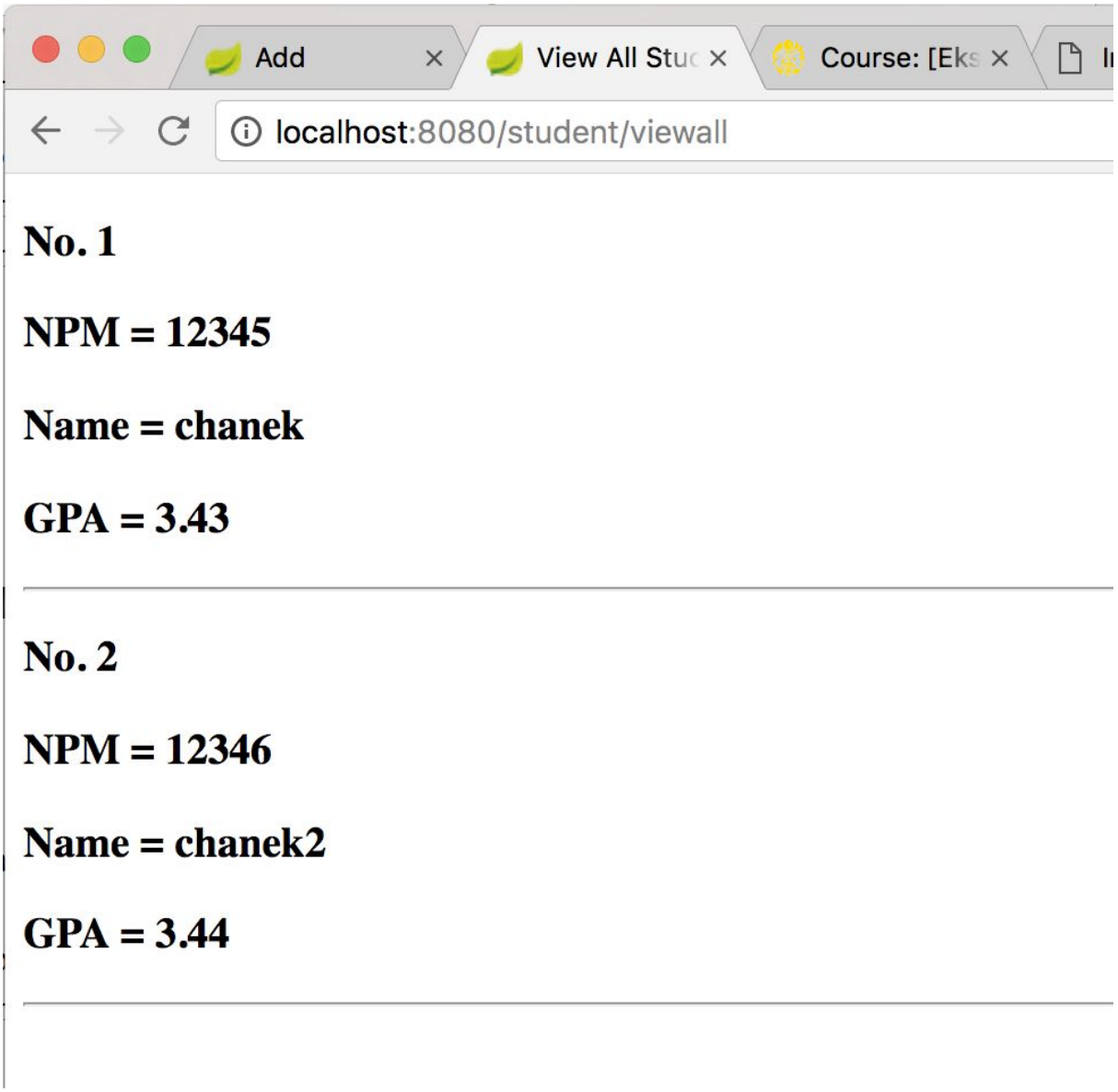
Berikut method delete student berdasarkan npm. Pada method ini ditambahkan kondisi jika npm yang digunakan sebagai parameter tidak ditemukan dalam list student maka akan diredirect ke sebuah halaman web dengan pesan error “Student not found”

```
@RequestMapping(value = {"/student/delete/", "/student/delete/{npm}"})
   public String delete (Model model, @PathVariable Optional<String> npm){
        if (npm.isPresent()) {
                StudentModel student = studentService.selectStudent(npm.get());
                if (student != null) {
                        studentService.deleteStudent(npm.get());
                        return "delete";
                }else {
                        model.addAttribute("npm", npm);
                        return "not-found";
                }
        }else {
                model.addAttribute("npm", npm);
                return "not-found";
        }

   }
```

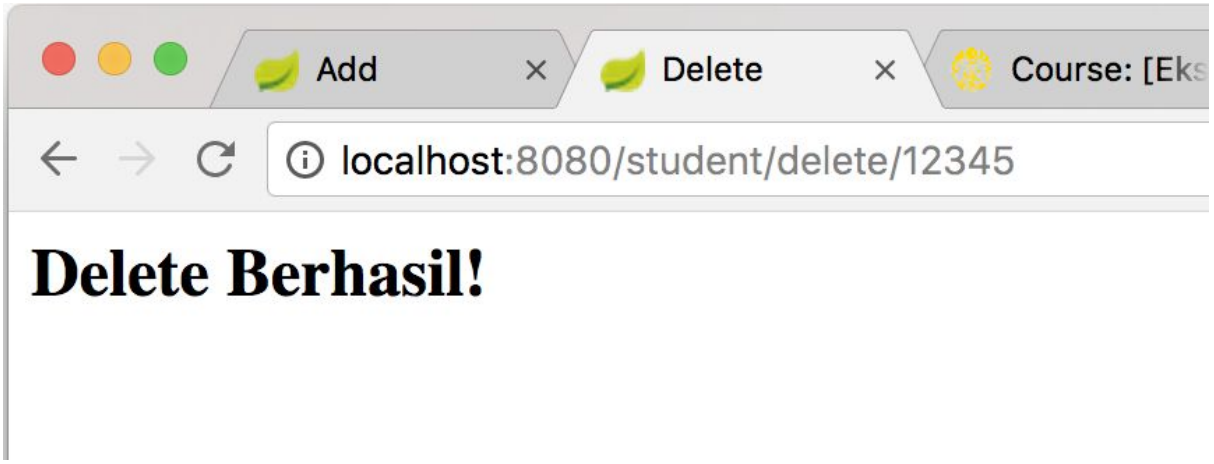
Berikut view dari method delete by npm

```
<html>
<head>
   <title>Delete</title>
</head>
<body>
   <h2>Delete Berhasil!</h2>
</body>
</html>
```

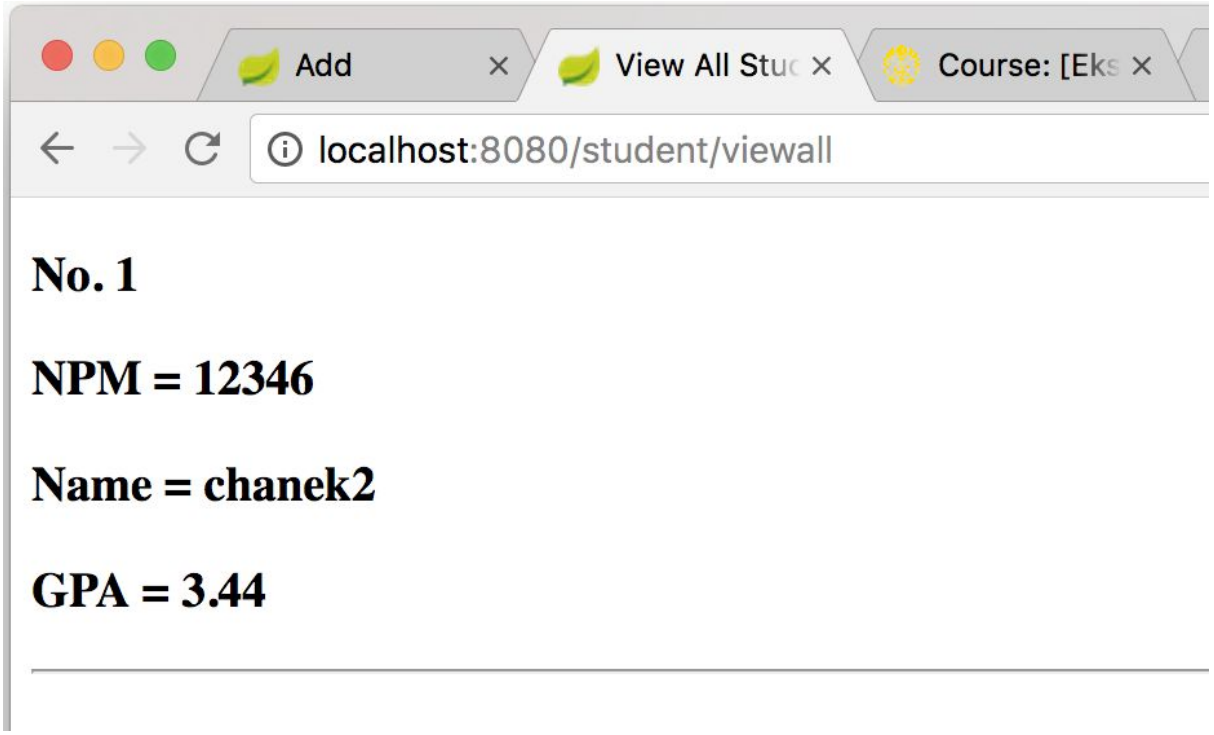
Setelah menambahkan dua data student dengan npm yang berbeda, berikut data student yang telah ditambahkan.



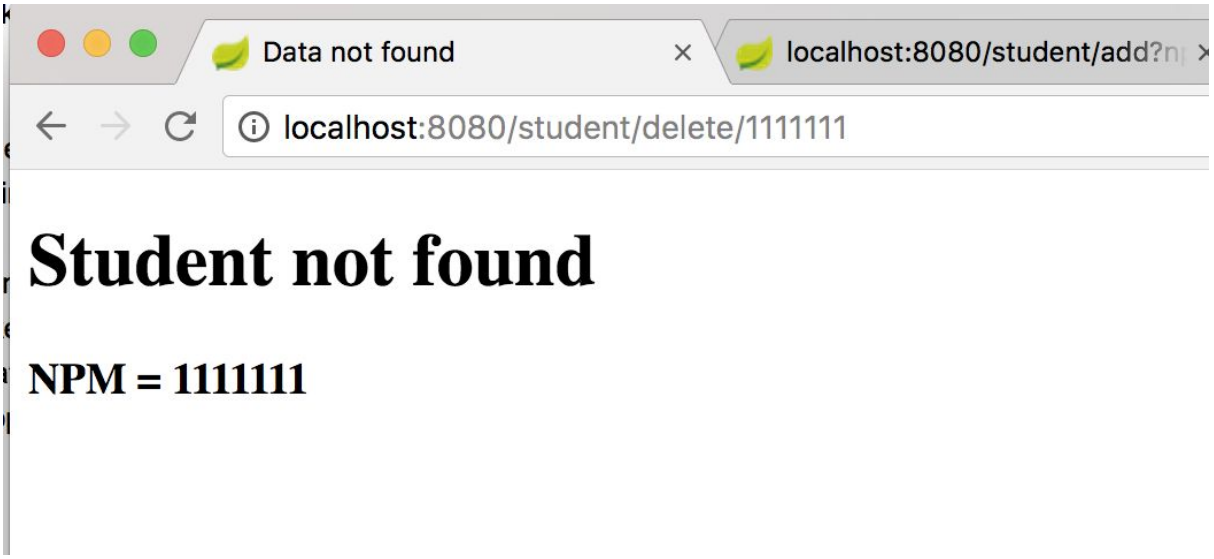
Kemudian salah satu data student dihapus dengan mengakses method delete pada studentController dan menyertakan npm student yang akan dihapus. URL : localhost:8080/student/delete/12345



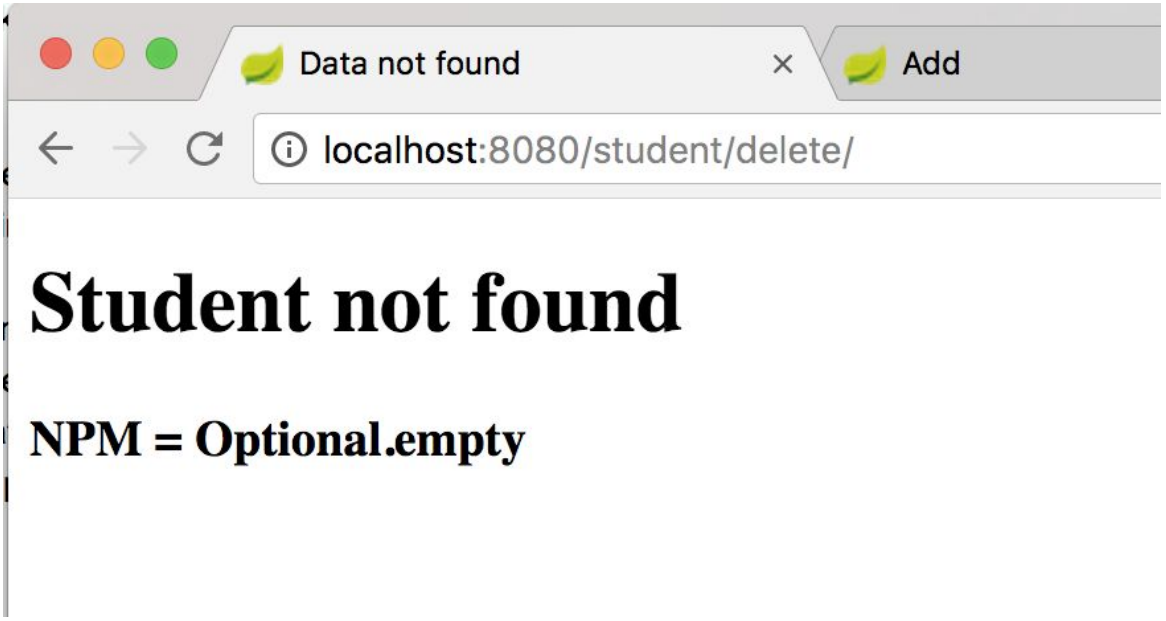
Setelah berhasil dihapus maka berikut tampilan data student yang tersisa, diakses dengan menggubakan method viewall student.



Jika method delete student diakses dengan menggunakan npm yang tidak ditemukan dalam list student, maka akan ditampilkan pesan error seperti gambar dibawah ini.



Jika method delete student diakses tanpa menggunakan parameter, maka akan ditampulkan pesan error seperti gambar dibawah ini.



Berikut code untuk tampilan data tidak ditemukan (not-found.html)

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
	<head>
	<title>Data not found</title>
	</head>
	<body>
	 <h1>Student not found</h1>
		<h3 th:text="'NPM = ' + ${npm}">Student NPM</h3>
	</body>
</html>
```