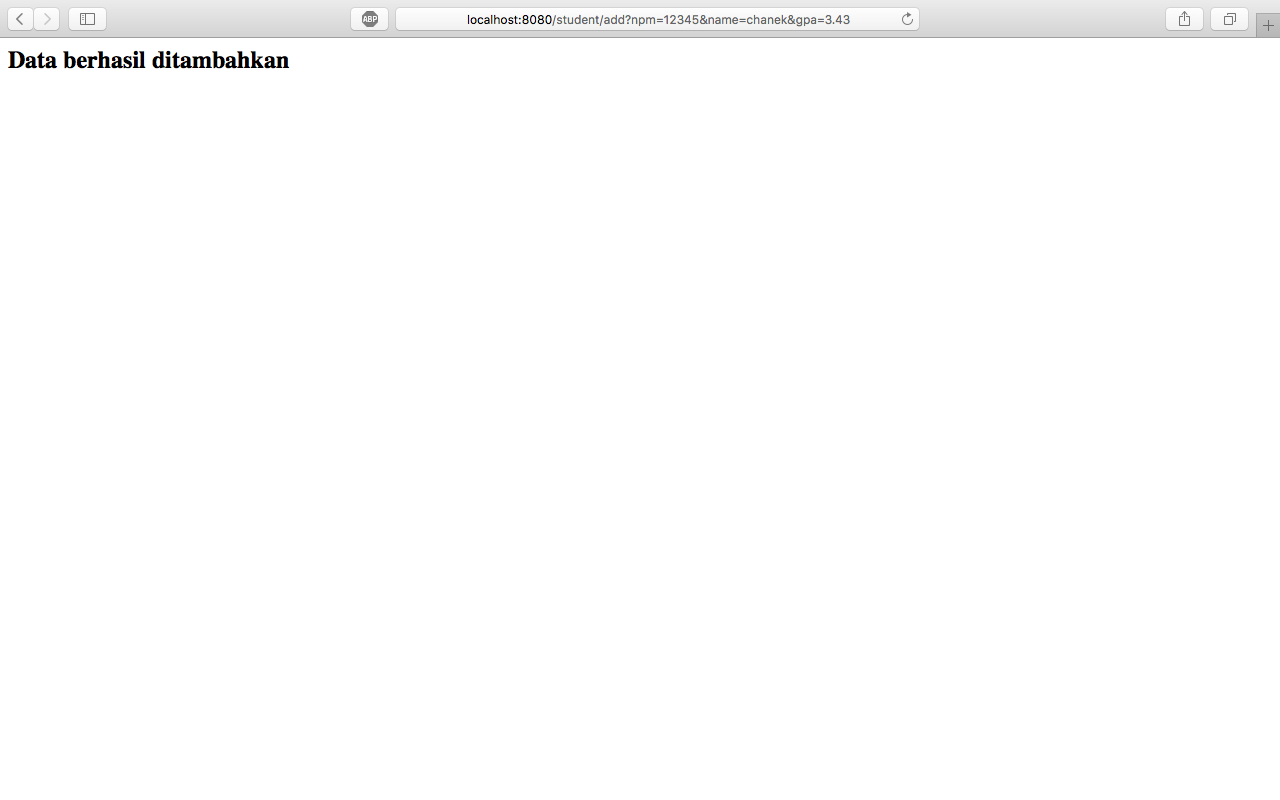
localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43

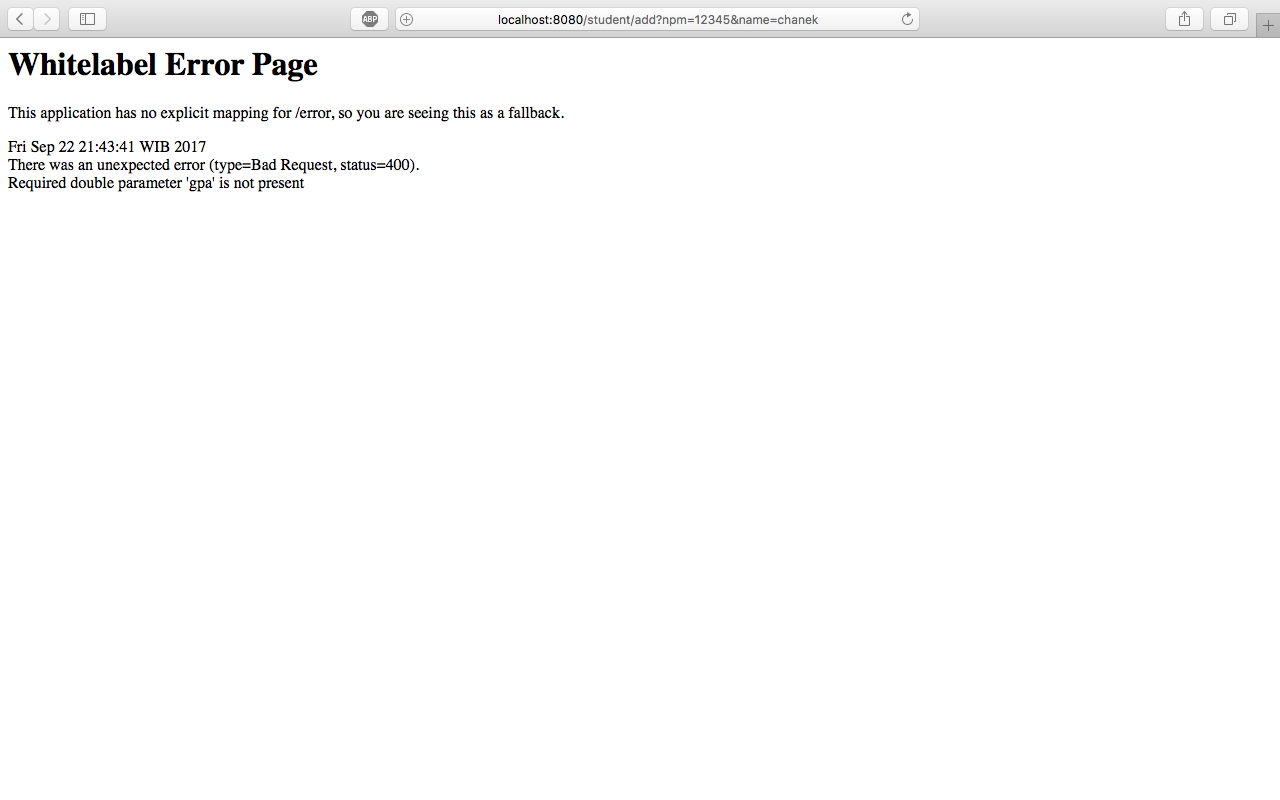
**Pertanyaan 1:** apakah hasilnya? Jika error, tuliskan penjelasan Anda.

****

**Hasilnya adalah halaman add.html ditampilkan pada browser.**

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

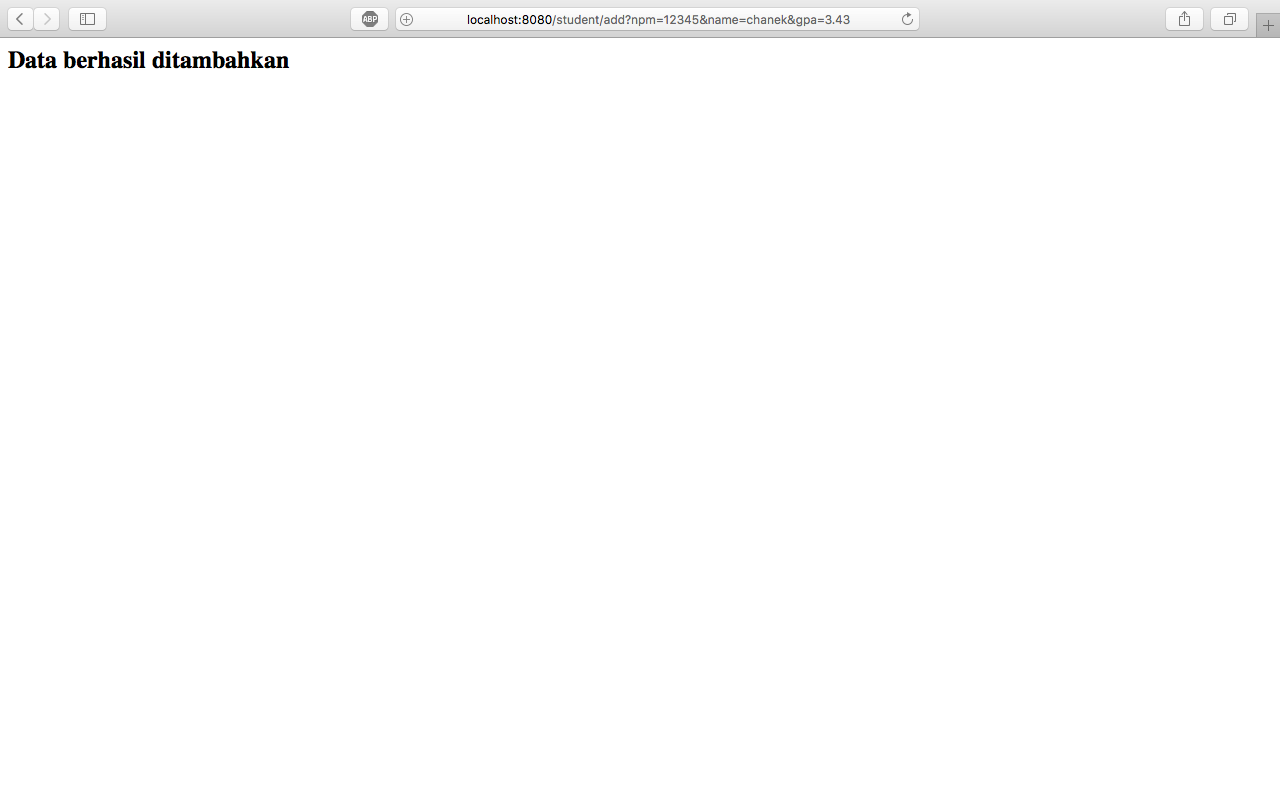
**Pertanyaan 2:** apakah hasilnya? Jika error, tuliskan penjelasan Anda.

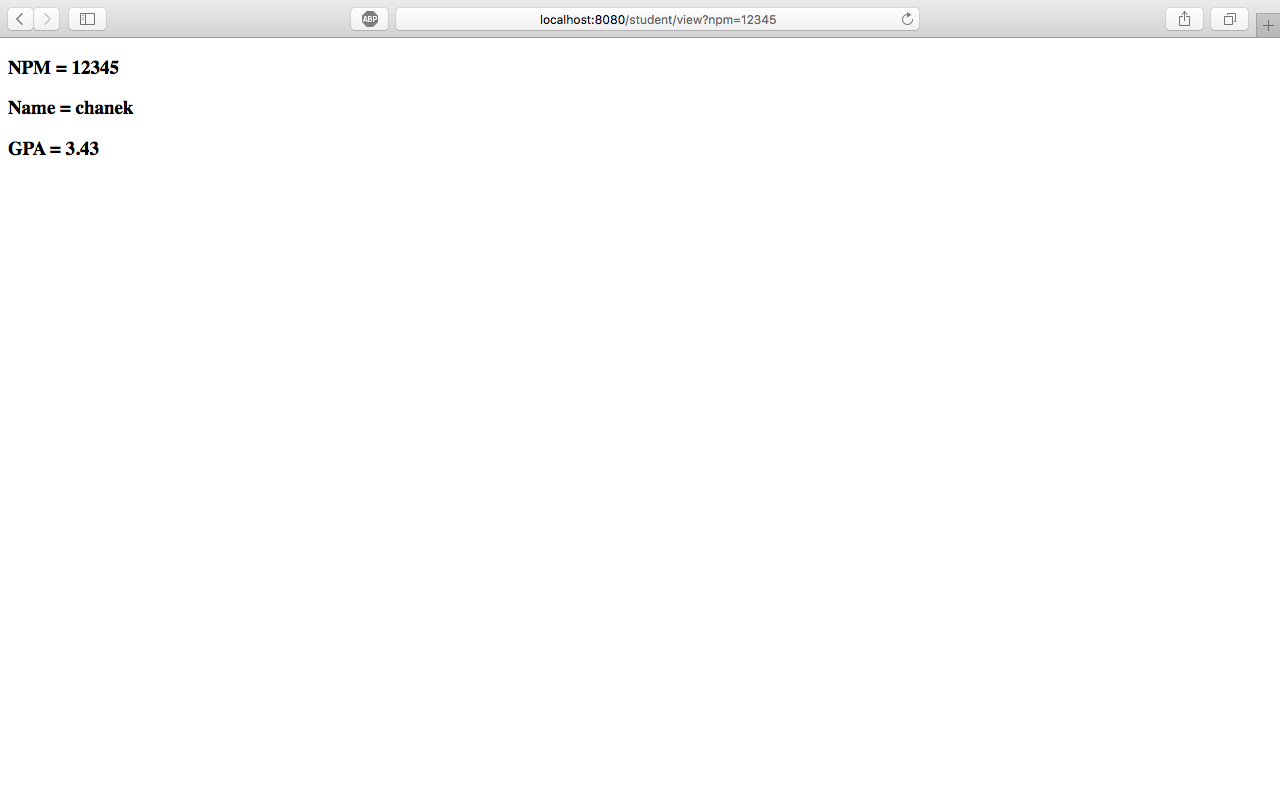
****

**Hasilnya adalah halaman Whitelabel Error, di mana hal ini terjadi karena parameter pada url (@RequestParam) yang bersifat wajib atau required tidak diisi yang dijelaskan pada kalimat “Required double parameter ‘gpa’ is not present”.**

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?

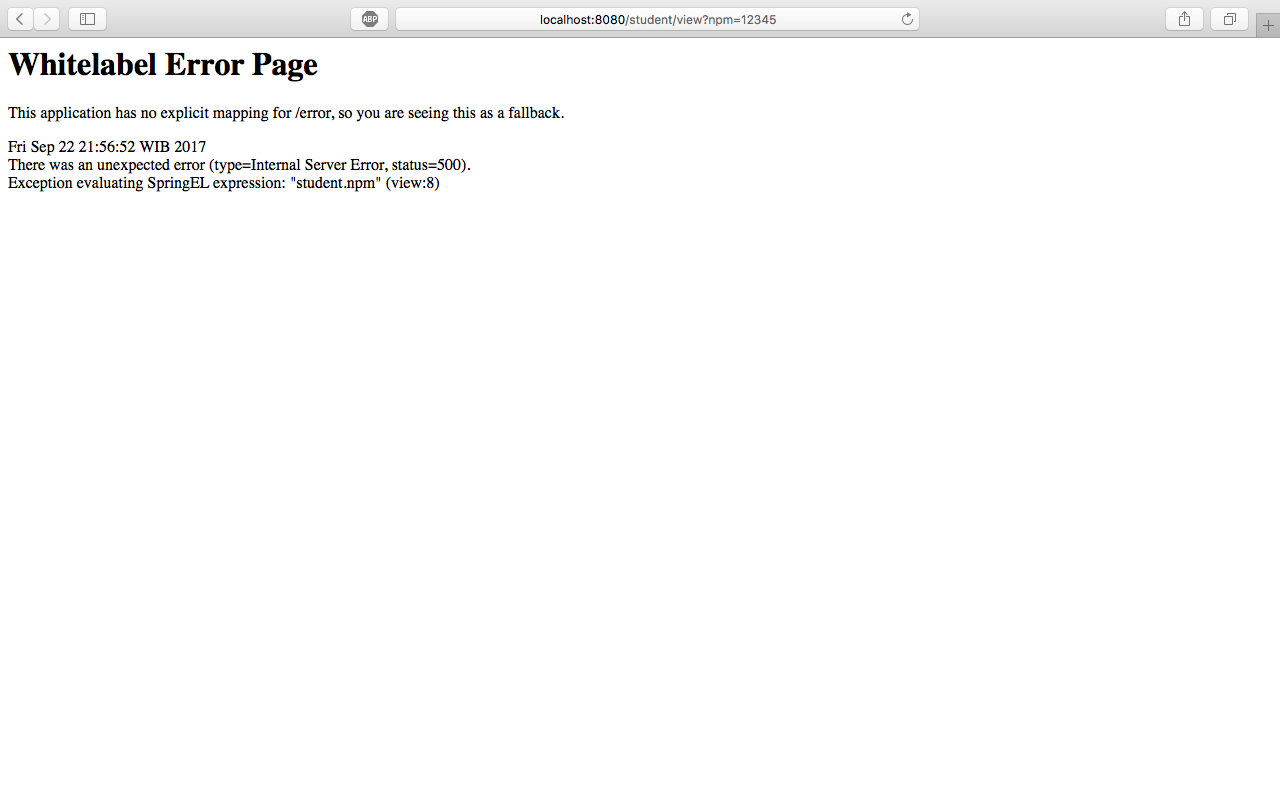




**Data student tersebut muncul pada halaman view.**

localhost:8080/student/view?npm=12345

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

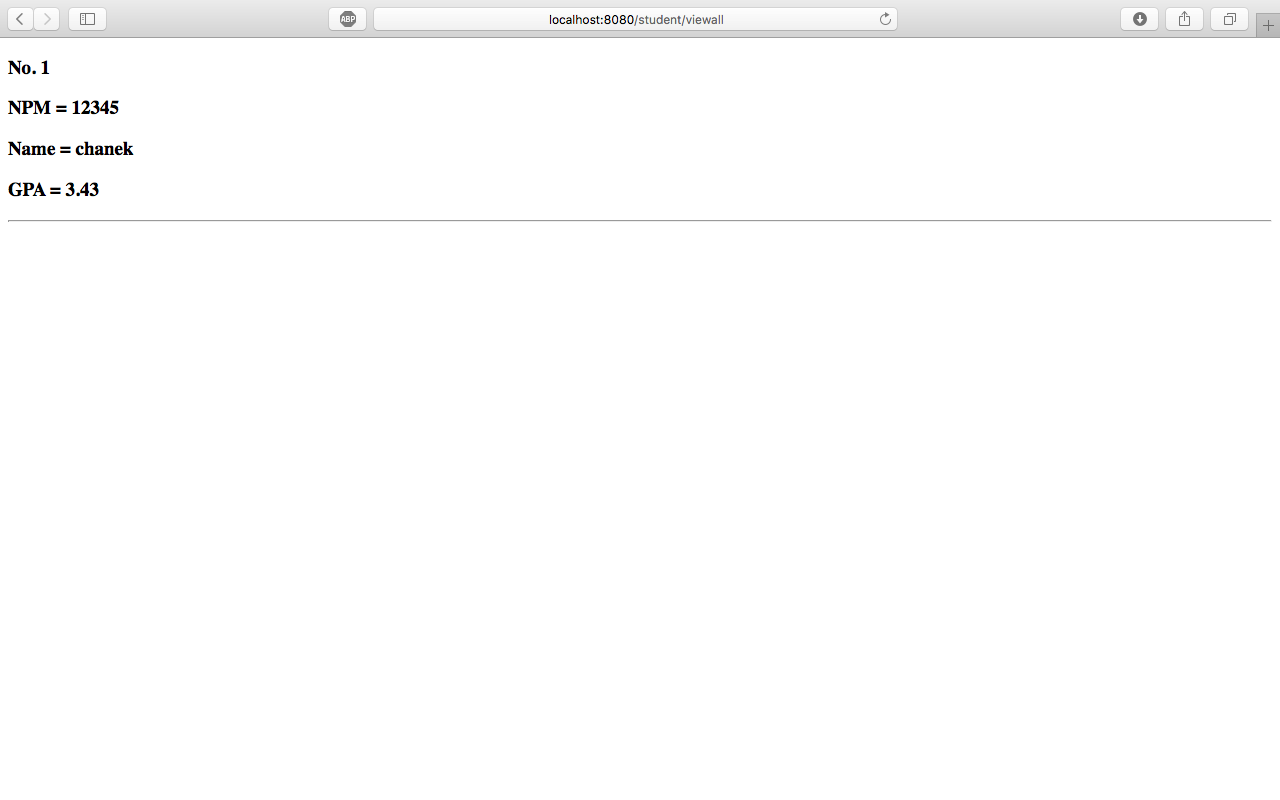


**Data student tersebut tidak muncul karena data yang telah disimpan oleh arraylist studentList telah di-reset setelah program di-running ulang. Hal ini juga disebabkan karena arraylist ybs hanya menyimpan data dalam satu waktu running sehingga ketika program di-running ulang, data yang telah disimpan sebelumnya akan terhapus dan arrraylist kembali kosong.**

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

localhost:8080/student/viewall

**Pertanyaan 5:** apakah data student tersebut muncul?

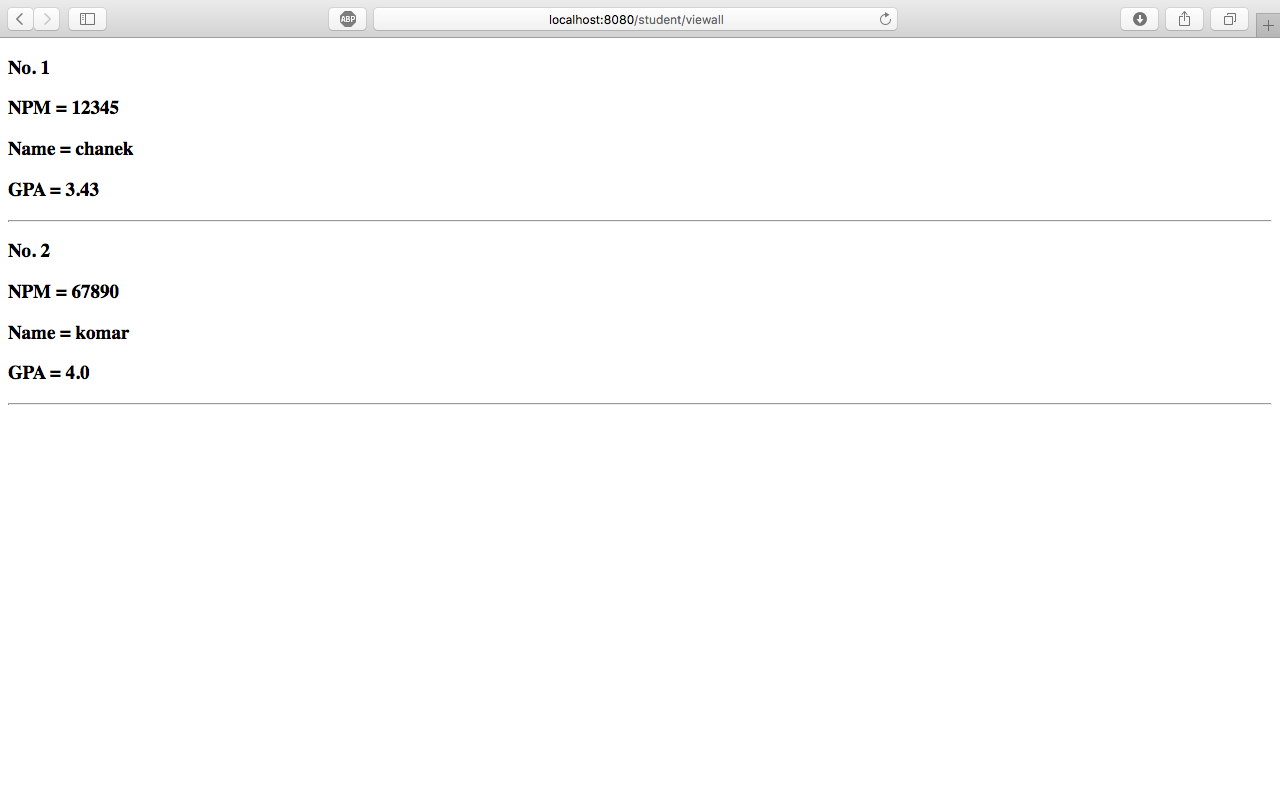


**Data student tersebut muncul pada halaman view all.**

tambahkan data Student lainnya dengan NPM yang berbeda

localhost:8080/student/viewall

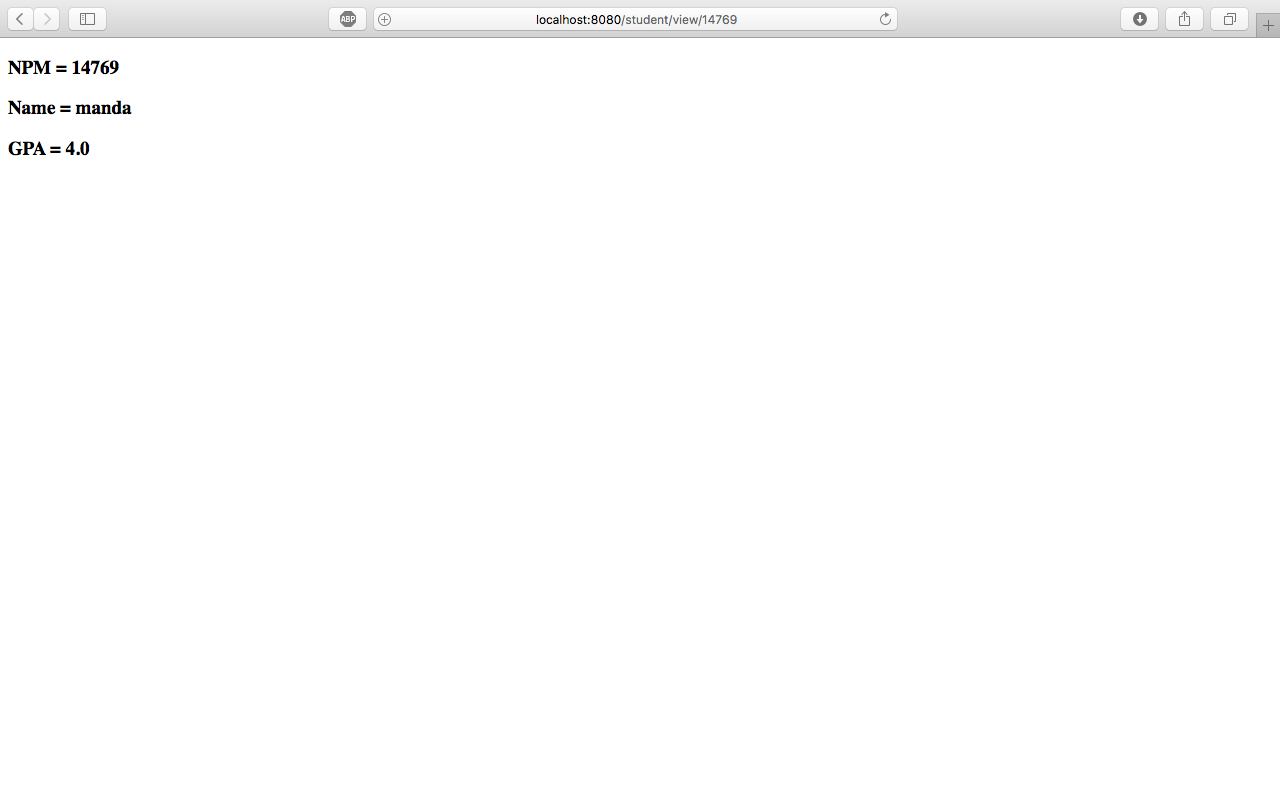
**Pertanyaan 5:** apakah semua data student muncul?

****

**Ya, data semua student muncul pada halaman view all.**

**Latihan**

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



**Tahap-tahap yang dilakukan:**

1. **Membuat method baru pada controller**

@RequestMapping("/student/view/{npm}")

public String viewStudent(Model model, @PathVariable String npm) {

StudentModel student = studentService.selectStudent(npm);

if(student == null) {

return "empty-data";

}

else {

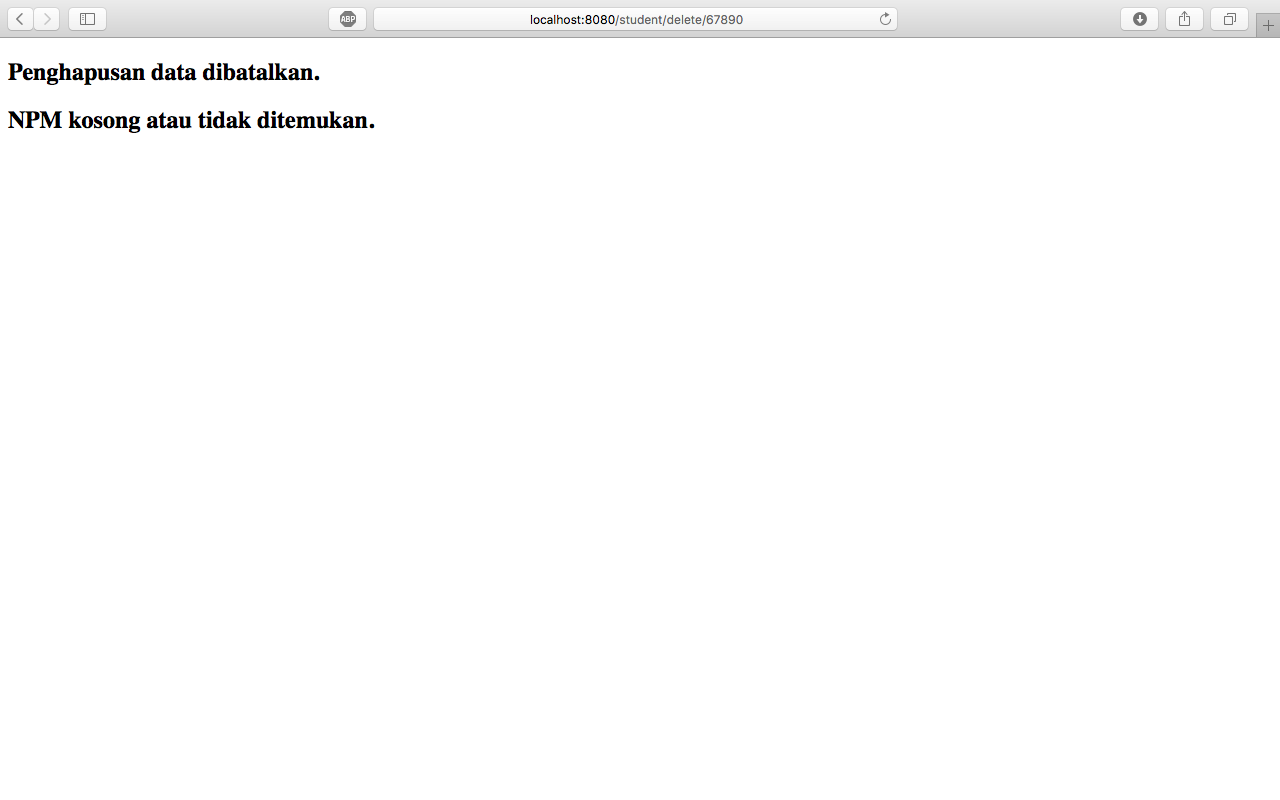
model.addAttribute("student", student);

return "view";

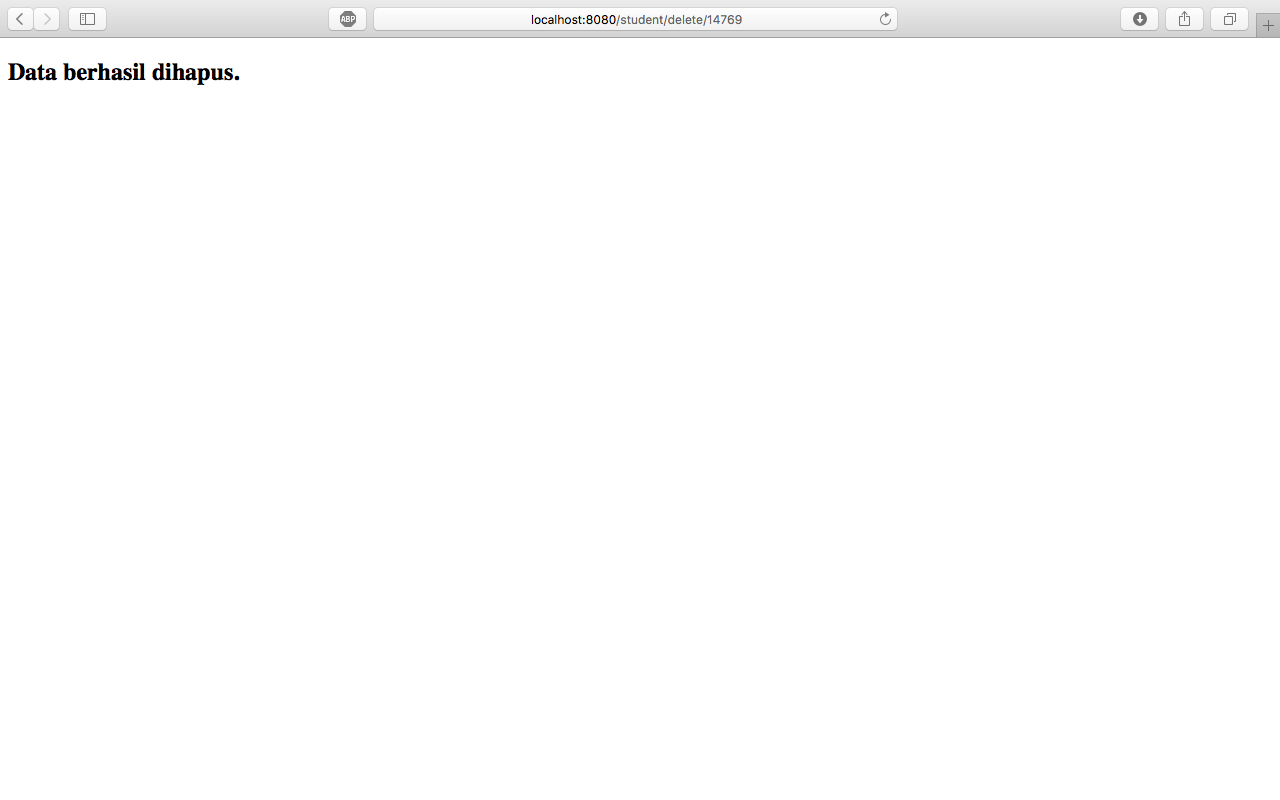
}

}

1. **Membuat halaman baru empty-data.html untuk menampilkan informasi bahwa NPM yang ingin dilihat tidak ditemukan atau kosong.**
2. Tambahkan fitur untuk melakukan delete Student berdasarkan NPM. Misalnya, setelah melakukan add Student pada soal nomor 1, cobalah untuk melakukan delete data tersebut dengan mengakses halaman localhost:8080/student/delete/14769. Tampilkan sebuah halaman yang memberikan informasi bahwa data tersebut telah berhasil dihapus.



**Halaman jika NPM kosong atau tidak ditemukan.**



**Halaman data berhasil dihapus.**

**Tahap-tahap yang dilakukan:**

1. **Membuat method baru pada controller**
2. **Membuat halaman delete.html untuk menampilkan halaman jika data berhasil dihapus.**
3. **Membuat halaman delete-cancel.html untuk menampilkan halaman jika NPM kosong atau tidak ditemukan.**

Hal yang telah saya pelajari pada tutorial kali ini adalah saya dapat membuat program dengan beberapa layer untuk masing-masing fungsinya, mulai dari model hingga controller. Hal ini terkait dengan arsitektur MVC yang telah dipelajari di kelas.

Method selectStudent():

public StudentModel selectStudent(String npm) {

if (studentList.isEmpty()) {

return null;

} else {

for (int i = 0; i < studentList.size(); i++) {

if (npm.equalsIgnoreCase(studentList.get(i).getNpm())) {

return studentList.get(i);

}

}

return null;

}

}

Method deleteStudent():

@RequestMapping(value={"/student/delete", "/student/delete/{npm}"})

public String delete(@PathVariable Optional<String> npm) {

if(npm.isPresent()) {

StudentModel student = studentService.selectStudent(npm.get());

if (student == null) {

return "delete-cancel";

} else {

studentService.deleteStudent(npm.get());

return "delete";

}

}

else {

return "delete-cancel";

}

}

Pada awalnya, method ini mengecek apakah npm pada path variabel diisi atau tidak. Kemudian jika terisi, model student diambil dan di cek apakah model tersebut null atau tidak.

Jika model tersebut dikembalikan, data dengan npm tersebut akan dihapus dari arraylist.

Jika model yang dikembalikan adalah null, halaman akan di direct ke halaman delete-cancel.html, sama halnya dengan kondisi bahwa npm tidak diisi.