Deliverables

a. Ringkasan dari materi yang telah Anda pelajari di tutorial kali ini.

Materi yang telah dipelajari pada tutorial kali ini, adalah bagaimana menggunakan kelas interface, controller, tampilan untuk html dan kelas untuk membuat beragam fitur hasil dari override kelas interface(service).

Service bertugas untuk menghubungkan controller dengan database. Pada kelas service ini, disimpan berbagai logic yang digunakan untuk mengolah data yang terdapat pada database.

Model merupakan sebuah objek yang mereprensentasikan dan menyimpan informasi, atribut yang dimiliki si objek.

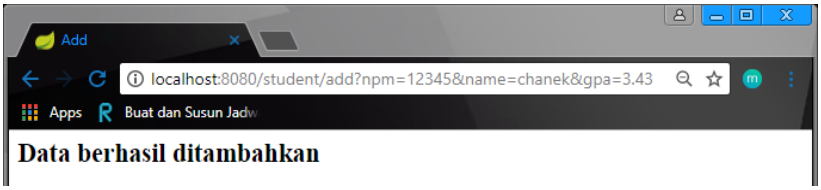
b. Hasil jawaban dari setiap poin pada bagian tutorial (dapat didukung dengan *screenshoot*)

❖ Membuat Controller dan Fungsi Add

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

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

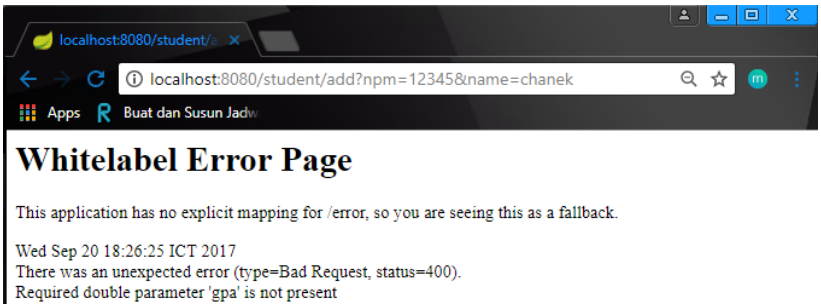
Tidak. Hasilnya seperti gambar di bawah ini. Data berhasil ditambahkan karena seluruh *value* pada url sesuai dengan *value* yang diminta pada *@RequestParam.*



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

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

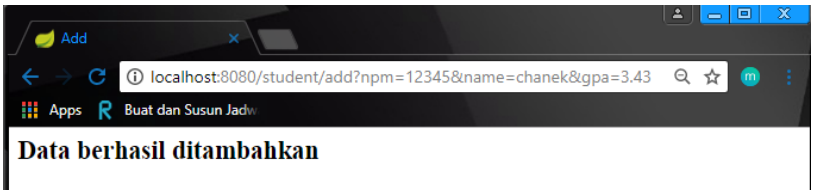
Ya. Karena tidak ada *value* gpa pada url, sementara pada method *add* di *class* StudentController, ada *@RequestParam* dengan *value* gpa dan *required* nya *true*.



❖ Method View by NPM

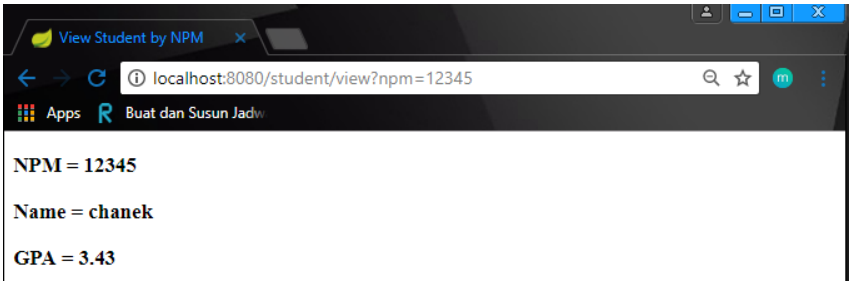
3. Jalankan program dan buka

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



localhost:8080/student/view?npm=12345

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

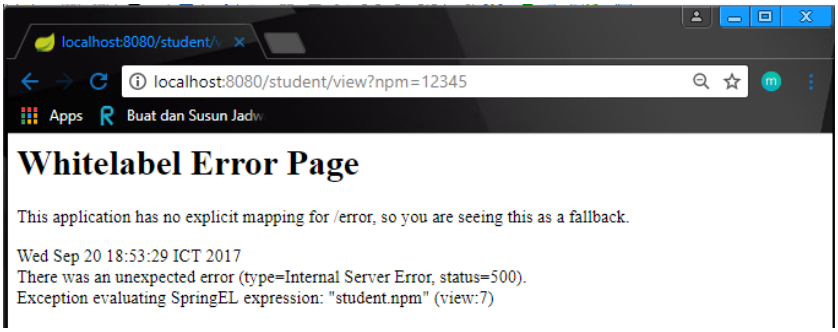


Muncul. Karena sebelum *load page view*, saya sudah menambahkan *student* yang npm nya 12345 dengan method *add*. Method *add* tersebut, langsung menambahkan *student* dengan npm x ke dalam *arrayList* bernama *studentList.*

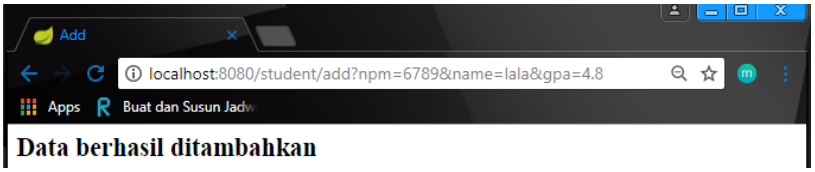
4. Coba matikan program dan jalankan kembali serta buka localhost:8080/student/view?npm=12345

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

Tidak. Karena sebelumnya tidak ada data *student* yang ditambahkan. Sementara langsung membuka *page view*. Maka tidak ada yang ditampilkan.



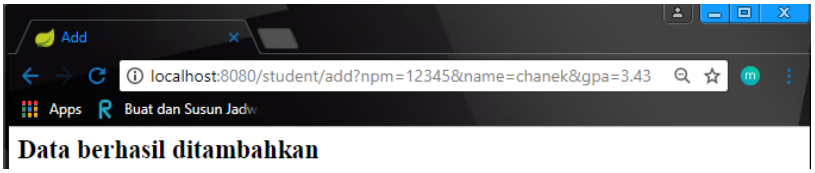
5. Coba tambahkan data Student lainnya dengan NPM yang berbeda.



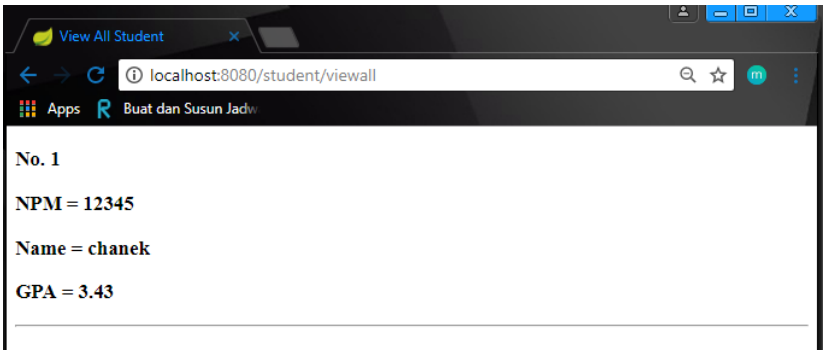
Menambahkan data *student* dengan npm=6789, name=lala, gpa=4.8

❖ Method View All

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



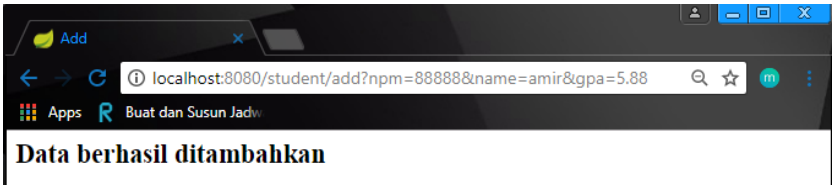
localhost:8080/student/viewall



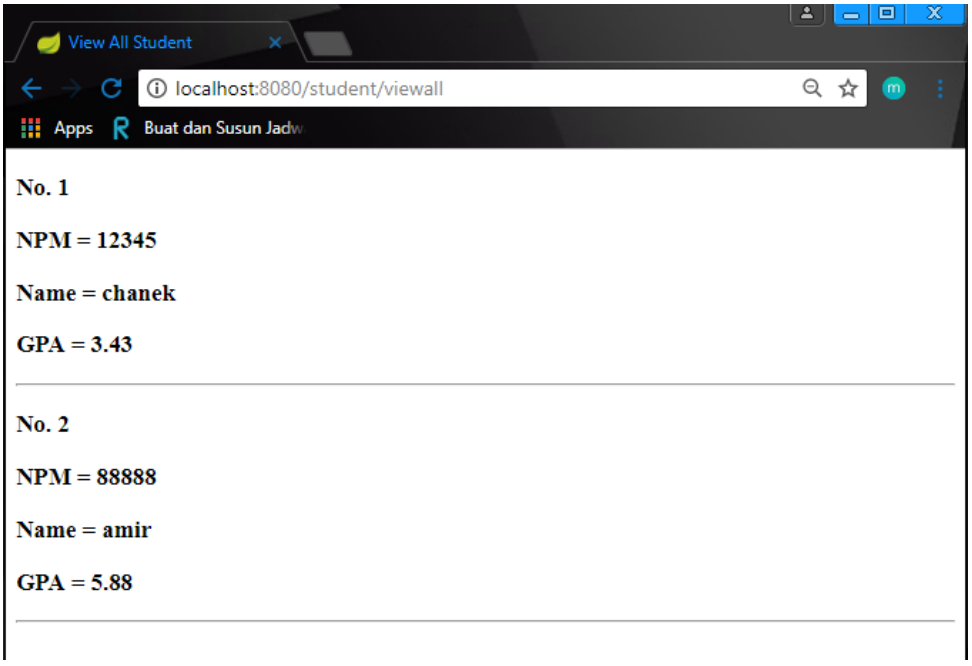
Pertanyaan 5: apakah data Student tersebut muncul?

Ya. Karena sudah ada data yang ditambahkan pada *arrayList studentList* dan data tersebut telah disimpan jadi dapat langsung ditampilkan.

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



Pertanyaan 6: Apakah semua data Student muncul?

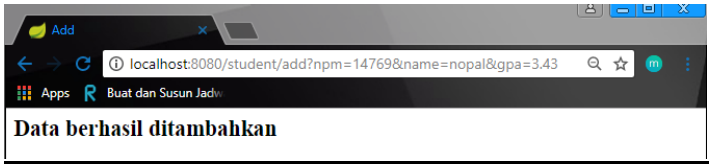


Ya. Semua data yang sebelumnya ditambahkan maka dapat ditampilkan sesuai nomor urut data yang ditambahkan.

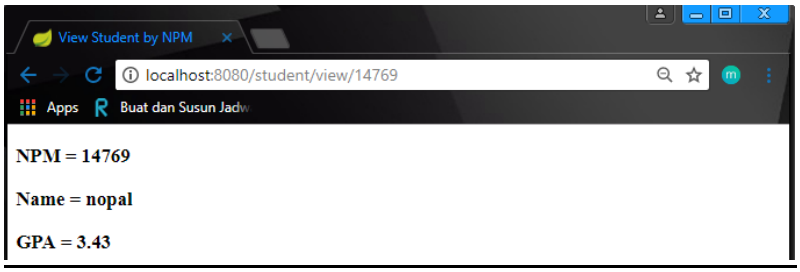
LATIHAN

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.**

   - Memasukkan data seorang Student yang memiliki npm = 14769

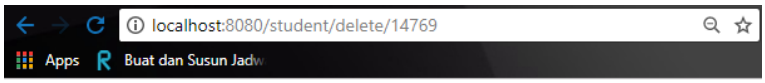


   - Melihat data yang baru dimasukkan dengan *Path Variable.* **localhost:8080/student/view/14769.**



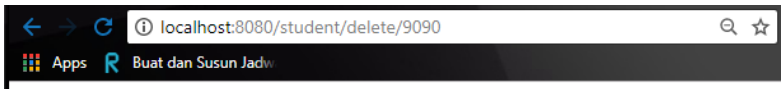
Jika nomor NPM tidak diberikan atau tidak ditemukan kembalikan halaman error yang berisi informasi bahwa nomor NPM kosong atau tidak ditemukan.

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.



Data berhasil dihapus

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



Maaf, NPM tidak ditemukan dan proses delete dibatalkan

c. Method selectStudent yang Anda implementasikan

Method selectStudent :

```
@Override
	public StudentModel selectStudent (String npm) {

		for(int i=0; i<studentList.size(); i++) {
			StudentModel student = studentList.get(i);
			if(student.getNpm().equals(npm)) {
				return student;
			}
		}

		return null;
	}
```

d. Penjelasan fitur *delete* yang Anda buat pada bagian latihan

Pertama, saya memiliki ide untuk membuat fitur *delete* langsung pada kelas StudentController.java. Saya sudah mengimplementasikan method selectStudent(). Dimana method ini bertugas untuk me-select student dengan npm sesuai value yang diberikan oleh user. Caranya, looping data student sebanyak ukuran arrayList si studentList. Ketika npm ada di arrayList tersebut, ambil npm nya. Kemudian dicek, apakah npm yang diambil itu sesuai dengan npm yang user berikan. Jika sesuai, maka return student dengan npm tersebut. Ketika tidak sesuai, return null. Yakni tidak ada nilai yang dapat diambil, tidak mengembalikan apapun.

Jadi, untuk delete student, gunakan method selectStudent() untuk mengambil data student dengan npm yang diinginkan user. Kemudian, data tersebut disimpan pada variable *student*. Setelah ada data yang disimpan, cek terlebih dahulu, apakah variable *student* tadi ada isinya atau tidak. Jika ada, maka select semua student dengan method selectAllStudent(), simpan pada arrayList dengan variable *students*. Lalu hapus data yang telah tersimpan pada arrayList *students* tadi. Kemudian panggil

halaman html yang akan menampilkan pesan bahwa data berhasil dihapus. Kondisi lain, yaitu jika variable *student* tidak ada isinya, maka akan dipanggil halaman html undelete yang menampilkan pesan bahwa npm tidak ditemukan dan proses delete dibatalkan.

Namun, ternyata cara seperti ini menimbulkan sedikit keganjilan. Cara ini dapat digunakan tanpa masalah ketika kita memakai localhost. Beda halnya dengan menggunakan database (bukan localhost). Jika di localhost, ketika menjalankan program ini (dengan @RequestMapping), data hanya bersifat sementara. Misal tambah data student → data berhasil ditambahkan, lalu buka halaman view → menampilkan data yang telah ditambahkan sebelumnya. Coba mematikan program, dan menjalankannya lagi, maka data yang tersimpan tadi hanya bersifat sementara. Jadi program yang langsung diimplementasikan pada @RequestMapping tidak berfungsi dengan maksimal nantinya. Jadi, saya coba buat sesuai tutorial.

Kedua, ide saya kali ini adalah mengimplementasikan method override delete di kelas InMemoryStudentService.java. Langkah pertama, buat interface deleteStudent di kelas StudentService.java dengan type boolean karena kita mau cek apakah data berhasil di delete atau tidak, jangan lupa parameternya adalah npm karena kita mau delete berdasarkan npm si student. Langkah kedua, buat override nya di kelas InMemoryStudentService.java. Fiturnya bekerja dengan cara: looping data student sebanyak arrayList nya. Lalu ambil si index npm nya dan simpan di variable *student.* Sekarang tinggal cek, jika npm yang diambil dari arrayList sama dengan npm yang user inginkan maka remove npm tadi dari arrayList dengan variable studentList, return true karena dia Boolean. Sementara jika npm tidak sama, return false.

Selanjutnya, beralih ke kelas StudentController.java. Di kelas ini, tinggal bagaimana cara menampilkan pesan yang diminta sesuai kondisi delete yang dijalankan pada method deleteStudent() di kelas InMemoryStudentService.java. Caranya, gunakan anotasi requestMapping dengan path variable untuk memanggil halaman html yang dibutuhkan. Setelah menuliskan syntax value untuk url yang diinginkan seperti apa, kita panggil method deleteStudent() dengan parameter npm yang akan di delete. Disimpan ke variable isDelete. Kenapa studentService.deleteStudent()? Karena kita perlu memanggil objek yang memiliki method deleteStudent itu. Lalu kita cek, jika tidak isDelete, yakni npm dari student yang mau di delete tidak ditemukan atau tidak ada, maka return “Undelete”, return ini akan mengacu ke halaman html Undelete yang bertugas menampilkan pesan bahwa npm tidak ditemukan dan proses delete dibatalkan. Selain itu, berarti npm yang ada di delete ada, maka panggil halaman html deleted. Bertugas menampilkan pesan bahwa npm berhasil dihapus.

Sekian, demo fitur delete dari saya. Terima kasih.