Tutorial 2

APAP

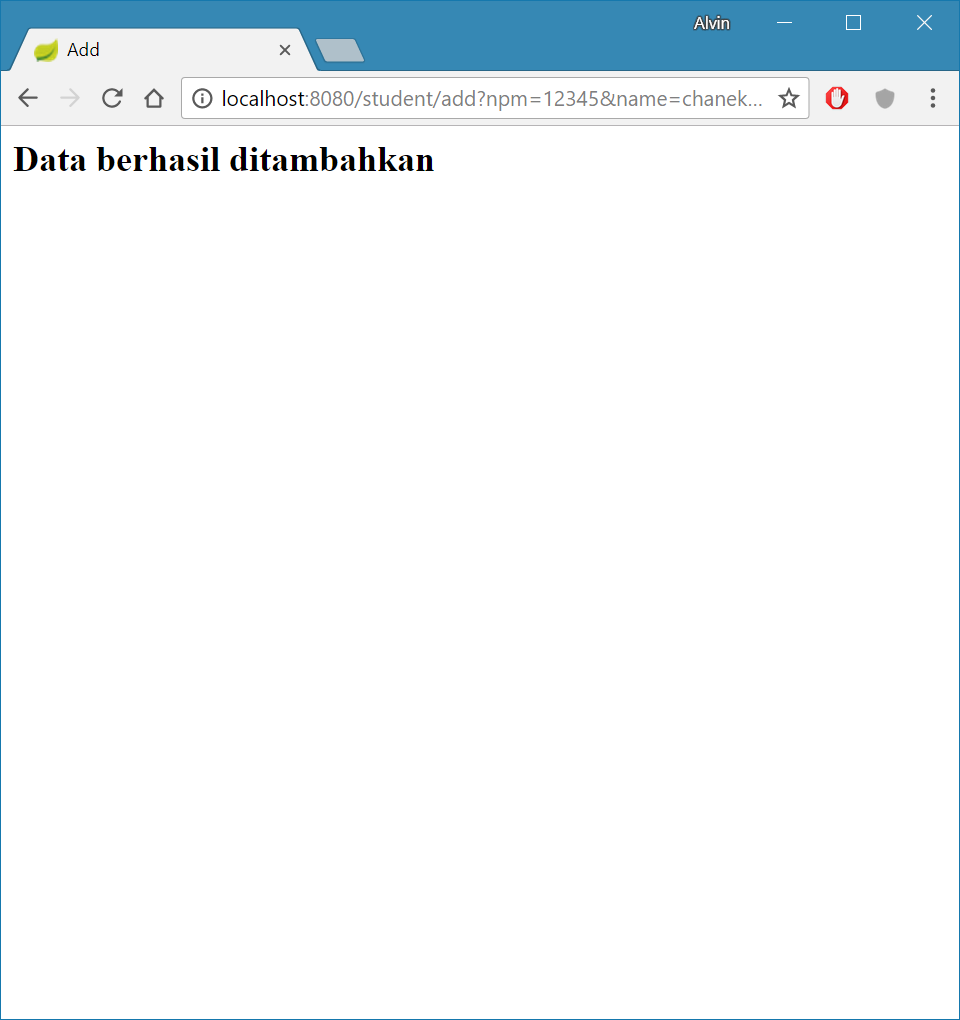
Tutorial

1. Jalankan program dan buka

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

Pertanyaan 1:

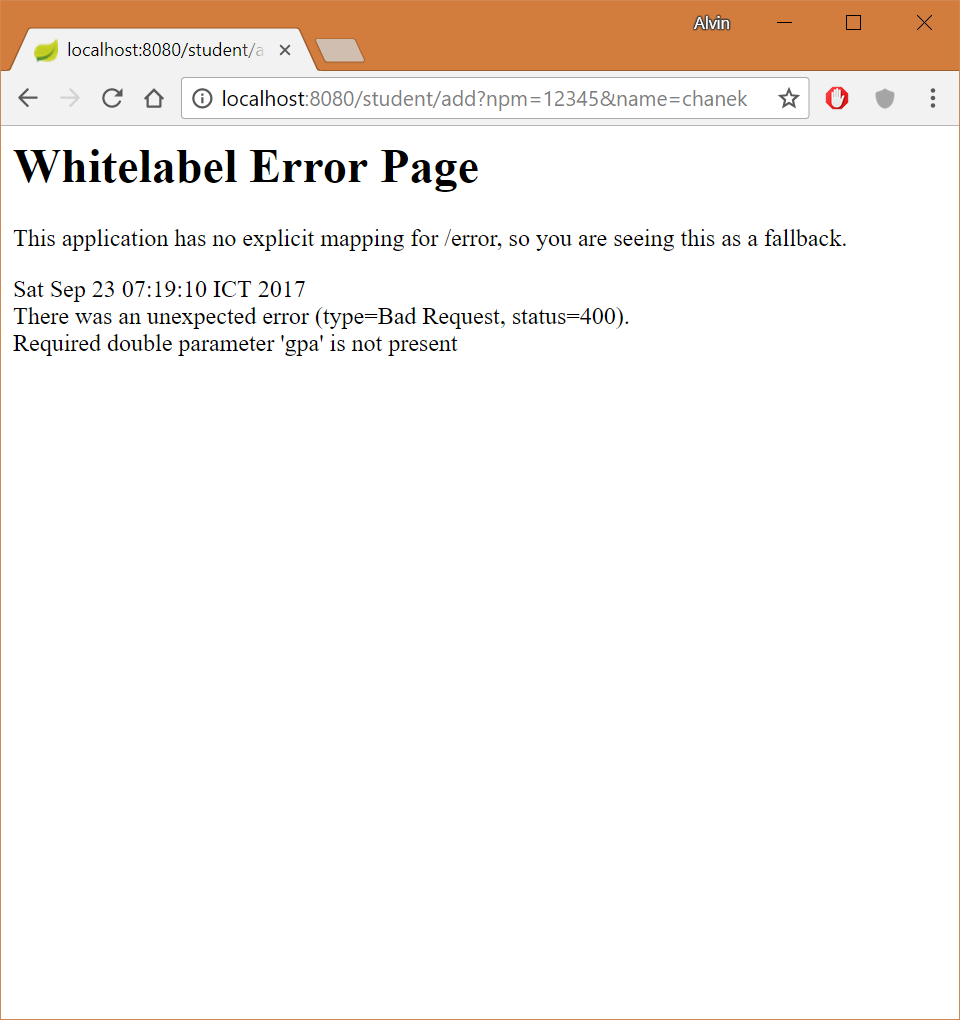
* Apakah hasilnya? Jika error, tuliskan penjelasan Anda. **Tidak error. Tampil tulisan “Data berhasil ditambahkan”**



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

Pertanyaan 2:

* Apakah hasilnya? Jika error, tuliskan penjelasan Anda. **Error. Karena parameter gpa nya tidak ada padahal required.**



2.

Method selectStudent():

@Override

**public** StudentModel selectStudent(String npm) {

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

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

**return** *studentList*.get(i);

}

}

**return** **null**;

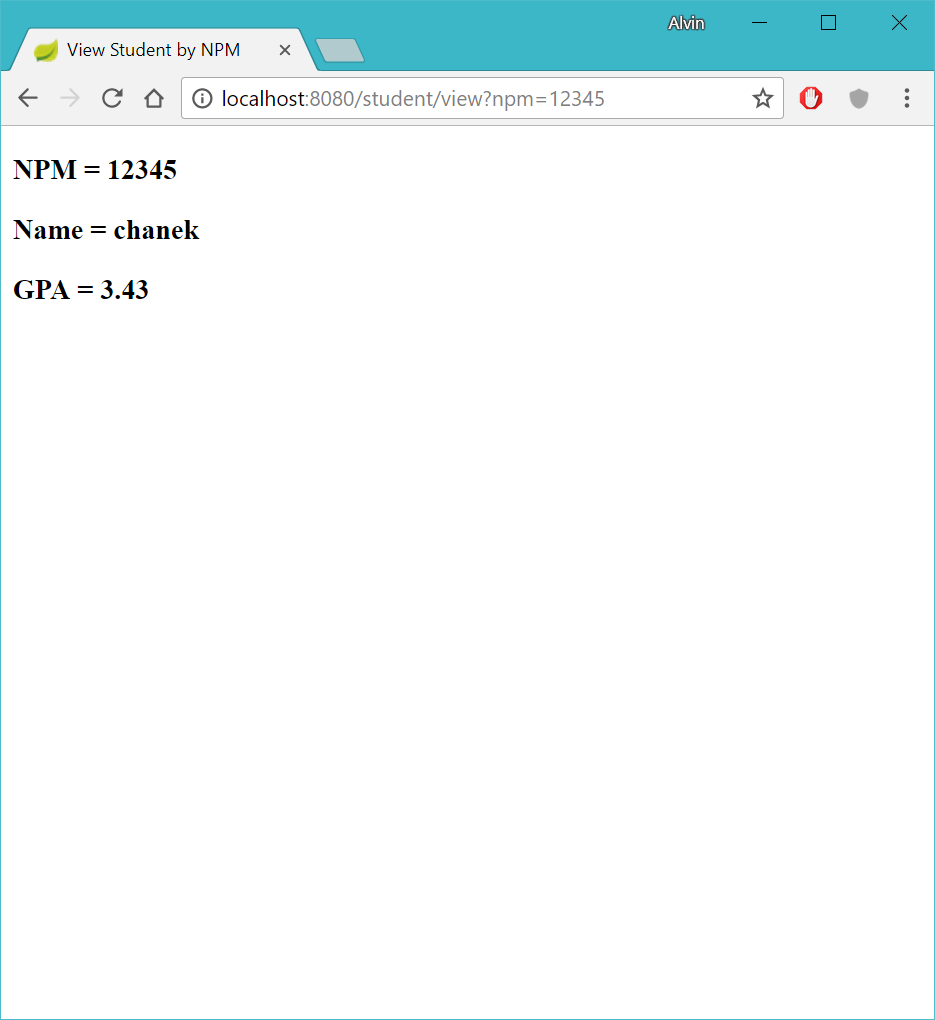
}

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? **Muncul**

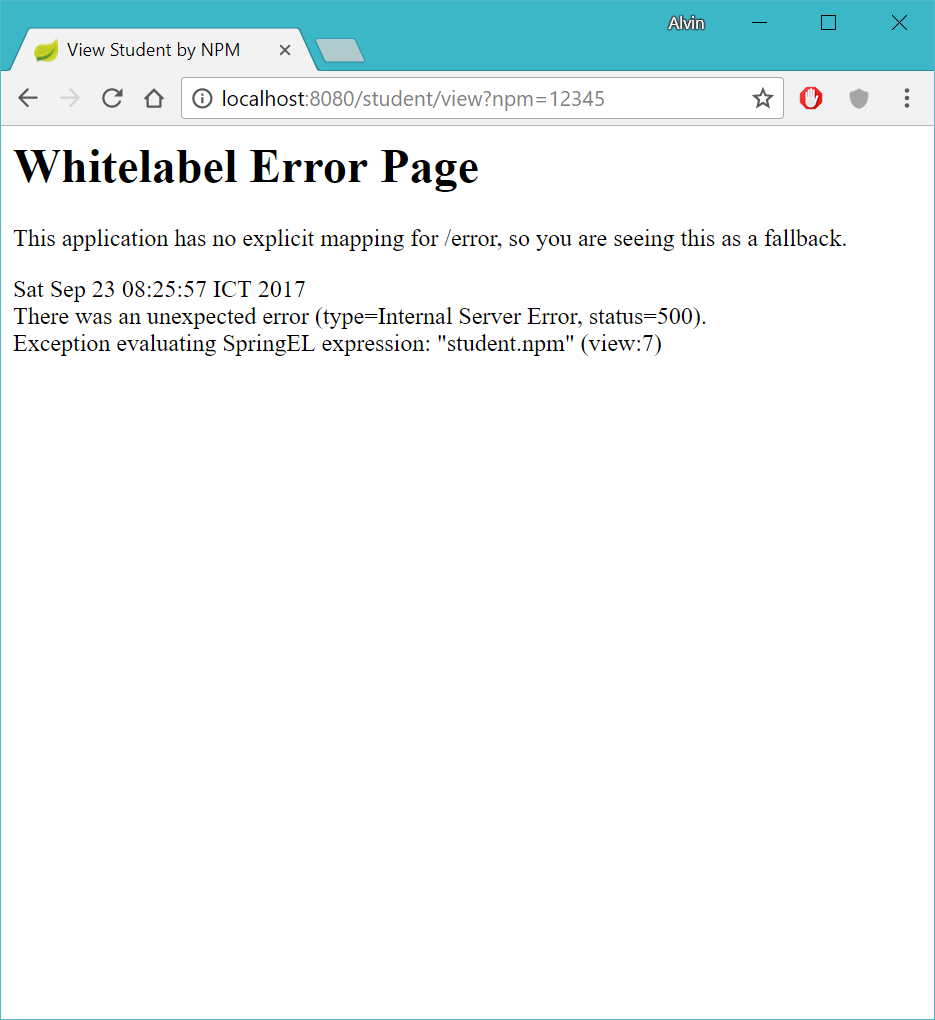


Coba matikan program dan jalankan kembali serta buka

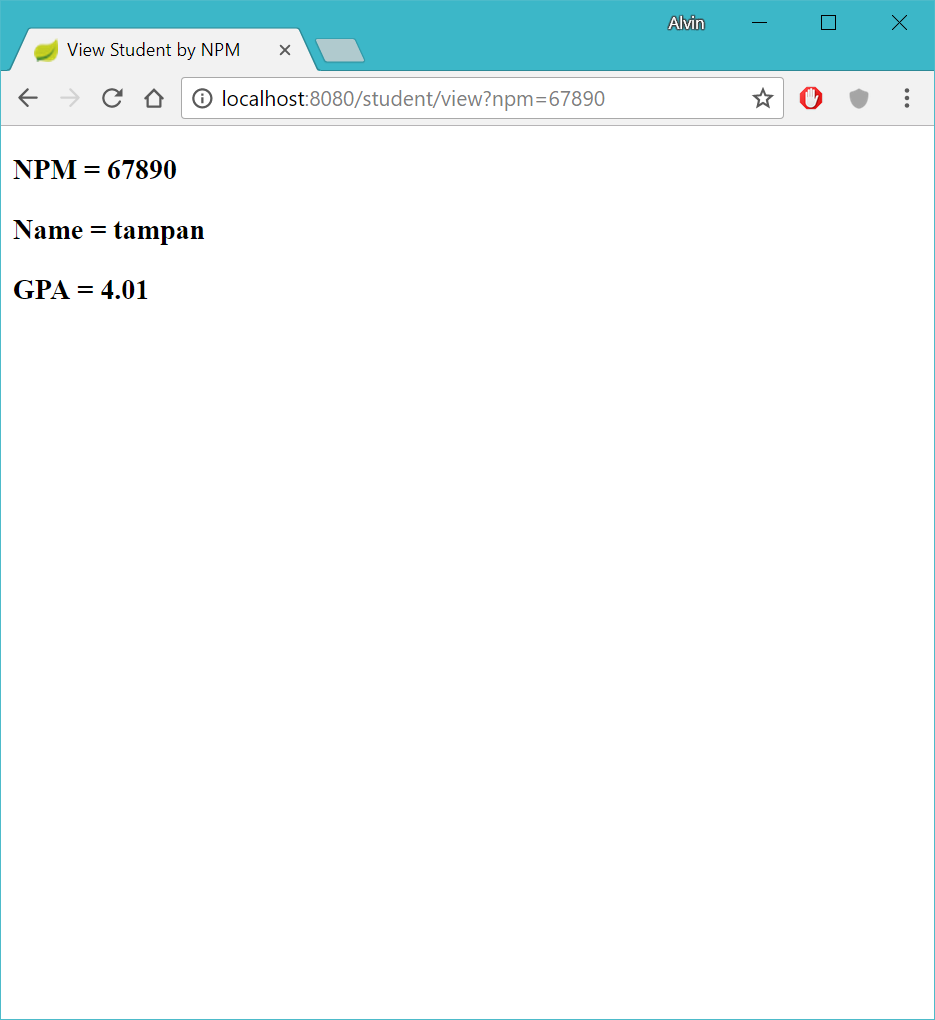
Localhost:8080/student/view?npm=12345

Pertanyaan 4:

* Apakah data Student tersebut muncul? Jika tidak, mengapa? **Tidak muncul. Karena saat view request data melalui Model, model tersebut tidak exist Karena list nya kosong.**



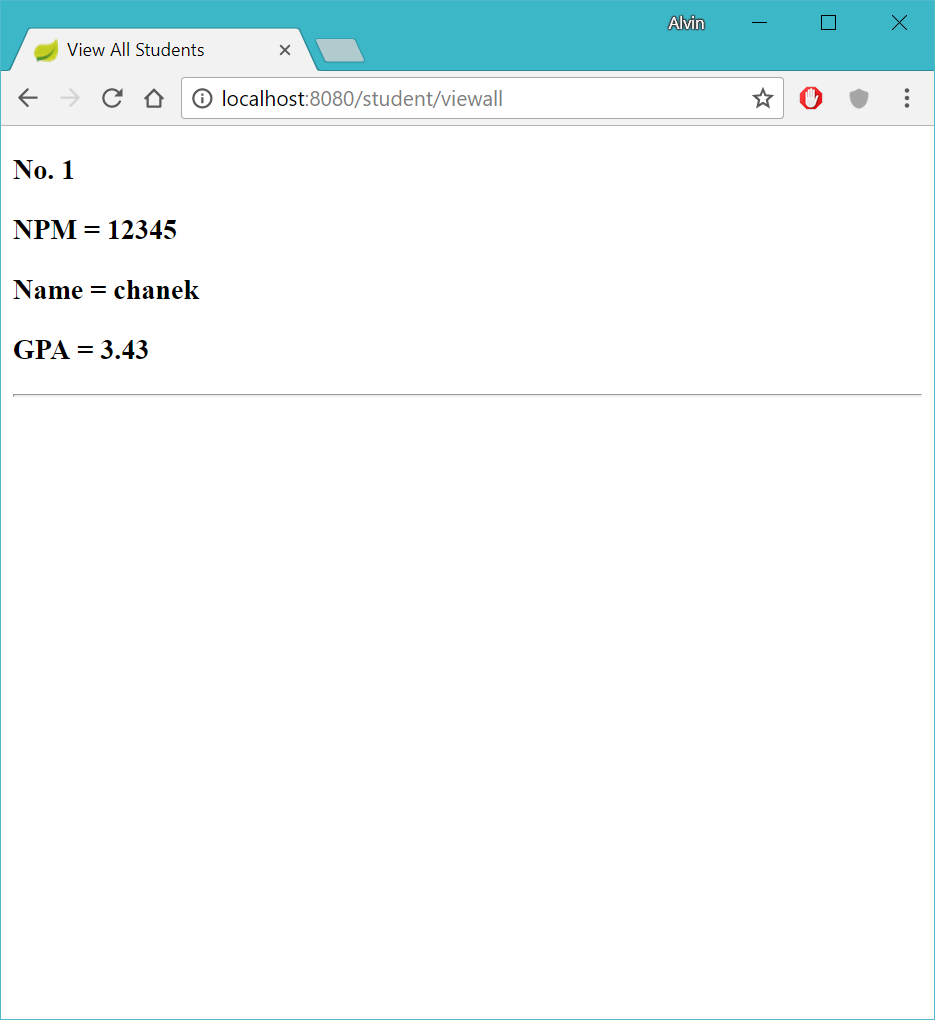
Coba tambahkan data Student lainnya dengan NPM yang berbeda.



3. Jalankan program dan buka

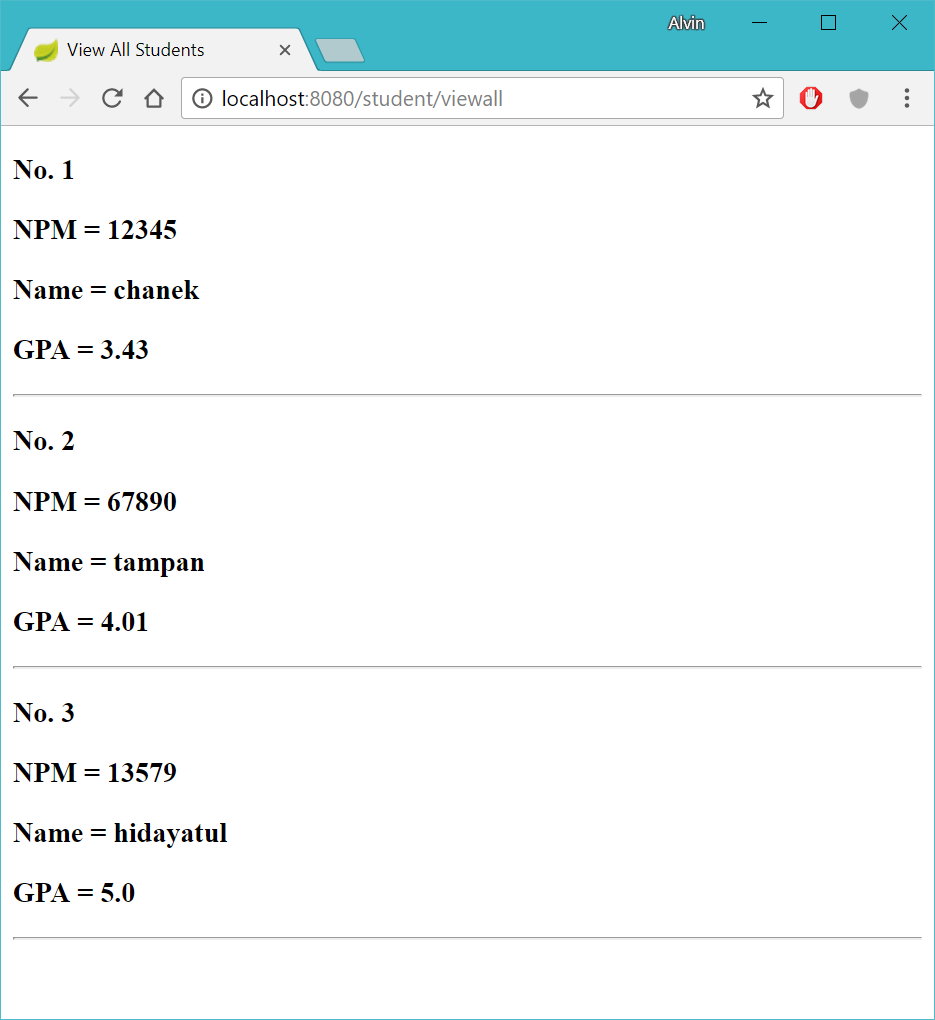
Localhost:8080/student/add?npm=12345&name=chanek&gpa=3.43 lalu buka localhost:8080/student/viewall

Pertanyaan 5: apakah data Student tersebut muncul? **Muncul**



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

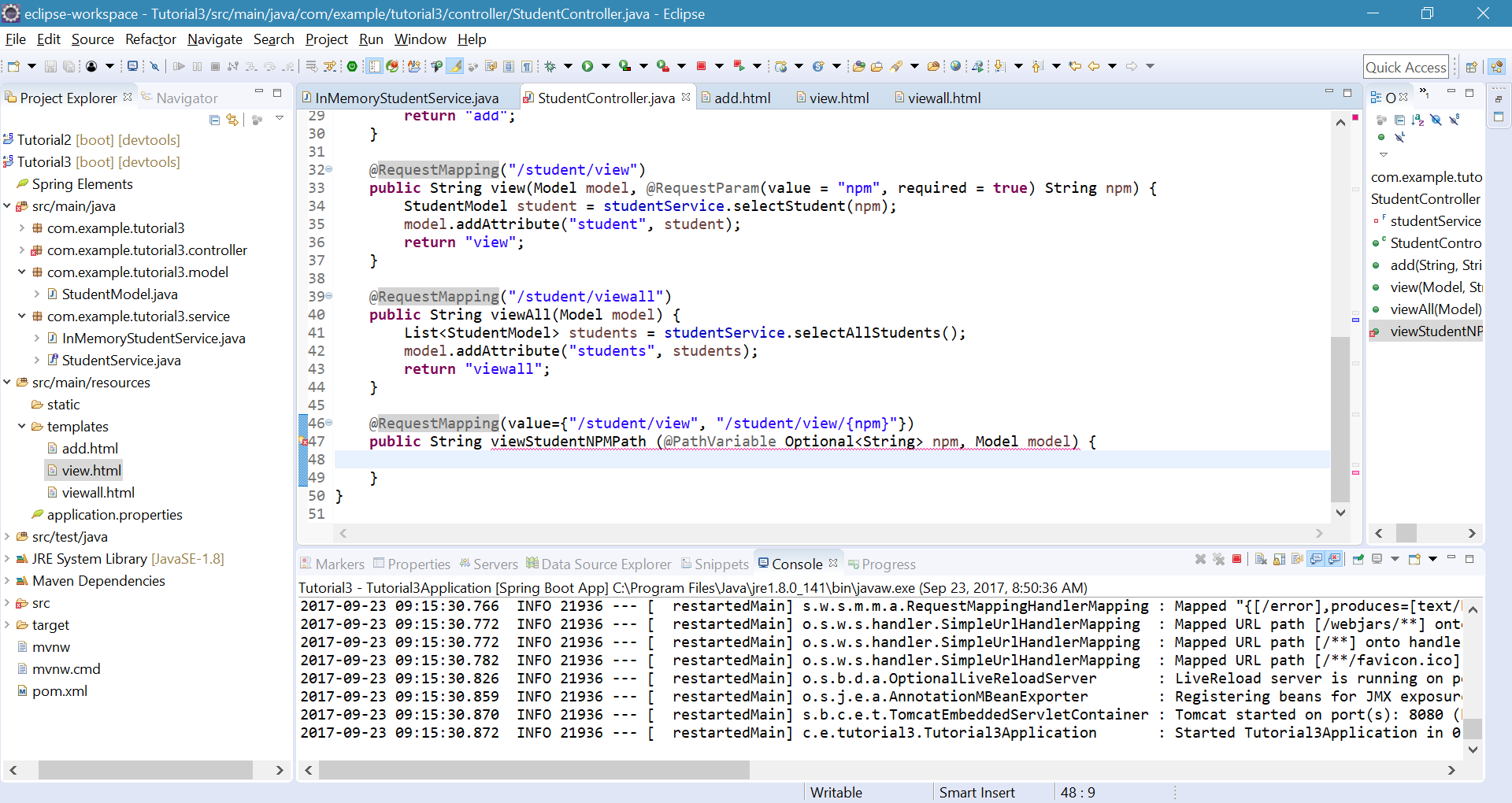
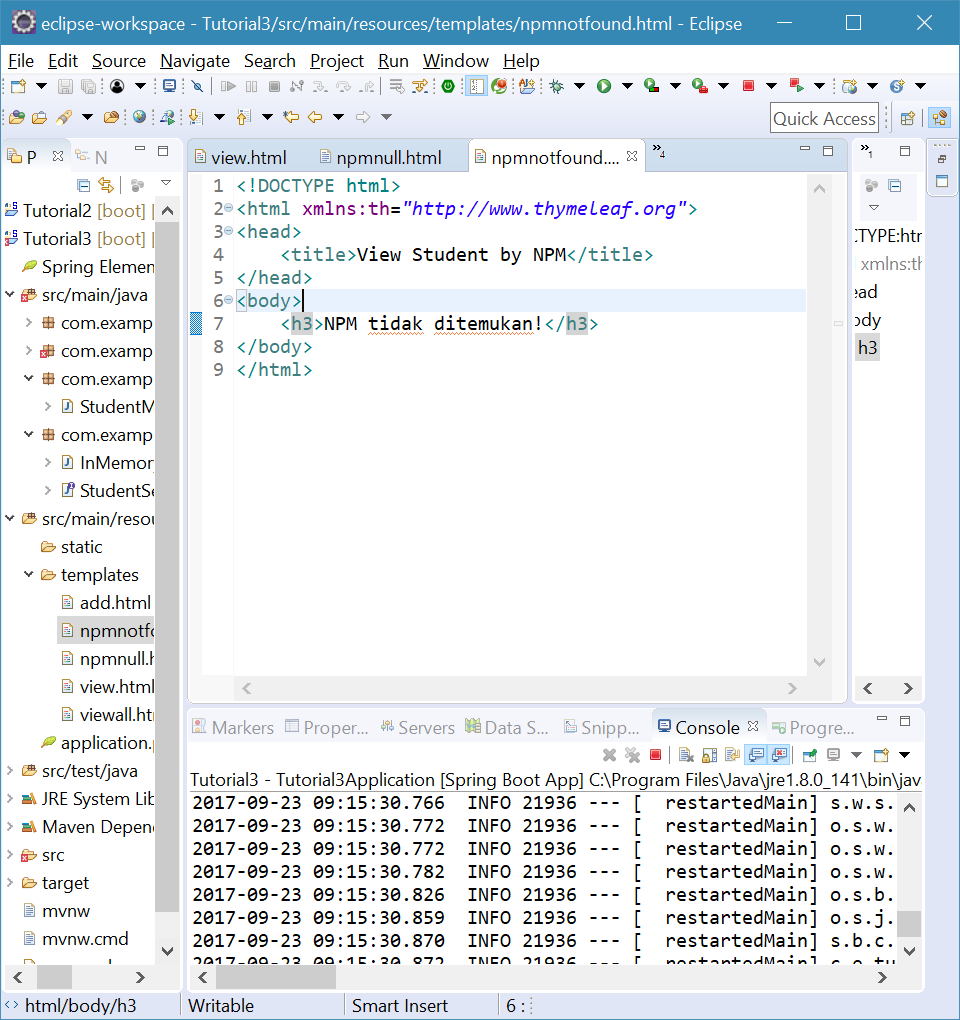
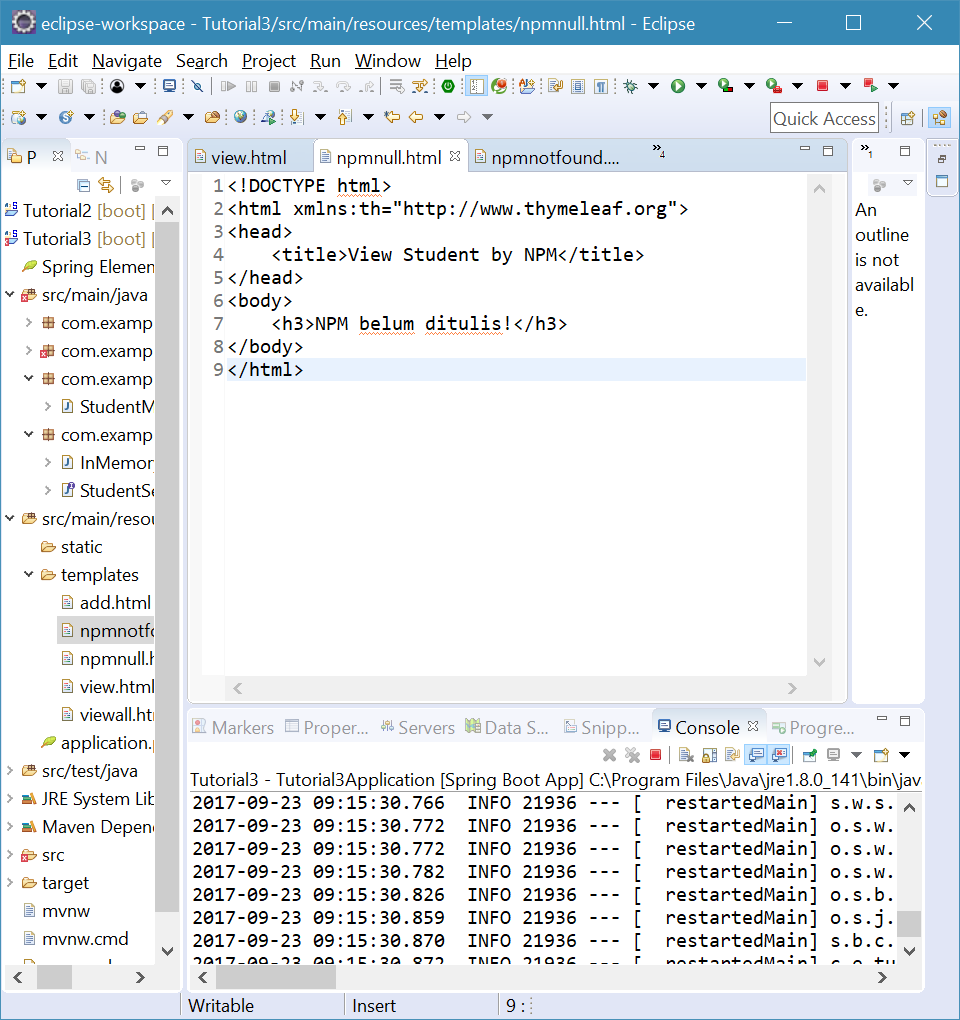
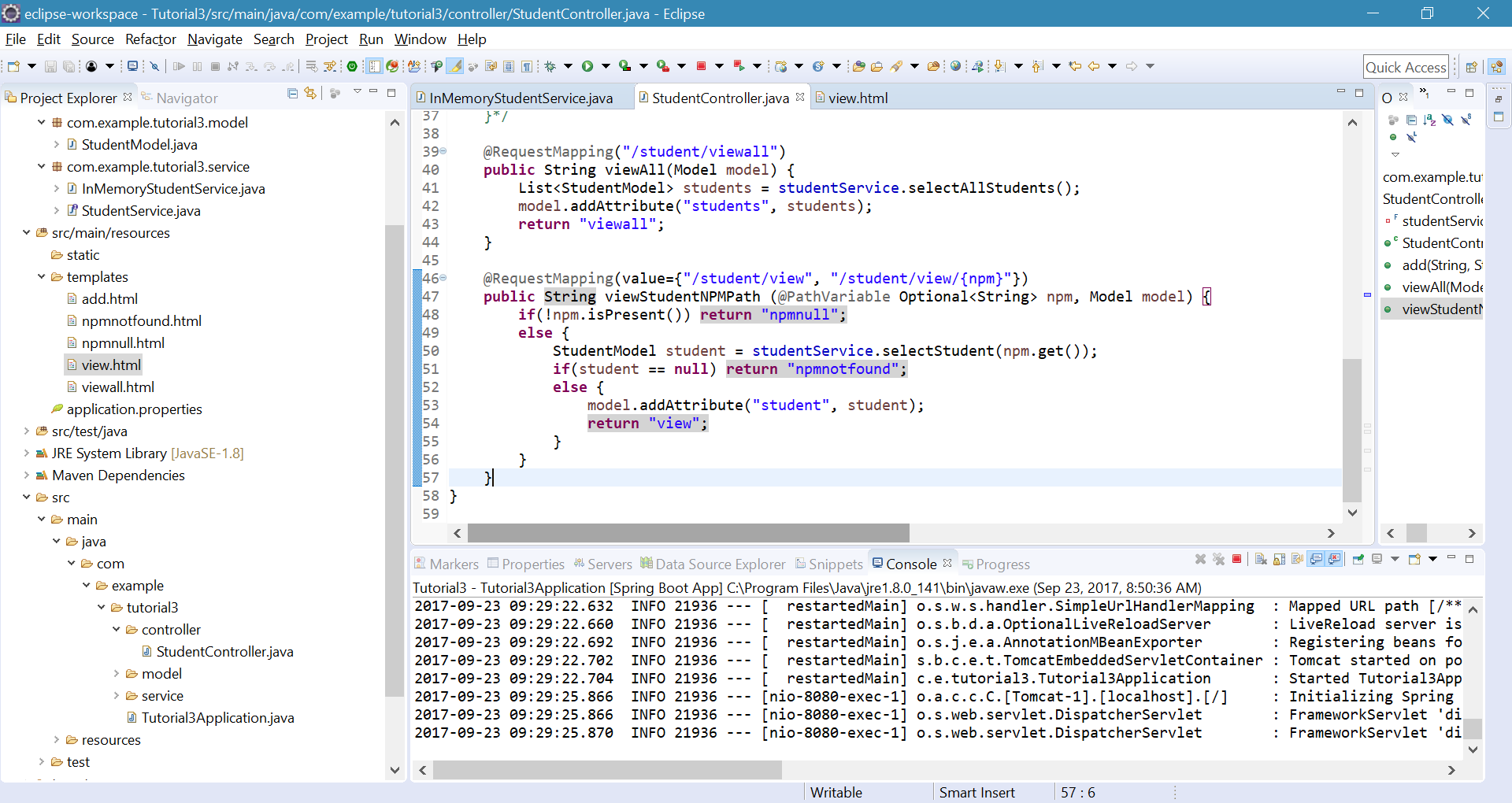
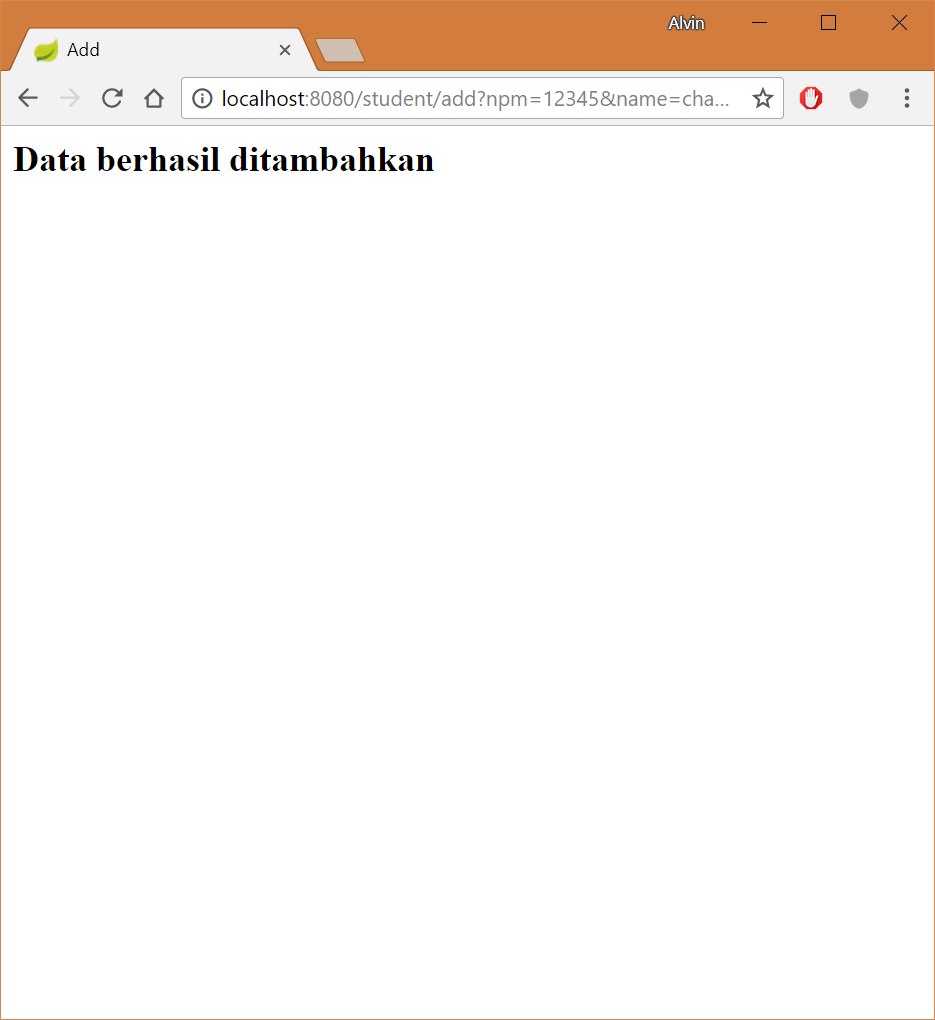
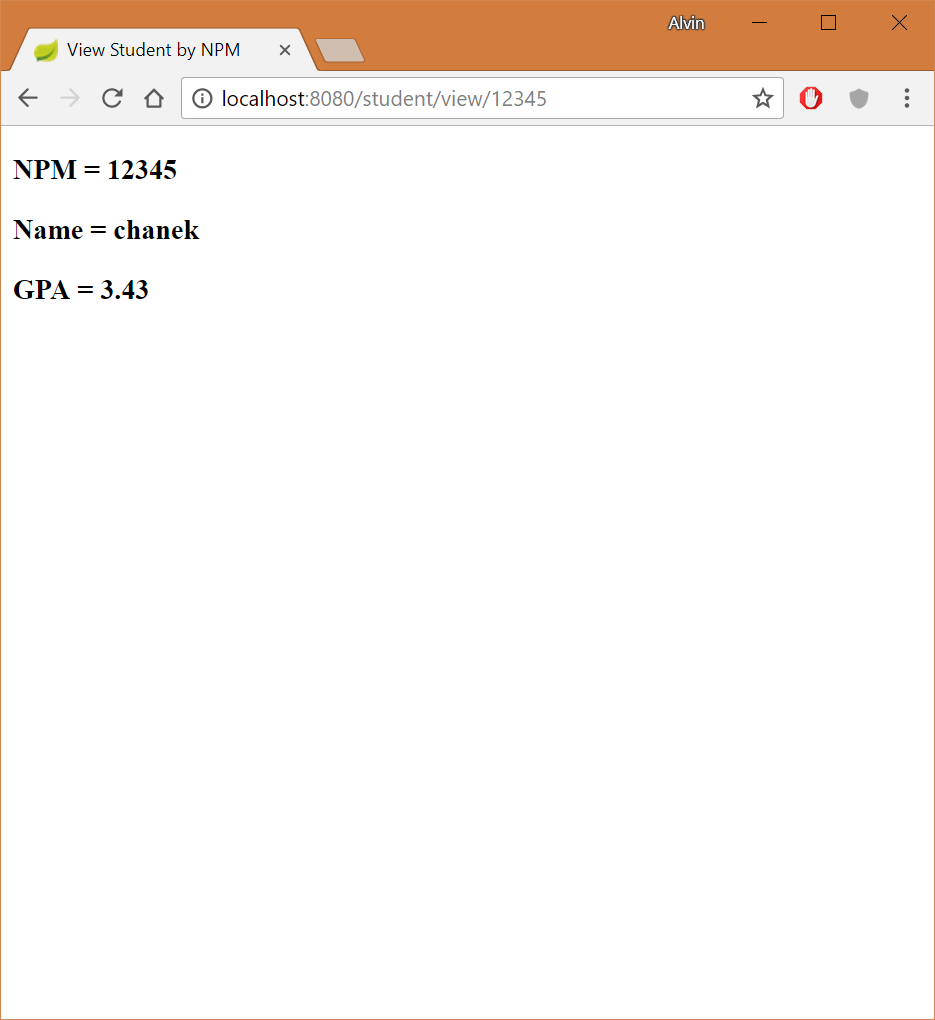
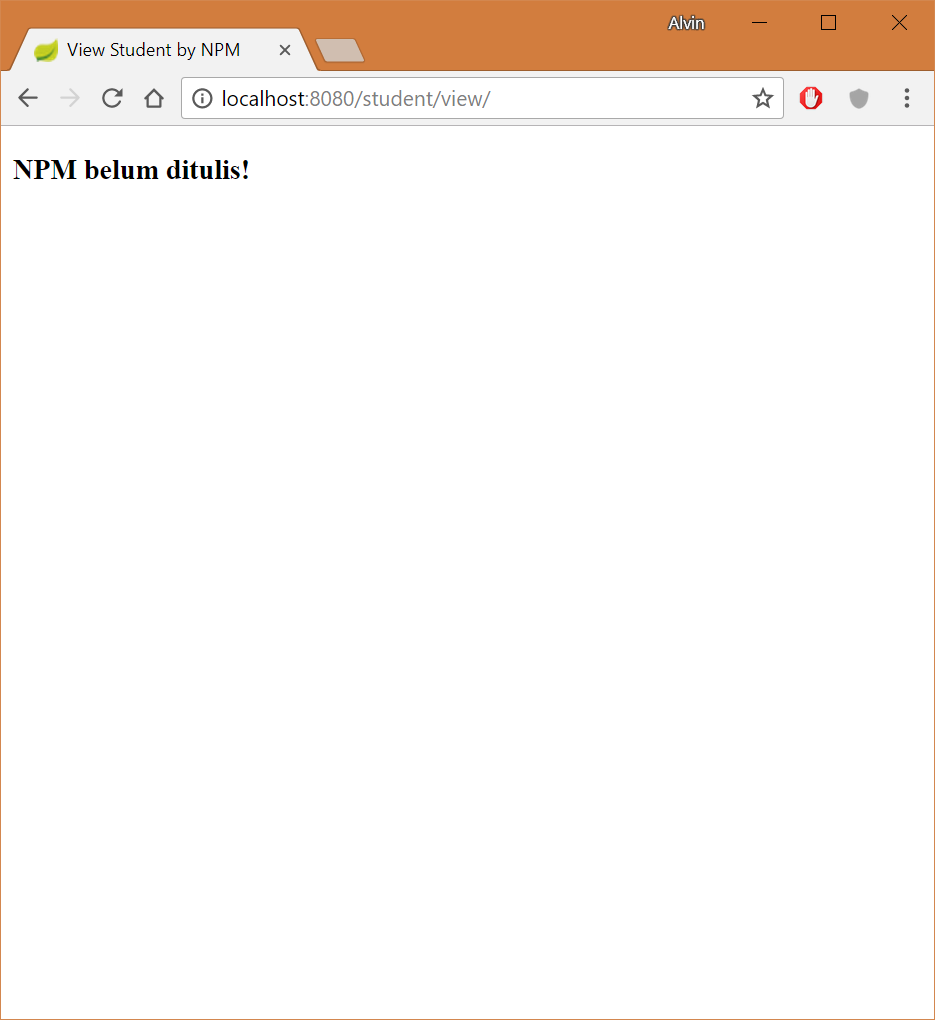
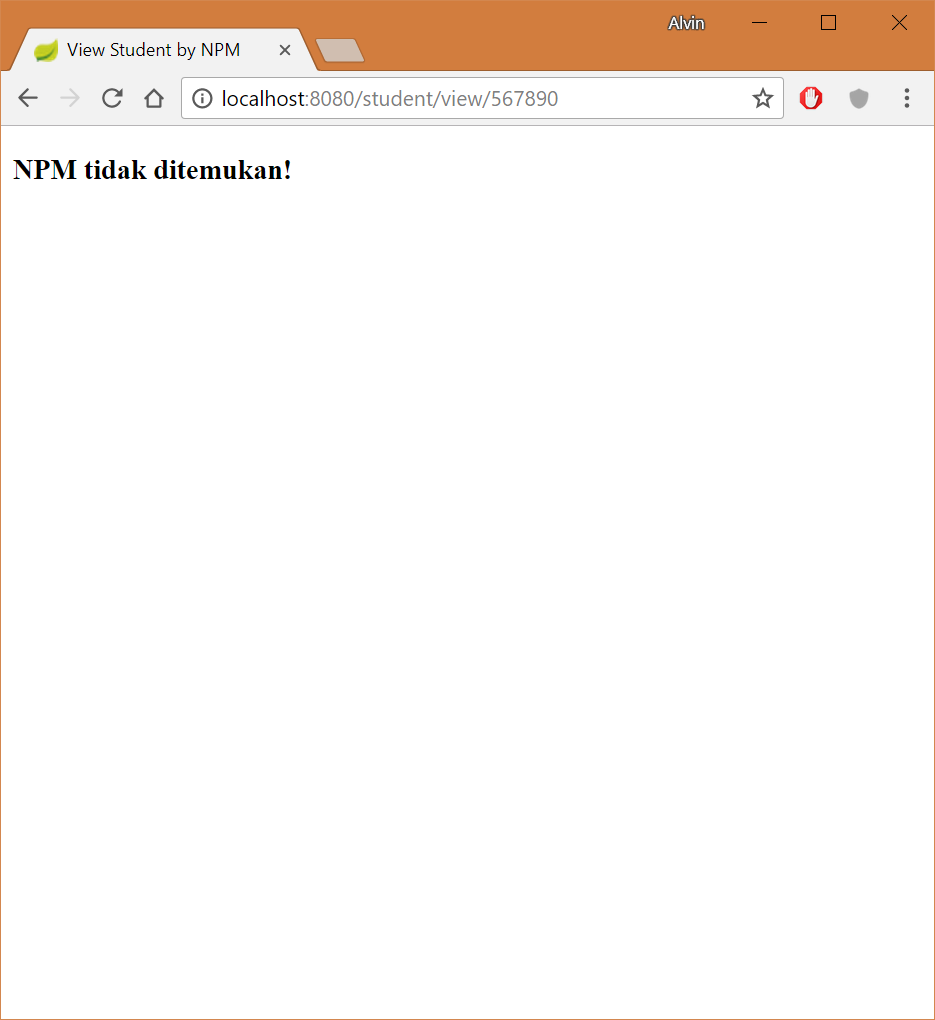
Pertanyaan 6: Apakah semua data Student muncul? **Muncul**



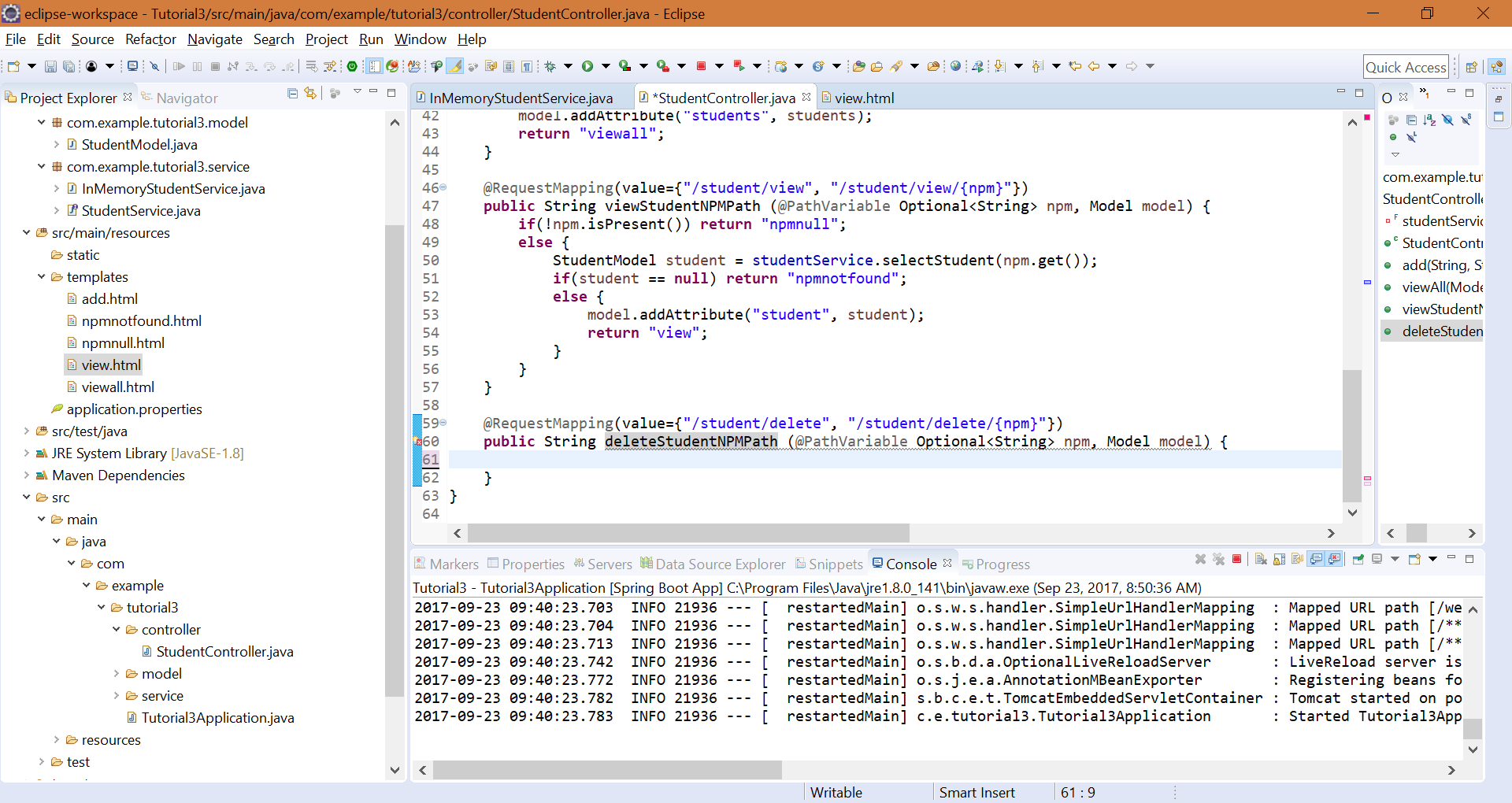
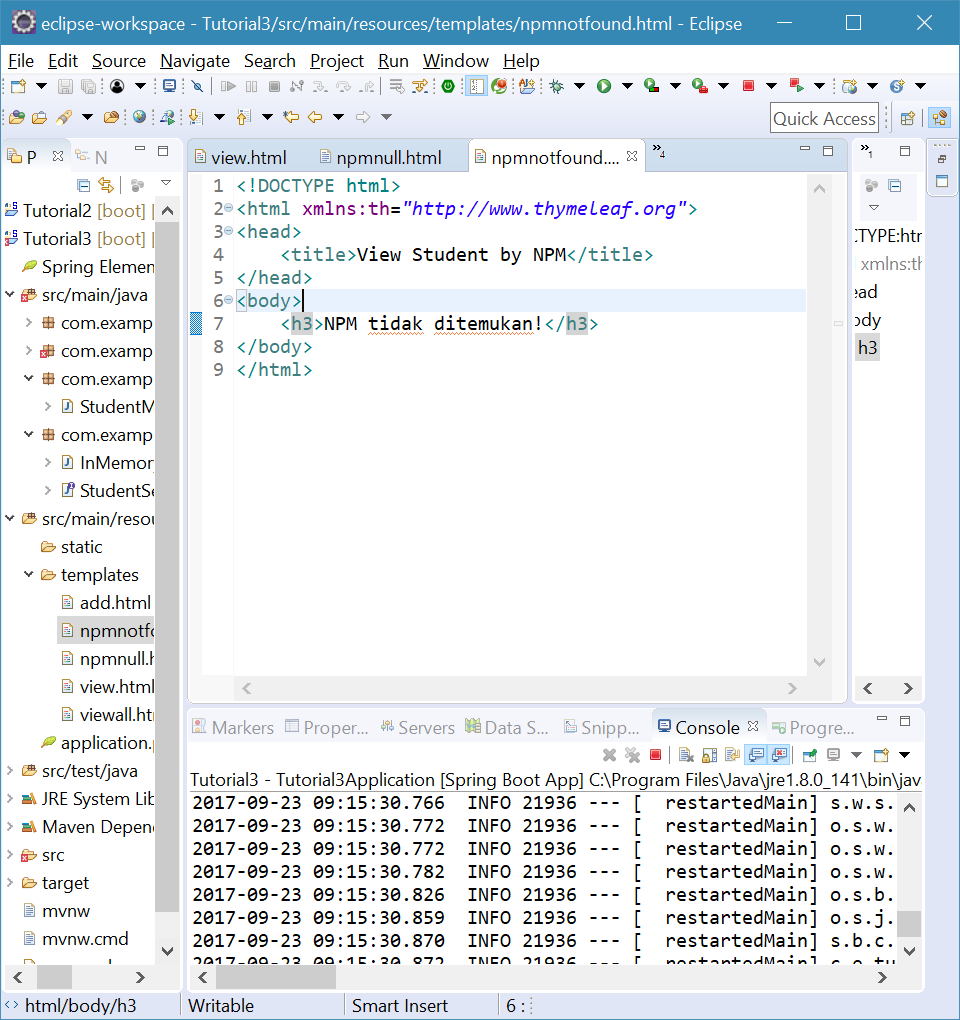
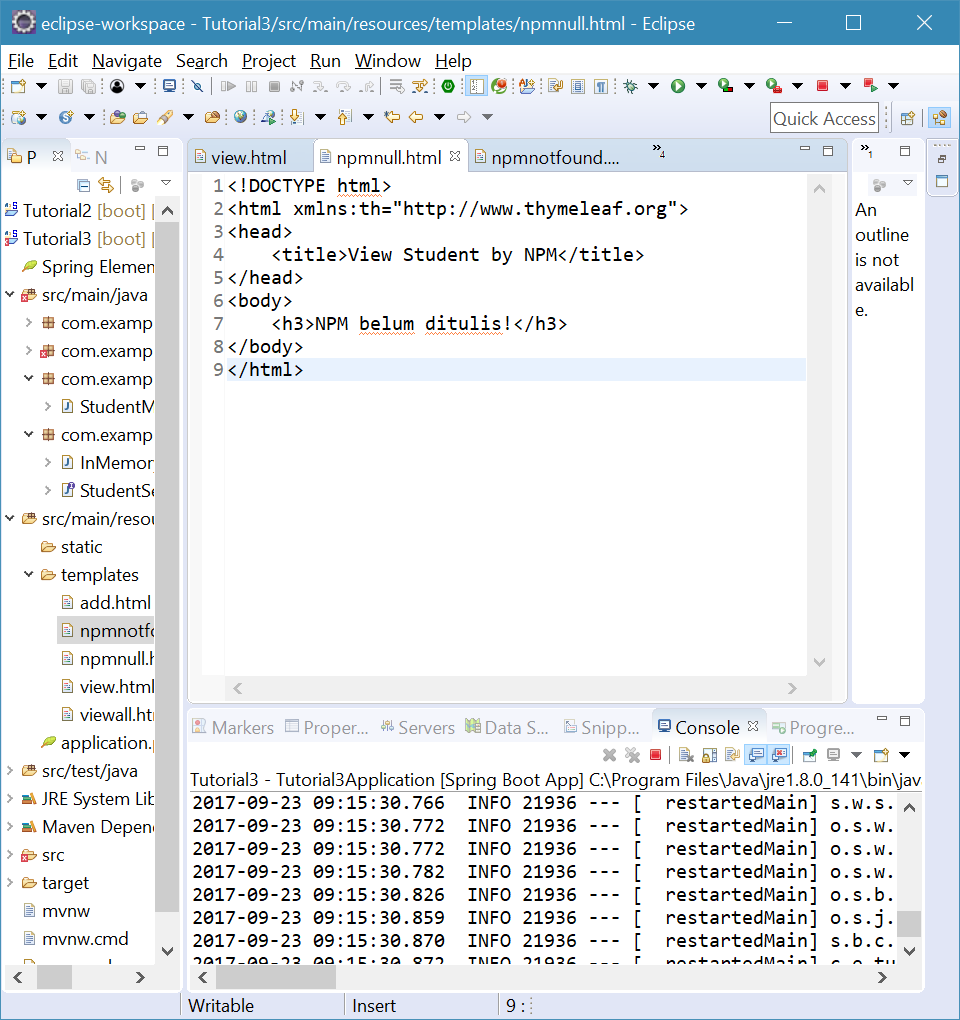
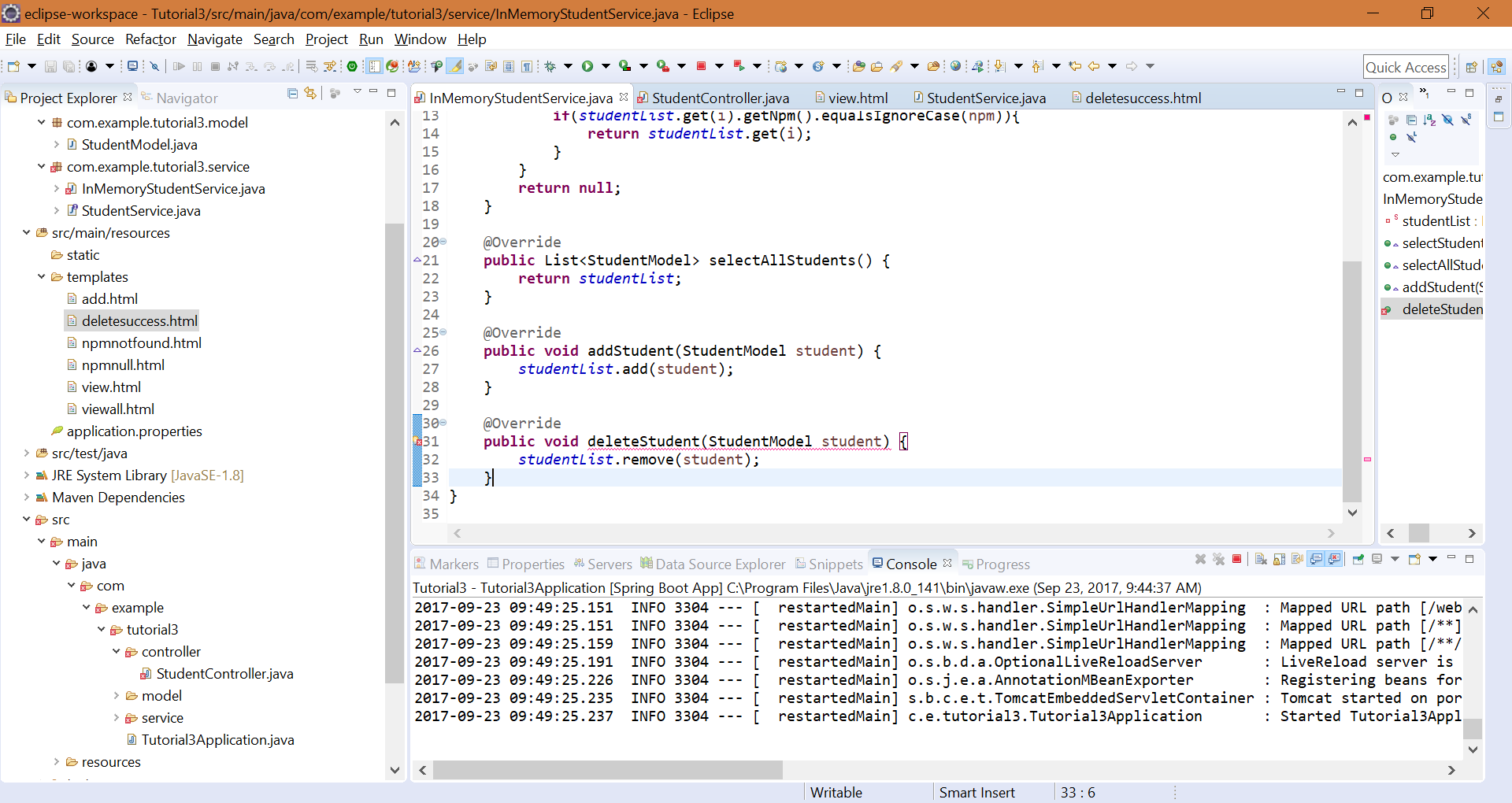
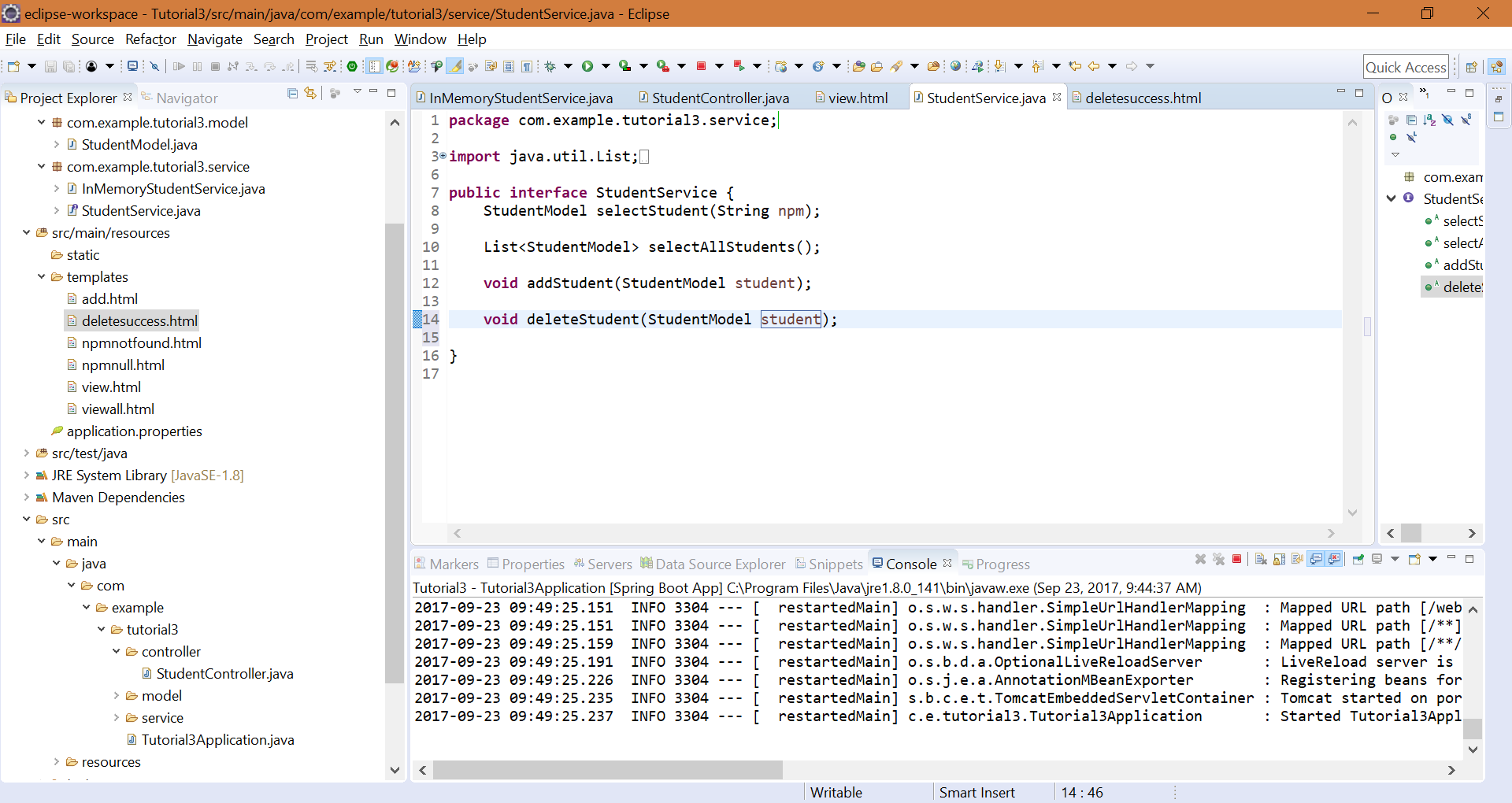
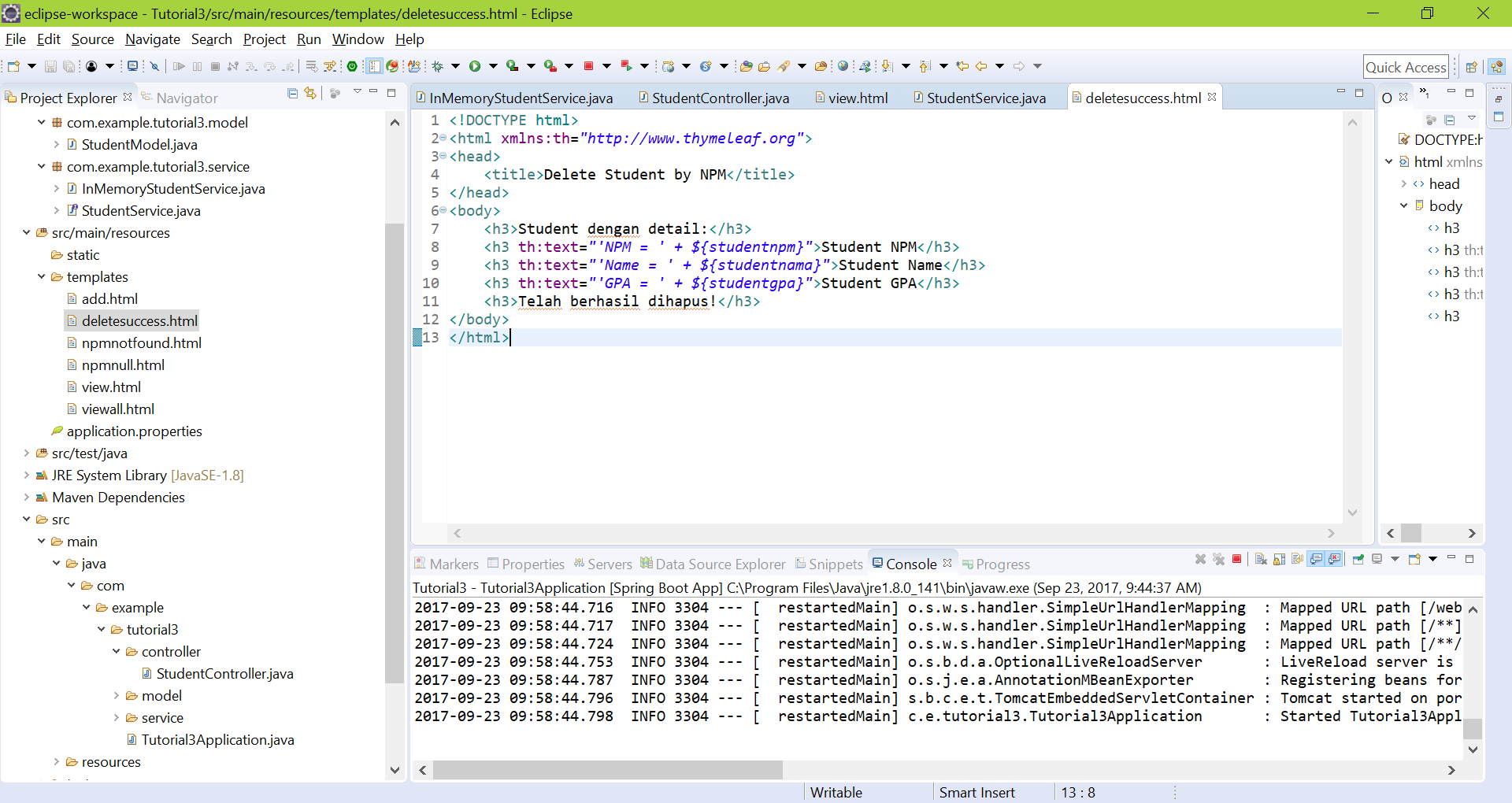
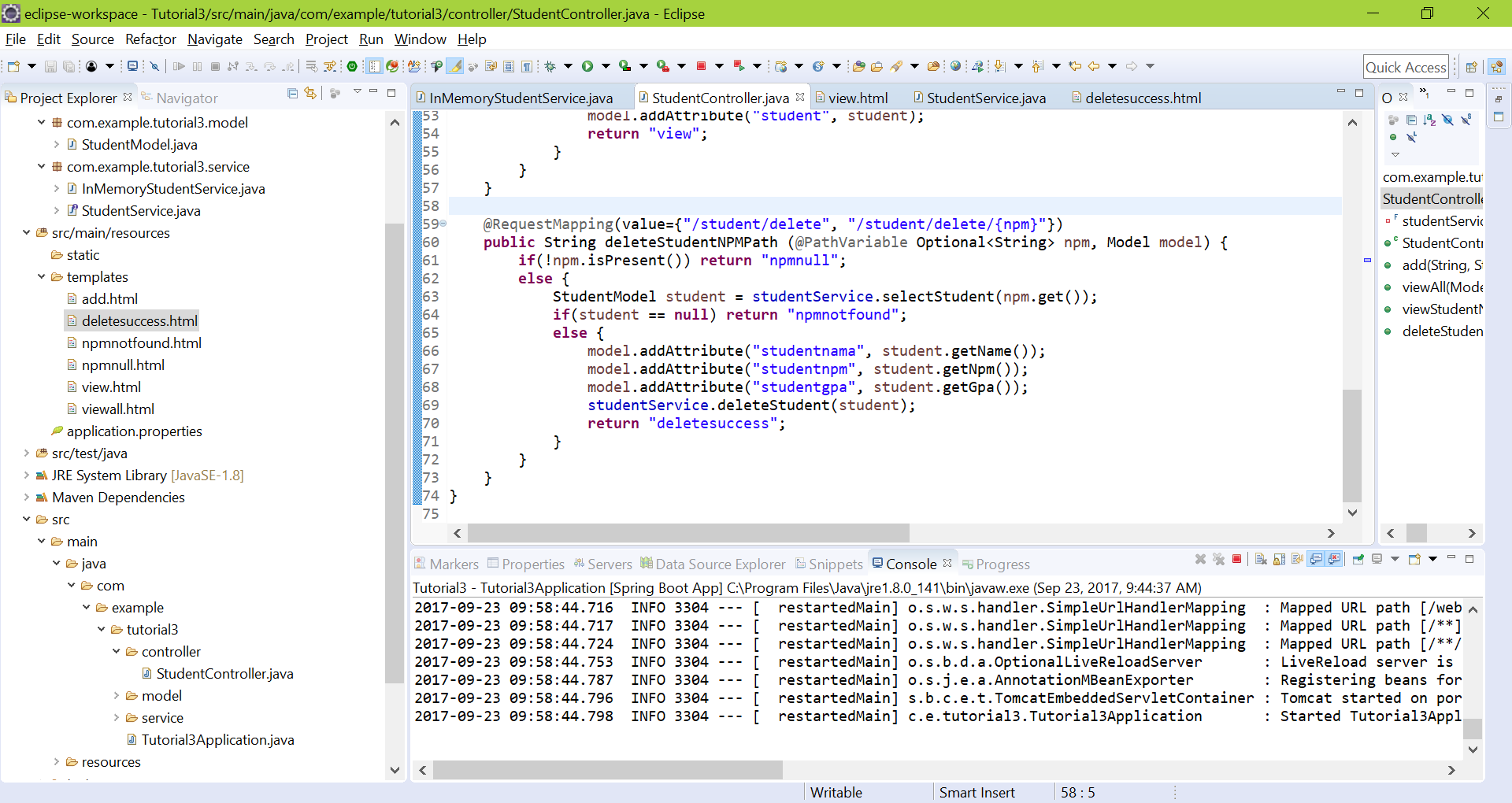
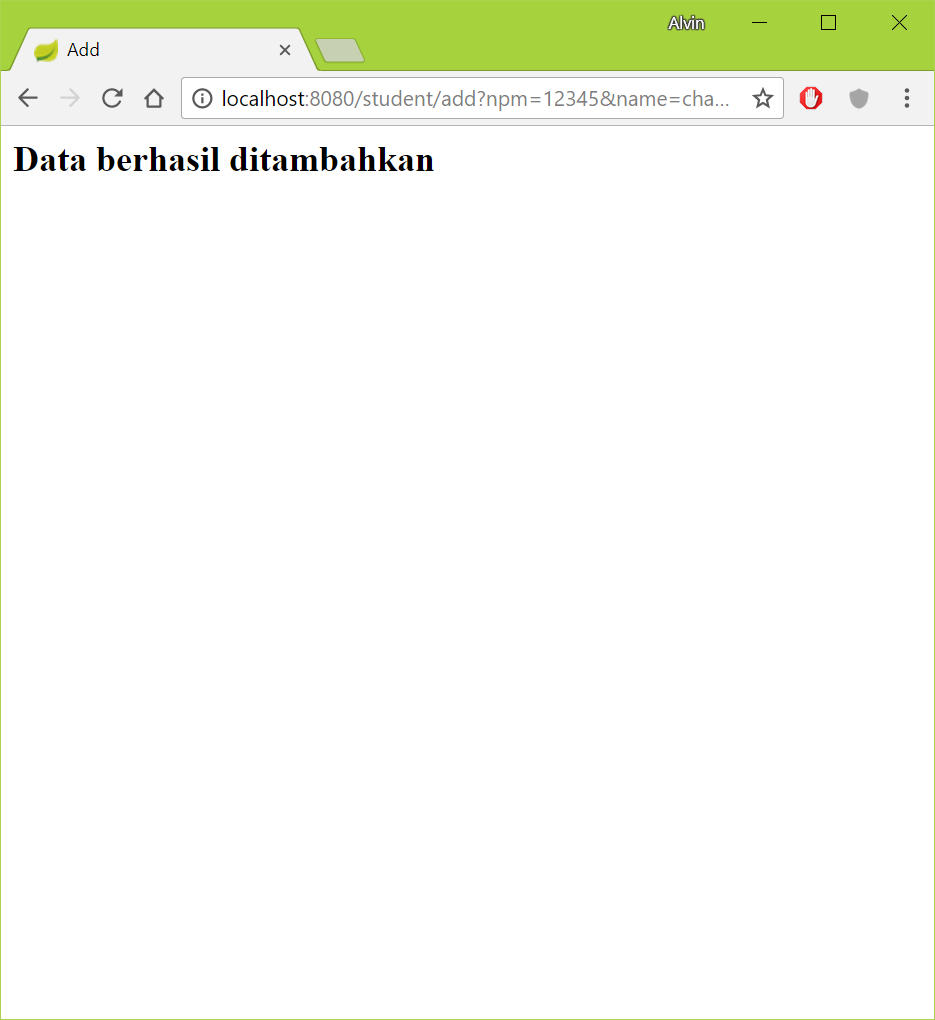
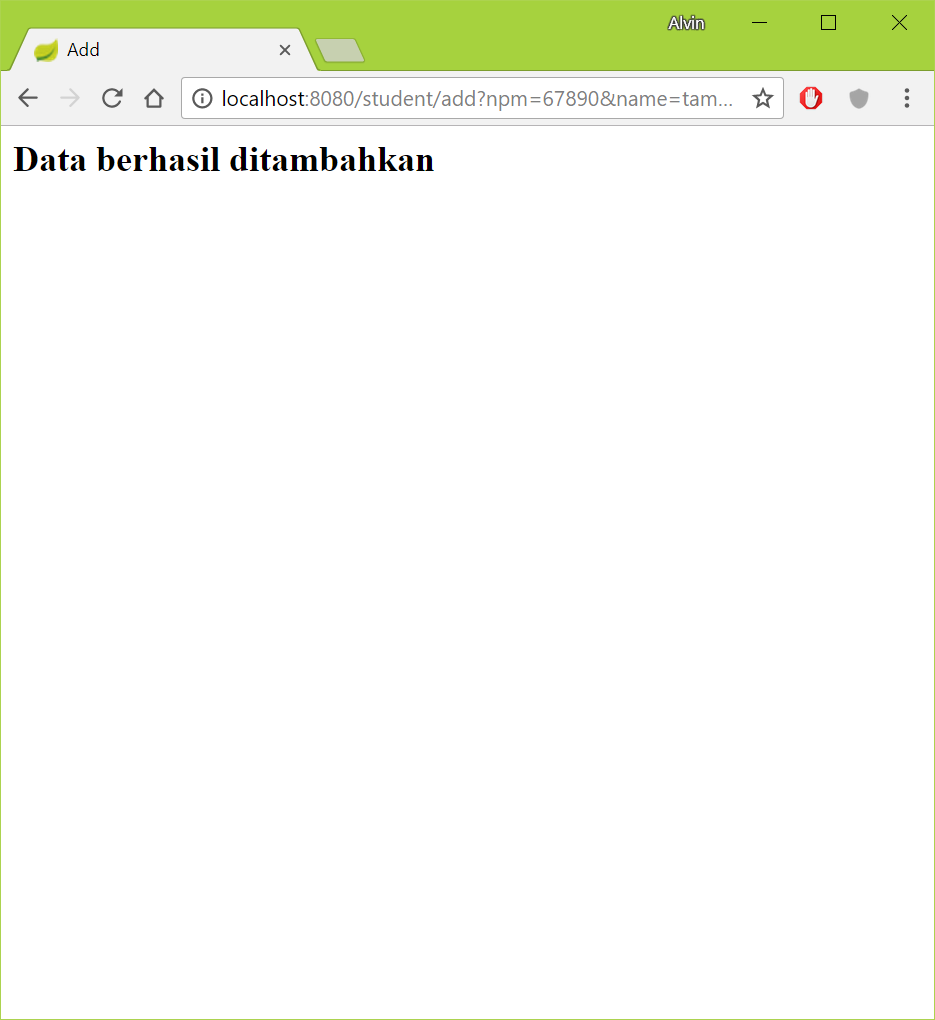
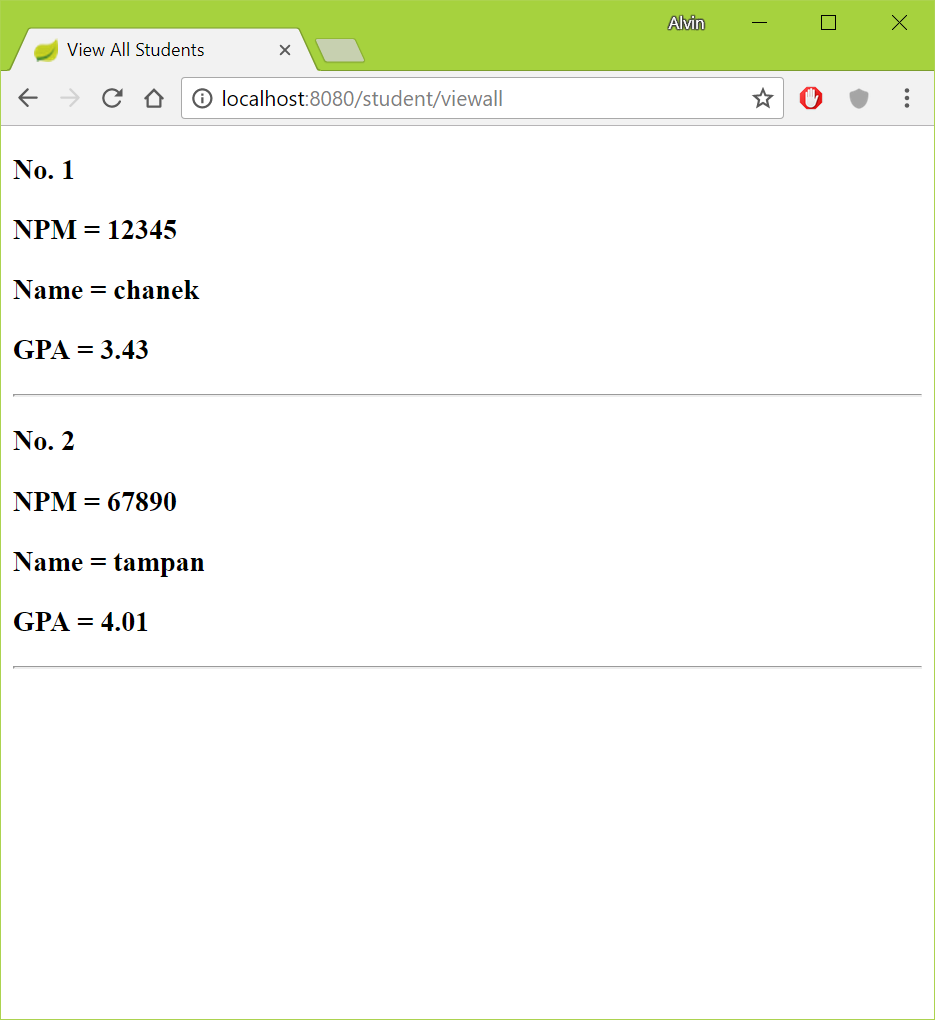
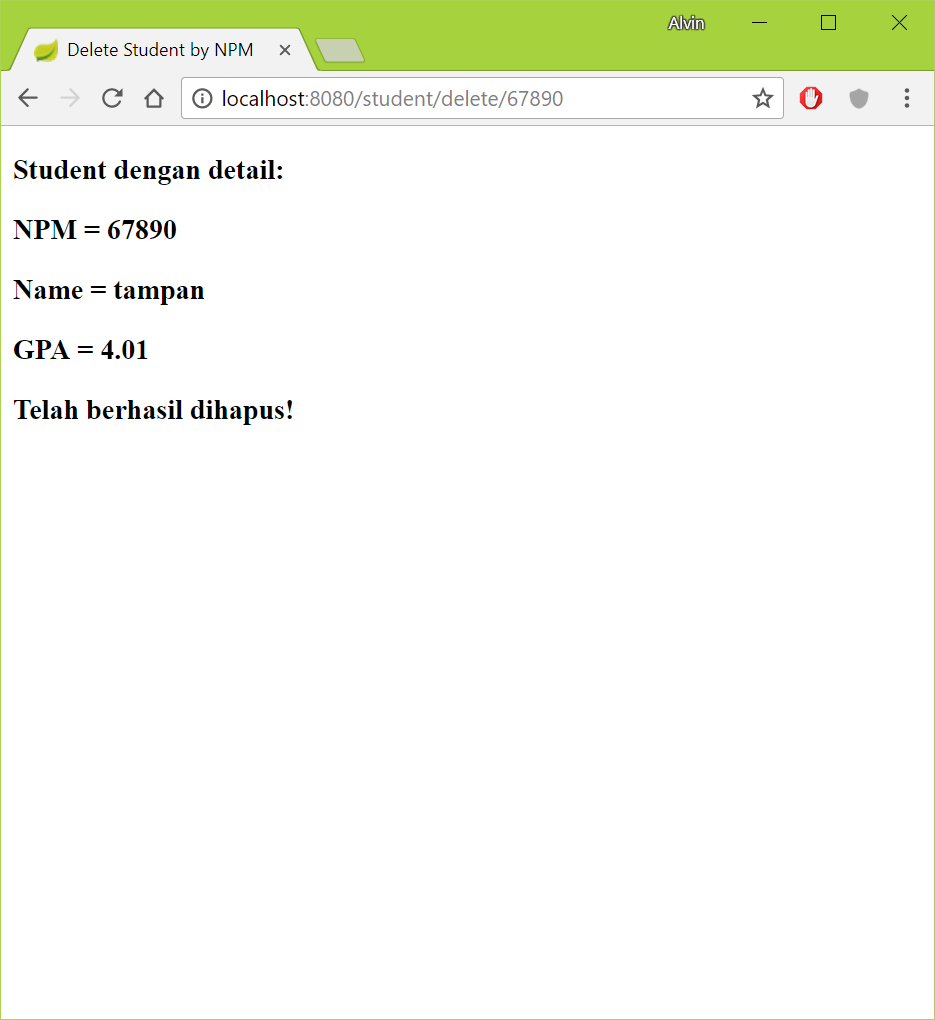
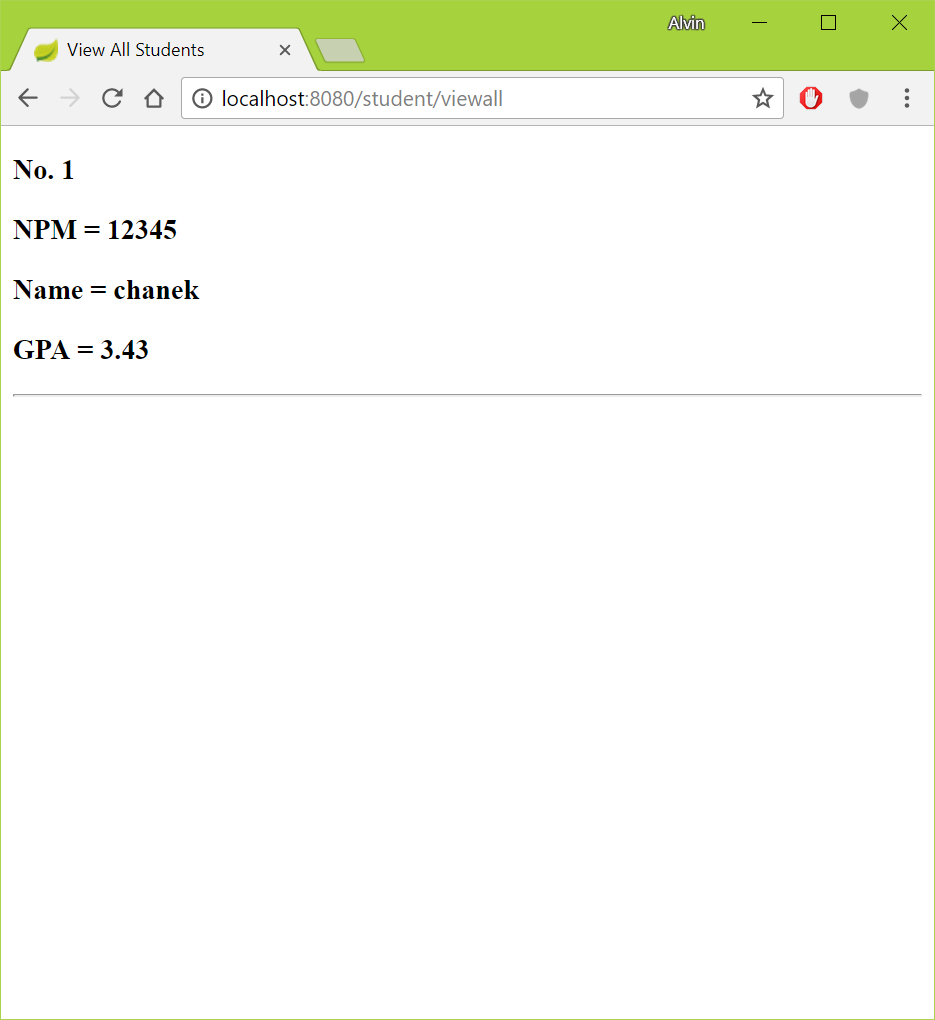
**Latihan**

1. Pada **StudentController** tambahkan sebuah method view Student dengan menggunakan **Path Variable**. Jika nomor NPM tidak diberikan atau tidak ditemukan kembalikan halaman error yang berisi informasi bahwa nomor NPM kosong atau tidak ditemukan.

Langkah:

* Buat method baru terlebih dahulu yang menerima Path Variable
  + 
* Buat 2 halaman yang akan di-return jika terjadi kondisi error. Yaitu yang pertama pada saat NPM tidak diberikan atau NPM tidak ditemukan
  + 
* Implementasi method viewStudentNPMPath
  + 
* Jalankan program. Tambahkan satu entri terlebih dahulu.
  + 
* Coba lihat data yang baru saja dientri menggunakan *path variable*
  + 
* Jika NPM tidak disertakan, maka akan muncul halaman berikut
  + 
* Jika NPM tidak ditemukan, maka akan muncul halaman berikut
  + 

2. Tambahkan fitur untuk melakukan delete Student berdasarkan NPM. Jika nomor NPM tidak diberikan atau tidak ditemukan kembalikan halaman error yang berisi informasi bahwa nomor NPM kosong atau tidak ditemukan dan proses delete dibatalkan.

* Buat method baru terlebih dahulu yang menerima Path Variable
  + 
* Buat 2 halaman yang akan di-return jika terjadi kondisi error. Yaitu yang pertama pada saat NPM tidak diberikan atau NPM tidak ditemukan
  + 
* Saya menambahkan method deleteStudent pada class InMemoryStudentService.java
  + 
* Namun, terjadi error karena di interface StudentService belum terimplementasi method tersebut. Jadi, pada interface tersebut akan saya tambahkan.
  + 
* Kemudian, saya membuat halaman untuk menunjukkan bahwa proses delete berhasil.
  + 
* Lalu implementasi method pada StudentController
  + 
* Jalankan program. Tambahkan beberapa student
  + 
  + 
* Lihat seluruh students
  + 
* Delete salah satu student. Contohnya saya akan delete student tampan melalui NPM nya (67890)
  + 
* Saat melihat kembali daftar students, student tampan tidak ada lagi karena sudah terhapus
  + 
* Penjelasan Fitur: Method delete yang saya implementasi yaitu menggunakan fitur remove() pada interface List, dengan gabungan implementasi Spring agar bisa men-delete entri melalui URL.

**Ringkasan**:

Yang telah saya pelajari melalui tutorial 3 APAP adalah saya telah mengetahui bagaimana class Model dalam MVC. Selain itu, saya juga mempelajari bagaimana menggunakan interface dengan MVC. Selain itu, saya juga mempelajari bagaimana data bisa tersimpan pada Springboot yang bisa mengkolaborasikan List dengan Springboot agar bisa mengakses melalui URL.