Ringkasan

Pada tutorial kali ini, saya belajar menggunakan logging untuk melakukan debugging suatu method. Saya belajar menjalankan query sql untuk langsung melakukan input, update, dan delete data dari database

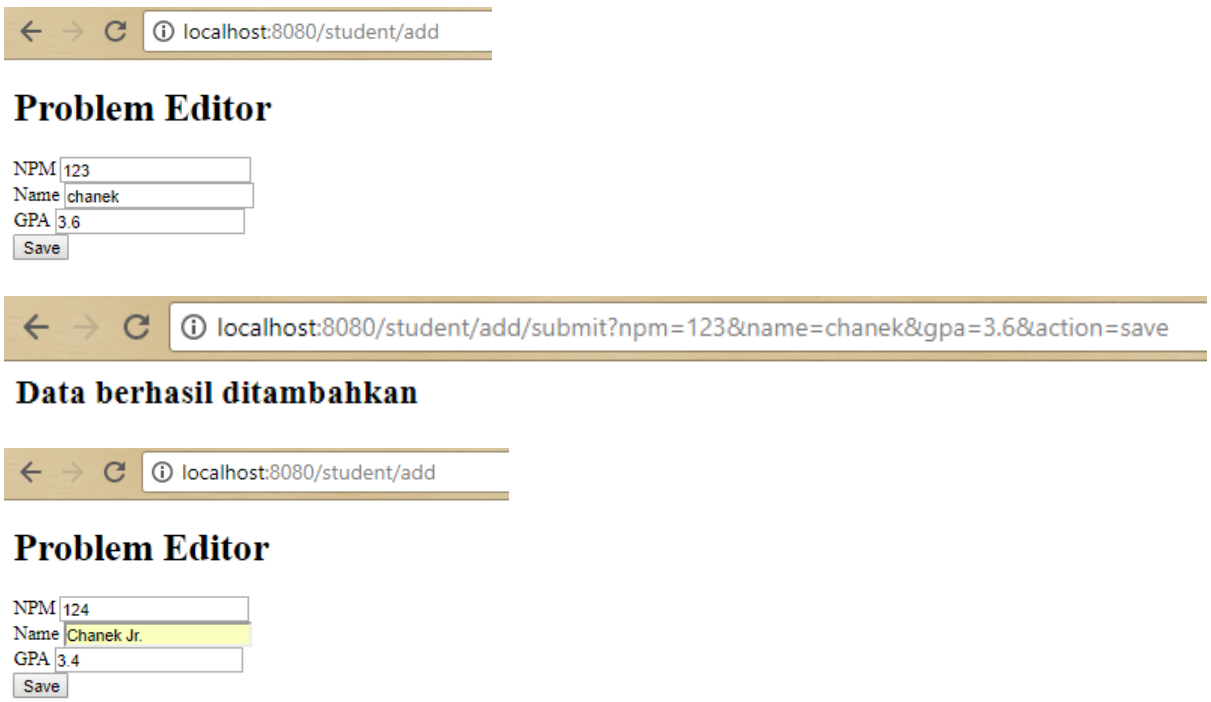
Pertanyaan

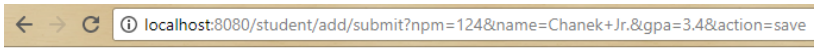
1. Jika menggunakan Object sebagai parameter pada form POST, bagaimana caranya melakukan validasi input yang optional dan input yang required?
   Apakah validasi diperlukan? Asumsikan input pada form Anda tidak menggunakan attribute required sehingga butuh validasi di backend.

   Dapat dengan menggunakan @Valid dan BindingRequest yang akan melakukan pengecekan apabila terjadi error atau tidak. Untuk validasi dapat dilakukan dengan menggunakan if statement untuk menanggulangi setiap validasi yang diinginkan.

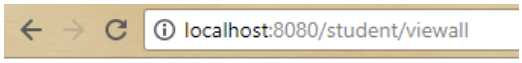
2. Menurut Anda, mengapa form submit biasanya menggunakan POST method dibanding GET method? Apakah perlu penanganan berbeda di header atau body method di controller jika form di post dikirim menggunakan method berbeda?
   Karena apabila menggunakan POST, hasil dari form yang diinput akan muncul pada link / url setelah submit.
   Secara best practice, diperlukan dan dapat dilakukan dengan RequestMethod
3. Apakah mungkin satu method menerima lebih dari satu jenis request method, misalkan menerima GET sekaligus POST?
   Tidak, method GET dan method POST memiliki ide yang bertolak belakang

Latihan Menambahkan Delete





Data berhasil ditambahkan



All Students

No. 1

NPM = 123

Name = chanek

GPA = 3.6

Delete Data
Update Data

No. 2

NPM = 124

Name = Chanek Jr.

GPA = 3.4

Delete Data
Update Data

Data berhasil dihapus

All Students

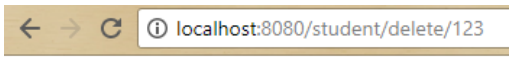
No. 1

NPM = 124

Name = Chanek Jr.

GPA = 3.4

Delete Data
Update Data

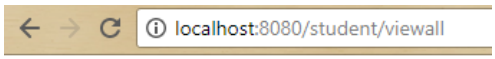


Student not found

NPM = 123

Pada method delete yang baru, saya menambahkan method deleteStudent dan memberikan annotation @Delete untuk menjalankan query ke database apabila method deleteStudent dipanggil. Apabila npm tidak ditemukan, maka akan terlempar ke not-found.html. Sedangkan apabila npm ditemukan, maka akan memanggil method deleteStudent dan akan menghapus object student dari database sesuai dengan npm yang ingin dihapus dan re-direct ke halaman delete.html

Latihan Menambahkan Update



All Students

No. 1

NPM = 124

Name = Chanek Jr.

GPA = 3.4

Delete Data
Update Data

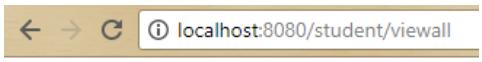
Student Editor

NPM 124
Name Chanek Jr.
GPA 3.4
Save

Student Editor

NPM 124
Name Chanek Jr.
GPA 3.5
Save

Data berhasil diupdate



All Students

No. 1

NPM = 124

Name = Chanek Jr.

GPA = 3.5

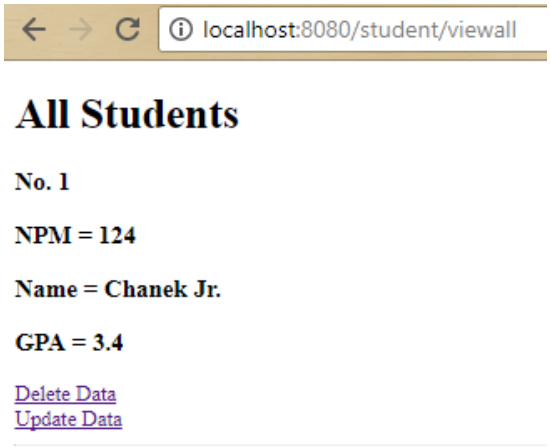
Delete Data
Update Data

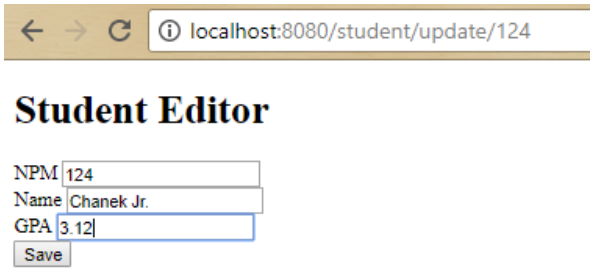
Student not found

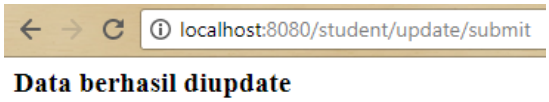
NPM = 123

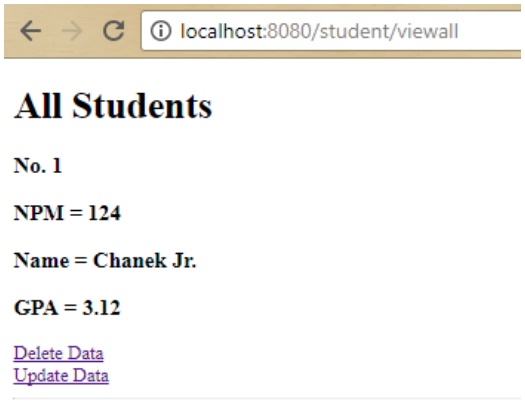
Pada method update, dibuat method updateStident pada StudentMapper dengan parameter StudentModel. Buat annotation @Update untuk menjalankan query update apabila method updateStudent dijalankan. Lalu implementasikan method updateStudent pada StudentService dan StudentServiceDatabase yang akan menjalankan method updateStudent dengan querynya. Lalu lakukan edit pada input di form-update.html sesuai dengan instruksi soal. Jika update berhasil dilakukan maka akan langsung redirect ke success-update.html. Pada method update di StudentController. Lakukan pencarian pada npm yang ingin diupdate, apabila ditemukan maka akan ke redirect ke halaman form-update.html, sebaliknya jika tidak ditemukan maka akan diredirect ke not-found.html. Setelah itu buat method updateSubmit dengan request mapping dan parameter mapping sesuai dengan instruksi. Pada method ini akan dipanggil updateStudent yang akan menjalankan query untuk melakukan update langsung pada database.

Latihan Menggunakan Object Sebagai Parameter









Pada latihan ini, parameter untuk update diubah menjadi object StudentModel. Object dapat dijadikan parameter dengan menggunakan @ModelAttribute. Sehingga setelah menggunakan @ModelAttribute, kita tidak perlu menggunakan parameter dan membuat object student dalam method dan dapat langsung memanggil updateStudent untuk menjalankan query update.