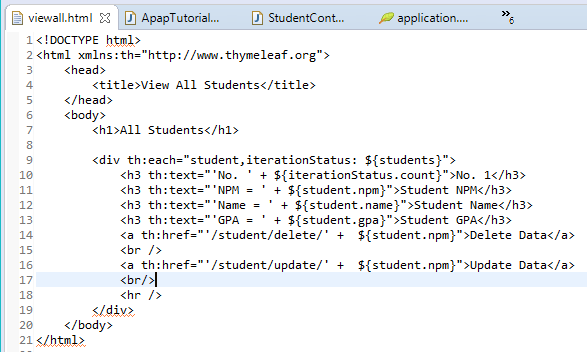
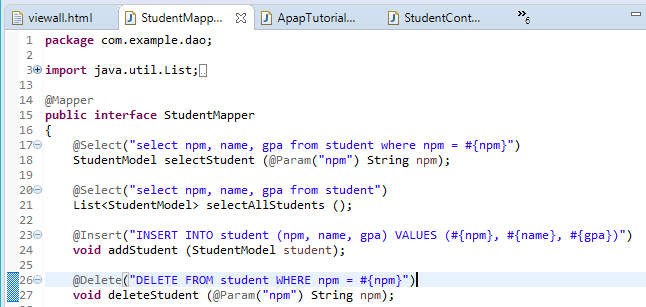
**Latihan​ ​Menambahkan​ ​Delete**

1. **Pada​ ​viewall.html​ ​tambahkan**

<a​​ ​​th:href​=​"'/student/delete/'​ ​+​ ​${student.npm}"​​ ​​>​​ ​Delete​ ​Data​</a><br/>



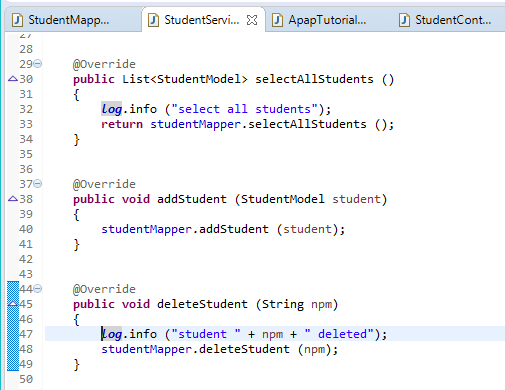
1. **Tambahkan​ ​method​ ​​deleteStudent​ ​​yang​ ​ada​ ​di​ ​class​ ​​StudentMapper**
2. **Tambahkan​ ​method​ ​delete​ ​student​ ​yang​ ​menerima​ ​parameter​ ​NPM.**
3. **Tambahkan​ ​annotation​ ​delete​ ​di​ ​atas​ ​dan​ ​SQL​ ​untuk​ ​menghapus @Delete(“[LENGKAPI]”)**



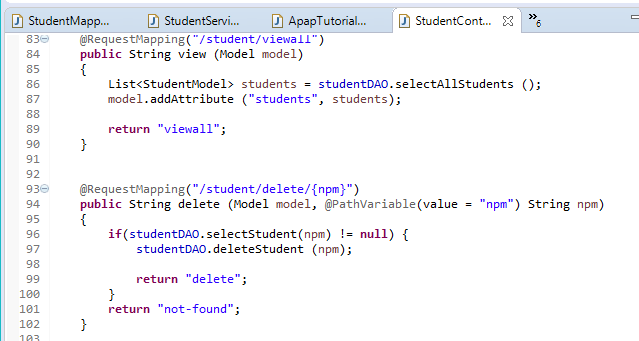
1. **Lengkapi​ ​method​ ​​deleteStudent​​ ​yang​ ​ada​ ​di​ ​class​ ​​StudentServiceDatabase**

**a. Tambahkan​ ​log​ ​untuk​ ​method​ ​tersebut​ ​dengan​ ​cara​ ​menambahkan log​.​info​ ​​(​"student​ ​"​​ ​​+​​ ​npm​ ​​+​​ ​​"​ ​deleted"​);**

**b. Panggil​ ​method​ ​delete​ ​student​ ​yang​ ​ada​ ​di​ ​Student​ ​Mapper**



1. **Lengkapi​ ​method​ ​​delete​​ ​pada​ ​class​ ​​StudentController**
2. **Tambahkan​ ​validasi​ ​agar​ ​jika​ ​mahasiswa​ ​tidak​ ​ditemukan​ ​tampilkan​ ​view not-found**



1. **Jika​ ​berhasil​ ​delete​ ​student​ ​dan​ ​tampilkan​ ​view​ ​delete**



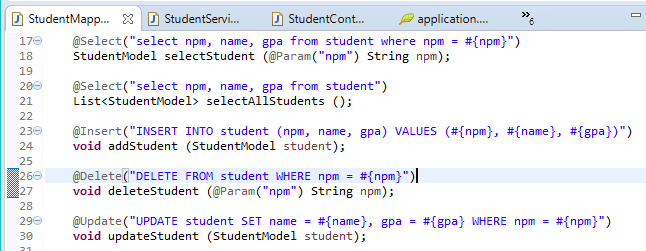
**Latihan​ ​Menambahkan​ ​Update**

1. **Tambahkan​ ​method​ ​​updateStudent​​ ​pada​ ​class​ ​​StudentMapper**

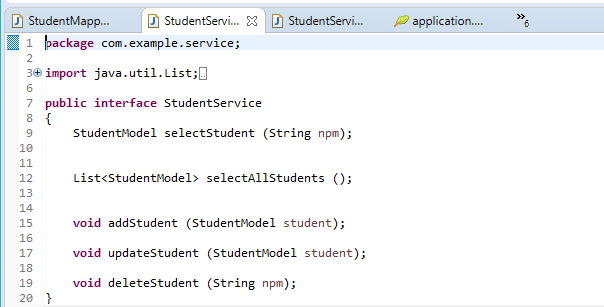
**a. Parameternya​ ​adalah​ ​StudentModel​ ​student**

**b. Annotationnya​ ​adalah​ ​@Update**

**c. Lengkapi​ ​SQL​ ​​update ​ -nya Hint:​ ​Query​ ​SQL​ ​untuk​ ​update**

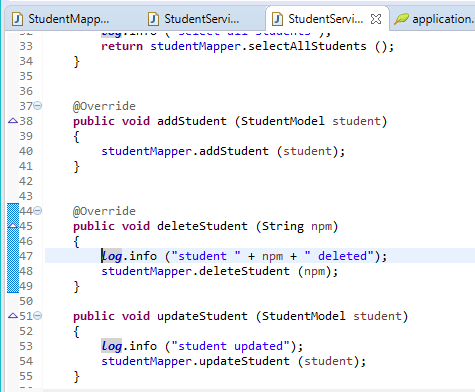


1. **Tambahkan​ ​method​ ​​updateStudent​ ​​pada​ ​interface​ ​​StudentService**

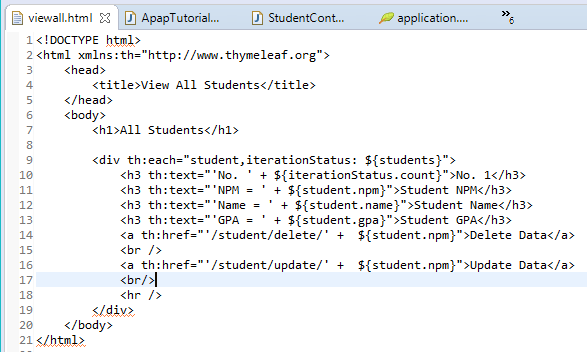


1. **Tambahkan​ ​implementasi​ ​method​ ​​updateStudent​ ​​pada​ ​class​ ​​StudentServiceDatabase.**

**Jangan​ ​lupa​ ​tambahkan​ ​logging​ ​pada​ ​method​ ​ini**



1. **Tambahkan​ ​link​ ​Update​ ​Data​ ​pada​ ​​viewall.html**



1. **Copy​ ​view​ ​form-add.html​ ​menjadi​ ​​form-update.html​.**

**a. Ubah​ ​info-info​ ​yang​ ​diperlukan​ ​seperti​ ​title,​ ​page-header,​ ​tombol​ ​menjadi​ ​update dll.**

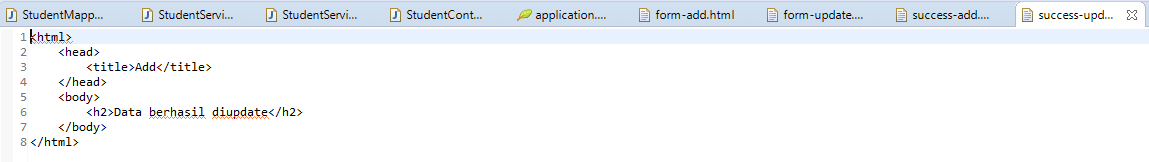
**b. Ubah​ ​action​ ​form​ ​menjadi​ ​/student/update/submit**

**c. Ubah​ ​method​ ​menjadi​ ​post**



1. **Copy​ ​view​ ​success-add.html​ ​menjadi​ ​​success-update.html​.**

**a. Ubah​ ​keterangan​ ​seperlunya**

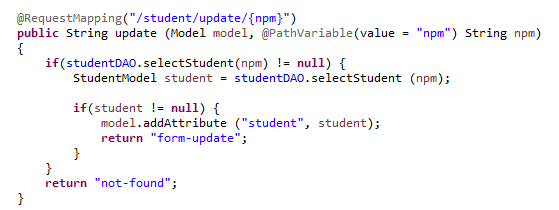


1. **Tambahkan​ ​method​ ​​update​​ ​pada​ ​class​ ​​StudentController**

**a. Request​ ​mapping​ ​ke​ ​/student/update/{npm}**

**b. Sama​ ​halnya​ ​seperti​ ​delete,​ ​lakukan​ ​validasi.**

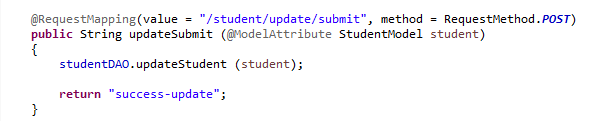
**c. Jika​ ​student​ ​dengan​ ​npm​ ​tidak​ ​ada​ ​tampilkan​ ​view​ ​not-found,​ ​jika​ ​ada​ ​tampilkan view​ ​form-update**



1. **Tambahkan​ ​method​ ​​updateSubmit​​ ​pada​ ​class​ ​​StudentController**

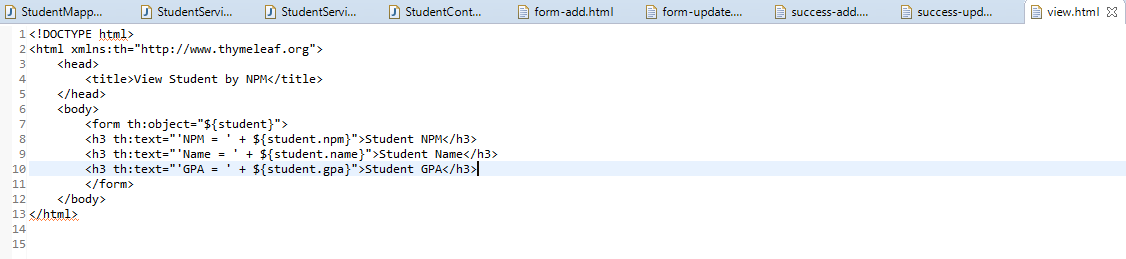
**a. Karena​ ​menggunakan​ ​post​ ​method​ ​maka​ ​request​ ​mappingnya​ ​adalah​ ​sebagai berikut:**

**b. Lengkapi​ ​method​ ​dengan​ ​memanggil​ ​method​ ​update​ ​Student​ ​dan​ ​kembalikan view​ ​success-update**

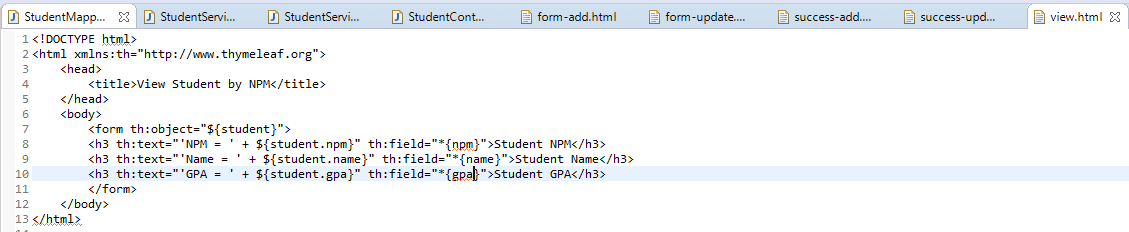


**Latihan​ ​Menggunakan​ ​Object​ ​Sebagai​ ​Parameter**

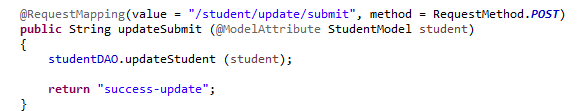
1. **Menambahkan​ ​th:object=”${student}”​ ​pada​ ​tag​ ​​<form>​​ ​di​ ​view**



1. **Menambahkan​ ​​ ​th:field=”\*{[nama\_field]}”​ ​pada​ ​setiap​ ​input**



1. **Ubah​ ​method​ ​updateSubmit​ ​pada​ ​StudentController​ ​yang​ ​hanya​ ​menerima​ ​parameter berupa​ ​StudentModel**



1. **Jelaskan​ ​apa​ ​saja​ ​hal​ ​yang​ ​Anda​ ​pelajari​ ​dari​ ​tutorial​ ​ini.**

Menggunakan database untuk input dan update data.

1. **Method​ ​yang​ ​Anda​ ​buat​ ​pada​ ​Latihan​ ​Menambahkan​ ​Delete,​ ​jelaskan**

Pada soal dijelaskan untuk buat method delete student pada StudentController.java, method menerima parameter yaitu npm. Lalu mengecek npm di database ada atau tidak dengan selectStudent, jika ada akan memunculkan delete.html.

1. **Method​ ​yang​ ​Anda​ ​buat​ ​pada​ ​Latihan​ ​Menambahkan​ ​Update,​ ​jelaskan**

Pada dijelaskan untuk buat method updateStudent pada kelas studentMapper.

Pada kelas StudentController saaya membuat 2 method update, yaitu update dimana method ini akan menerima parameter npm kemudian di cek di database apa data npm yang diminta ada atau tidak, jika ada maka akan memunculkan ke form-update.html. pada file tersebut user akan memperbaharui, lalu akan dikembalikan ke method updateSubmit yang ada di kelas StudentController, method ini akan melakukan pengecekan terhadap input user, input user berupa npm, nama, dan gpa. Lalu jika di cek semua terisi maka method ini akan melakukan update dan dimasukkan ke dalam database.

1. **​Method​ ​yang​ ​Anda​ ​buat​ ​pada​ ​Latihan​ ​Menggunakan​ ​Object​ ​Sebagai​ ​Parameter,​ ​jelaskan**

Perbedaan terletak pada method updateObjectSubmit, method ini hanya menerima parameter berupa StudentModel student dan langsung memperbaharui di database.

.