

Nama : Olvi Lora S

NPM : 1606954943

Fitur-fitur yang dikerjakan pada Tugas 1 ini adalah sebagai berikut:

1. Menampilkan Data Mahasiswa

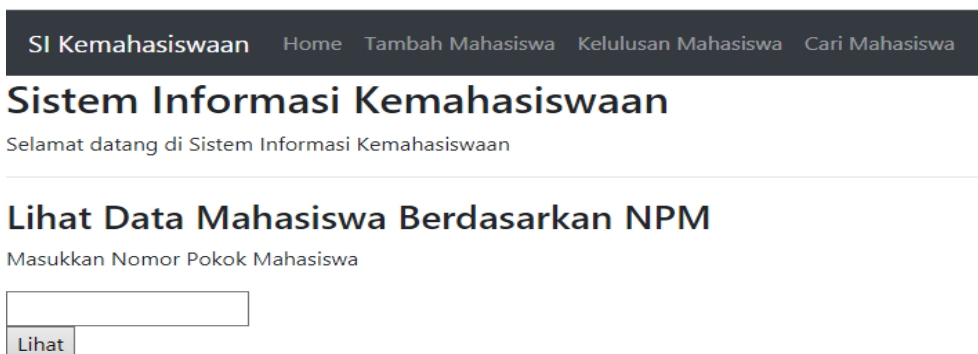
- Data mahasiswa ditampilkan dengan mengirimkan npm yang di-input pada field view yang diakses melalui url <http://localhost:8080>
- Di class StudentController terdapat method index yang meng-handle url tersebut

```
@RequestMapping("/")
public String index() {
    return "index";
}
```

- Method tersebut mengembalikan string index, artinya akan di-load view dengan nama index.html

```
1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymeleaf.org">
3 <head>
4     <link rel="stylesheet" href="/css/bootstrap.min.css" />
5     <title>Index</title>
6 </head>
7 <body>
8     <div th:replace="fragments/fragment :: header"></div>
9     <h2>Sistem Informasi Kemahasiswaan</h2>
10    <p>Selamat datang di Sistem Informasi Kemahasiswaan </p>
11    <hr>
12    <h3>Lihat Data Mahasiswa Berdasarkan NPM</h3>
13    <p>Masukkan Nomor Pokok Mahasiswa</p>
14    <form action="/mahasiswa" method="get">
15        <input type="text" name="npm" th:value="${npm}" />
16        <div>
17            <button> Lihat </button>
18        </div>
19    </form>
20
21    <!-- <div th:replace="fragments/fragment :: footer"></div> -->
22 </body>
23 </html>
```

- Apabila diakses, maka akan ditampilkan view seperti gambar di bawah ini



The screenshot shows a web application with a dark navigation bar at the top containing the text 'SI Kemahasiswaan' and links for 'Home', 'Tambah Mahasiswa', 'Kelulusan Mahasiswa', and 'Cari Mahasiswa'. Below the navigation bar is a main heading 'Sistem Informasi Kemahasiswaan' followed by the subtitle 'Selamat datang di Sistem Informasi Kemahasiswaan'. The main content area has a heading 'Lihat Data Mahasiswa Berdasarkan NPM' and a prompt 'Masukkan Nomor Pokok Mahasiswa'. There is a text input field for entering the NPM, and a button labeled 'Lihat' next to it.

- Dari tampilan di atas masukkan npm mahasiswa yang akan ditampilkan dan klik tombol Lihat. Akan dilakukan action yang memanggil url

<http://localhost:8080/mahasiswa?npm=162011454001> sesuai yang dideskripsikan di index.html

- Di class StudentController terdapat method lihatMahasiswa yang meng-handle url tersebut

```
@RequestMapping(value = "/mahasiswa", method = RequestMethod.GET)
public String lihatMahasiswa(@RequestParam(value = "npm") String npm, Model model) {
    StudentModel student = studentDAO.selectMahasiswa(npm);
    System.out.println("student : " + student);
    if (student != null) {
        model.addAttribute("student", student);
        return "detail-mahasiswa";
    } else {
        return "not-found";
    }
}
```

- Method tersebut melakukan query sql select mahasiswa berdasarkan npm yang telah didefinisikan di class StudentMapper dan mengembalikan parameter yang diperlukan.

```
@Select("select * from mahasiswa where npm = #{npm}")
@Results( value = {
    @Result(property = "id", column = "id"),
    @Result(property = "npm", column = "npm"),
    @Result(property = "nama", column = "nama"),
    @Result(property = "tempat_lahir", column = "tempat_lahir"),
    @Result(property = "tanggal_lahir", column = "tanggal_lahir"),
    @Result(property = "jenis_kelamin", column = "jenis_kelamin"),
    @Result(property = "agama", column = "agama"),
    @Result(property = "golongan_darah", column = "golongan_darah"),
    @Result(property = "status", column = "status"),
    @Result(property = "tahun_masuk", column = "tahun_masuk"),
    @Result(property = "jalur_masuk", column = "jalur_masuk"),
    @Result(property = "id_prodi", column = "id_prodi"),
    @Result(property = "prodi", column = "id_prodi", one = @One(select = "selectProdi"))
})
StudentModel selectMahasiswa (@Param("npm") String npm);
```

- Untuk bisa menggunakan method tersebut, harus didefinisikan terlebih dahulu di interface StudentService dan class StudentServiceDatabase

Interface StudentService : mendeskripsikan method selectMahasiswa yang akan diimplementasikan

```
StudentModel selectMahasiswa (String npm);
```

StudentServiceDatabase : memanggil method selectMahasiswa yang sudah didefinisikan di StudentMapper

```
public StudentModel selectMahasiswa (String npm)
{
    log.info ("select student with npm {}", npm);
    return studentMapper.selectMahasiswa (npm);
}
```

- Apabila mahasiswa dengan npm yang di-input ada di database, maka akan ditampilkan view detail-mahasiswa.html

```

1 <html xmlns="http://www.w3.org/1999/xhtml"
2   xmlns:th="http://www.thymeleaf.org">
3 <head>
4   <title>Detail Mahasiswa</title>
5   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
6   <link rel="stylesheet" href="/css/bootstrap.min.css" />
7   <script src="/js/bootstrap.min.js"></script>
8 </head>
9 <body>
10  <div th:replace="fragments/fragment :: header"></div>
11  <h3 th:text="'Lihat Data Mahasiswa - ' + ${student.npm}"></h3>
12  <table id="tblStudent">
13    <thead>
14      <tr>
15        <th width="20%">NIK</th>
16        <td th:text="${student.npm}"></td>
17      </tr>
18      <tr>
19        <th>Nama</th>
20        <td th:text="${student.nama}"></td>
21      </tr>
22      <tr>
23        <th>Tempat/Tanggal Lahir</th>
24        <td
25          th:text="${student.tempat_lahir} + ' ' + ${student.tanggal_lahir}">
26        </td>
27      </tr>
28      <tr>
29        <th>Program Studi</th>
30        <td
31          th:text="${student.prodi.nama_prodi}">
32        </td>
33      </tr>
34      <tr>
35        <th>Fakultas</th>
36        <td
37          th:text="${student.prodi.fakultas.nama_fakultas}">
38        </td>
39      </tr>
40      <tr>
41        <th>Universitas</th>
42        <td
43          th:text="${student.prodi.fakultas.univ.nama_univ}">
44        </td>
45      </tr>
46      <tr>
47        <th>Jenis Kelamin</th>
48        <td th:text="${student.jenis_kelamin}"></td>
49      </tr>
50      <tr>
51        <th>Agama</th>
52        <td th:text="${student.agama}"></td>
53      </tr>
54      <tr>
55        <th>Golongan Darah</th>
56        <td th:text="${student.golongan_darah}"></td>
57      </tr>
58      <tr>
59        <th>Tahun Masuk</th>
60        <td th:text="${student.tahun_masuk}"></td>
61      </tr>
62      <tr>
63        <th>Jalur Masuk</th>
64        <td th:text="${student.jalur_masuk}"></td>
65      </tr>
66      <tr>
67        <th>Status</th>
68        <td th:text="${student.status}"></td>
69        <td>
70          <button class="btn-success">Lulus</button>
71          <button class="btn-danger">Non-Aktifkan</button>
72        </td>
73      </tr>
74    </thead>
75  </table>
76 </body>
77 </html>

```

- Di browser, tampilan untuk detail-mahasiswa adalah sebagai berikut.

Lihat Data Mahasiswa - 162011454001

NIK	162011454001
Nama	Kelsi Mohr
Tempat/Tanggal Lahir	North Eunaside 1982-05-05
Program Studi	Fisika
Fakultas	Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas	Institut Teknologi Bandung
Jenis Kelamin	0
Agama	Protestan
Golongan Darah	AB-
Tahun Masuk	2016
Jalur Masuk	Undangan Reguler/SNMPTN
Status	Drop Out

Lulus

Non-Aktifkan

- Apabila npm yang di-input tidak ada di database maka akan ditampilkan view not-found.html

Mahasiswa tidak ditemukan

2. Menambahkan Data Mahasiswa

- Dengan meng-klik menu Tambah Mahasiswa akan memanggil url <http://localhost:8080/mahasiswa/tambah> (didefinisikan di header fragment.html yang di-include di setiap menu)

```
<li class="nav-item">
  <a class="nav-link" href="/mahasiswa/tambah">Tambah Mahasiswa</a>
</li>
```

- Di class StudentController terdapat method tambahMahasiswa index yang meng-handle url tersebut

```
@RequestMapping(value = "/mahasiswa/tambah")
public String tambahMahasiswa() {
    return "form-tambah";
}
```

- Method tersebut mengembalikan string form-tambah yang akan menampilkan view form-tambah.html

```

1 <!DOCTYPE HTML>
2 <html xmlns="http://www.w3.org/1999/xhtml"
3   xmlns:th="http://www.thymeleaf.org">
4 <head>
5
6 <title>Add student</title>
7 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
8 </head>
9
10 <body>
11   <div th:replace="fragments/fragment :: header"></div>
12   <h1 class="page-header">Tambah Mahasiswa</h1>
13
14   <form action="/mahasiswa/tambah/submit" method="post">
15     <div>
16       <label for="nama">Nama Lengkap</label> <input type="text" name="nama" />
17     </div>
18     <div>
19       <label for="tempat_lahir">Tempat Lahir</label> <input type="text" name="tempat_lahir" />
20     </div>
21     <div>
22       <label for="tanggal_lahir">Tanggal Lahir</label> <input type="text" name="tanggal_lahir" />
23     </div>
24     <div>
25       <label for="jenis_kelamin">Jenis Kelamin</label>
26       <select id="jenis_kelamin" name="jenis_kelamin">
27         <option value="0">Laki-Laki </option>
28         <option value="1">Perempuan </option>
29       </select>
30     </div>
31     <div>
32       <label for="agama">Agama</label>
33       <select id="agama" name="agama">
34         <option value="Protestan">Protestan </option>
35         <option value="Katolik">Katolik </option>
36         <option value="Budha">Budha </option>
37         <option value="Konghucu">Konghucu </option>
38         <option value="Hindu">Hindu </option>
39         <option value="Islam">Islam </option>
40       </select>
41     </div>
42     <div>
43       <label for="gol_darah">Golongan Darah</label>
44       <select id="gol_darah" name="gol_darah">
45         <option value="AB-">AB- </option>
46         <option value="A-">A- </option>
47         <option value="B-">B- </option>
48         <option value="O+">O+ </option>
49         <option value="B+">B+ </option>
50         <option value="O-">O- </option>
51         <option value="A+">A+ </option>
52         <option value="AB+">AB+ </option>
53       </select>
54     </div>
55     <div>
56       <label for="tahun_masuk">Tahun Masuk</label> <input type="text" name="tahun_masuk" />
57     </div>
58     <div>
59       <label for="jalur_masuk">Jalur Masuk</label>
60       <select id="jalur_masuk" name="jalur_masuk">
61         <option value="Undangan Reguler/SNMPTN">Undangan Reguler/SNMPTN </option>
62         <option value="Ujian Tulis Mandiri">Ujian Tulis Mandiri </option>
63         <option value="Undangan Paralel/PPKB">Undangan Paralel/PPKB </option>
64         <option value="Ujian Tulis Bersama/SBMPTN">Ujian Tulis Bersama/SBMPTN </option>
65         <option value="Undangan Olimpiade">Undangan Olimpiade </option>
66       </select>
67     </div>
68     <div>
69       <label for="id_prodi">Id Program Studi</label> <input type="text" name="id_prodi" />
70     </div>
71     <div>
72       <button type="submit" name="action" value="save">Save</button>
73     </div>
74   </form>
75
76 </body>

```

- Di browser tampilan form-add.html adalah sebagai berikut.

SI Kemahasiswaan
Home
Tambah Mahasiswa
Kelulusan Mahasiswa
Cari Mahasiswa

Tambah Mahasiswa

Nama Lengkap

Tempat Lahir

Tanggal Lahir

Jenis Kelamin

Agama

Golongan Darah

Tahun Masuk

Jalur Masuk

Id Program Studi

- Setelah mengisi semua field di form-add tersebut, akan dipanggil url <http://localhost:8080/mahasiswa/tambah/submit> sesuai form action yang didefinisikan di view form-add tersebut.
- Di class StudentController terdapat method addSubmit index yang meng-handle url tersebut

```

@RequestMapping("/mahasiswa/tambah/submit")
public String addSubmit(@RequestParam(value = "nama", required = false) String nama,
    @RequestParam(value = "tempat_lahir", required = false) String tempat_lahir,
    @RequestParam(value = "tanggal_lahir", required = false) String tanggal_lahir,
    @RequestParam(value = "jenis_kelamin", required = false) String jenis_kelamin,
    @RequestParam(value = "agama", required = false) String agama,
    @RequestParam(value = "gol_darah", required = false) String gol_darah,
    @RequestParam(value = "tahun_masuk", required = false) String tahun_masuk,
    @RequestParam(value = "jalur_masuk", required = false) String jalur_masuk,
    @RequestParam(value = "id_prodi", required = false) String id_prodi, Model model) {
    String npm_fix, order;
    String thnMasuk = tahun_masuk.substring(2);
    StudentModel student = studentDAO.selectMahasiswaByProdi(id_prodi);
    String kodeUniv = student.getProdi().getFakultas().getUniv().getKode_univ() + "";
    String kodeProdi = student.getProdi().getKode_prodi();
    String kode_jalur = getJalurMahasiswa(jalur_masuk);
    String npm = '%' + thnMasuk + kodeUniv + kodeProdi + kode_jalur + '%';
    System.out.println("npm = " + npm);
    StudentModel student_n = studentDAO.selectMahasiswaByNpm(npm);
    if (student_n == null) {
        order = "001";
    } else {
        String id = student_n.getNpm();
        String no_input = id.substring(9);
        int temp_order = Integer.parseInt(no_input);
        temp_order = temp_order + 1;
        order = temp_order + "";
    }
    npm_fix = thnMasuk + kodeUniv + kodeProdi + kode_jalur + order;
    System.out.println("npm_fix = " + npm_fix);
    StudentModel new_student = new StudentModel(0, npm_fix, nama, tempat_lahir, tanggal_lahir, jenis_kelami
        gol_darah, "Aktif", tahun_masuk, jalur_masuk, id_prodi, null);
    studentDAO.tambahMahasiswa(new_student);
    model.addAttribute("student", new_student);
    return "success-add";
}

```

- Method tersebut menerima semua parameter yang dikirim dari view, kemudian dilakukan proses dalam membentuk npm yang baru yaitu:
 - Mengambil 2 digit terakhir dari tahun masuk yang di-input
 - Mendapatkan kodeUniv dan kodeProdi dari method selectMahasiswaByProdi. Method ini diimplementasikan di class StudentMapper. Method tersebut select semua field berdasarkan parameter id_prodi yang dikirim dan mengembalikan field yang diperlukan.

```
@Select("select * from mahasiswa where id_prodi = #{id_prodi} order by id desc limit 1")
@Results( value = {
    @Result(property = "id", column = "id"),
    @Result(property = "npm", column = "npm"),
    @Result(property = "nama", column = "nama"),
    @Result(property = "tempat_lahir", column = "tempat_lahir"),
    @Result(property = "tanggal_lahir", column = "tanggal_lahir"),
    @Result(property = "jenis_kelamin", column = "jenis_kelamin"),
    @Result(property = "agama", column = "agama"),
    @Result(property = "golongan_darah", column = "golongan_darah"),
    @Result(property = "status", column = "status"),
    @Result(property = "tahun_masuk", column = "tahun_masuk"),
    @Result(property = "jalur_masuk", column = "jalur_masuk"),
    @Result(property = "prodi", column = "id_prodi", one = @One(select = "selectProdi"))
})
StudentModel selectMahasiswaByProdi (@Param("id_prodi") String id_prodi);
```

- Untuk bisa menggunakan method tersebut, harus didefinisikan terlebih dahulu di interface StudentService dan class StudentServiceDatabase
Interface StudentService : mendeksripsikan method selectMahasiswaByProdi yang akan diimplementasikan

```
StudentModel selectMahasiswaByProdi (String id_prodi);
```

Class

StudentServiceDatabase : memanggil method selectMahasiswaByProdi yang sudah didefinisikan di StudentMapper

```
public StudentModel selectMahasiswaByProdi (String id_prodi)
{
    log.info ("select student with id prodi {}", id_prodi);
    return studentMapper.selectMahasiswaByProdi (id_prodi);
}
```

- Mendapatkan kode jalur melalui method getJalurMahasiswa. Method ini mengembalikan kode jalur berdasarkan nama jalur yang dipilih di drop down view form-add.html

```
public String getJalurMahasiswa(String nama_jalur) {
    if (nama_jalur.equalsIgnoreCase("Undangan Reguler/SNMPTN")) {
        return "54";
    } else if (nama_jalur.equalsIgnoreCase("Ujian Tulis Mandiri")) {
        return "62";
    } else if (nama_jalur.equalsIgnoreCase("Undangan Paralel/PPKB")) {
        return "55";
    } else if (nama_jalur.equalsIgnoreCase("Ujian Tulis Bersama/SBMPTN")) {
        return "57";
    } else if (nama_jalur.equalsIgnoreCase("Undangan Olimpiade")) {
        return "53";
    }
    return "";
}
```

- Untuk mendapatkan kode urut mahasiswa saya melakukan query select mahasiswa dengan npm yang mirip dengan tahunMasuk+kodeUniv+kodeProdi+kodeJalur sort npm

tertinggi dan batasnya 1 ke database. Saya menambahkan method selectMahasiswaByNpm di class StudentMapper

```
@Select("select * from mahasiswa where npm LIKE #{npm} order by npm desc limit 1")
@Results( value = {
    @Result(property = "id", column = "id"),
    @Result(property = "npm", column = "npm")
})
StudentModel selectMahasiswaByNpm (@Param("npm") String npm);
```

- Kemudian diambil 3 digit terakhir dari npm yang terpilih dan ditambahkan 1
- Setelah semua syarat pembentuk npm selesai, maka dilakukan insert data mahasiswa ke database.
- Menambahkan method tambahMahasiswa yang melakukan insert ke database di class StudentMapper.

```
@Insert("INSERT INTO mahasiswa (npm, nama, tempat_lahir, tanggal_lahir, jenis_kelamin, agama, golongan_darah, "
+ "status, tahun_masuk, jalur_masuk, id_prodi) VALUES (#{npm}, #{nama}, #{tempat_lahir}, #{tanggal_lahir}, "
+ " #{jenis_kelamin}, #{agama}, #{golongan_darah}, #{status}, #{tahun_masuk}, #{jalur_masuk}, #{id_prodi})")
void tambahMahasiswa (StudentModel student);
```

- Untuk bisa menggunakan method tersebut, harus didefinisikan terlebih dahulu di interface StudentService dan class StudentServiceDatabase
- Interface StudentService : mendeksripsikan method tambahMahasiswa yang akan diimplementasikan

```
void tambahMahasiswa (StudentModel student);
```

StudentServiceDatabase : memanggil method tambahMahasiswa yang sudah didefinisikan di StudentMapper

```
public void tambahMahasiswa (StudentModel student)
{
    log.info ("insert student with npm {}", student.getNpm());
    studentMapper.tambahMahasiswa (student);
}
```

- Setelah insert berhasil, maka data mahasiswa yang dimasukkan ada di database

id	npm	nama	tempat_lahir	tanggal_lahir	jenis_kelamin	agama	golongan_darah	status	tahun_masuk	jalur_masuk	id_prodi
1	5009 17201115534	windiany	balige	1995-03-14	1	Konghucu B-		Aktif	2017	Undangan Paralel/PPKB	1

- Kemudian ditampilkan view success-add.html

```
1 <html>
2   <head>
3     <title>Add</title>
4   </head>
5   <body>
6     <div th:replace="fragments/fragment :: header"></div>
7     <h2>Sukses</h2>
8     <p th:text="'Mahasiswa dengan NPM ' + ${student.npm} + ' berhasil ditambahkan'"></p>
9   </body>
10 </html>
```

- Apabila diakses dari browser tampilannya adalah sebagai berikut.

SI Kemahasiswaan Home Tambah Mahasiswa Kelulusan Mahasiswa Cari Mahasiswa

Sukses

Mahasiswa dengan NPM 17201115534 berhasil ditambahkan

3. Mengubah Data Mahasiswa

- Mengubah data mahasiswa diakses dengan menggunakan URL <http://localhost:8080/mahasiswa/ubah/172011357204>
- Di class StudentController terdapat method ubahMahasiswa index yang meng-handle url tersebut

```
@RequestMapping("/mahasiswa/ubah/{npm}")
public String ubahMahasiswa(Model model, @PathVariable(value = "npm") String npm) {
    StudentModel student = studentDAO.selectMahasiswa(npm);
    if (student != null) {
        model.addAttribute("student", student);
        return "form-update";
    } else {
        return "not-found";
    }
}
```

- Method tersebut melakukan pengecekan mahasiswa yang akan di-update berdasarkan parameter npm yang dikirimkan dari view menggunakan selectMahasiswa yang telah dideskripsikan di interface StudentService, diimplementasikan di class StudentServiceDatabase dan StudentMapper. Apabila ada, maka akan ditampilkan form-update.html

```
1 <!DOCTYPE HTML>
2 <html xmlns="http://www.w3.org/1999/xhtml"
3     xmlns:th="http://www.thymeleaf.org">
4 <head>
5
6 <title>Add student</title>
7 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
8 </head>
9
10 <body>
11 <div th:replace="fragments/fragment :: header"></div>
12 <h1 class="page-header">Update Mahasiswa</h1>
13
14 <form action="/mahasiswa/ubah/submit" method="post" th:object="${student}" th:value="${student}">
15 <input type="hidden" name="npm" th:value="${student.npm}" th:field="${student.npm}" />
16 <div>
17 <label for="nama">Nama Lengkap</label> <input type="text" name="nama" th:value="${student.nama}"
18     th:field="${student.nama}" />
19 </div>
20 <div>
21 <label for="tempat_lahir">Tempat Lahir</label> <input type="text" name="tempat_lahir"
22     th:value="${student.tempat_lahir}" th:field="${student.tempat_lahir}" />
23 </div>
24 <div>
25 <label for="tanggal_lahir">Tanggal Lahir</label> <input type="text" name="tanggal_lahir"
26     th:value="${student.tanggal_lahir}" th:field="${student.tanggal_lahir}" />
27 </div>
28 <div>
29 <label for="jenis_kelamin">Jenis Kelamin</label>
30 <select id="jenis_kelamin" name="jenis_kelamin" th:value="${student.jenis_kelamin}"
31     th:field="${student.jenis_kelamin}">
32 <option value="0"> Laki-Laki </option>
33 <option value="1"> Perempuan </option>
34 </select>
35 </div>
36 <div>
37 <label for="agama">Agama</label>
38 <select id="agama" name="agama" th:value="${student.agama}" th:field="${student.agama}">
```

```

38 <select id="agama" name="agama" th:value="${student.agama}" th:field="${student.agama}">
39 <option value="Protestan"> Protestan </option>
40 <option value="Katolik"> Katolik </option>
41 <option value="Budha"> Budha </option>
42 <option value="Konghucu"> Konghucu </option>
43 <option value="Hindu"> Hindu </option>
44 <option value="Islam"> Islam </option>
45 </select>
46 </div>
47 <div>
48 <label for="agama">Golongan Darah</label>
49 <select id="gol_darah" name="gol_darah" th:value="${student.golongan_darah}"
50 th:field="${student.golongan_darah}">
51 <option value="AB-"> AB- </option>
52 <option value="A-"> A- </option>
53 <option value="B-"> B- </option>
54 <option value="O+"> O+ </option>
55 <option value="B+"> B+ </option>
56 <option value="O-"> O- </option>
57 <option value="A+"> A+ </option>
58 <option value="AB+"> AB+ </option>
59 </select>
60 </div>
61 <div>
62 <label for="tanggal_lahir">Tahun Masuk</label> <input type="text" name="tahun_masuk"
63 th:value="${student.tahun_masuk}" th:field="${student.tahun_masuk}" />
64 </div>
65 <div>
66 <label for="jalur_masuk">Jalur Masuk</label>
67 <select id="jalur_masuk" name="jalur_masuk" th:value="${student.jalur_masuk}"
68 th:field="${student.jalur_masuk}">
69 <option value="Undangan Reguler/SNMPTN"> Undangan Reguler/SNMPTN </option>
70 <option value="Ujian Tulis Mandiri"> Ujian Tulis Mandiri </option>
71 <option value="Undangan Paralel/PPKB"> Undangan Paralel/PPKB </option>
72 <option value="Ujian Tulis Bersama/SBMPTN">Ujian Tulis Bersama/SBMPTN </option>
73 <option value="Undangan Olimpiade"> Undangan Olimpiade </option>
74 </select>
75 </div>
76 <div>
77 <label for="id_prodi">Id Program Studi</label> <input type="text" name="id_prodi"
78 th:value="${student.id_prodi}" th:field="${student.id_prodi}" />
79 </div>
80 <div>
81 <button type="submit" name="action" value="save">Update</button>
82 </div>
83 </form>
84
85 </body>
86
87 </html>

```

- Apabila diakses dari database akan ditampilkan sebagai berikut:

Update Mahasiswa

Nama Lengkap

Tempat Lahir

Tanggal Lahir

Jenis Kelamin

Agama

Golongan Darah

Tahun Masuk

Jalur Masuk

Id Program Studi

- Ketika data yang akan diubah telah dituliskan di field yang tersedia dan mengklik tombol Update maka akan dipanggil URL <http://localhost:8080/mahasiswa/ubah/submit> sesuai action yang didefinisikan di view form-update.html
- Di class StudentController terdapat method updateMahasiswaSubmit index yang meng-handle url tersebut.

```
@RequestMapping(value = "/mahasiswa/ubah/submit", method = RequestMethod.POST)
public String updateMahasiswaSubmit(@ModelAttribute StudentModel student, Model model)
{
    String npm_fix, order;
    String thnMasuk = student.getTahun_masuk().substring(2);
    StudentModel student_ = studentDAO.selectMahasiswaByProdi(student.getId_prodi());
    String kodeUniv = student_.getProdi().getFakultas().getUniv().getKode_univ() + "";
    String kodeProdi = student_.getProdi().getKode_prodi();
    String kode_jalur = getJalurMahasiswa(student.getJalur_masuk());
    String npm = '%' + thnMasuk + kodeUniv + kodeProdi + kode_jalur + '%';
    StudentModel student_n = studentDAO.selectMahasiswaByNpm(npm);
    if (student_n == null) {
        order = "001";
    } else {
        String id = student_n.getNpm();
        String no_input = id.substring(9);
        int temp_order = Integer.parseInt(no_input);
        temp_order = temp_order + 1;
        order = temp_order + "";
    }
    npm_fix = thnMasuk + kodeUniv + kodeProdi + kode_jalur + order;
    String new_npm = npm_fix;
    System.out.println("npm_fix = " + npm_fix);
    studentDAO.updateMahasiswa(student);
    studentDAO.updateNPMMahasiswa(student.getNpm(), new_npm);
    model.addAttribute("student", student);
    return "success-update";
}
```

- Method tersebut akan melakukan update data mahasiswa ke database dengan memanggil method `updateMahasiswa` yang telah didefinisikan di `StudentMapper`. Kemudian mengubah nilai npm mahasiswa yang di-update dengan data baru yang dimasukkan dengan memanggil method `updateNPMMahasiswa`.

```
36 @Update("UPDATE mahasiswa SET nama = #{nama}, tempat_lahir = #{tempat_lahir}, tanggal_lahir = #{tanggal_lahir}, "
37 + "jenis_kelamin = #{jenis_kelamin}, agama = #{agama}, golongan_darah = #{golongan_darah}, "
38 + "tahun_masuk = #{tahun_masuk}, jalur_masuk = #{jalur_masuk}, id_prodi = #{id_prodi} where npm = #{npm}")
39 void updateMahasiswa(StudentModel student);
```

```
@Update("UPDATE mahasiswa SET npm = #{new_npm} where npm = #{npm}")
void updateNPMMahasiswa(@Param("npm") String npm, @Param("new_npm") String new_npm);
```

- Untuk bisa menggunakan method tersebut, harus didefinisikan terlebih dahulu di interface `StudentService` dan class `StudentServiceDatabase`
Interface `StudentService` : mendeksripsikan method `updateMahasiswa` yang akan diimplementasikan

```
void updateMahasiswa (StudentModel student);
```

```
void updateNPMMahasiswa (String npm, String new_npm);
```

`StudentServiceDatabase` : memanggil method `updateMahasiswa` yang sudah didefinisikan di `StudentMapper`

```
@Override
public void updateMahasiswa(StudentModel student) {
    log.info ("student " + student + " updated");
    studentMapper.updateMahasiswa(student);
}
```

```
@Override
public void updateNPMMahasiswa(String npm, String new_npm) {
    log.info ("student " + npm + " updated");
    studentMapper.updateNPMMahasiswa(npm, new_npm);
}
```

- Setelah di-update maka data di database akan berubah

5007 172011357204	mega christy	siantar city	1995-03-28	1 Budha	B-	Aktif	2016	Undangan Paralel/PPKB	1
-------------------	--------------	--------------	------------	---------	----	-------	------	-----------------------	---

- Kemudian akan ditampilkan view `success-update.html`

```
1 <html xmlns="http://www.w3.org/1999/xhtml"
2   xmlns:th="http://www.thymeleaf.org">
3   <head>
4     <title>Update</title>
5   </head>
6   <body>
7     <div th:replace="fragments/fragment :: header"></div>
8     <h2>Sukses</h2>
9     <p th:text="'Mahasiswa dengan NPM ' + ${student.npm} + ' berhasil diubah'"></p>
10  </body>
11 </html>
```

- Apabila diakses melalui browser, tampilannya adalah sebagai berikut.

Sukses

Mahasiswa dengan NPM 172011357204 berhasil diubah

- Apabila mengupdate mahasiswa dengan npm yang tidak ada di database akan ditampilkan view not-found.html

```

1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymeleaf.org">
3   <head>
4     <title>Student not found</title>
5   </head>
6   <body>
7     <div th:replace="fragments/fragment :: header"></div>
8     <h1>Mahasiswa tidak ditemukan</h1>
9   </body>
10 </html>
11

```

- Apabila diakses melalui browser, tampilannya adalah sebagai berikut.

Mahasiswa tidak ditemukan

4. Menampilkan Persentase Kelulusan Mahasiswa pada Tahun dan Prodi Tertentu

- Dengan meng-klik menu Kelulusan Mahasiswa akan memanggil url <http://localhost:8080/kelulusan> (didefinisikan di header fragment.html yang di-include di setiap menu)

```

<li class="nav-item">
  <a class="nav-link" href="/kelulusan">Kelulusan Mahasiswa</a>
</li>

```

- Di class StudentController terdapat method ubahMahasiswa index yang meng-handle url tersebut

```

@RequestMapping(value = "/kelulusan", method = RequestMethod.GET)
public String lihatKelulusan() {
    return "form-persentasi-kelulusan";
}

```

- Method tersebut kan mengembalikan string form-persentasi-kelulusan yang akan membuka file form-persentasi-kelulusan.html

```

1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymeleaf.org">
3 <head>
4     <link rel="stylesheet" href="/css/bootstrap.min.css" />
5     <title>Index</title>
6 </head>
7 <body>
8     <div th:replace="fragments/fragment :: header"></div>
9     <h2>Persentase Kelulusan</h2>
10    <h3>Lihat Data Mahasiswa Berdasarkan NPM</h3>
11    <p>Masukkan Nomor Pokok Mahasiswa</p>
12    <form action="/kelulusan/submit" method="get">
13        <div>
14            <label for="tahun_masuk"><b>Tahun</b></label>
15            <div>
16                <input type="text" name="tahun_masuk" />
17            </div>
18        </div>
19        <div>
20            <label for="id_prodi"><b>Id Program Studi</b></label>
21            <div>
22                <input type="text" name="id_prodi" />
23            </div>
24        </div>
25        <div>
26            <button type="submit" name="action" value="save">Lihat</button>
27        </div>
28    </form>
29 </body>
30 </html>

```

- Apabila diakses melalui browser, tampilannya adalah sebagai berikut

SI Kemahasiswaan
Home
Tambah Mahasiswa
Kelulusan Mahasiswa
Cari Mahasiswa

Persentase Kelulusan

Lihat Data Mahasiswa Berdasarkan NPM

Masukkan Nomor Pokok Mahasiswa

Tahun

Id Program Studi

Lihat

- Setelah mengisi field tahun dan id program studi dan meng-klik button Lihat maka akan dipanggil URL
http://localhost:8080/kelulusan/submit?tahun_masuk=2017&id_prodi=1&action=save
sesuai action yang didefinisikan di view form-persentasi-kelulusan.html
- Di controller terdapat method submitKelulusan yang meng-handle URL tersebut.

```

@RequestMapping(value = "/kelulusan/submit", method = RequestMethod.GET)
public String submitKelulusan(Model model, @RequestParam(value = "tahun_masuk", required = false)
String tahun_masuk, @RequestParam(value = "id_prodi", required = false) String id_prodi) {
    StudentModel student = studentDAO.selectMahasiswaByProdi(id_prodi);
    String prodi = student.getProdi().getNama_prodi() + "";
    String nama_fakultas = student.getProdi().getFakultas().getNama_fakultas() + "";
    String universitas = student.getProdi().getFakultas().getUniv().getNama_univ() + "";
    String jlhAll = studentDAO.selectAktifAllMahasiswa(tahun_masuk, id_prodi);
    String jlh = studentDAO.selectAktifMahasiswa(tahun_masuk, id_prodi);
    double a = Double.parseDouble(jlh);
    double n = Double.parseDouble(jlhAll);
    double persentasi = a/n*100;
    int persent = (int) persentasi;
    int jlhLulus = (int) a;
    int jlhAll_ = (int) n;
    System.out.println("jumlah = "+a);
    System.out.println("jumlah all = "+n);
    System.out.println("persentasi1 = "+persentasi);
    model.addAttribute("tahun_masuk", tahun_masuk);
    model.addAttribute("prodi", prodi);
    model.addAttribute("fakultas", nama_fakultas);
    model.addAttribute("universitas", universitas);
    model.addAttribute("jlhLulus", jlhLulus);
    model.addAttribute("jlhAll", jlhAll_);
    model.addAttribute("persentasi", persent);
    return "view-persentasi-kelulusan";
}

```

- Pada method tersebut dilakukan pemanggilan method selectMahasiswaByProdi sesuai parameter id prodi yang dikirim dari view
- Kemudian didapatkan nama fakultas dan universitasnya
- Setelah itu dilakukan pengecekan persentasi dengan mengecek jumlah keseluruhan mahasiswa yang masuk berdasarkan tahun dan id prodi yang diisi dan mengecek jumlah mahasiswa yang lulus berdasarkan tahun dan id prodi yang diisi.

- **Pengecekan jumlah keseluruhan mahasiswa** dengan membuat method selectAktifAllMahasiswa di StudentMapper

```

@Select("SELECT COUNT(*) as count from mahasiswa where tahun_masuk = #{tahun_masuk} and id_prodi = #{id_prodi}")
String selectAktifAllMahasiswa(@Param("tahun_masuk") String tahun_masuk, @Param("id_prodi") String id_prodi);

```

- Untuk bisa menggunakan method tersebut, harus didefinisikan terlebih dahulu di interface StudentService dan class StudentServiceDatabase
Interface StudentService : mendeksripsikan method selectAktifAllMahasiswa yang akan diimplementasikan

```
String selectAktifAllMahasiswa(String tahun_masuk, String id_prodi);
```

StudentServiceDatabase : memanggil method selectAktifAllMahasiswa yang sudah didefinisikan di StudentMapper

```

public String selectAktifAllMahasiswa (String tahun_masuk, String id_prodi)
{
    log.info ("select student with tahun_masuk {}", tahun_masuk);
    return studentMapper.selectAktifAllMahasiswa (tahun_masuk, id_prodi);
}

```

- **Pengecekan jumlah keseluruhan mahasiswa yang lulus** dengan membuat method selectAktifMahasiswa di StudentMapper

```

@Select("SELECT COUNT(*) as count from mahasiswa where tahun_masuk = #{tahun_masuk} and id_prodi = #{id_prodi} "
+ "and status = 'Lulus'")
String selectAktifMahasiswa(@Param("tahun_masuk") String tahun_masuk, @Param("id_prodi") String id_prodi);

```

- Untuk bisa menggunakan method tersebut, harus didefinisikan terlebih dahulu di interface StudentService dan class StudentServiceDatabase

Interface StudentService : mendeksripsikan method selectAktifMahasiswa yang akan diimplementasikan

```
String selectAktifMahasiswa(String tahun_masuk, String id_prodi);
```

StudentServiceDatabase : memanggil method selectAktifAllMahasiswa yang sudah didefinisikan di StudentMapper

```
public String selectAktifMahasiswa (String tahun_masuk, String id_prodi)
{
    log.info ("select student with tahun_masuk {}", tahun_masuk);
    return studentMapper.selectAktifMahasiswa (tahun_masuk, id_prodi);
}
```

- Kemudian didapatkan persentasi mahasiswa yang lulus
- Informasi persentasi kelulusan tersebut ditampilkan di view view-persentasi-kelulusan.html

```
1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymeleaf.org">
3 <head>
4     <link rel="stylesheet" href="/css/bootstrap.min.css" />
5     <title>Index</title>
6     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
7 </head>
8 <body>
9     <div th:replace="fragments/fragment :: header"></div>
10    <h2>Persentase Kelulusan</h2>
11    <table width="700px">
12        <tr>
13            <td> Tahun </td>
14            <td th:text="{tahun_masuk}"> </td>
15            <td th:text="{persentasi}+'%'" rowspan="2"><h3> 95 %</h3></td>
16        </tr>
17        <tr>
18            <td> Program Studi </td>
19            <td th:text="{prodi}"> Akuntansi </td>
20        </tr>
21        <tr>
22            <td> Fakultas </td>
23            <td th:text="{fakultas}"> Fakultas Ekonomi dan Bisnis </td>
24            <td th:text="{jhlLulus}+' dari '+{jhlAll}+' Mahasiswa telah lulus'" rowspan="2"> 95 dari 100 Mahasiswa telah lulus
25        </td>
26        </tr>
27        <tr>
28            <td> Universitas </td>
29            <td th:text="{universitas}"> Universitas Indonesia </td>
30        </tr>
31    </table>
32 </body>
33 </html>
```

- Apabila diakses dari browser, maka tampilannya sebagai berikut

SI Kemahasiswaan Home Tambah Mahasiswa Kelulusan Mahasiswa Cari Mahasiswa		
Persentase Kelulusan		
Tahun	2017	24%
Program Studi	Teknik Geologi	
Fakultas	Fakultas Ilmu dan Teknologi Kebumian	6 dari 25 Mahasiswa telah lulus
Universitas	Institut Teknologi Bandung	

5. Menampilkan Data Mahasiswa Berdasarkan Universitas, Fakultas dan Program Studi Tertentu

- Dengan meng-klik menu Kelulusan Mahasiswa akan memanggil url <http://localhost:8080/kelulusan> (didefinisikan di header fragment.html yang di-include di setiap menu)

```
<li class="nav-item">
  <a class="nav-link" href="/mahasiswa/cari">Cari Mahasiswa</a>
</li>
```

- Di class StudentController terdapat method pilihUniversitas index yang meng-handle url tersebut

```
@RequestMapping(value = "/mahasiswa/cari", method = RequestMethod.GET)
public String pilihUniversitas(Model model) {
    List<UniversitasModel> universitas = studentDAO.selectAllUniversitas();
    model.addAttribute("universitas", universitas);
    return "pilih-universitas";
}
```

- Method ini akan mengambil data semua universitas yang ada di database yang dibuat di method selectAllUniversitas di class Mapper

```
@Select("select * from universitas")
@Results( value = {
    @Result(property = "id", column = "id"),
    @Result(property = "kode_univ", column = "kode_univ"),
    @Result(property = "nama_unv", column = "nama_unv"),
})
List<UniversitasModel> selectAllUniversitas();
```

- Untuk bisa menggunakan method tersebut, harus didefinisikan terlebih dahulu di interface StudentService dan class StudentServiceDatabase

Interface StudentService : mendeksripsikan method selectAktifMahasiswa yang akan diimplementasikan

```
List<UniversitasModel> selectAllUniversitas();
```

- StudentServiceDatabase : memanggil method selectAktifAllMahasiswa yang sudah didefinisikan di StudentMapper

```
@Override
public List<UniversitasModel> selectAllUniversitas ()
{
    log.info ("select all students");
    return studentMapper.selectAllUniversitas ();
}
```

- Kemudian dikembalikan string pilih-universitas yang akan memanggil view pilih-universitas.html

```

1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymeleaf.org">
3 <head>
4   <link rel="stylesheet" href="/css/bootstrap.min.css" />
5   <title>Index</title>
6 </head>
7 <body>
8   <div th:replace="fragments/fragment :: header"></div>
9   <h3>Cari Mahasiswa</h3>
10  <form action="/mahasiswa/cari/submit" method="get" th:object="${universitas}" th:value="${universitas}">
11    <p><b>Universitas</b></p>
12    <select id="universitas" name="universitas" th:value="${universitas}" th:field="${universitas}">
13      <div th:each="universitas, iterationStatus: ${universitas}" th:class="${iterationStatus.odd} ? 'odd'">
14        <option id="kode_univ" name="kode_univ" th:text="${universitas.nama_univ}"
15          th:value="${universitas.kode_univ}"></option>
16      </div>
17    </select>
18    <div>
19      <button> Cari </button>
20    </div>
21  </form>
22
23  <!-- <div th:replace="fragments/fragment :: footer"></div> -->
24 </body>
25 </html>

```

- Apabila diakses dari browser, tampilannya sebagai berikut.

SI Kemahasiswaan
Home
Tambah Mahasiswa
Kelulusan Mahasiswa
Cari Mahasiswa

Cari Mahasiswa

Universitas

Institut Teknologi Bandung

Cari

6. Menambahkan Form Validation

- Tambah Mahasiswa

Tambah Mahasiswa

Nama Lengkap

Tempat Lahir

Tanggal Lahir

Jenis Kelamin

Agama

Golongan Darah

Tahun Masuk

Jalur Masuk

Id Program Studi

- Update Mahasiswa

Update Mahasiswa

Nama Lengkap

Tempat Lahir

Tanggal Lahir

Jenis Kelamin

Agama

Golongan Darah

Tahun Masuk

Jalur Masuk

Id Program Studi

- Kelulusan Mahasiswa

SI Kemahasiswaan Home Tambah Mahasiswa Kelulusan Mahasiswa Cari Mahasiswa

Persentase Kelulusan

Lihat Data Mahasiswa Berdasarkan NPM

Masukkan Nomor Pokok Mahasiswa

Tahun



Please fill in this field.

Lihat

- Cari mahasiswa berdasarkan NPM

SI Kemahasiswaan Home Tambah Mahasiswa Kelulusan Mahasiswa Cari Mahasiswa

Persentase Kelulusan

Lihat Data Mahasiswa Berdasarkan NPM

Masukkan Nomor Pokok Mahasiswa

Tahun



Please fill in this field.

Lihat