

**LAPORAN PRAKTIKUM**  
**REKAYASA SOFTWARE BERBASIS KOMPONEN**



**DISUSUN OLEH :**

**NAMA : RIO KISNA EKA PUTRA**  
**NIM : 21120116130060**  
**KELOMPOK : 32**

**LABORATORIUM SOFTWARE ENGINEERING**  
**DEPARTEMENT TEKNIK KOMPUTER**  
**FAKULTAS TEKNIK**  
**UNIVERSITAS DIPONEGORO**  
**SEMARANG**  
**2019**

## LEMBAR PENGESAHAN

### Laboratorium Rekayasa Perangkat Lunak

Teknik Komputer Universitas Diponegoro  
D204 Gedung Kuliah Bersama Fakultas Teknik  
Universitas Diponegoro Semarang



### LEMBAR PENGESAHAN

#### PRAKTIKUM REKAYASA SOFTWARE BERBASIS KOMPONEN 2019

Nama : Rio Kisna Eka Putra  
NIM : 21120116130060

  
Guntur Dwi Cahyono  
21120116120005

  
Jeremy Karisma Mesalinri  
21120116140078

  
Muhammad Ikhsan  
21120116120033

  
Agyan Atma Villantya  
21120116140071

  
Yogie Mesya Tama  
21120116130061

  
Fanny Hasbi  
21120116140068

  
Adam Maulidani  
21120116130065

Mengetahui  
Koordinator Praktikum

  
Guntur Dwi Cahyono  
21120116120005



Scanned with  
CamScanner

## **KATA PENGANTAR**

Puji syukur ke hadirat Tuhan Yang Maha Esa atas berkah, rahmat, dan karunia-Nya sehingga Laporan Praktikum Rekayasa Software Berbasis Komponen 2019 ini dapat diselesaikan dan disusun dengan baik. Diharapkan laporan ini dapat membantu dalam mempelajari dan memahami segala sesuatu yang berkaitan Rekayasa Software.

Ucapan terima kasih diberikan kepada :

1. Tuhan Yang Maha Esa yang telah memberikan rahmat dan karunia-Nya.
2. Kedua orang tua yang selalu memberikan dukungan dan doa.
3. Ibu Ike Pertiwi, S.T., M.T, selaku dosen pengampu mata kuliah Rekayasa Software Berbasis Komponen.
4. Seluruh asisten praktikum yang telah membimbing praktikan dalam praktikum.
5. Seluruh teman Teknik Komputer angkatan 2016 dan semua pihak yang terlibat lainnya.

Semoga laporan ini dapat bermanfaat bagi yang membacanya. Kritik dan saran sangat diterima demi lebih baiknya laporan ini. Akhir kata saya mengucapkan terimakasih

Semarang, 14 Desember 2019

Penulis

Rio Kisna Eka Putra  
NIM: 21120116130060

## DAFTAR ISI

|  |     |
|--|-----|
| LEMBAR PENGESAHAN .....                                    | ii  |
| KATA PENGANTAR .....                                       | iii |
| DAFTAR ISI.....  | iv  |
| DAFTAR GAMBAR .....  | vi  |
| ABSTRAK.....   | xii |
| BAB I PENDAHULUAN .....                                    | 1   |
| 1.1 Latar Belakang .....                                   | 1   |
| 1.2 Tujuan Praktikum .....                                 | 2   |
| 1.3 Sistematika Penulisan.....                             | 3   |
| BAB II JAVABEAN .....                                      | 4   |
| 2.1 Tujuan.....  | 4   |
| 2.2 Alat dan Bahan .....                                   | 5   |
| 2.3 Dasar Teori .....                                      | 8   |
| 2.4 Langkah Kerja .....                                    | 11  |
| 2.5 Tugas dan Pembahasan.....                              | 28  |
| 2.6 Kesimpulan.....  | 37  |
| BAB III EJB <i>SESSION BEAN</i> .....                      | 38  |
| 3.1 Tujuan.....  | 38  |
| 3.3 Alat dan Bahan .....                                   | 39  |
| 3.3 Dasar Teori .....                                      | 42  |
| 3.4 Langkah Kerja .....                                    | 46  |
| 3.5 Hasil Percobaan .....                                  | 59  |
| 3.6 Tugas dan Pembahasan.....                              | 66  |
| 3.7 Kesimpulan.....  | 75  |
| BAB IV Aplikasi CRUD menggunakan EJB, JPA, dan MySQL ..... | 76  |
| 4.1 Tujuan.....  | 76  |
| 4.2 Dasar Teori .....                                      | 77  |
| 4.3 Langkah Kerja .....                                    | 81  |
| 4.4 Hasil Percobaan .....                                  | 112 |

|                                      |     |
|--------------------------------------|-----|
| 4.5 Tugas dan Pembahasan.....        | 136 |
| 4.6 Kesimpulan.....                  | 151 |
| BAB V JAVA SERVER FACES.....         | 152 |
| 5.1 Tujuan.....                      | 152 |
| 5.2 Dasar Teori .....                | 153 |
| 5.3 Langkah Kerja .....              | 157 |
| 5.4 Hasil dan Analisa.....           | 179 |
| 5.5 Tugas dan Pembahasan.....        | 183 |
| 5.6 Kesimpulan.....                  | 190 |
| BAB VI PENUTUP .....                 | 191 |
| 6.1 Kesimpulan .....                 | 191 |
| 6.2 Saran.....                       | 192 |
| DAFTAR PUSTAKA .....                 | 193 |
| LAMPIRAN 1 (LEMBAR ASSISTENSI) ..... | 194 |

## DAFTAR GAMBAR

|   |    |
|---|----|
| Gambar 2.1 Laptop.....  | 5  |
| Gambar 2.2 NetBeans .....   | 5  |
| Gambar 2.3 Komponen BumperSticker.jar.....                            | 6  |
| Gambar 2.4 Github.....  | 6  |
| Gambar 2.5 Git.....   | 7  |
| Gambar 2.6 Membuat <i>project</i> baru .....                          | 11 |
| Gambar 2.7 Memberi nama project .....                                 | 11 |
| Gambar 2.8 Membuat JFrame <i>Form</i> .....                           | 12 |
| Gambar 2.9 Memberi nama JFrame <i>Form</i> .....                      | 13 |
| Gambar 2.10 Tampilan pada snapframe .....                             | 13 |
| Gambar 2.11 <i>Connection mode</i> .....                              | 14 |
| Gambar 2.12 Memilih <i>action</i> .....                               | 15 |
| Gambar 2.13 Memilih keluaran pada target komponen.....                | 15 |
| Gambar 2.14 Isi value label .....                                     | 16 |
| Gambar 2.15 Source code jButtonActionPerformed .....                  | 16 |
| Gambar 2.16 Clean and Build Project .....                             | 17 |
| Gambar 2.17 Import beans.....   | 18 |
| Gambar 2.18 Menambahkan komponen dari .jar yang sudah ada .....       | 19 |
| Gambar 2.19 Mencari lokasi bumperSticker.jar .....                    | 20 |
| Gambar 2.20 Memilih bumperSticker .....                               | 21 |
| Gambar 2.21 Memilih folder tujuan.....                                | 22 |
| Gambar 2.22 Selesai mengimport komponen .....                         | 23 |
| Gambar 2.23 Menambahkan bumpersticker ke <i>form</i> .....            | 24 |
| Gambar 2.24 Menambahkan <i>button</i> .....                           | 25 |
| Gambar 2.25 Memilih <i>actionPerformed</i> sebagai <i>event</i> ..... | 25 |
| Gambar 2.28 Kondisi awal.....   | 29 |
| Gambar 2.29 Menampilkan identitas kelompok .....                      | 30 |
| Gambar 2.30 Kondisi wajah sedih .....                                 | 30 |
| Gambar 2.31 Kondisi awal program .....                                | 34 |

|  |    |
|--|----|
| Gambar 2.32 Ketika Button di tekan sekali .....                | 35 |
| Gambar 2.33 Gambar di tekan kedua kalinya.....                 | 35 |
| Gambar 3.1 Laptop.....   | 39 |
| Gambar 3.2 NetBeans .....                                      | 39 |
| Gambar 3.3 Java <i>Enterprise Edition</i> .....                | 40 |
| Gambar 3.4 Jawa SDK.....                                       | 40 |
| Gambar 3.5 Glassfish Server.....                               | 41 |
| Gambar 3.6 New <i>Project Java EE</i> .....                    | 46 |
| Gambar 3.7 Memberi nama <i>Project</i> .....                   | 46 |
| Gambar 3.8 Memilih GlassFish Server untuk Web Server.....      | 47 |
| Gambar 3.9 Membuat <i>Session Bean</i> .....                   | 47 |
| Gambar 3.10 Memberi nama <i>Session Bean</i> .....             | 48 |
| Gambar 3.11 Insert <i>Code</i> .....                           | 48 |
| Gambar 3.12 Generate <i>code</i> .....                         | 49 |
| Gambar 3.13 Menambah fungsi login.....                         | 49 |
| Gambar 3.14 Menambah fungsi isLoginStatus .....                | 50 |
| Gambar 3.15 Menambah fungsi setLoginStatus .....               | 50 |
| Gambar 3.16 Membuat <i>Session Bean Validasi Session</i> ..... | 51 |
| Gambar 3.17 Membuat JSP Baru.....                              | 53 |
| Gambar 3.18 Membuat LoginView.jsp.....                         | 53 |
| Gambar 3.19 Membuat Servlet Baru .....                         | 55 |
| Gambar 3.20 Membuat LoginServlet.....                          | 55 |
| Gambar 3.21 Memberi URL Pattern.....                           | 56 |
| Gambar 3.22 Call <i>Enterprise Bean</i> .....                  | 56 |
| Gambar 3.23 Memanggil EJB ke servlet .....                     | 57 |
| Gambar 3.24 index.html.....                                    | 64 |
| Gambar 3.25 Halaman Login.....                                 | 64 |
| Gambar 3.26 Memasukan inputan salah .....                      | 65 |
| Gambar 3.27 Memasukan Inputan Benar.....                       | 65 |
| Gambar 3.28 index.html tugas .....                             | 71 |

|  |     |
|--|-----|
| Gambar 3.29 Halaman pencarian .....  | 71  |
| Gambar 3.30 Mencari mahasiswa yang terdaftar.....                            | 72  |
| Gambar 3.31 Mencoba mencari mahasiswa yang belum terdaftar .....             | 72  |
| Gambar 3.32 Mahasiswa tidak ditemukan .....                                  | 73  |
| Gambar 3.33 Menambahkan mahasiswa baru .....                                 | 73  |
| Gambar 3.34 Mahasiswa baru telah di tambahkan .....                          | 74  |
| Gambar 4.1 Membuat <i>database</i> baru .....                                | 81  |
| Gambar 4.2 Mengimport <i>file sql</i> .....                                  | 82  |
| Gambar 4.3 Membuat project web application .....                             | 83  |
| Gambar 4.4 Memberi nama project .....  | 83  |
| Gambar 4.5 Memilih GlassFish <i>Server</i> .....                             | 83  |
| Gambar 4.6 <i>Finish</i> pembuatan project.....                              | 84  |
| Gambar 4.7 Memilih other untuk menambahkan <i>file connection pool</i> ..... | 84  |
| Gambar 4.8 Membuat connection pool baru.....                                 | 85  |
| Gambar 4.9 Memilih koneksi untuk <i>database MySQL</i> .....                 | 85  |
| Gambar 4.10 Mengkonfigurasi koneksi.....                                     | 86  |
| Gambar 4.11 Memilih other untuk membuat <i>file JDBC Resource</i> .....      | 86  |
| Gambar 4.12 Membuat JDBC Resource baru.....                                  | 87  |
| Gambar 4.13 Memilih Connecction pool yang sudah dibuat.....                  | 87  |
| Gambar 4.14 Membuat Persistance Unit baru .....                              | 88  |
| Gambar 4.15 Memilih JDBC resource yang sudah dibuat.....                     | 88  |
| Gambar 4.16 Membuat <i>file JSP</i> baru .....                               | 89  |
| Gambar 4.17 Contoh membuat JSP <i>home</i> .....                             | 89  |
| Gambar 4.18 <i>File JSP</i> yang telah dibuat.....                           | 90  |
| Gambar 4.19 Menambahkan Asset ke project .....                               | 96  |
| Gambar 4. 20 Membuat Java <i>Package</i> baru.....                           | 96  |
| Gambar 4. 21 Contoh membuat Java <i>Package</i> com.controlletr .....        | 96  |
| Gambar 4. 22 Membuat Entity <i>Class</i> baru.....                           | 97  |
| Gambar 4. 23 Contoh membuat Entity <i>Class Student</i> .....                | 97  |
| Gambar 4. 24 Membuat session bean baru .....                                 | 100 |

|   |     |
|---|-----|
| Gambar 4. 25 Contoh membuat Session Bean <i>UserDAO</i> .....     | 101 |
| Gambar 4. 26 Membuat Servlet baru .....                           | 105 |
| Gambar 4. 27 Contoh membuat <i>RegisterServlet</i> .....          | 105 |
| Gambar 4. 28 Mengubah pattern url <i>RegisterServlet</i> .....    | 106 |
| Gambar 4. 29 Mengubah isi dari file web.xml.....                  | 112 |
| Gambar 4.30 Mengganti halaman awal web menjadi <i>login</i> ..... | 112 |
| Gambar 4.31 Hasil halaman <i>Login</i> .....                      | 132 |
| Gambar 4.32 Hasil halaman <i>Register</i> .....                   | 133 |
| Gambar 4.33 Hasil halaman <i>Home</i> .....                       | 133 |
| Gambar 4.34 Hasil percobaan tambah data.....                      | 134 |
| Gambar 4.35 Hasil percobaan <i>edit</i> data .....                | 134 |
| Gambar 4.36 Hasil percobaan search data .....                     | 135 |
| Gambar 4.37 Hasil percobaan <i>delete</i> data.....               | 135 |
| Gambar 4.38 Tampilan <i>login</i> .....                           | 148 |
| Gambar 4.39 Tampilan <i>register</i> .....                        | 148 |
| Gambar 4.40 Tampilan Halaman <i>Home</i> .....                    | 149 |
| Gambar 4.41 Tampilan halaman <i>About</i> .....                   | 149 |
| Gambar 4.42 Tampilan halaman <i>error</i> .....                   | 150 |
| Gambar 5.1 Membuka XAMPP .....                                    | 157 |
| Gambar 5.2 Mengakses phpmyadmin.....                              | 157 |
| Gambar 5.3 Membuat Database .....                                 | 157 |
| Gambar 5.4 Mengimport ke database baru .....                      | 157 |
| Gambar 5.5 Mengimport file sql.....                               | 158 |
| Gambar 5.6 Mengeksekusi file sql.....                             | 158 |
| Gambar 5.7 Membuat user baru terhadap database .....              | 159 |
| Gambar 5.8 Memasukan informasi login.....                         | 159 |
| Gambar 5.9 Mengatur hak user .....                                | 160 |
| Gambar 5.10 Membuat project baru .....                            | 160 |
| Gambar 5.11 Project Java Web Application .....                    | 160 |
| Gambar 5.12 Memberi nama project .....                            | 161 |

|   |     |
|---|-----|
| Gambar 5.13 Menambah instance server .....                  | 161 |
| Gambar 5.14 Lokasi instalasi dan detail login server .....  | 162 |
| Gambar 5.15 Memilih Apache Tomcat sebagai server.....       | 162 |
| Gambar 5.16 Memilih JSF sebaagai <i>framework</i> .....     | 163 |
| Gambar 5.17 Menambahkan file jar.....                       | 163 |
| Gambar 5.18 Menambahkan connector .....                     | 164 |
| Gambar 5.19 Membuat Java Package baru .....                 | 164 |
| Gambar 5.20 Membuat Java Class baru.....                    | 164 |
| Gambar 5.21 Membuat class koneksi .....                     | 165 |
| Gambar 5.22 <i>Fix import</i> .....                         | 166 |
| Gambar 5.23 Mengubah informasi login database .....         | 166 |
| Gambar 5.24 Coba run file koneksi .....                     | 167 |
| Gambar 5.25 Koneksi berhasil dibuat.....                    | 167 |
| Gambar 5.26 Membuat JSF Managed Bean baru .....             | 167 |
| Gambar 5.27 Membuat JSF Managed Bean Mahasiswa .....        | 168 |
| Gambar 5.28 File index.xhtml .....                          | 172 |
| Gambar 5.29 isi file index.xhtml.....                       | 172 |
| Gambar 5.30 Mencoba run file index.xhtml .....              | 174 |
| Gambar 5.31 Hasil run file index.xhtml .....                | 174 |
| Gambar 5.32 Membuat JSF Page baru.....                      | 175 |
| Gambar 5.33 Membuat JSF Page Edit .....                     | 175 |
| Gambar 5.34 Membuat JSF Page Tambah.....                    | 176 |
| Gambar 5.35 Hasil praktikum halaman index.....              | 179 |
| Gambar 5.36 Hasil praktikum tambah data.....                | 179 |
| Gambar 5.37 Hasil praktikum berhasil menambah data .....    | 180 |
| Gambar 5.38 Hasil praktikum edit mahasiswa .....            | 180 |
| Gambar 5.39 Hasil praktikum halaman edit mahasiswa.....     | 181 |
| Gambar 5.40 Hasil praktikum edit mahasiswa berhasil.....    | 182 |
| Gambar 5.41 Hasil praktikum delete mahasiswa berhasil ..... | 182 |
| Gambar 5.42 Tabel penjurusan .....                          | 183 |

|   |     |
|---|-----|
| Gambar 5.43 Tabel mahasiswa baru .....                  | 184 |
| Gambar 5.44 Constraint foreign key .....                | 184 |
| Gambar 5.45 tabel view .....                            | 185 |
| Gambar 5.46 Hasil tugas halaman index.....              | 187 |
| Gambar 5.47 Hasil tugas pesan tambah mahasiswa.....     | 188 |
| Gambar 5.48 Hasil tugas tambah mahasiswa berhasil ..... | 188 |
| Gambar 5.49 Hasil tugas data tertambah .....            | 189 |
| Gambar 5.50 Hasil tugas pesan error edit mahasiswa..... | 189 |
| Gambar 5.51 Hasil tugas edit data berhasil.....         | 189 |

## **ABSTRAK**

*Di masa yang sudah tidak terpisahkan dengan teknologi seperti sekarang ini, manusia semakin tidak bisa lepas dari komputer. Komputer pun sekarang sudah beralih menjadi semakin kecil, hingga muncullah mobile device atau smartphone yang lebih mudah dibawa-bawa dan memiliki fungsi yang hampir sama dengan sebuah komputer. Bidang rekayasa perangkat lunak-pun semakin berkembang mengikuti perkembangan teknologi komputer. Oleh karena itu muncullah banyak jenis aplikasi untuk pengembangan perangkat lunak berbasis komponen pada kehidupan sehari-hari, seperti pembuatan aplikasi sederhana, semisal konversi suhu dengan lingkungan pengembang Netbeans. Pada Laporan Praktikum Rekayasa Software Berbasis Komponen ini berisi mengenai materi pendalamannya dari aplikasi berbasis desktop dan web. Dengan menggunakan komponen seperti Java Persistence API dan Java Server Faces.*

**Kata Kunci :** Komponen, RSBK, rekayasa perangkat lunak, Netbeans

## **BAB I**

### **PENDAHULUAN**

#### **1.1 Latar Belakang**

Praktikum Rekayasa Software Berbasis Komponen adalah praktikum yang dilaksanakan pada semester 7 oleh mahasiswa di Fakultas Teknik Jurusan Teknik Komputer Universitas Diponegoro. Praktikum ini merupakan praktikum implementasi dari mata kuliah RSBK pada semester sebelumnya. Praktikum ini dilaksanakan agar mahasiswa dapat mengetahui tentang Rekayasa Software Berbasis Komponen dengan baik dan jelas.

## **1.2 Tujuan Praktikum**

1. Agar mahasiswa mengetahui konsep RSBK dengan baik.
2. Menambah wawasan mahasiswa mengenai sistem RSBK.
3. Praktikan mampu membuat aplikasi berbasis komponen.
4. Mengetahui berbagai macam pengaplikasian komponen pada pemrograman dan pengimplementasiannya.
5. Melatih mahasiswa untuk memahami pembuatan software berbasis komponen di Departemen Teknik Komputer.

### **1.3 Sistematika Penulisan**

Laporan akhir ini dibuat dengan sistematika sebagai berikut :

Halaman Cover

Halaman Pengesahan

Kata Pengantar

Daftar Isi

Daftar Gambar

Abstrak

BAB I PENDAHULUAN

    1.1. Latar Belakang

    1.2. Tujuan

    1.3. Sistematika Penulisan

BAB II Java Bean

BAB III EJB Session Bean

BAB IV CRUD Menggunakan EJB, JPA dan MySQL

BAB V Java Server Faces

BAB VI PENUTUP

Daftar Pustaka

LAMPIRAN I (Lembar Asistensi)

## **BAB II**

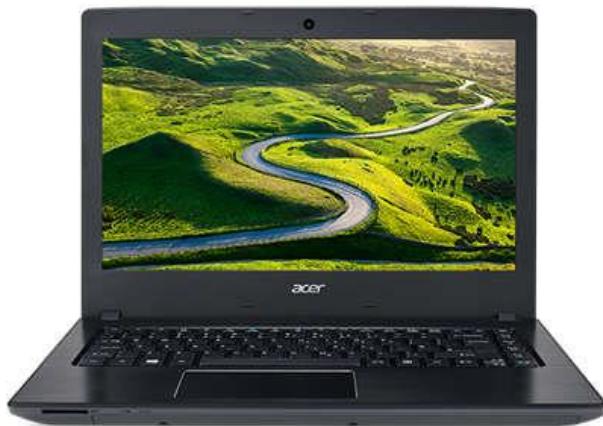
### **JAVABEAN**

#### **2.1 Tujuan**

1. Praktikan dapat mengetahui dasar Java Bean
2. Praktikan dapat mengetahui fungsi Java Bean
3. Praktikan dapat mengetahui penggunaan *command* Git dan Github
4. Praktikan dapat memahami konsep rekayasa komponen berupa *reuse* dan *compose*

## 2.2 Alat dan Bahan

1. Laptop



Gambar 2.1 Laptop

Laptop digunakan sebagai media instalasi keperluan praktikum seperti NetBeans.

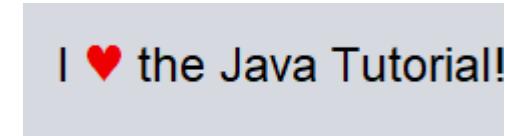
2. NetBeans



Gambar 2.2 NetBeans

NetBeans adalah suatu serambi pengembangan perangkat lunak yang ditulis dalam bahasa pemrograman Java. Serambi Pada NetBeans, pengembangan suatu aplikasi dapat dilakukan dimulai dari setelan perangkat lunak modular bernama *modules*.

3. BumperSticker.Jar



Gambar 2.3 Komponen BumperSticker.jar

Berkas ini adalah komponen yang digunakan saat praktikum yang dapat digunakan berulang-ulang

4. Akun Github yang sudah terverifikasi



Gambar 2.4 Github

GitHub adalah layanan penginangan web bersama untuk proyek pengembangan perangkat lunak yang menggunakan sistem pengontrol versi Git dan layanan hosting internet. Hal ini banyak digunakan untuk kode komputer. Ini memberikan kontrol akses dan beberapa fitur kolaborasi seperti pelacakan *bug*, permintaan fitur, manajemen tugas, dan wiki untuk setiap proyek.

## 5. Git



Gambar 2.5 Git

Git adalah perangkat lunak pengontrol versi atau proyek manajemen kode perangkat lunak yang diciptakan oleh Linus Torvalds, yang pada awalnya ditujukan untuk pengembangan kernel Linux.

## 2.3 Dasar Teori

### 2.3.1 Java Bean

Java Beans merupakan komponen model java yang mendukung prinsip *reusability* pada proses perangkat lunak. Kelebihan yang lainnya, dapat dimanipulasi secara visual menggunakan *builder tool* dan memungkinkan pengguna untuk membangun aplikasi secara mudah.

Java beans sifatnya *portable*, sehingga tidak harus *diinstall* pada sistem operasi. Arsitekturnya dibangun melalui kolaborasi antar industri dan mengijinkan developer untuk menulis ulang komponen ke dalam bahasa pemrograman java.

Java beans dapat berupa visual bean (*button, text box*) dan non visual bean (FTP, SMTP, Zip *code validator*). Sedang fungsionalitas yang didukung oleh java beans adalah :

- Event; yakni suatu *message* yang dari suatu obyek ke obyek yang lain, memberitahukan ke penerima bahwa telah terjadi 'sesuatu', atau simplenya untuk komunikasi antar bean. Untuk itu, *event* dibedakan atas 3 *event*, yakni *event source, event listener, event object*.
- Property; mendefenisikan karakteristik dari suatu bean, atau method untuk *get/set property values*. Cotoh "Public void set(value);". Property sendiri dibedakan atas 4 jenis, *simple property, indexed property, bound property, dan constrained property*.
- Persistence; memungkinkan beans untuk menyimpan dan *merestore*, atau sebagai *development tool* menyimpan Java Bean di dalam *hardisk* dan dapat di *load* pada suatu waktu. Selain itu, memelihara nilai *property* tanpa tergantung apakah Java Beans dapat menyimpan memori atau harddisk.
- *Object serialization*, yang memungkinkan *persistence* diperoleh, yakni dengan menyimpan semua *content* dari suatu *object* pada data *stream* dan meng *generate* kembali *object* ketika membuat dari data *stream*.

- *Introspection*, untuk mengetahui *property*, *events* dan *method*. Misalnya dibuat *class* XXXBeanInfo, khusus untuk menjelaskan *class* XXX secara detail. BeanInfo ini akan menjelaskan informasi bean seperti *icon*, *property*, *method* dan informasi lainnya.
- Java *Reflection API*, yakni Java API yang berfungsi untuk menemukan metode, *field field*, *konstruktor*, *superclasses* pada RUNTIME. API ini juga digunakan untuk menulis *development tools* yang lain yakni *debuggers*, *class browsers*, *GUI builders*.

(Sumber : <http://lea.si.fti.unand.ac.id/2013/11/javabeans/>)

### 2.3.2 Swing/AWT

AWT adalah singkatan dari *Abstract Window Toolkit*. Ini adalah API untuk mengembangkan aplikasi berbasis GUI atau Windows di Java. Ini membutuhkan objek OS asli untuk mengimplementasikan fungsionalitas. Juga, komponen AWT adalah kelas berat dan membutuhkan lebih banyak ruang memori. Apalagi mereka butuh waktu untuk mengeksekusi. Selanjutnya, programmer harus mengimpor paket javax.awt untuk mengembangkan GUI berbasis AWT. Tombol, *scrollbars*, bidang teks, daftar, dialog, dan panel adalah beberapa komponen AWT. Setelah membuat objek, mereka ditempatkan dalam sebuah wadah. Juga, ia menyediakan ruang yang diperlukan untuk memuat komponen. Biasanya, aplikasi AWT dalam satu OS mungkin terlihat berbeda di OS lain.

Swing adalah *toolkit* widget GUI untuk Java. Ini dibangun di atas API AWT. Juga, ini adalah bagian dari Java Foundation Classes (JFC) Oracle. Selanjutnya, Swing menyediakan komponen dasar seperti label, kotak teks, tombol, dll. Serta komponen lanjutan seperti panel tab, tabel, dan, pohon. Oleh karena itu, Swing menyediakan komponen yang lebih canggih daripada AWT. Di sini, programmer harus mengimpor paket javax.swing untuk menulis aplikasi Swing. Paket ini menyediakan sejumlah kelas seperti JButton, JTable, JList, JTextArea, dan, JCheckBox. Swing merupakan platform-independent dan komponennya ringan. Selanjutnya, komponen

membutuhkan ruang memori minimum. Oleh karena itu, aplikasi Swing mengeksekusi lebih cepat. Salah satu pola desain umum dalam pengembangan adalah pola Model, Tampilan, Pengendali (MVC). Ayunan mengikuti pola ini. Ini membantu menjaga kode dengan mudah.

(Sumber : <https://perbedaan.budisma.net/perbedaan-awt-dan-swing.html>)

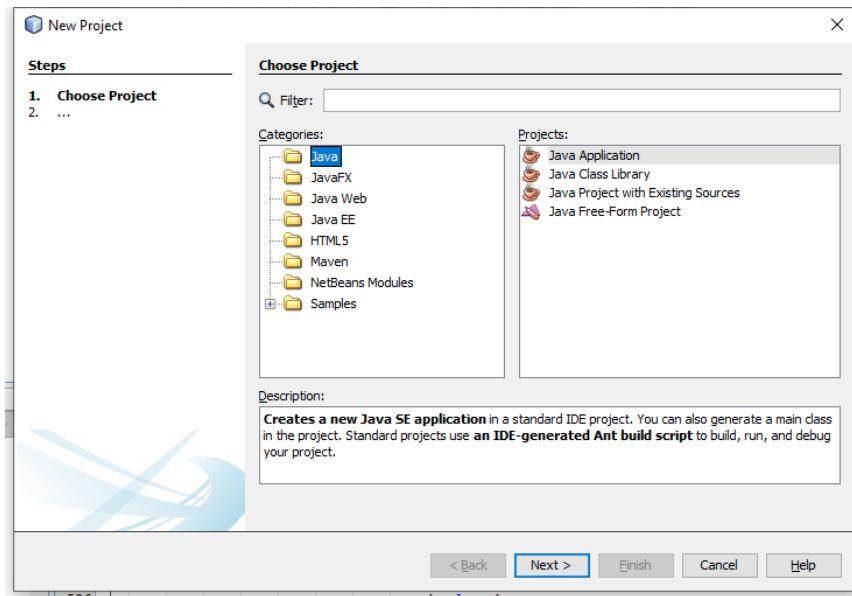
### **2.3.3 Prinsip Rekayasa Komponen (*Reuse* dan *Compose*)**

Java Beans merupakan bahasa pemrograman yang berorientasi objek, di mana semua komponen yang ada di dalamnya merupakan sebuah objek yang di satukan membentuk satu objek tunggal yang kompleks. Sehingga, Java Beans memungkinkan pengguna untuk membuat komponen dan menggunakan ulang komponen tersebut, atau lebih dikenal dengan prinsip *Reuse* dan *Compose*. Dengan prinsip ini, terdapat kelebihan, yaitu mengembangkan aplikasi akan lebih praktis, mudah dan cepat.

## 2.4 Langkah Kerja

- Buat *file* baru : *File* → *New Project* → *Java* → *Java application*

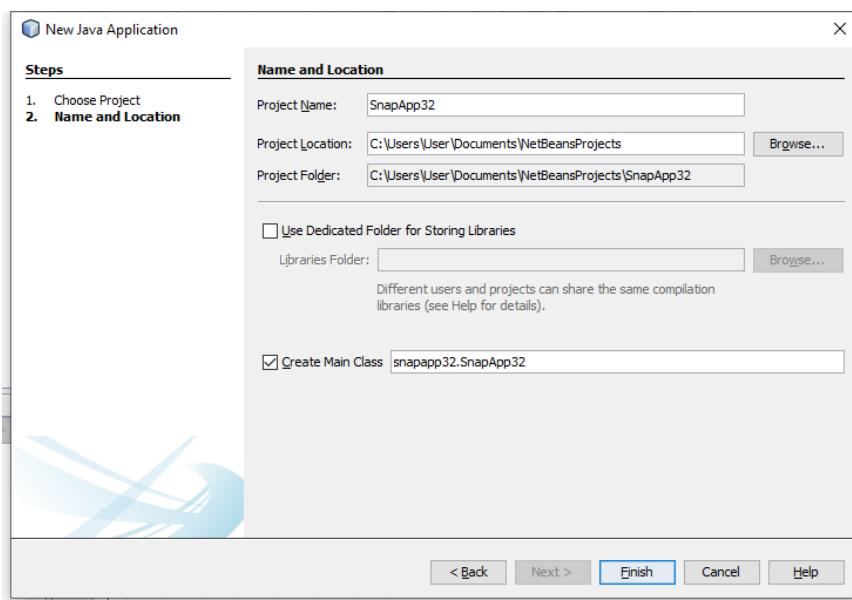
Langkah ini akan membuat *project* baru



Gambar 2.6 Membuat *project* baru

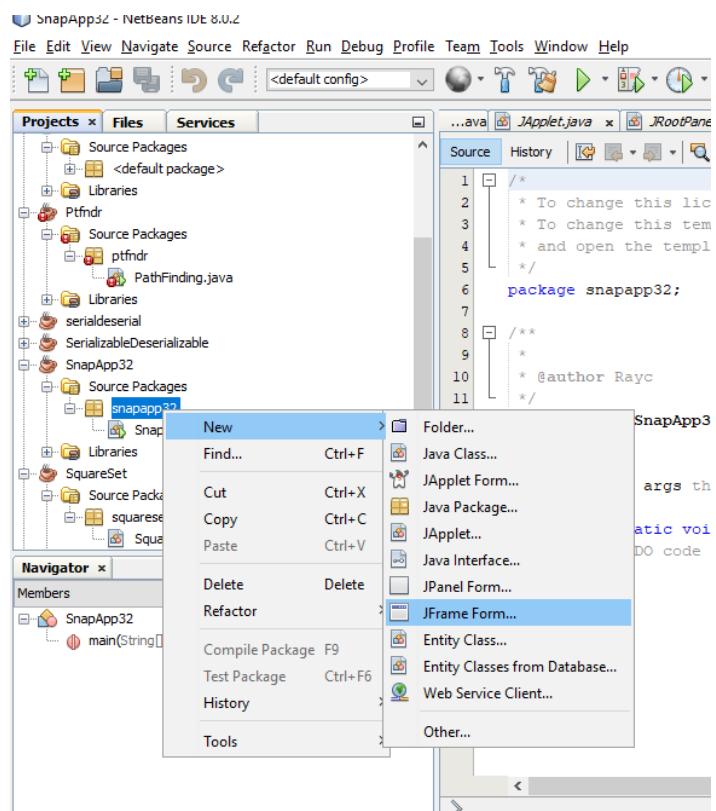
- Beri nama SnapApp32

Nama dan lokasi *project* digunakan untuk mempermudah akses di lain waktu



Gambar 2.7 Memberi nama project

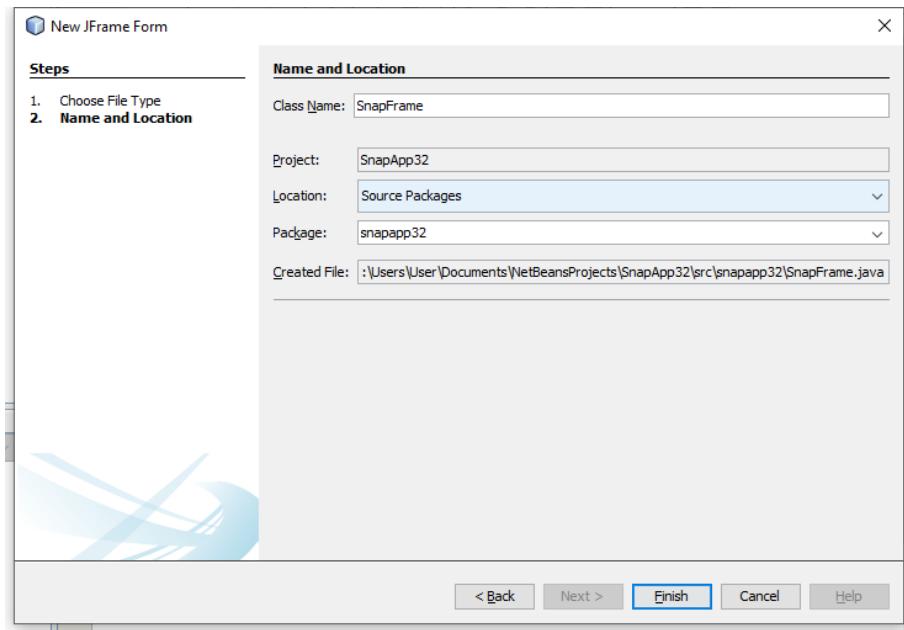
3. Klik kanan pada package SnapApp → pilih New → pilih JFrame Form  
JFrame Form adalah sebuah “halaman” dari aplikasi yang dibuat



Gambar 2.8 Membuat JFrame Form

4. Beri nama SnapFrame

Nama yang diberikan adalah nama *class* untuk halaman aplikasi



Gambar 2.9 Memberi nama JFrame Form

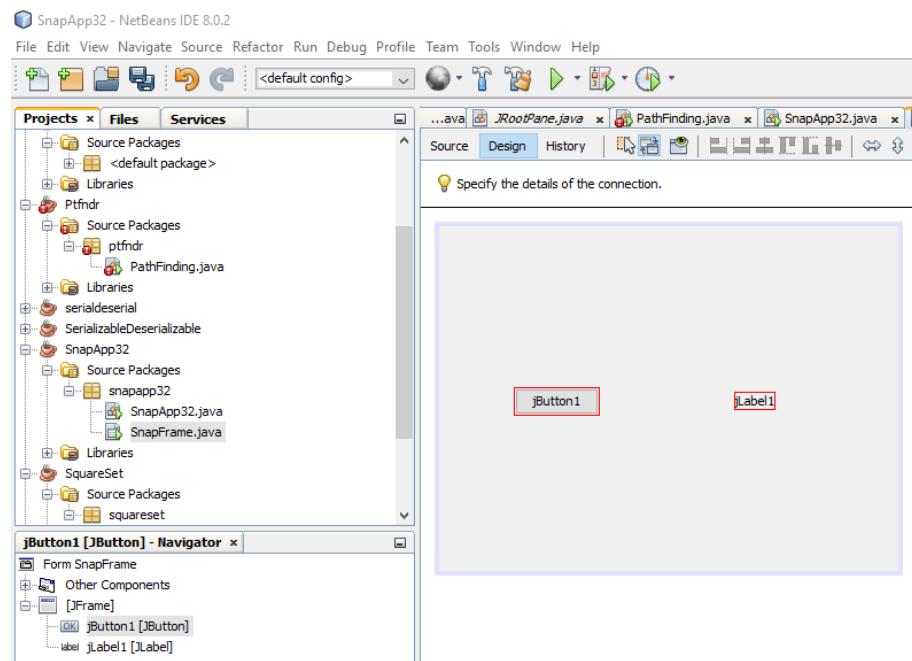
5. Buat tampilan seperti berikut ini

Digunakan sebuah *button* dan sebuah *label*



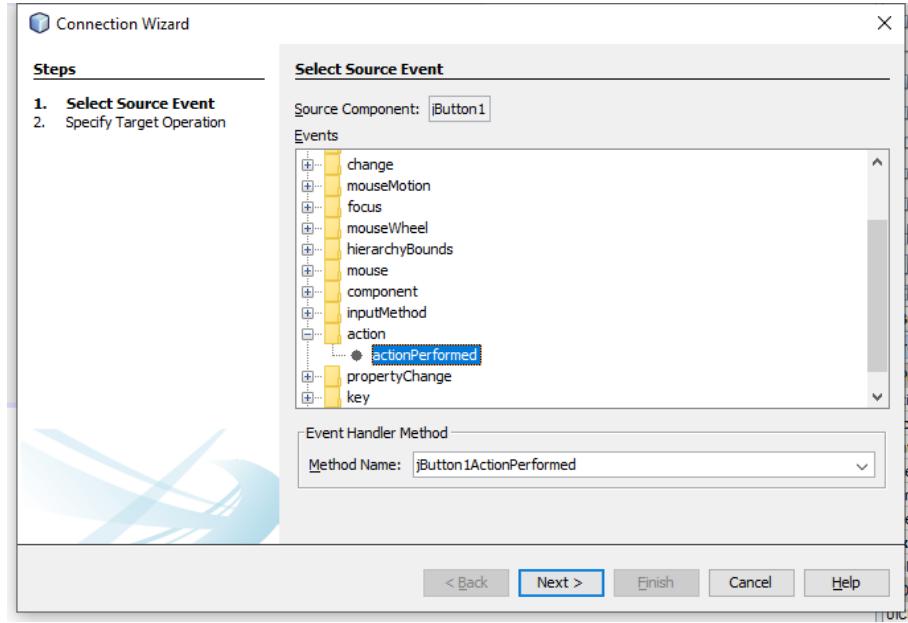
Gambar 2.10 Tampilan pada snapframe

6. Klik *Connection Mode*, kemudian klik pada tombol, kemudian klik pada label. *Connection Mode* digunakan untuk memberikan nilai pada label ketika *button* diberi *action*.  
*Connection Mode* pada dasarnya adalah memanipulasi suatu elemen jika terdapat aksi yang dilakukan pada elemen lain



Gambar 2.11 *Connection mode*

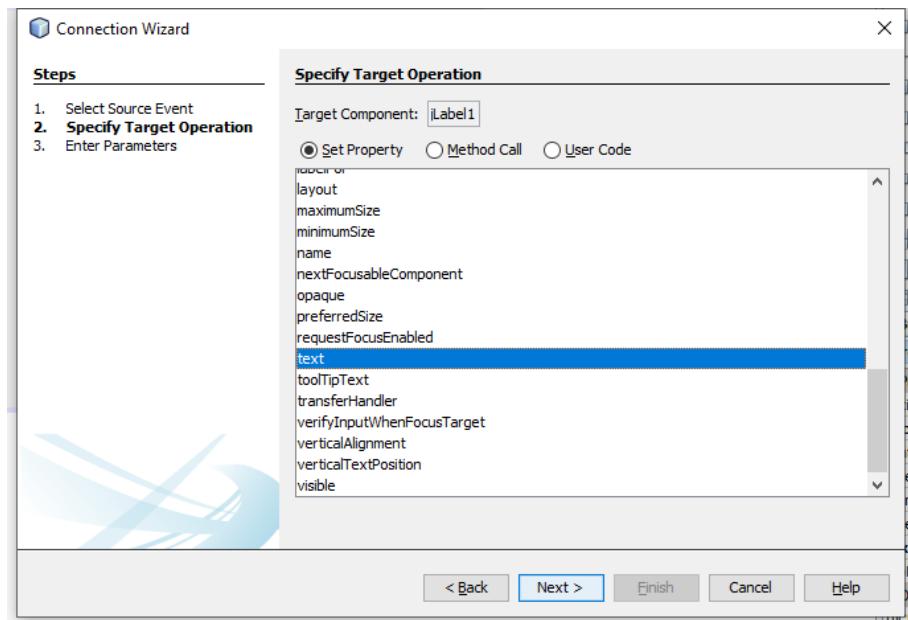
7. Setelah itu akan muncul *connection Wizard*, pilih *action* → *actionPerformed*. Langkah ini akan membuat *button* ketika diberi *action* akan menjalankan *events actionPerformed*. Event *actionPerformed* akan menangkap aksi yang dilakukan pada *button* sebagai *input* untuk diolah



Gambar 2.12 Memilih *action*

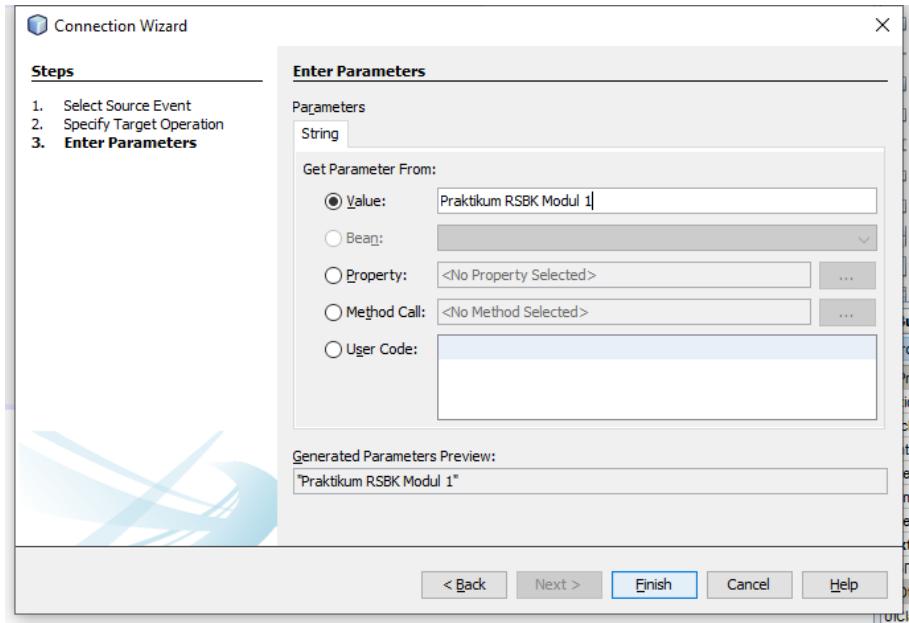
8. Sekarang pengaturan pada labelnya, pilih set *property* → *Text*. Hal ini bertujuan untuk memilih *action* yang akan terjadi pada label.

*Set property* akan membuat label menjadi objek dari *output* yang dihasilkan



Gambar 2.13 Memilih keluaran pada target komponen

9. Isi Value “praktikum rsbk modul 1”, kemudian *finish*. Ini akan menghasilkan *output* “praktikum rsbk modul 1” pada label ketika *button* diberi *action*.  
*Value* akan mendeklarasikan *output* yang ditaruh pada objek yaitu label



Gambar 2.14 Isi value label

10. Secara otomatis value akan muncul seperti source code berikut

Source code dari pengolahan action yang telah dibuat secara GUI

```

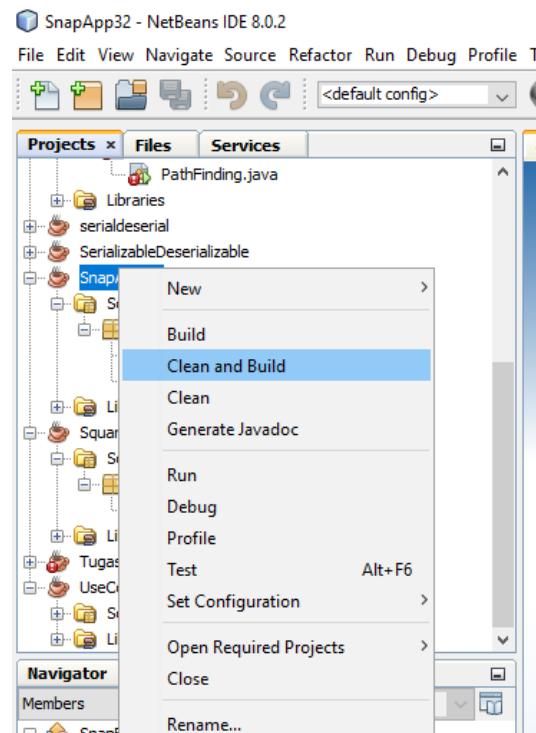
67
68  private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
69      jLabel1.setText("Praktikum RSBK Modul 1");
70  }
71

```

Gambar 2.15 Source code jButtonActionPerformed

## 11. Clean and Build

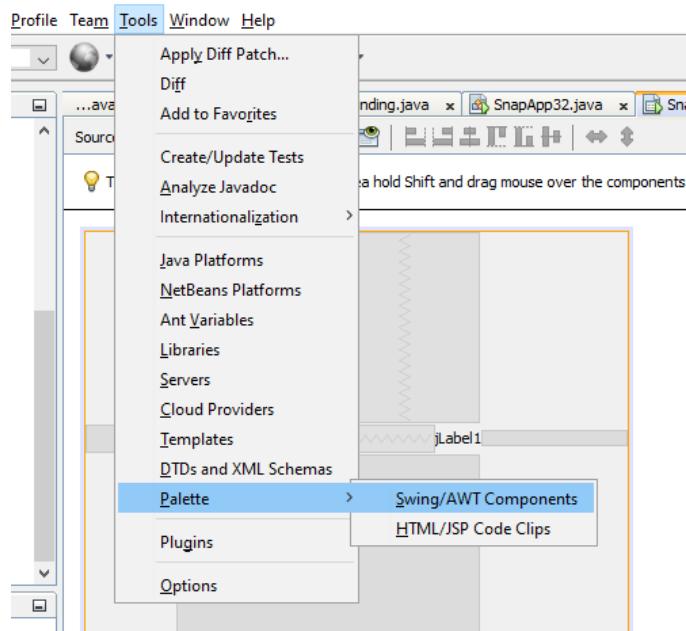
Agar project tercompile



Gambar 2.16 Clean and Build Project

12. Klik Tool → Pallette → Swing / AWT Components untuk mengimport component baru.

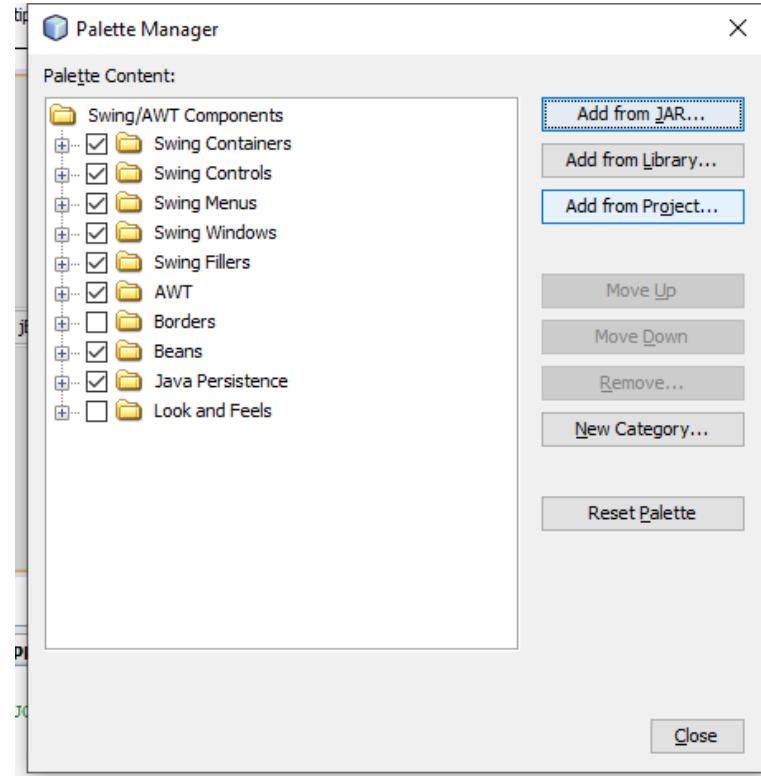
Digunakan untuk mengimpor komponen Swing/AWT



Gambar 2.17 Import beans

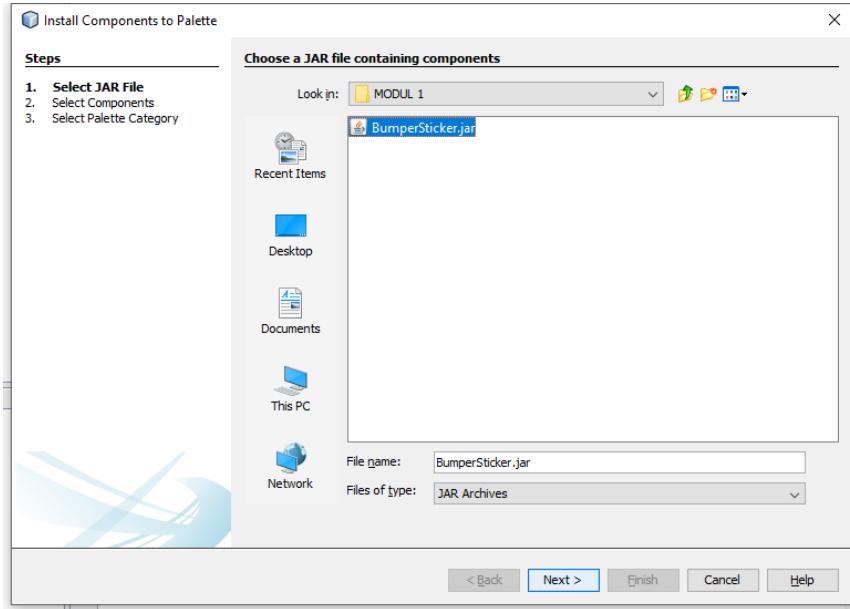
13. Klik Add From Jar untuk mulai memilih package component.

Untuk memilih package dari suatu komponen berupa file berekstensi .jar



Gambar 2.18 Menambahkan komponen dari .jar yang sudah ada

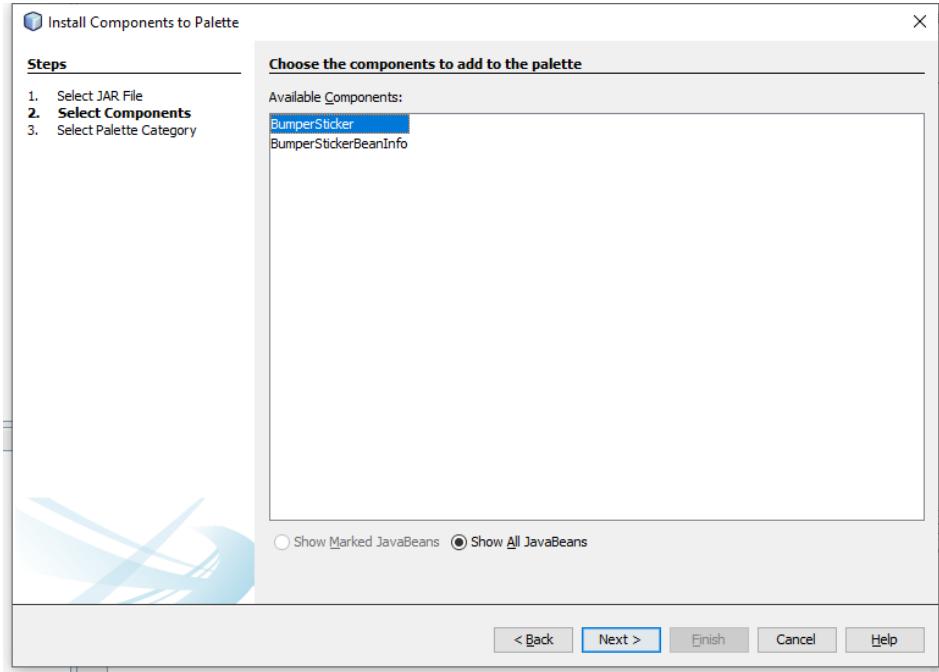
14. Kemudian cari lokasi berkas BumperSticker, pilih berkasnya BumperSticker.jar  
Memilih *file package* komponen dalam suatu *direktori*



Gambar 2.19 Mencari lokasi bumperSticker.jar

### 15. Pilih bumperSticker

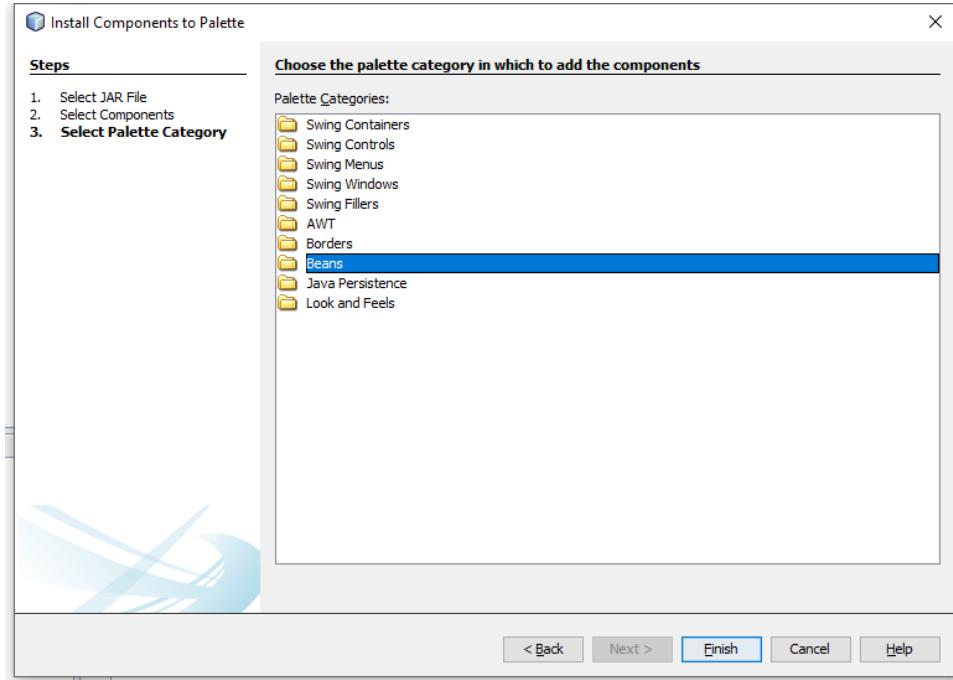
Memilih komponen dari suatu *package* yang akan dipakai



Gambar 2.20 Memilih bumperSticker

### 16. Pilih Beans yang berarti meletakkan *component* baru tersebut ke folder beans.

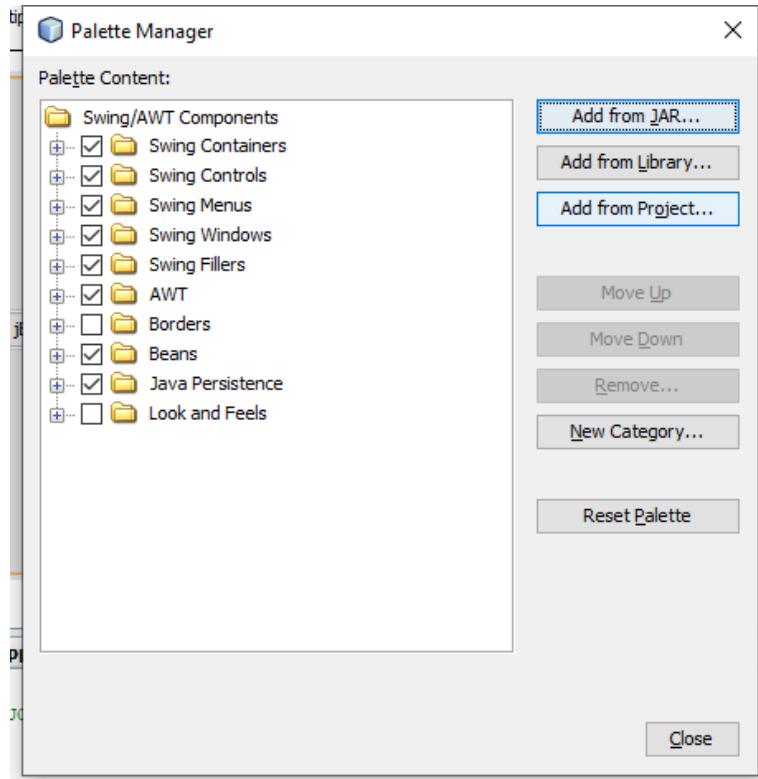
Komponen tersebut diletakkan ke dalam folder Beans, yang merupakan salah satu kategori komponen



Gambar 2.21 Memilih folder tujuan

17. Klik *close* setelah selesai *mengimport component*.

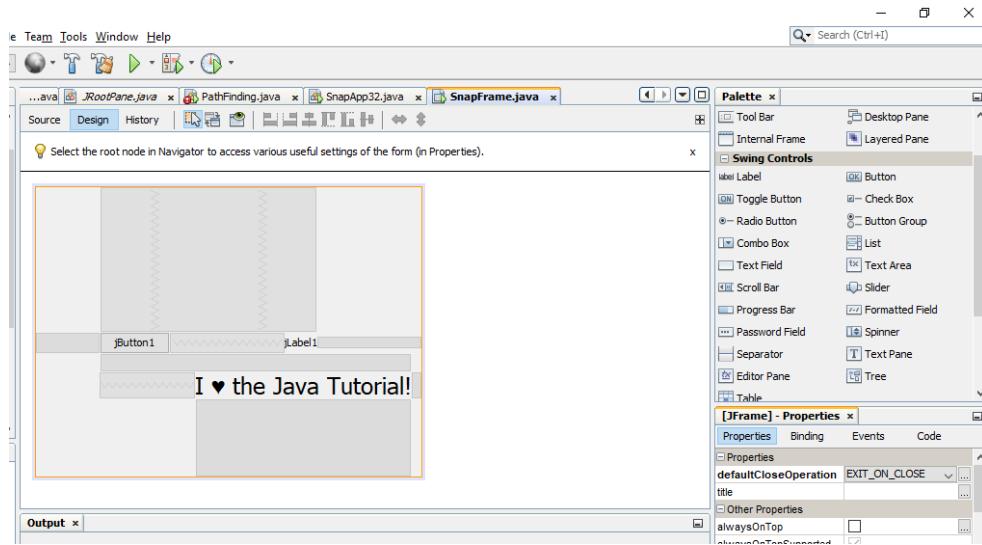
Menutup jendela impor komponen



Gambar 2.22 Selesai mengimport komponen

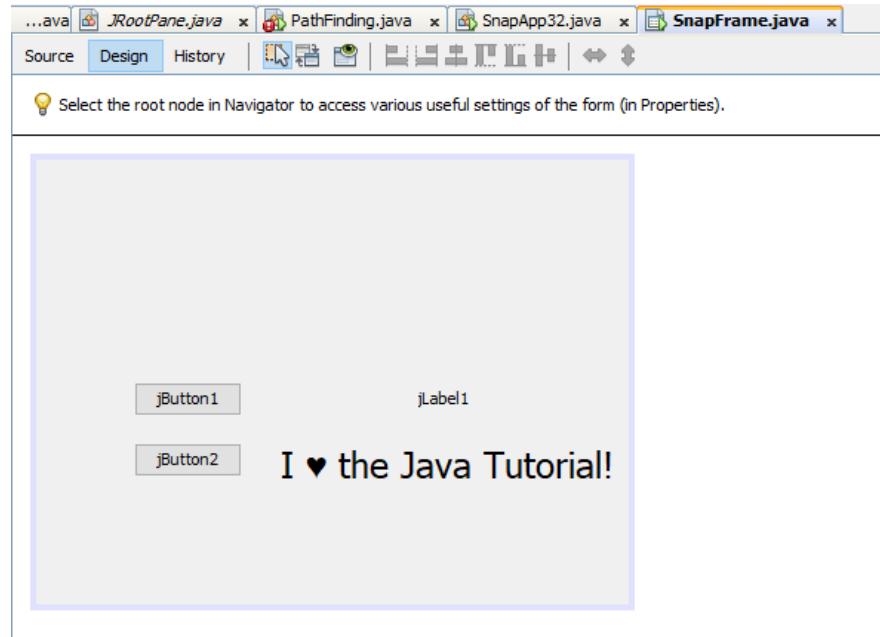
18. *Drag Beans* → BumperSticker ke *Form*

Menggunakan komponen di dalam *form*



Gambar 2.23 Menambahkan bumpersticker ke form

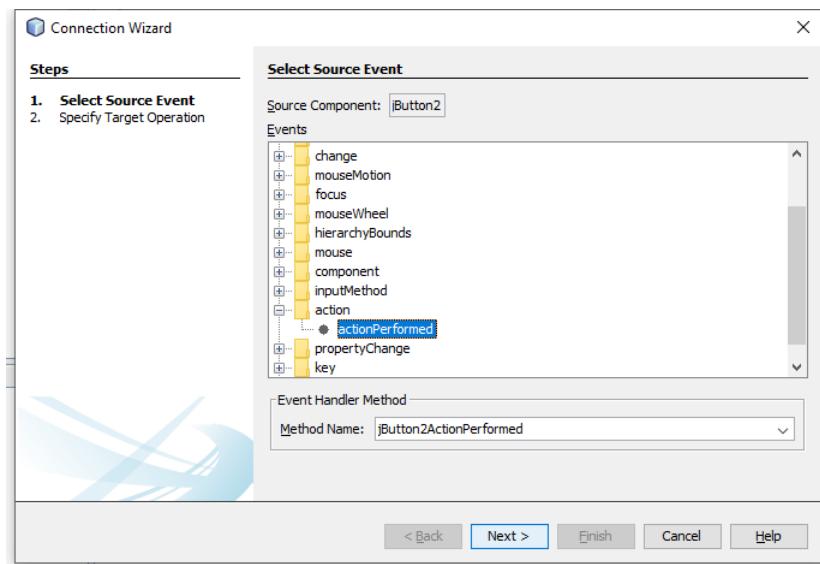
Tambahkan satu *button* lagi untuk menjalankan animasinya,



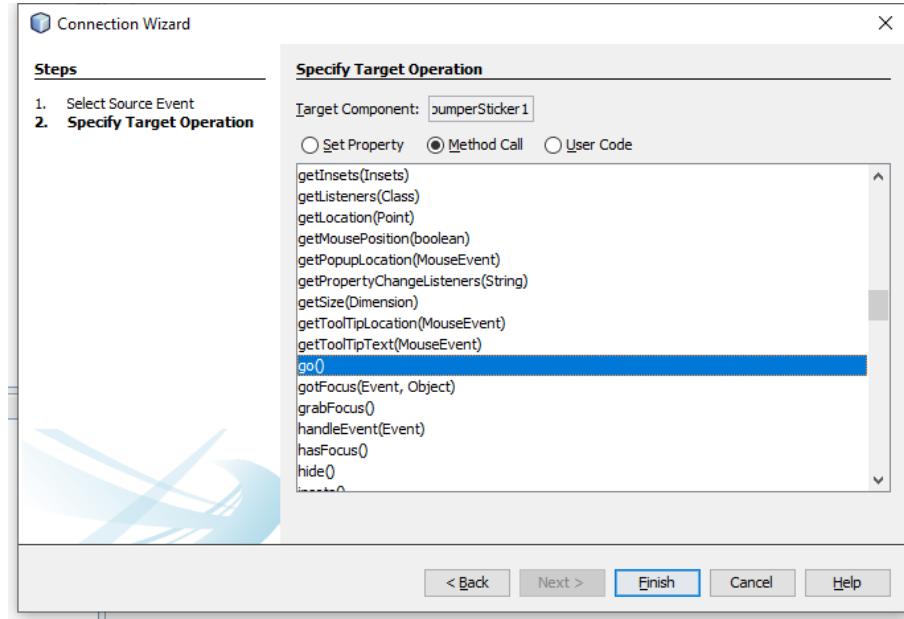
Gambar 2.24 Menambahkan *button*

19. Koneksikan Btton 2 dengan Bean BumperSticker, gunakan *actionperformed*, pilih MethodCall, pilih method go()

Menangkap *action* yang digunakan sebagai *input* untuk diolah menjadi *output*



Gambar 2.25 Memilih *actionPerformed* sebagai *event*



Gambar 2.26 *Connection mode* antara bumpersticker dan button

20. Setelah itu run projectnya. Jika jButton1 ditekan maka label akan menampilkan teks “praktikum rsbk modul 1” dan jika jButton2 ditekan maka gambar *love* akan berkedip merah hitam.



Gambar 2.27 Hasil run program

## 2.5 Tugas dan Pembahasan

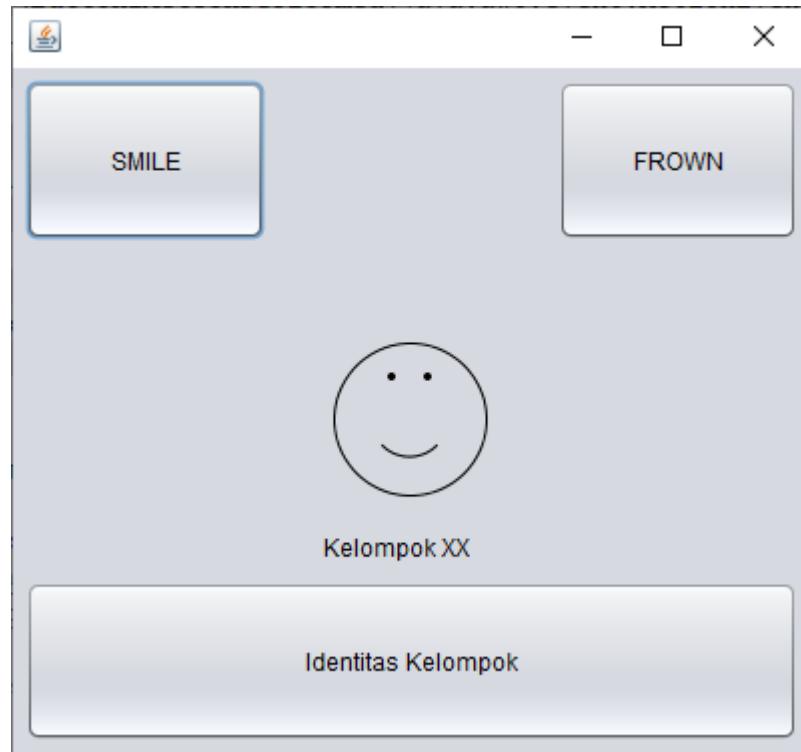
### 2.5.1 Tugas 1

*Source code* pembentuk wajah

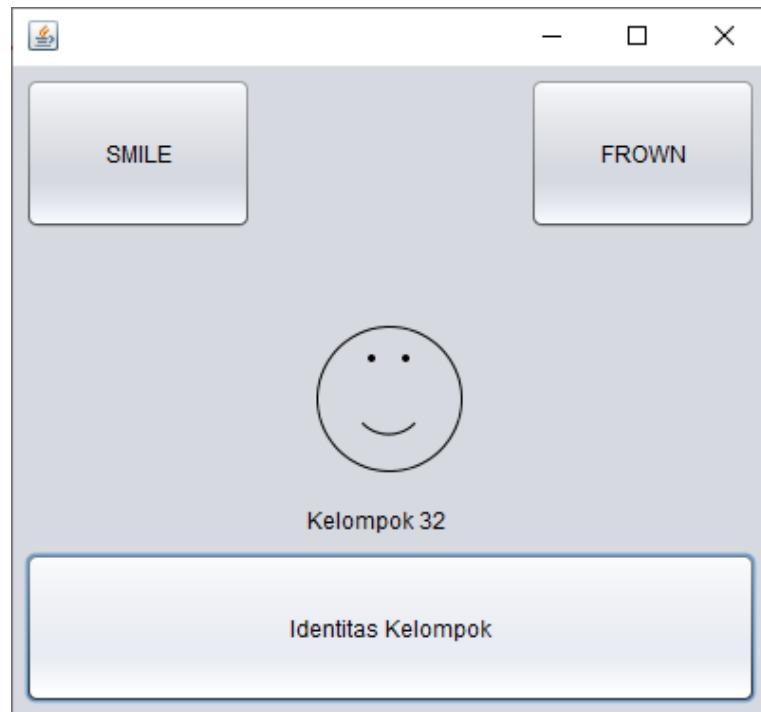
```
public void paint(Graphics g) {
    Graphics2D g2 = (Graphics2D)g;
    g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
                        RenderingHints.VALUE_ANTIALIAS_ON);
    // Face
    int w = getWidth();
    int h = getHeight();
    int pad = 12;
    int cw = w - pad * 2;
    int ch = h - pad * 2;
    g2.setColor(getBackground());
    g2.fillArc(pad, pad, cw, ch, 0, 360);
    g2.setColor(getForeground());
    g2.drawArc(pad, pad, cw, ch, 0, 360);
    // Mouth
    int sw = cw / 2;
    int sh = ch / 2;
    if (mSmile == true)
        g2.drawArc(w / 2 - sw / 2, h / 2 - sh / 2, sw, sh, 270
- mMouthWidth / 2, mMouthWidth);
    else
        g2.drawArc(w / 2 - sw / 2, h / 2 + sh / 3, sw, sh, 90 -
mMouthWidth / 2, mMouthWidth);
    // Eyes
    int er = 4;
    g2.fillArc(w / 2 - cw * 1 / 8 - er / 2, h / 2 - ch / 4 -
er, er, er, 0, 360);
    g2.fillArc(w / 2 + cw * 1 / 8 - er / 2, h / 2 - ch / 4 -
er, er, er, 0, 360);
}
```

Pada *source code* di atas, adalah fungsi untuk membuat sebuah *graphics* atau gambar pada sebuah *canvas*. Gambar ini dibuat berdasarkan fungsi bawaan dari *graphics* seperti *setColor* digunakan untuk memberi warna. *fillArc* digunakan untuk mengisi objek berupa lengkungan, *drawArc* digunakan untuk menggambar objek berupa lengkungan. Dengan perhitungan yang sesuai, dapat dibuat sebuah gambar menggunakan fungsi-fungsi tersebut. Kemudian pada pengkondisian *mSmile*, jika *mSmile* *true* maka akan dijalankan sebuah *code* dimana garis yang dibuat berupa garis

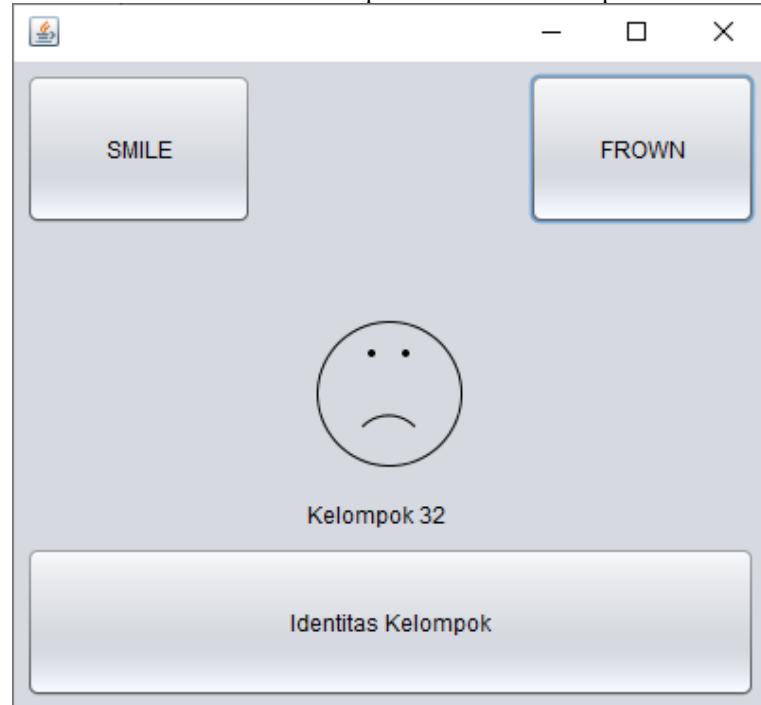
melengkung membentuk senyuman. Namun jika *false*, garis yang dibuat berupa garis melengkung membentuk raut sedih.



Gambar 2.28 Kondisi awal



Gambar 2.29 Menampilkan identitas kelompok



Gambar 2.30 Kondisi wajah sedih

Terdapat 3 buah *button* pada program ini, yaitu *button smile* untuk mengubah ekspresi wajah pada komponen yang di masukan. Fungsi yang di jalankan adalah fungsi *smile* yang di dalamnya memasukan nilai mSmile sebagai *true*. *Button Frown* adalah button yang akan menjalankan fungsi *frown* yang di dalamnya memasukan nilai mSmile sebagai *false*, sehingga wajah menjadi sedih. *Button* ketiga adalah *button* untuk menunjukan identitas kelompok.

## 2.5.2 Tugas 2

Source code Colors

```
package Colors;
import java.awt.*;
import java.awt.event.*;
import java.io.*;
public class Colors extends Canvas implements Serializable{
    private Color color;
    private boolean rect;
    public Colors(){
        rect=false; setSize(100,100);
        change();
    }
    public boolean getRect(){
        return rect;
    }
    public void setRect(boolean flag){
        this.rect=flag; repaint();
    }
    public void change(){
        color = randomColor();
        repaint();
    }
    private Color randomColor(){
        int r=(int) (255*Math.random());
        int g =(int) (255*Math.random());
        int b=(int) (255*Math.random());
        return new Color(r,g,b);
    }
    public void paint(Graphics g){
        Dimension d = getSize();
        int h=d.height;
        int w=d.width;
        g.setColor(color);
        if(rect){
            g.fillRect(0,0,w-1,h-1);
        }else{
            g.fillOval(0,0,w-1,h-1);
        }
    }
}
```

Pada *source code* Colors, dideklarasikan *method* randomColor() yang akan menghasilkan warna secara acak. Setelahnya, digunakan *method* repaint() untuk mengaplikasikan warna atau mengubah warna yang sudah ada di dalam objek. Selain itu, juga dideklarasikan komponen *rectangle* yang berukuran panjang dan lebar masing-masing sebesar 100px. Namun, komponen *rectangle* mempunyai nilai boolean false yang berarti terdapat komponen *rectangle* pada *form* namun tidak ditampilkan.

### Source code ColorsTriangle

```
package Colors;
import java.awt.*;
import java.awt.event.*;
import java.io.*;
public class ColorsTriangle extends Canvas implements
Serializable{
    private Color color;
    private boolean rect;
    public ColorsTriangle(){
        rect=false; setSize(100,100);
        change();
    }
    public boolean getRect(){
        return rect;
    }
    public void setRect(boolean flag){
        this.rect=flag; repaint();
    }
    public void change(){
        color = randomColor();
        repaint();
    }
    private Color randomColor(){
        int r=(int) (255*Math.random());
        int g =(int) (255*Math.random());
        int b=(int) (255*Math.random());
        return new Color(r,g,b);
    }
    public void paint(Graphics g){
        Dimension d = getSize();
        int h=d.height;
        int w=d.width;
        g.setColor(color);
        if(rect){
            g.fillPolygon(new int[] {0,99,99}, new int[]{50,99,1}, 3);
        }else{
            g.fillPolygon(new int[] {0,99,99}, new int[]{50,99,1}, 3);
        }
    }
}
```

### Source code ColorsTriangle2

```

package Colors;
import java.awt.*;
import java.awt.event.*;
import java.io.*;
public class ColorsTriangle2 extends Canvas implements
Serializable{
    private Color color;
    private boolean rect;
    public ColorsTriangle2(){
        rect=false; setSize(100,100);
        change();
    }
    public boolean getRect(){
        return rect;
    }
    public void setRect(boolean flag){
        this.rect=flag; repaint();
    }
    public void change(){
        color = randomColor();
        repaint();
    }
    private Color randomColor(){
        int r=(int) (255*Math.random());
        int g =(int) (255*Math.random());
        int b=(int) (255*Math.random());
        return new Color(r,g,b);
    }
    public void paint(Graphics g){
        Dimension d = getSize();
        int h=d.height;
        int w=d.width;
        g.setColor(color);
        if(rect){
            g.fillPolygon(new int[] {99,0,0}, new int[]{50,99,1}, 3);
        }else{
            g.fillPolygon(new int[] {99,0,0}, new int[]{50,99,1}, 3);
        }
    }
}

```

Pada *source code* ColorsTriangle dan ColorsTriangle2, dideklarasikan ukuran segitiga yaitu dengan panjang dan lebar masing-masing sebesar 100px. Kemudian, digunakan *method* repaint() untuk mengaplikasikan warna acak dari *method* randomColor() yang telah dideklarasikan di dalam *method call* change() setelah button mendapat action berupa ditekan.

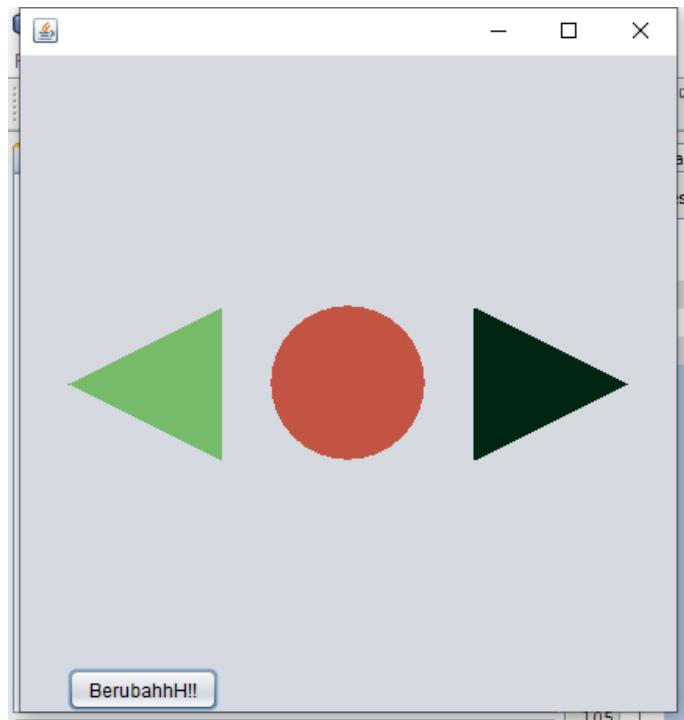
### Source code Action Button

```

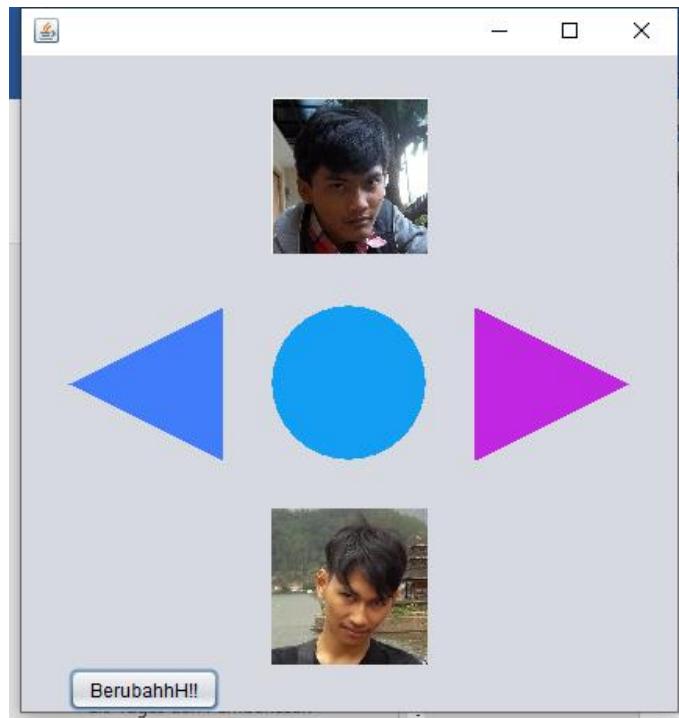
colors1.change();
colorsTriangle1.change();

```

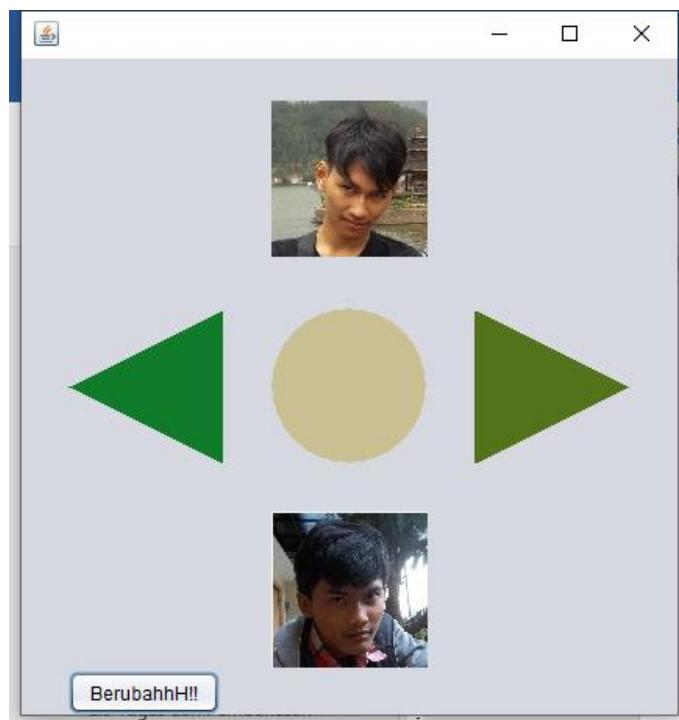
```
colorsTriangle21.change();
if (cek){
    jLabel1.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/tugaspraktikum
rsbk1/riokisna100p.jpg")));
    jLabel2.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/tugaspraktikum
rsbk1/daud100p.jpg")));
    cek = false;
}
else {
    jLabel2.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/tugaspraktikum
rsbk1/riokisna100p.jpg")));
    jLabel1.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/tugaspraktikum
rsbk1/daud100p.jpg")));
    cek = true;
}
```



Gambar 2.31 Kondisi awal program



Gambar 2.32 Ketika Button di tekan sekali



Gambar 2.33 Gambar di tekan kedua kalinya

Ketika tombol berubah mendapat *action* berupa ditekan, tombol akan memberikan *method call* berupa change(). Pada *source code* ColorsTriangle dan ColorsTriangle2, telah didefinisikan *argument* di dalam method change yaitu repaint() komponen dengan warna dari *method* randomColor(). Namun pada komponen *rectangle*, *argument* didefinisikan di dalam *source code* Action *Button*. Ketika *button* ditekan, maka nilai boolean dari *rectangle* akan berubah dari *false* menjadi *true*, sehingga method repaint() dapat digunakan, *Method* repaint() pada *rectangle* digunakan bukan untuk mengubah warna dari *rectangle* menggunakan *method* randomColor() seperti pada ColorsTriangle dan ColorsTriangle2, namun akan mengimpor gambar *icon* pada direktori dari *project* yang telah diatur. Sehingga, akan muncul gambar *icon*. Agar gambar *icon* dapat bertukar tempat setiap kali *button* ditekan, digunakan pengkondisian *if else* dengan parameter cek=*true* atau cek=*false*, sesuai dengan *state button* setelah ditekan dan argumen mengimpor gambar *icon* dengan posisi saling berkebalikan.

Link Github : <https://github.com/riokisna/Praktikum-RSBK-Kelompok32>

## 2.6 Kesimpulan

1. Dengan menggunakan Java Beans, mengembangkan aplikasi dapat dibuat secara mudah dan praktis karena setiap komponen dianggap sebagai objek.
2. Dengan mengembangkan aplikasi menggunakan Java Beans, penulisan *source code* secara manual berkurang karena sebagian besar pengembangannya menggunakan GUI
3. *Connection Mode* digunakan untuk menghubungkan dua komponen yang bekerjasama mengolah input untuk menghasilkan *output*.
4. Komponen dapat digunakan sebagai objek untuk menerima *input* dengan cara memberi *event* yang beragam pada *source event connection mode wizard*.
5. Komponen juga dapat digunakan sebagai objek untuk menampilkan output dengan cara memilih komponen menjadi target *operation* pada *connection mode wizard*.
6. Komponen dapat dibuat dan digunakan ulang dengan cara mengimpor *package component*.

## **BAB III**

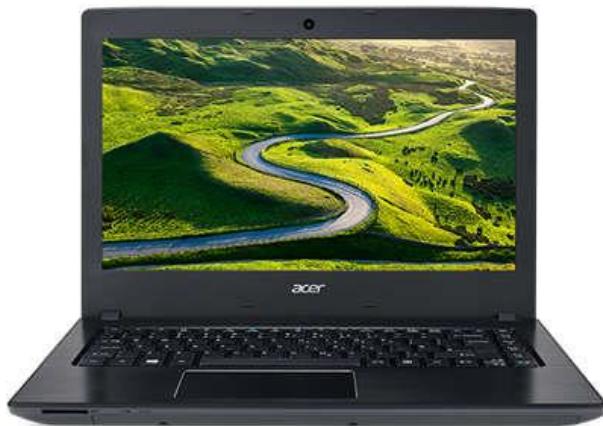
### **EJB SESSION BEAN**

#### **3.1 Tujuan**

1. Praktikan mengetahui apa itu *session* bean
2. Praktikan mengenal glashfish server pada netbeans
3. Praktikan mengetahui perbedaan *Stateless* dan *Statefull*
4. Praktikan dapat mengetahui fungsi dari *class* Servlet

### 3.3 Alat dan Bahan

#### 1. Laptop



Gambar 3.1 Laptop

Laptop digunakan sebagai media instalasi keperluan praktikum seperti NetBeans.

#### 2. NetBeans IDE



Gambar 3.2 NetBeans

NetBeans adalah suatu serambi pengembangan perangkat lunak yang ditulis dalam bahasa pemrograman Java. Serambi Pada NetBeans, pengembangan suatu aplikasi dapat dilakukan dimulai dari setelan perangkat lunak modular bernama *modules*.

#### 3. Java EE SDK



Gambar 3.3 Java *Enterprise Edition*

Java EE adalah plugin netbeans untuk membuat sebuah *Project* web yang berisi modul EJB dan modul WAR.

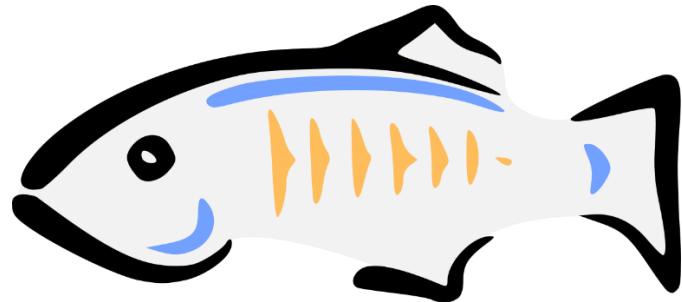
#### 4. Java SDK 7



Gambar 3.4 Jawa SDK

Java Standard Edition Development Kit adalah implementasi salah satu dari paket Edisi standar yang ditujukan untuk pengembangan java di Windows.

## 5. Glassfish Server



Gambar 3.5 Glassfish Server

Glassfish server adalah salah satu web server yang kerap digunakan dalam pengembangan aplikasi web J2EE.

### **3.3 Dasar Teori**

#### **3.3.1 Java Enterprise Edition (Java EE)**

Java *Enterprise* Edition ( Java EE ), sebelumnya Java 2 Platform, *Enterprise* Edition ( J2EE ), saat ini berganti nama menjadi Jakarta EE, adalah seperangkat spesifikasi, memperluas Java SE 8<sup>[1]</sup> dengan spesifikasi untuk fitur-fitur perusahaan seperti komputasi terdistribusi dan layanan web .<sup>[2]</sup> Aplikasi Java EE dijalankan pada runtimes referensi , yang dapat berupa server layanan mikro atau server aplikasi , yang menangani transaksi, keamanan, skalabilitas, konkurensi , dan pengelolaan komponen yang dikerahkannya.

Java EE didefinisikan oleh spesifikasinya . Spesifikasi mendefinisikan API (antarmuka pemrograman aplikasi) dan interaksinya. Seperti spesifikasi Proses Komunitas Java lainnya , penyedia harus memenuhi persyaratan kesesuaian tertentu untuk menyatakan produk mereka sebagai yang *sesuai* dengan *Java EE* .

Java EE mencakup beberapa spesifikasi yang melayani berbagai tujuan, seperti menghasilkan halaman web, membaca dan menulis dari database dengan cara transaksional, mengelola antrian yang didistribusikan. Java EE APIs mencakup beberapa teknologi yang memperluas fungsionalitas Java SE API dasar , seperti *Enterprise JavaBeans* , konektor , servlets , JavaServer Pages, dan beberapa teknologi layanan web .

*Sumber :* [\*https://en.wikipedia.org/wiki/Java\\_Platform,\\_Enterprise\\_Edition\*](https://en.wikipedia.org/wiki/Java_Platform,_Enterprise_Edition)

#### **3.3.2 Component Statefull**

*Stateful* adalah disaat informasi yang diberikan sebelumnya disimpan dan mempengaruhi konten/informasi/data yang akan diberikan setelahnya. Berikut ini adalah karakteristik *Statefull* komponen:

1. Server menyimpan informasi tentang *file* yang terbuka, dan posisi *file* sekarang (current position),
2. Open (dibuka) sebelum access dan kemudian ditutup, dan
3. Menyediakan *file locks*.

(Sumber : <https://medium.com/@wafaakamilahmaulanihermawan/Statefull-vs-Stateless-application-6d600eddefe3>)

### **3.3.3 Component Stateless**

*Stateless* adalah disaat informasi tidak disimpan sehingga tampilan web akan sama saja kalau dilihat oleh anda berulang kali atau oleh orang lain.

1. Server tidak menyimpan state informasi,
2. Operasi *file* harus mengandung semua yang diperlukan (memuat pesan yang lengkap),
3. Dapat dengan mudah di-recovery apabila terjadi client ataupun server crash, dan
4. Membutuhkan extra lock server untuk mempertahankan state.

(Sumber : <https://medium.com/@wafaakamilahmaulanihermawan/Statefull-vs-Stateless-application-6d600eddefe3>)

### **3.3.4 Java Server Page (JSP)**

JSP adalah sebuah bahasa pemrograman berbasis Java yang diperuntukkan untuk membuat sebuah website. JSP sangat sesuai dan tangguh untuk menangani kebutuhan akan sebuah website. Sama seperti PHP, bahasa yang satu ini dikhususkan untuk berkutat di bagian backend. Artinya, bagian static yang berupa tag-tag HTML akan terpisah dari kode JSP. Kita dapat membuat halaman web static dengan html dan css seperti biasanya, kemudian akan disisipi dengan JSP untuk membuat halaman web menjadi dinamis.

Ada beberapa alasan penting yang membuat JSP cukup terkenal dalam pengembangan aplikasi web:

Yang pertama, JSP menggunakan bahasa Java. Bagi sebagian orang yang sudah terbiasa dengan Java pasti tidak akan kesulitan saat mempelajari ataupun menggunakan bahasa yang satu ini.

Yang kedua, JSP mendukung multiplatform sama seperti Java. Dalam hal ini, JSP memungkinkan kode dapat dipindah-pindahkan ke berbagai platform tanpa perlu melakukan perubahan apapun.

Dan yang ketiga, JSP cenderung memiliki perfomansi yang lebih baik, karena dalam JSP, dilakukan proses compile terlebih dahulu menjadi servlet. Servlet adalah *class* Java yang siap dijalankan oleh web server.

(Sumber : <https://www.mahircoding.com/pengenalan-jsp/>)

### 3.3.5 Java Servlet

Servlet adalah bahasa pemrograman Java kelas digunakan untuk memperluas kemampuan dari server yang tuan aplikasi mengakses melalui model pemrograman request-respon. Meskipun servlet dapat menanggapi setiap jenis permintaan, mereka biasanya digunakan untuk memperpanjang aplikasi host oleh server Web. Dengan demikian, dapat dianggap sebagai Java Applet yang berjalan pada server bukan browser.

Servlet adalah berbasis Java server-side teknologi web. Sesuai namanya, melayani permintaan klien dan menerima respon dari server. Secara teknis, Servlet adalah kelas Java dalam Java EE yang sesuai dengan Java Servlet API, sebuah protokol di mana kelas Java mungkin merespon permintaan. Mereka tidak terikat dengan sebuah protokol client-server khusus, tetapi yang paling sering digunakan dengan protokol HTTP. Oleh karena itu, kata “Servlet” sering digunakan dalam arti “Servlet HTTP”. [2] Dengan demikian, seorang pengembang perangkat lunak dapat menggunakan servlet untuk menambahkan konten dinamis ke server Web menggunakan platform Java. Isi dihasilkan umumnya HTML, tetapi mungkin data lain seperti XML. Servlets adalah mitra Jawa non-Jawa teknologi konten web yang dinamis seperti PHP dan

ASP.NET. Servlets dapat mempertahankan negara dalam variabel sesi transaksi di server banyak dengan menggunakan cookie HTTP, atau menulis ulang URL. Untuk menyebarkan dan menjalankan Servlet, wadah Web harus digunakan. Sebuah wadah Web (juga dikenal sebagai wadah Servlet) pada dasarnya adalah komponen Web server yang berinteraksi dengan servlet. Wadah Web bertanggung jawab untuk mengelola siklus hidup servlets, pemetaan URL ke servlet tertentu dan memastikan bahwa URL pemohon memiliki hak akses yang benar.

API servlet, yang terkandung dalam hirarki javax.servlet paket Java, mendefinisikan interaksi yang diharapkan dari wadah Web dan servlet [2]. Servlet adalah sebuah objek yang menerima permintaan dan menghasilkan respon berdasarkan permintaan itu. Paket servlet mendefinisikan dasar Jawa objek untuk mewakili servlet permintaan dan tanggapan, serta sebagai objek untuk mencerminkan konfigurasi servlet parameter dan lingkungan eksekusi. Paket javax.servlet.http mendefinisikan HTTP spesifik subclass dari elemen servlet generik, termasuk objek manajemen sesi yang melacak beberapa permintaan dan tanggapan antara Web server dan klien. Servlets dapat dikemas dalam *file* WAR sebagai aplikasi Web. Servlets dapat dihasilkan secara otomatis dari JavaServer Pages (JSP) oleh kompilator JavaServer Pages. Perbedaan antara Servlets dan JSP adalah bahwa Servlets biasanya menanamkan HTML di dalam kode Java, sedangkan JSP embed kode Java dalam HTML. Sedangkan penggunaan langsung dari Servlets untuk menghasilkan HTML (seperti yang ditunjukkan pada contoh di bawah) telah menjadi langka, tingkat yang lebih tinggi MVC kerangka web di Java EE (JSF) masih secara eksplisit menggunakan teknologi Servlet untuk penanganan permintaan / tanggapan tingkat rendah melalui FacesServlet yang . Sebuah penggunaan agak lebih tua adalah dengan menggunakan servlet bersama dengan JSP dalam pola yang disebut “Model 2”, yang merupakan rasa pola model-view-controller.

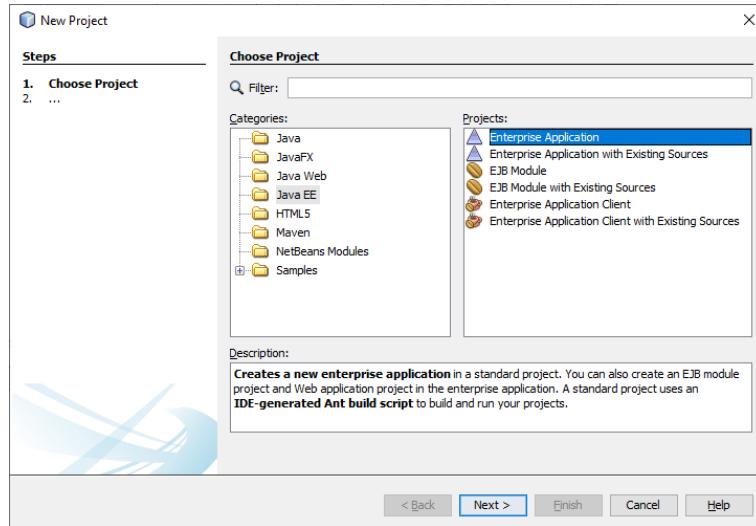
(Sumber : <https://aeroyid.wordpress.com/2012/07/16/javamemperkenalkan-java-servlet-dan-membuat-helloworld/>)

### 3.4 Langkah Kerja

1. Buka aplikasi NETBEANS IDE.

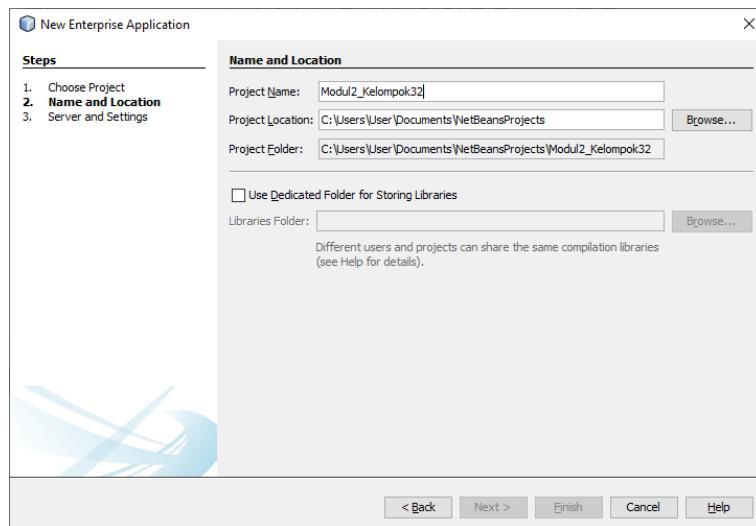
Di sini menggunakan Netbeans 8.0.2

2. Buat *Project* baru pilih Java EE > *Enterprise Application* lalu klik next  
Java EE adalah *Project* yang digunakan untuk membuat web service



Gambar 3.6 New Project Java EE

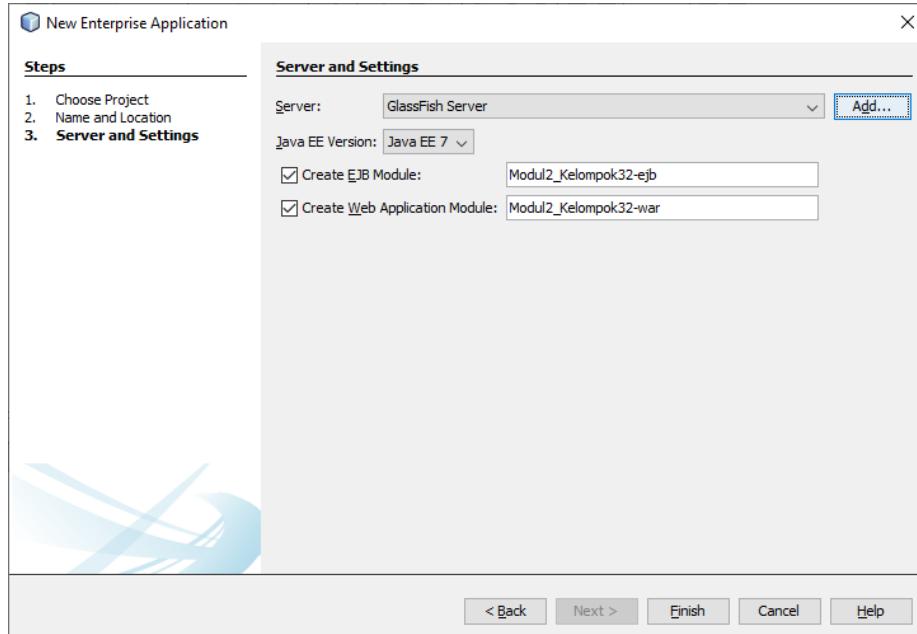
3. Pilih location *Project* lalu klik next



Gambar 3.7 Memberi nama *Project*

4. Pilih server glashfish server yang, ikuti seperti gambar.

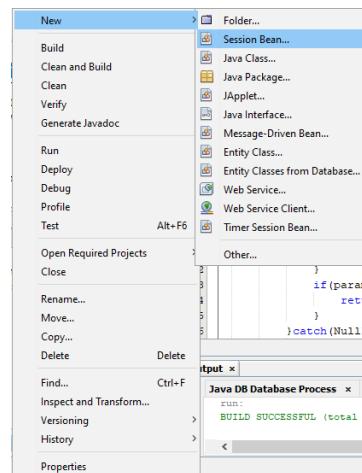
GlassFish Server adalah web server yang digunakan untuk *Project* ini.



Gambar 3.8 Memilih GlassFish Server untuk Web Server

5. Buat *file Session Been* baru dengan cara klik kanan *Project* EJB pilih New>Session Bean.

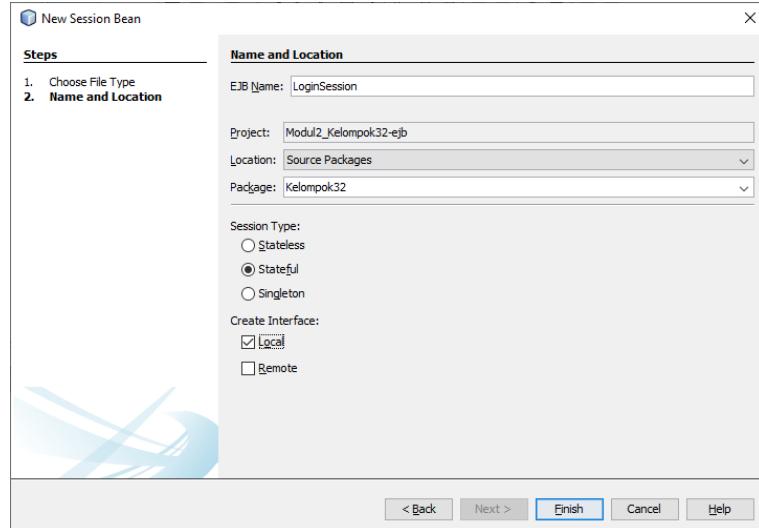
*Session bean* adalah sebuah *file java* yang dapat di panggil ke dalam servlet nantinya.



Gambar 3.9 Membuat Session Bean

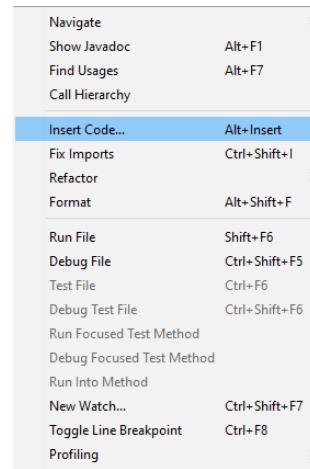
6. Pada Isikan LoginSession pada **EJB Name**. Pada **Session Type** pilih *Stateful* dan centang pilihan Local pada **Create Interface**.

*Stateful* artinya data dapat disimpan secara sementara, sedangkan local interface akan membuat file interface yang terhubung dengan session bean ini.



Gambar 3.10 Memberi nama Session Bean

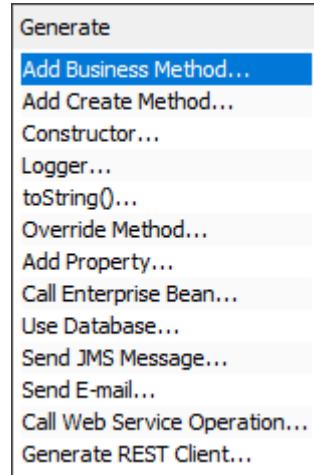
7. Klik kanan pada text editor dan pilih **Insert Code**



Gambar 3.11 Insert Code

8. Pilih **Add Business Methode**.

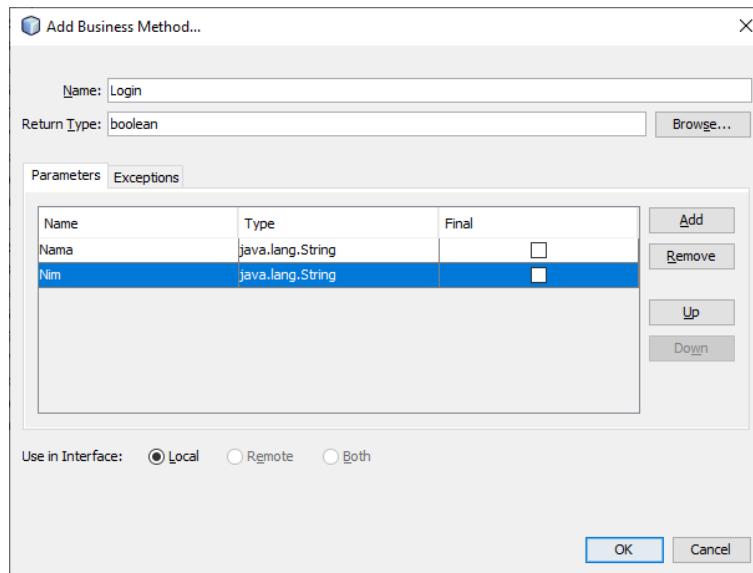
Fitur ini digunakan untuk memasukan fungsi yang terhubung langsung ke interface dengan mudah.



Gambar 3.12 Generate code

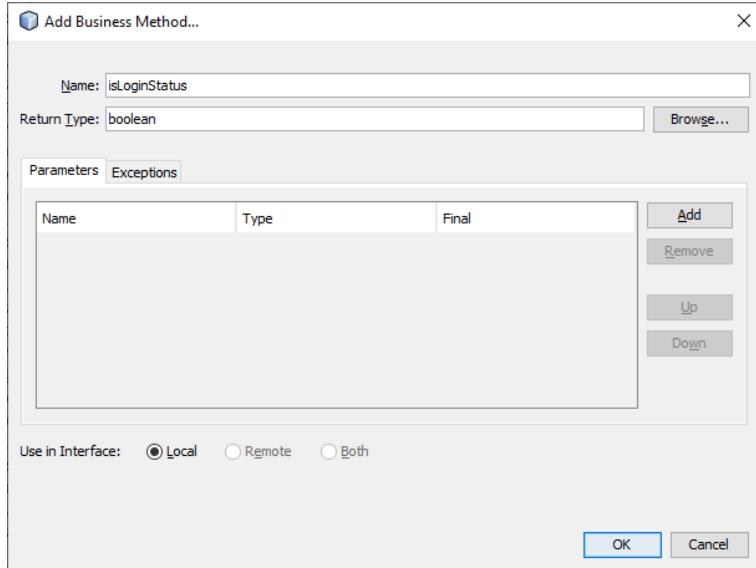
9. Masukan nama Login dengan **Return Type** boolean. Tambahkan 2 parameter dengan nama **Nama** dan **Nim**.

Parameter di masukan beserta nama fungsinya, yang otomatis akan membuat sebuah fungsi return.

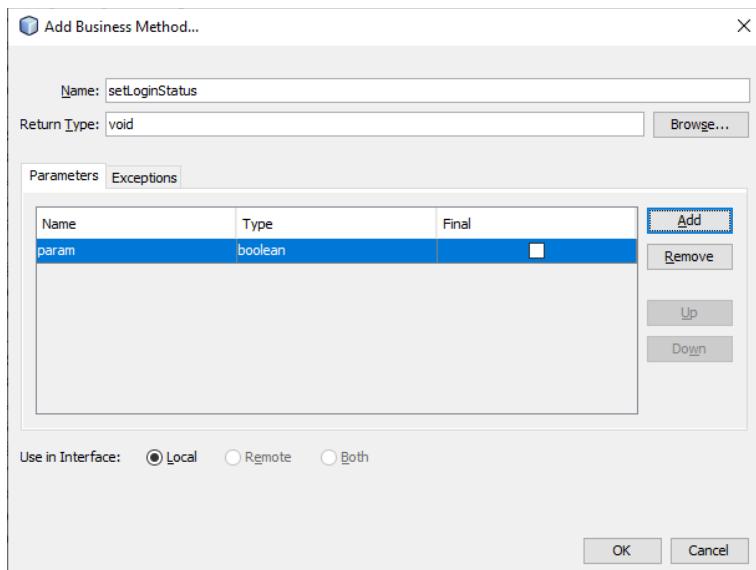


Gambar 3.13 Menambah fungsi login

10. Lakukan hal yang sama untuk *Method isLoginStatus dan setLoginStatus*



Gambar 3.14 Menambah fungsi isLoginStatus



Gambar 3.15 Menambah fungsi setLoginStatus

11. Tambahkan *source code* berikut didalam *class LoginSession*

Ini adalah *code* utama dari fungsi java.

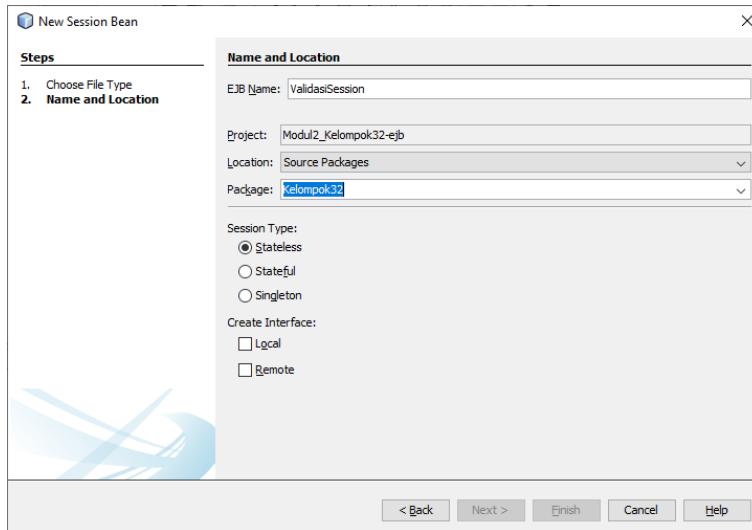
```
private final String [] Nama = {"Praktikum RSBK"};
```

```

private final String [] Nim = {"21120116000001"};
private boolean loginStatus = false;
@Override
public boolean Login(String Nama, String Nim) {
    return Nama.equals(this.Nama[0]) &&
Nim.equals(this.Nim[0]);
}
@Override
public boolean isLoginStatus() {
    return loginStatus;
}
@Override
public void setLoginStatus(boolean param) {
    this.loginStatus = param;
}

```

12. Ikuti langkah 5 untuk membuat *Session Bean* baru
13. Pada Isikan ValidasiSession pada **EJB Name**. Pada **Session Type** pilih *Stateless*. *Stateless* artinya tidak menyimpan data.



Gambar 3.16 Membuat *Session Bean* Validasi *Session*

14. Tambakan *source code* berikut didalam class **ValidasiSession**.

Ini adalah *source code* yang digunakan untuk memvalidasi *session login*.

```

public boolean nama(String param) {
    try{
        if(param.isEmpty()) {
            return false;
        }
        if(param.length()<=5) {
            return false;
        }
    }
}

```

```

        }
    }catch(NullPointerException e) {
        return false;
    }
    return true;
}

public boolean nim(String param) {
    try{
        Long.parseLong(param);
    }catch(NumberFormatException e) {
        return false;
    }
    if(param.isEmpty()) {
        return false;
    }
    if(param.length()<14) {
        return false;
    }
    return true;
}

```

15. Buka file index.html di directory **Modul2\_Kelompok32-war>Web Page** dan timpa *source code* yang sudah ada dengan *source code* berikut.

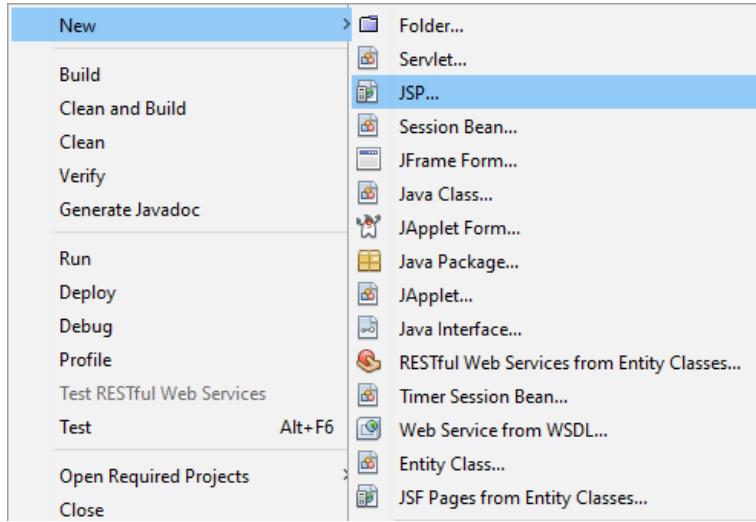
*File index.html* berisikan baris kode untuk tampilan halaman awal sebuah web service.

```

<!DOCTYPE html>
<html lang="en" dir="ltr">
<head>
    <meta charset="utf-8">
    <title>2020 Gelar ST</title>
</head>
<body>
    <h1 style="text-align:center"> 2020 GELAR ST</h1>
    <h3 style="text-align:center">Tahun depan kita wisuda</h3>
    <h3 style="text-align:center">Pelajari program berikut biar bisa mewujudkannya</h3>
    <h3 style="text-align:center;"><a href="Login" style="text-decoration:none"><< Klik Disini >></a></h3>
    <h4 style="text-align:center">&copy; Copyright Praktikum Rekayasa Perangkat Lunak 2019</h4>
</body>
</html>

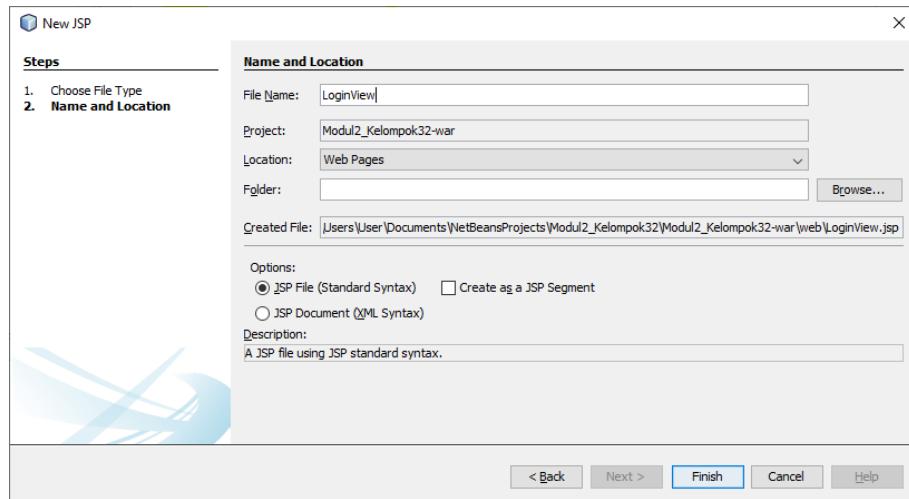
```

16. Buat file JSP dengan cara klik kanan pada *Project war* pilih **New>JSP**  
*File Javascript* berguna sebagai tampilan untuk halaman baru.



Gambar 3.17 Membuat JSP Baru

17. Beri nama LoginView lalu klik finish



Gambar 3.18 Membuat LoginView.jsp

18. Paste *source code* berikut ke dalam *file JSP* yang dibuat.

*Source code* ini berisi sebuah tampilan untuk login.

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html lang="en" dir="ltr">
    <head>
        <meta charset="utf-8">
        <title>Tahun Depan Wisuda</title>
```

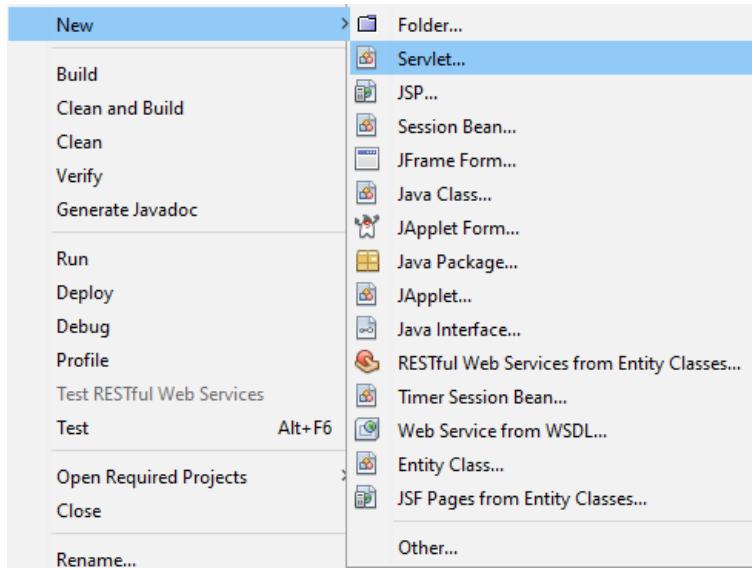
```

<meta name="viewport" content="width=device-width, initial-
scale=1.0">
</head>
<body>
    <header>
        <h1 style="text-align:center">SEMANGAT, TAHUN DEPAN
WISUDA</h1>
    </header>
    <main style="display:flex;justify-content:center;flex-
direction: column;">
        <% String stat = (String)request.getAttribute("status");
           if(stat != "true"){%>
            <form action="Login" Method="post"
style="display:flex;justify-content:center;flex-direction:
column;align-items: center;">
                <label for="nama" style="margin-bottom:5px">Nama
Mahasiswa</label>
                <input id="nama" type="text" name="nama"
placeholder="Nama Mahasiswa">
                <span style="color:red">${namaEr}</span>
                <br>
                <label for="nim" style="margin-bottom:5px">NIM
Mahasiswa</label>
                <input id="nim" type="text" name="nim"
placeholder="Nim Mahasiswa">
                <span style="color:red">${nimEr}</span>
                <span style="color:red">${userEr}</span>
                <br>
                <input type="submit" name="submit" value="Cari"
style="width:150px">
            </form>
        <%} else {%
            <h2 style="text-align: center">Hallo
${namaMahasiswa}, Semoga cepat lulus ya</h2>
            <form action="Login" Method="get"
style="display:flex;justify-content:center;flex-direction:
column;align-items: center;">
                <input type="submit" name="keluar"
value="Kembali" style="width:150px">
            </form>
        <%}%>
    </main><br>
    <footer style="text-align:center">&copy; Copyright Praktikum
Rekayasa Perangkat Lunak 2019</footer>
    </body>
</html>

```

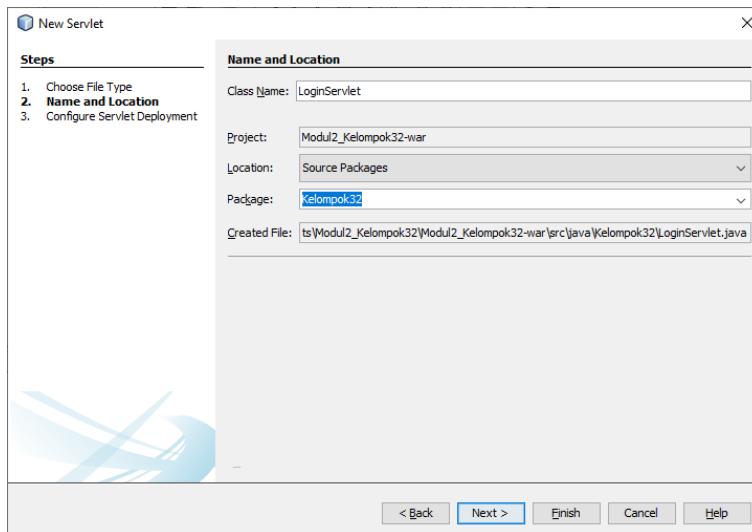
19. Buat *file* Sevlet dengan cara klik kanan pada *Project* web lalu pilih **New>Sevlet**.

*File* Servlet adalah sebuah penghubung antara *code* pada java ke web service.



Gambar 3.19 Membuat Servlet Baru

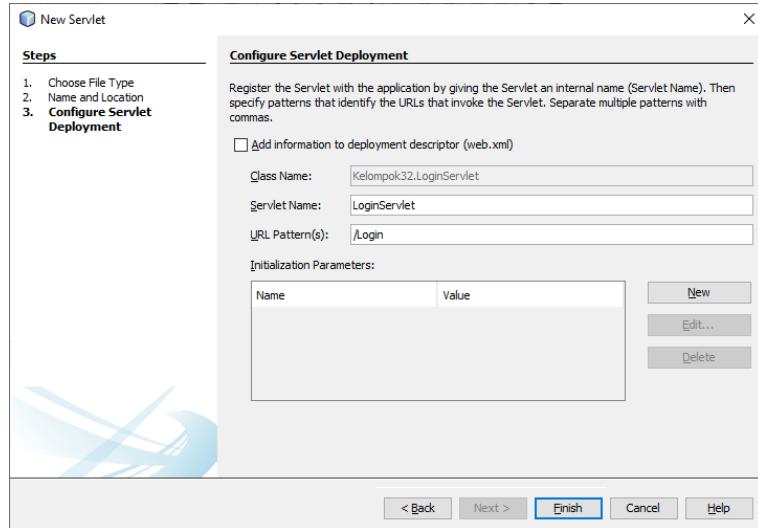
20. Beri nama LoginSevlet pada kolom **Class Name** dan Kelompok32 pada package.



Gambar 3.20 Membuat LoginServlet

21. Pastikan **Add information** di uncheck dan ganti URL Pattern(s) menjadi /Login

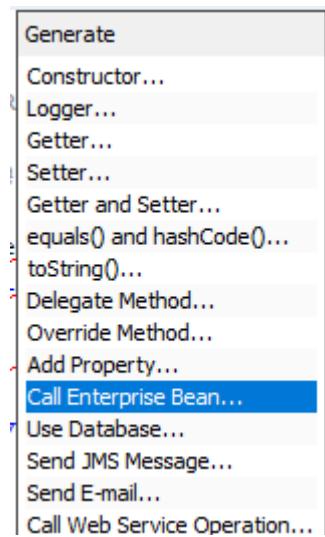
Dengan mengatur URL Pattern, kita dapat menentukan keluaran URL dari file tersebut.



Gambar 3.21 Memberi URL Pattern

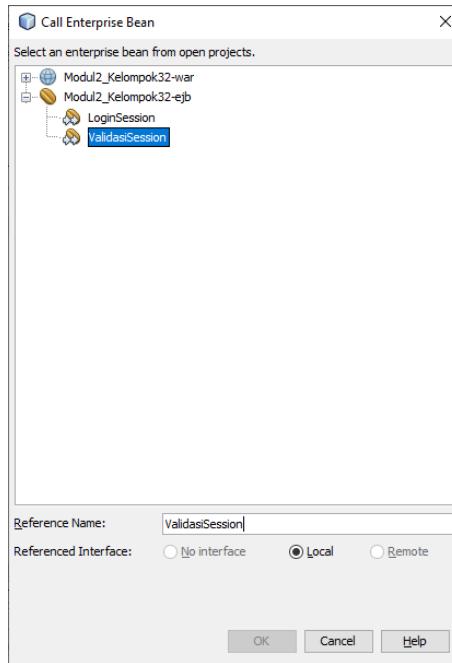
22. Import `LoginSession` dan `ValidasiSession` dengan cara klik kanan pada text editor dan pilih *Call Enterprise Bean*.

*Call Enterprise Bean* berfungsi untuk memanggil fungsi *Session Bean* yang telah dibuat.



Gambar 3.22 *Call Enterprise Bean*

23. Pilih *LoginSession* lalu klik OK dan ulang dari langkah 22 untuk impor *ValidasiSession*.



Gambar 3.23 Memanggil EJB ke servlet

24. Hapus *source code* pada *Method processRequest*, lalu tambahkan *source code* berikut.

*Source code* ini berfungsi untuk memuat file *LoginView.jsp*

```
validasiSession = new ValidasiSession();

request.setAttribute("status", "false");
RequestDispatcher rd =
getServletContext().getRequestDispatcher("/LoginView.jsp");
rd.forward(request, response);
```

25. Tambahkan *source code* berikut didalam *Method doGet*.

*Source code* ini adalah *source code* logout atau kembali dengan menghapus *session*

```
if("Keluar".equals(request.getParameter("keluar"))){
    loginSession.setLoginStatus(false);
    request.setAttribute("nimEr", "");
    request.setAttribute("namaEr", "");
}
```

26. Tambahkan *source code* berikut didalam *Method doPost*.

*Source code* ini adalah *source code* yang dieksekusi ketika menekan button Login.

```
String nama = request.getParameter("nama");
String nim = request.getParameter("nim");
boolean stNama, stNim = false;
stNama = validasiSession.nama(nama);
stNim = validasiSession.nim(nim);
if (stNim && stNama) {
    if (loginSession.Login(nama, nim)) {
        loginSession.setLoginStatus(true);
    } else {
        request.setAttribute("userEr", "Mahasiswa tidak
terdaftar");
    }
} else{
    if (!stNim) request.setAttribute("nimEr", "Inputan
Salah");
    if (!stNama) request.setAttribute("namaEr", "Inputan
Salah");
}

if (loginSession.isLoginStatus()) {
    request.setAttribute("status", "true");
    RequestDispatcher rd =
getServletContext().getRequestDispatcher("/LoginView.jsp");
    rd.forward(request, response);
} else {
    request.setAttribute("status", "false");
    RequestDispatcher rd =
getServletContext().getRequestDispatcher("/LoginView.jsp");
    rd.forward(request, response);
}
```

Jalankan *Project Enterprise* dengan cara klik kanan *Project Enterprise* lalu klik pilih run.

### 3.5 Hasil Percobaan

*Source code LoginSession.java*

```
private final String [] Nama = {"Rio Kisna Eka Putra"};
private final String [] Nim = {"21120116130060"};
private boolean loginStatus = false;
@Override
public boolean Login(String Nama, String Nim) {
    return Nama.equals(this.Nama[0]) &&
Nim.equals(this.Nim[0]);
}
@Override
public boolean isLoginStatus() {
    return loginStatus;
}
@Override
public void setLoginStatus(boolean param) {
    this.loginStatus = param;
}
```

*Source code* diatas adalah kelas utama dari program ini yang berisi sebuah model untuk variabel nama dan nim yang sudah terisi. Kemudian juga ada pernyataan `loginStatus` untuk `session` dan beberapa fungsi seperti fungsi `login` untuk pembanding Nama dan nim, fungsi `isLoginStatus()` untuk mengambil nilai `loginStatus`, dan `setLoginStatus()` untuk mengatur nilai `loginStatus`.

*ValidasiSession.java*

```
public boolean nama(String param) {
    try{
        if(param.isEmpty()) {
            return false;
        }
        if(param.length()<=5) {
            return false;
        }
    }catch(NullPointerException e) {
        return false;
    }
    return true;
}
```

```

public boolean nim(String param) {
    try{
        Long.parseLong(param);
    }catch(NumberFormatException e){
        return false;
    }
    if(param.isEmpty()){
        return false;
    }
    if(param.length()<14) {
        return false;
    }
    return true;
}

```

Ini adalah *Source code* yang digunakan untuk validasi saat login dimana mengambil nilai dari kotak teks untuk nama dan nim. Dilakukan pengecekan dari variabel yang dimasukan tersebut pada fungsi nama dan nim di kelas ini.

### Index.html

```

<!DOCTYPE html>
<html lang="en" dir="ltr">
<head>
    <meta charset="utf-8">
    <title>2020 Gelar ST</title>
</head>
<body>
    <h1 style="text-align:center"> 2020 GELAR ST</h1>
    <h3 style="text-align:center">Tahun depan kita wisuda</h3>
    <h3 style="text-align:center">Pelajari program berikut biar bisa mewujudkannya</h3>
    <h3 style="text-align:center;"><a href="Login" style="text-decoration:none"><< Klik Disini ></a></h3>
    <h4 style="text-align:center">&copy; Copyright Praktikum Rekayasa Perangkat Lunak 2019</h4>
</body>
</html>

```

Ini adalah *source code* yang berfungsi untuk menampilkan halaman utama web yang dibuat. Hanya berisi teks-teks yang diantaranya ada sebuah teks yang ketika di klik akan mengarahkan ke LoginView.jsp.

### LoginView.jsp

```

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html lang="en" dir="ltr">
    <head>
        <meta charset="utf-8">
        <title>Tahun Depan Wisuda</title>
        <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    </head>
    <body>
        <header>
            <h1 style="text-align:center">SEMANGAT, TAHUN DEPAN
WISUDA</h1>
        </header>
        <main style="display:flex;justify-content:center;flex-
direction: column;">
            <% String stat = (String)request.getAttribute("status");
if(stat != "true"){%
            <form action="Login" Method="post"
style="display:flex;justify-content:center;flex-direction:
column;align-items: center;">
                <label for="nama" style="margin-bottom:5px">Nama
Mahasiswa</label>
                <input id="nama" type="text" name="nama"
placeholder="Nama Mahasiswa">
                <span style="color:red">${namaEr}</span>
                <br>
                <label for="nim" style="margin-bottom:5px">NIM
Mahasiswa</label>
                <input id="nim" type="text" name="nim"
placeholder="Nim Mahasiswa">
                <span style="color:red">${nimEr}</span>
                <span style="color:red">${userEr}</span>
                <br>
                <input type="submit" name="submit" value="Cari"
style="width:150px">
            </form>
            <%} else {%
                <h2 style="text-align: center">Hallo
${namaMahasiswa}, Semoga cepat lulus ya</h2>
                <form action="Login" Method="get"
style="display:flex;justify-content:center;flex-direction:
column;align-items: center;">
                    <input type="submit" name="keluar"
value="Kembali" style="width:150px">
                </form>
            <%} %>
        </main><br>
        <footer style="text-align:center">&copy; Copyright Praktikum
Rekayasa Perangkat Lunak 2019</footer>
    </body>
</html>

```

LoginView.jsp berisikan sebuah tampilan yang digunakan untuk login dan menampilkan sebuah teks ketika berhasil login. Didalamnya terdapat dua kotak masukan, dengan sebuah button. kotak masukan tersebut akan di kirimkan ke servlet untuk diolah ketika button di tekan, dan kemudian akan mengeksekusi sesuai pengkondisian yang ada di servlet.

#### processRequest.LoginServlet.java

```
response.setContentType("text/html; charset=UTF-8");
validasiSession = new ValidasiSession();

request.setAttribute("status", "false");
RequestDispatcher rd =
getServletContext().getRequestDispatcher("/LoginView.jsp");
rd.forward(request, response);
```

Fungsi processRequest bersisi RequestDispatcher yang berfungsi untuk memuat halaman LoginView.jsp.

#### doGet.LoginServlet.java

```
if("Kembali".equals(request.getParameter("keluar"))){
    loginSession.setLoginStatus(false);
    request.setAttribute("nimEr", "");
    request.setAttribute("namaEr", "");

}
processRequest(request, response);
```

Fungsi doGet di eksekusi ketika tombol kembali di tekan. Fungsi ini akan mengubah nilai loginStatus pada loginSession menjadi false dan menghapus nilai nimEr dan namaEr.

#### doPost.LoginServlet.java

```
String nama = request.getParameter("nama");
String nim = request.getParameter("nim");
boolean stNama, stNim = false;
stNama = validasiSession.nama(nama);
stNim = validasiSession.nim(nim);
if (stNim && stNama) {
    if (loginSession.Login(nama, nim)) {
```

```
        loginSession.setLoginStatus(true);
    } else {
        request.setAttribute("userEr", "Mahasiswa tidak
terdaftar");
    }
}
else{
    if (!stNim) request.setAttribute("nimEr", "Inputan
Salah");
    if (!stNama) request.setAttribute("namaEr",
"Inputan Salah");
}

if (loginSession.isLoginStatus()) {
    request.setAttribute("status", "true");
    request.setAttribute("namaMahasiswa", nama);
    RequestDispatcher rd =
getServletContext().getRequestDispatcher("/LoginView.jsp");
    rd.forward(request, response);
} else {
    request.setAttribute("status", "false");
    RequestDispatcher rd =
getServletContext().getRequestDispatcher("/LoginView.jsp");
    rd.forward(request, response);
}

processRequest(request, response);
```

Fungsi doPost dijalankan ketika menekan tombol login, terdapat pengkondisian berdasarkan nilai yang di masukan pada kotak masukan nama dan nim. Kemudian nilai tersebut akan di bandingkan dengan nilai yang sudah di inisialisasi pada LoginSession.java. jika sama maka akan mengubah statusLogin menjadi true, sehingga fungsi yang di jalankan pada LoginView.jsp akan berubah ke halaman yang telah *login*. Lalu nilai namaMahasiswa akan di set sesuai dengan variabel nama pada file LoginSession.java



Gambar 3.24 index.html

Ketika program di jalankan akan di alihkan ke halaman ini dan dapat menekan sebuah teks “Klik Disini”. Selanjutnya akan dialihkan ke halaman baru.



Gambar 3.25 Halaman Login



Gambar 3.26 Memasukan inputan salah

Jika masukan salah, akan di tampilkan petunjuk seperti diatas.



Gambar 3.27 Memasukan Inputan Benar

Ketika memasukan nama dan nim yang benar, akan keluar tampilan seperti diatas, dimana variabel nama akan ditampilkan

### 3.6 Tugas dan Pembahasan

#### Mahasiswa.java

```

String nama, nim;

public Mahasiswa(String nama, String nim) {
    this.nama = nama;
    this.nim = nim;
}

public String getNama() {
    return nama;
}

public void setNama(String nama) {
    this.nama = nama;
}

public String getNim() {
    return nim;
}

public void setNim(String nim) {
    this.nim = nim;
}

```

*Source code* ini adalah sebuah model Mahasiswa yang di gunakan sebagai object yang akan di masukan kedalam array list. Terdapat 2 attribute, yaitu nama dan nim dan beberapa fungsi setter getter.

#### cariMahasiswa.java

```

ArrayList<Mahasiswa> datar = new ArrayList<Mahasiswa>();
public void cekpicek(String nama, String nim) {
    datar.add(new Mahasiswa(nama, nim));
}

public cariMahasiswa() {
    cekpicek("Rio Kisna Eka Putra", "21120116130060");
    cekpicek("Alfian Aulia Firdaus", "21120116130035");
    cekpicek("Favo Perdana HS", "21120116120015");
    cekpicek("Farrell Denando", "21120116120026");
    cekpicek("Faisal Rizki R", "21120116120014");
}

```

```

private String checkNama(String param){
    for (int i = 0; i < datar.size(); i++){
        if(param.equals(datar.get(i).nama)){
            return "Nama Praktikan : "
+datar.get(i).nama+"("+datar.get(i).nim+")";
        }
    }
    return null;
}

private String checkNIM(String param){
    for (int i = 0; i < datar.size(); i++) {
        if(param.equals(datar.get(i).nim)){
            return "Nama Praktikan : "
+datar.get(i).nama+"("+datar.get(i).nim+")";
        }
    }
    return null;
}

public String search(String param){
    if (checkNama(param) != null) {
        return checkNama(param);
    }
    else if (checkNIM(param) !=null) {
        return checkNIM(param);
    }
    else {
        return "Nope";
    }
}
;
```

*Source code* diatas adalah kelas yang berisi fungsi-fungsi yang akan di panggil di servlet serta berbagai variabel yang sudah ditentukan. Pertama-tama adalah pembuatan array list, kemudian membuat sebuah fungsi yang digunakan untuk menambahkan mahasiswa ke dalam array list. Selanjutnya membuat constructor yang di dalamnya sudah membuat 5 objek mahasiswa. Kemudian ada fungsi checkNama untuk mengecek nama yang nanti dicari, sedangkan fungsi checkNim untuk mengecek nim yang nanti dicari. Lalu fungsi terakhir adalah mengambil informasi yang di gabungkan dalam sebuah string dari nama dan nim, yang nantinya akan digunakan sebagai string yang di tampilkan dalam label. Hanya satu *Method* public yang di buat, yaitu *Method* search, hal ini di karenakan untuk melindungi penggunaan *Method* lain yang bersifat tidak

lengkap. Selain itu akan lebih efektif jika hanya satu *Method* public yang di panggil di servlet nantinya.

### Search.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html lang="en" dir="ltr">
    <head>
        <meta charset="utf-8">
        <title>Mesin Pencari Mahasiswa</title>
        <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    </head>
    <body style="background-color: #b3ffec">
        <header>
            <h1 style="text-align:center; background-color:
#33ffcc;">MESIN PENCARIAN MAHASISWA YANG SEGERA WISUDA</h1>
        </header>
        <main style="display:flex;justify-content:center;flex-
direction: column;">
            <form action="search" Method="post"
style="display:flex;justify-content:center;flex-direction:
column;align-items: center;">
                <h2 for="nama" style="margin-bottom:5px">-----
-----Cari Mahasiswa-----</h2>
                <input id="nama" type="text" name="param"
style="background-color: white;color: #3CBC8D;border: none;width:
90%;padding: 12px 20px; margin: 8px 0; box-sizing: border-
box;border-radius: 3px;" placeholder="Masukan Nama / NIM">
                <span style="color:red ;font-size:
20px">${show}</span>
                <br>
                <input type="submit" name="submit" value="Cari"
style=" background-color: #4CAF50; border: none; color: white;
padding: 16px 32px; text-decoration: none; margin: 4px 2px; cursor:
pointer; width: 90%">
            </form>
            <br>
            <form action="search" Method="get"
style="display:flex;justify-content:center;flex-direction:
column;align-items: center;">
                <h2 for="namal" style="margin-bottom:5px">-----
-----Tambah Mahasiswa-----</h2>
                <input id="namal" type="text" name="namal"
style="background-color: white;color: #3CBC8D;border: none;width:
90%;padding: 12px 20px; margin: 8px 0; box-sizing: border-
box;border-radius: 3px;" placeholder="Masukan Nama">
                <br>
```

```

<input id="nim1" type="text" name="nim1"
style="background-color: white; color: #3CBC8D; border: none; width:
90%; padding: 12px 20px; margin: 8px 0; box-sizing: border-
box; border-radius: 3px;" placeholder="Masukan NIM">
<br>
<input type="submit" name="submit" value="Tambah"
style=" background-color: #4CAF50; border: none; color: white;
padding: 16px 32px; text-decoration: none; margin: 4px 2px; cursor:
pointer; width: 90%">
</form>
</main><br>
<footer style="text-align:center; background-color:
#33ffcc;">&copy; Copyright Kelompok 32 Praktikum RSBK 2019</footer>
</body>
</html>
```

Search.jsp adalah *file* yang berisikan sebuah *code* javascript yang digunakan untuk menampilkan sebuah halaman pencarian dan penambahan data mahasiswa. Pada *source code* diatas terdapat 2 buah action, yaitu post search dan get search. Post search digunakan untuk mencari data mahasiswa yang sudah terdaftar. Terdapat sebuah kotak masukan, serta sebuah button untuk mencari. Kemudian pada action get search, berisi 2 buah kotak masukan dan sebuah button.

Penggunaan request get disini dilakukan agar action untuk menambah data dapat dilakukan pada servlet yang sama. Karena servlet tidak dapat menyimpan data ketika beralih ke servlet lainnya, sehingga untuk menambahkan data pada servlet terkait harus dilakukan pada halaman ini juga.

#### processRequest.cariServlet.java

```

response.setContentType("text/html; charset=UTF-8");

RequestDispatcher rd =
getServletContext().getRequestDispatcher("/search.jsp");
rd.forward(request, response);
```

Fungsi processRequest berisi RequestDispatcher yang digunakan untuk memuat *file* search.jsp saat program dijalankan.

### doGet.cariServlet.java

```
cariMahasiswa.cekpicek(request.getParameter("nama1"),
request.getParameter("nim1"));
processRequest(request, response);
```

Fungsi doGet di eksekusi ketika action get search pada search.jsp di jalankan. Isinya berupa pemanggilan fungsi cekpicek dari *class* cariMahasiswa dengan parameter nama1 dan nim1 untuk menambahkan objek baru pada sebuah arraylist.

### doPost.cariServlet.java

```
String param = request.getParameter("param");
String cek = cariMahasiswa.search(param);
if (cek=="Nope") {
    request.setAttribute("show", "Mahasiswa tidak
terdaftar");
}
else {
    request.setAttribute("show", cek);
}

processRequest(request, response);
```

Fungsi doPost digunakan untuk memanggil fungsi search pada *class* cariMahasiswa dengan parameter param, kemudian nilai di simpan pada variabel cek. Variabel cek kemudian di bandingkan, jika bernilai “Nope” (nilai default diubah ke bentuk “Nope”), maka data mahasiswa tidak ada, sehingga atribut show di tuliskan demikian, namun jika ternyata nilai cek itu lainnya, maka segera di tuliskan nilai cek tersebut ke atribut “show”.



## Dashboard Data Mahasiswa

Mahasiswa Segera Wisuda

<< Klik Disini >>

© Copyright Kelompok 32 Praktikum RSBK 2019

Gambar 3.28 index.html tugas

Gambar 3.29 Halaman pencarian



Gambar 3.30 Mencari mahasiswa yang terdaftar



Gambar 3.31 Mencoba mencari mahasiswa yang belum terdaftar



Gambar 3.32 Mahasiswa tidak ditemukan



Gambar 3.33 Menambahkan mahasiswa baru



Gambar 3.34 Mahasiswa baru telah di tambahkan

Link Github : <https://github.com/riokisna/Praktikum-RSBK-Kelompok32>

### 3.7 Kesimpulan

1. *Stateful* komponen berguna untuk membuat *session* yang dapat menyimpan informasi secara sementara.
2. *Call Enterprise* Bean memudahkan pemrogram untuk memasukan sebuah fungsi *Session Bean* ke dalam Servlet.
3. Servlet tidak dapat menyimpan data ketika ditutup atau berpindah ke servlet lainnya.
4. RequestDispatcher berfungsi untuk memuat sebuah *file JSP*, sehingga bahasa permograman untuk tampilan web bisa di lakukan di *file* tersebut.
5. Local Interface digunakan digunakan untuk menghubungkan fungsi dan *Method* yang ada di *session bean* ke sebuah Servlet

## **BAB IV**

### **Aplikasi CRUD menggunakan EJB, JPA, dan MySQL**

#### **4.1 Tujuan**

1. Praktikan dapat menggunakan operasi CRUD dengan *database* MySQL
2. Praktikan dapat mengimplementasikan EJB
3. Praktikan dapat mengimplementasikan JPA
4. Praktikan dapat menggunakan GlassFish *server* dalam pengembangan web menggunakan Bahasa pemrograman Java.

## 4.2 Dasar Teori

### 4.2.1 Java Persistence API

Java Persistence API merupakan *tool* untuk mengolah ataupun pengatur data relational dalam platform Java Standard Edition dan Java Enterprise Edition. JPA sendiri merupakan alat dalam pembuatan aplikasi berbentuk framework dalam pemrograman java dengan pendekatan *Object Relational Mapping* (ORM). ORM sendiri merupakan sebuah konsep yang berdiri sendiri, tidak terkait dengan Java. Namun hubungan ORM dengan JPA sangat dekat karena JPA merupakan standart ORM dalam Java, dan harus diikuti oleh pengguna ORM di Java agar ada standart yang sama antara ORM di Java dengan yang diluar Java. Dengan menggunakan JPA, memungkinkan manipulasi data tanpa menggunakan query, namun bukan berarti tanpa menggunakan query sama sekali, tetapi ada penggunaan query disana. Cara JPA ini dinilai lebih baik dari teknik manipulasi data dengan jdbc. Jika kita menggunakan JPA, maka cara kita terhubung ke *database* sama semua, baik pakai MySQL, SQL Server ataupun PostgreSQL. API JPA terdapat dalam *package* javax.persistence. Di dalamnya mengandung Query khusus yang disebut (JPQL)Java Persistence Query Language. Beberapa Library yang mengimplementasikan JPA antara lain adalah Hibernate dan EclipseLink. Kelebihan JPA yang cukup bermanfaat adalah tidak perlu membuat query untuk manipulasi data. Selain itu kita dapat dengan mudah mengelola transaksi dengan API. Kita bisa menghindari pembuatan *Data Access Object* yang rumit dan kompleks sekali jika menggunakan JPA ini. Dan yang cukup bagus, kita juga dapat mengelola Plain Old Java *Object* disini.

(Sumber : <https://dartoblog.wordpress.com/2012/08/01/pengenalan-jpa-java-persistence-api/>)

### 3.2.2 Session Bean

Session bean adalah EJB yang digunakan untuk mengeksekusi proses. Isi dari Session Bean ini biasanya berupa kata kerja (*transfer, pay, calculate, updateData, dll*).

Stateless Session Bean (SLSB) adalah Session Bean yang tidak menyimpan state (keadaan) pada setiap kali eksekusi. Berbeda dengan Statefull Session Bean (SFSB) yang dapat menyimpan state. State ini dapat kita gunakan misalnya untuk menyimpan informasi *user* atau barang-barang yang sudah dibeli (pada kasus online shop).

(Sumber : <https://suhearie.wordpress.com/2008/08/26/java-enterprise-mulai-dari-mana-part-2-netbeans-glassfish/>)

#### 4.2.3 Entity Unit

Entity Bean adalah EJB yang digunakan untuk mempermudah manipulasi *database*. Konsepnya adalah *Object – Relational Mapping* (ORM) yang berarti memetakan *object* dengan data di dalam *database*. Entity Bean sebenarnya adalah spesifikasi “bawaan” dari versi EJB sebelumnya yaitu EJB 2.1. Dalam EJB3, ada API lain yang lebih sederhana yaitu Java Persistence API (JPA). Sekarang orang lebih banyak menggunakan JPA dibandingkan Entity Bean.

(Sumber : <https://suhearie.wordpress.com/2008/08/26/java-enterprise-mulai-dari-mana-part-2-netbeans-glassfish/>)

#### 4.2.4 Java Servlet

Servlet adalah bahasa pemrograman Java kelas digunakan untuk memperluas kemampuan dari *server* yang tuan aplikasi mengakses melalui model pemrograman *request-respon*. Meskipun servlet dapat menanggapi setiap jenis permintaan, mereka biasanya digunakan untuk memperpanjang aplikasi host oleh *server* Web. Dengan demikian, dapat dianggap sebagai Java Applet yang berjalan pada *server* bukan browser.

Servlet berbasis Java *server-side* teknologi web. Sesuai namanya, melayani permintaan klien dan menerima respon dari *server*. Secara teknis, Servlet adalah kelas Java dalam Java EE yang sesuai dengan Java Servlet API, sebuah protokol di mana kelas Java mungkin merespon permintaan. Mereka tidak terikat dengan sebuah protokol client-*server* khusus, tetapi yang paling sering digunakan dengan protokol HTTP. Oleh karena itu, kata “Servlet” sering digunakan dalam arti “Servlet HTTP”.

[2] Dengan demikian, seorang pengembang perangkat lunak dapat menggunakan servlet untuk menambahkan konten dinamis ke *server* Web menggunakan platform Java. Isi dihasilkan umumnya HTML, tetapi mungkin data lain seperti XML. Servlets adalah mitra Jawa non-Jawa teknologi konten web yang dinamis seperti PHP dan ASP.NET. Servlets dapat mempertahankan negara dalam variabel sesi transaksi di *server* banyak dengan menggunakan cookie HTTP, atau menulis ulang URL. Untuk menyebarkan dan menjalankan Servlet, wadah Web harus digunakan. Sebuah wadah Web (juga dikenal sebagai wadah Servlet) pada dasarnya adalah komponen Web *server* yang berinteraksi dengan servlet. Wadah Web bertanggung jawab untuk mengelola siklus hidup servlets, pemetaan URL ke servlet tertentu dan memastikan bahwa URL pemohon memiliki hak akses yang benar. API servlet, yang terkandung dalam hirarki javax.servlet paket Java, mendefinisikan interaksi yang diharapkan dari wadah Web dan servlet [2]. Servlet adalah sebuah objek yang menerima permintaan dan menghasilkan respon berdasarkan permintaan itu. Paket servlet mendefinisikan dasar Jawa objek untuk mewakili servlet permintaan dan tanggapan, serta sebagai objek untuk mencerminkan konfigurasi servlet parameter dan lingkungan eksekusi. Paket javax.servlet.http mendefinisikan HTTP spesifik subclass dari elemen servlet generik, termasuk objek manajemen sesi yang melacak beberapa permintaan dan tanggapan antara Web *server* dan klien. Servlets dapat dikemas dalam file WAR sebagai aplikasi Web. Servlets dapat dihasilkan secara otomatis dari JavaServer Pages (JSP) oleh kompilator JavaServer Pages. Perbedaan antara Servlets dan JSP adalah bahwa Servlets biasanya menanamkan HTML di dalam kode Java, sedangkan JSP embed kode Java dalam HTML. Sedangkan penggunaan langsung dari Servlets untuk menghasilkan HTML (seperti yang ditunjukkan pada contoh di bawah) telah menjadi langka, tingkat yang lebih tinggi MVC kerangka web di Java EE (JSF) masih secara eksplisit menggunakan teknologi Servlet untuk penanganan permintaan / tanggapan tingkat rendah melalui FacesServlet yang . Sebuah penggunaan agak lebih tua adalah dengan menggunakan

servlet bersama dengan JSP dalam pola yang disebut “Model 2”, yang merupakan rasa pola *model-view-controlletr*.

(Sumber : <https://aeroyid.wordpress.com/2012/07/16/javamemperkenalkan-javaservlet-dan-membuat-helloworld/>)

#### 4.2.5 MySQL

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL (DBMS) yang multithread, dan multi-user. MySQL adalah implementasi dari sistem manajemen basisdata relasional (RDBMS). MySQL dibuat oleh TcX dan telah dipercaya mengelola sistem dengan 40 buah *database* berisi 10.000 tabel dan 500 di antaranya memiliki 7 juta baris. MySQL AB merupakan perusahaan komersial Swedia yang mensponsori dan yang memiliki MySQL. Pendiri MySQL AB adalah dua orang Swedia yang bernama David Axmark, Allan Larsson dan satu orang Finlandia bernama Michael “Monty”. Setiap pengguna MySQL dapat menggunakan secara bebas yang didistribusikan gratis dibawah lisensi GPL(General Public License) namun tidak boleh menjadikan produk turunan yang bersifat komersial. Pada saat ini MySQL merupakan *database server* yang sangat terkenal di dunia, semua itu tak lain karena bahasa dasar yang digunakan untuk mengakses *database* yaitu SQL. SQL (Structured Query Language) pertama kali diterapkan pada sebuah proyek riset pada laboratorium riset San Jose, IBM yang bernama system R. Kemudian SQL juga dikembangkan oleh Oracle, Informix dan Sybase. Dengan menggunakan SQL, proses pengaksesan *database* lebih *user-friendly* dibandingkan dengan yang lain, misalnya dBase atau Clipper karena mereka masih menggunakan perintah-perintah pemrograman murni. SQL dapat digunakan secara berdiri sendiri maupun di lekatkan pada bahasa pemograman seperti C, dan Delphi.

(Sumber : <https://upyes.wordpress.com/2013/02/06/pengertian-dan-sejarah-mysql/>)

### 4.3 Langkah Kerja

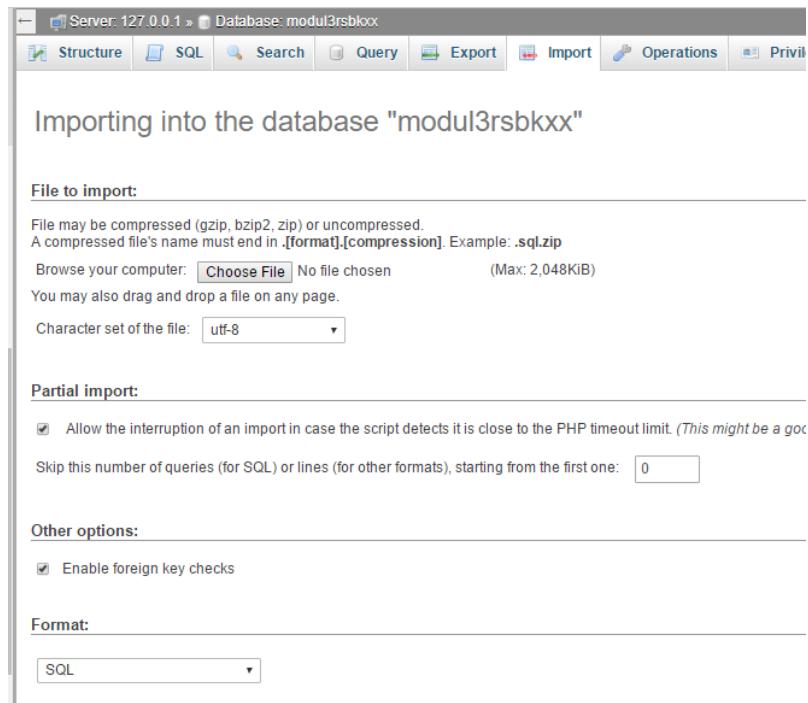
1. Jalankan XAMPP, jika ada bentrok dengan salah satu *PORT*, matikan *PORT* tersebut lalu nyalakan kembali (biasanya bentrok sama vmware/oracle) XAMPP yang di gunakan adalah MySQL dan Apache untuk konfigurasi *database*.
2. Buat *database* di MySQL. Beri nama “modul3rsbk32” ganti xx dengan kelompok.  
Ini adalah *database* yang akan digunakan untuk aplikasi nanti.

#### Databases



Gambar 4.1 Membuat *database* baru

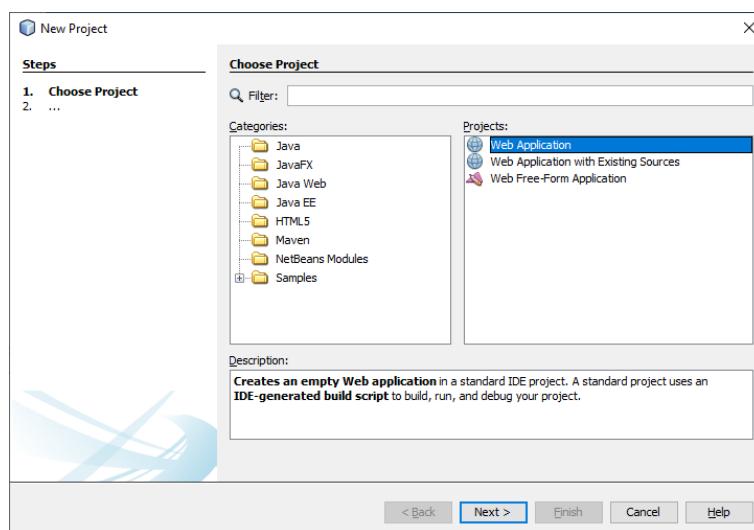
3. import file sql ke Mysql anda, pilih file .sql nya dan pilih go\file sql berisi query yang akan di eksekusi ketika mengimportnya ke dalam *database*.



Gambar 4.2 Mengimport file sql

4. Buka Netbeans lalu buat project baru pada Netbeans, pilih Java Web → Web Application

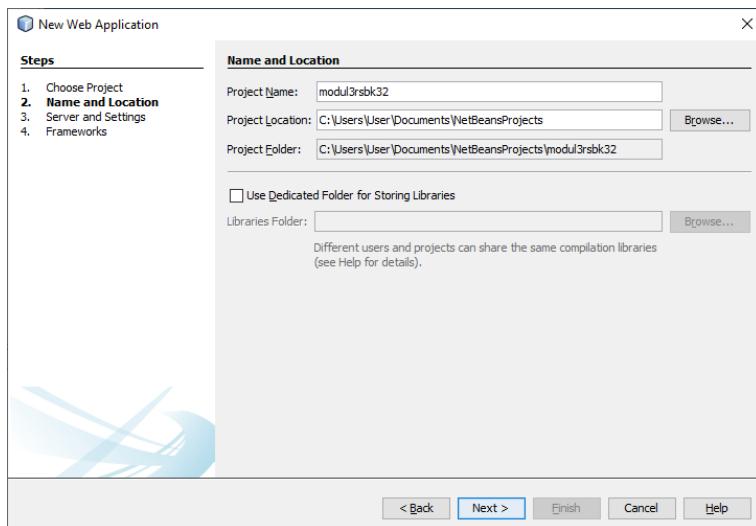
Java web application adalah sebuah project yang dapat digunakan untuk mengembangkan web berbasis Bahasa pemrograman Java.



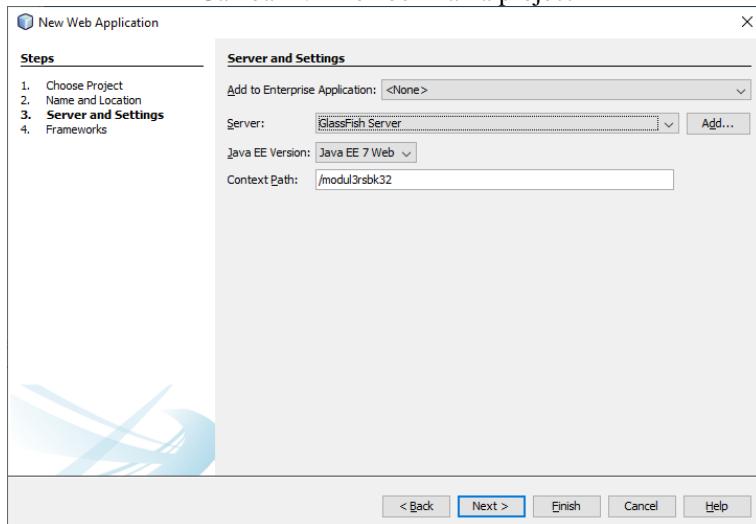
Gambar 4.3 Membuat project web application

5. Beri nama project “**modul3rsbk32**” dan pilih Glassfish Server (xx nomor kelompok).

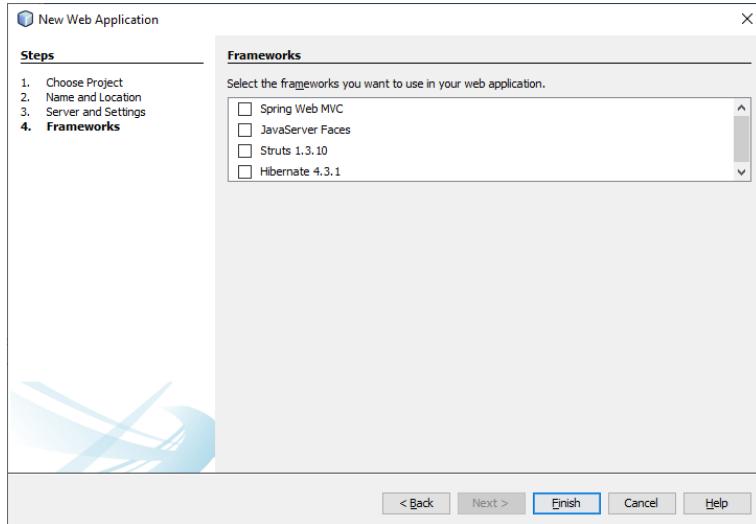
Glassfish server adalah sebuah web server yang digunakan untuk mengembangkan aplikasi berbasis web menggunakan Bahasa pemrograman Java.



Gambar 4.4 Memberi nama project

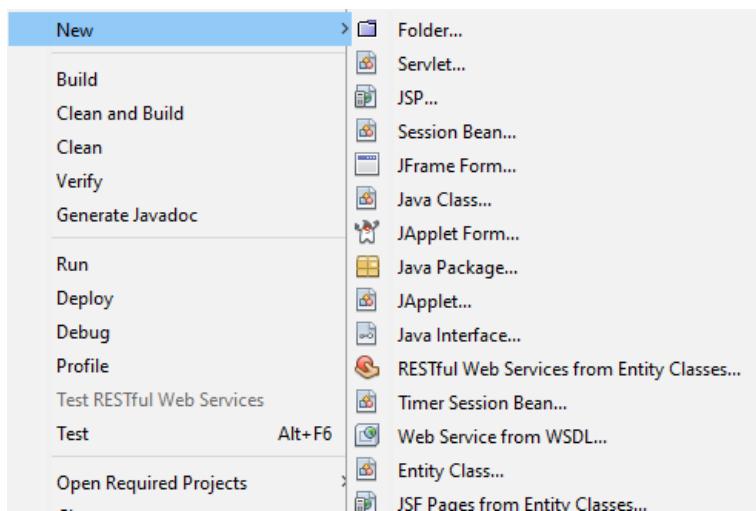


Gambar 4.5 Memilih GlassFish Server

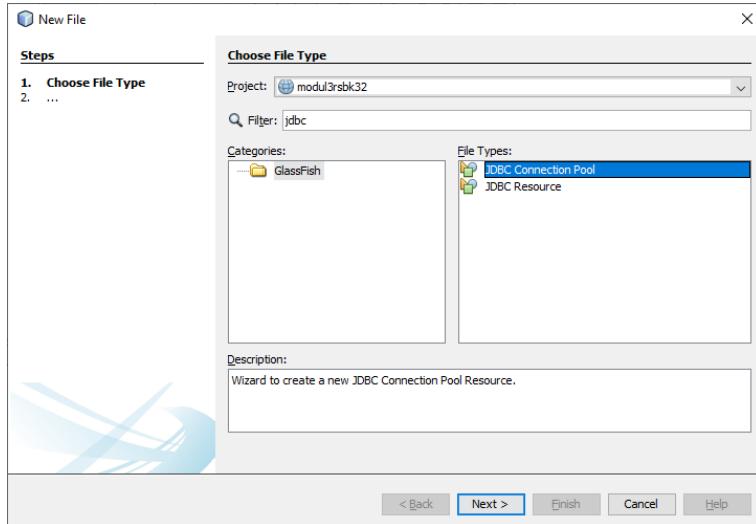


Gambar 4.6 *Finish* pembuatan project

6. Buat JDBC Connection Pool, klik kanan pada project, *new file*, other. Lalu masuk ke kategori GlassFish, pilih tipe *file* JDBC Connection Pool.
- JDBC Connection Pool adalah koneksi untuk menghubungkan Netbeans dengan *Database*.

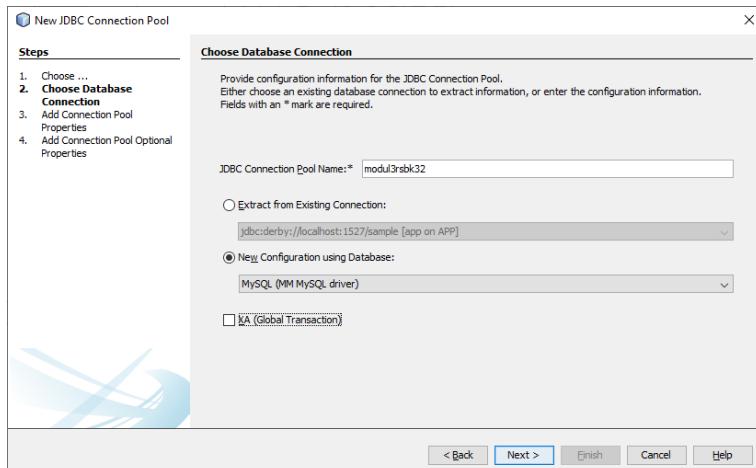


Gambar 4.7 Memilih other untuk menambahkan *file* connection pool



Gambar 4.8 Membuat connection pool baru

7. Beri nama connection pool (sama seperti nama projectnya, misal modul3rsbk32) dan pilih ‘*New Configuration using Database*’ → ‘MySQL (MM MySQL driver)



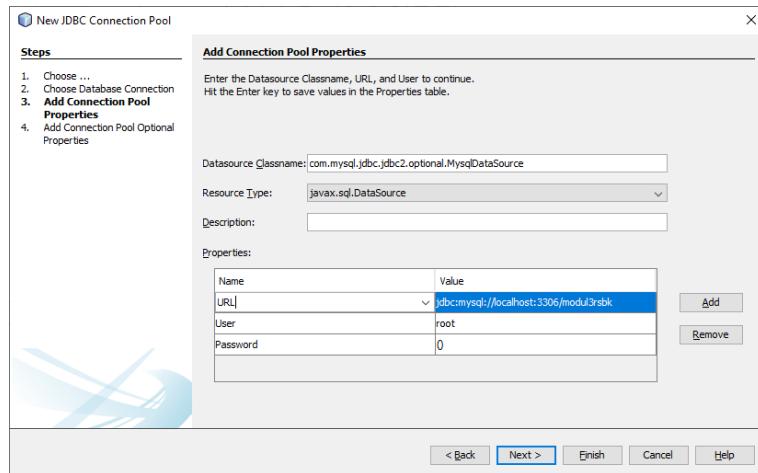
Gambar 4.9 Memilih koneksi untuk *database MySQL*

8. Masukkan URL, *User*, dan Password MySQL yang sudah dibuat. URL ke *localhost* dengan *port default MySQL 3306*, lalu nama *database* yang akan kita

gunakan (misal **modul3rsbkxx**). Untuk *user* pakai **root** dan passwordnya () . Kemudian pilih *Finish*.

Ini adalah konfigurasi untuk mengakses *database MySQL*.

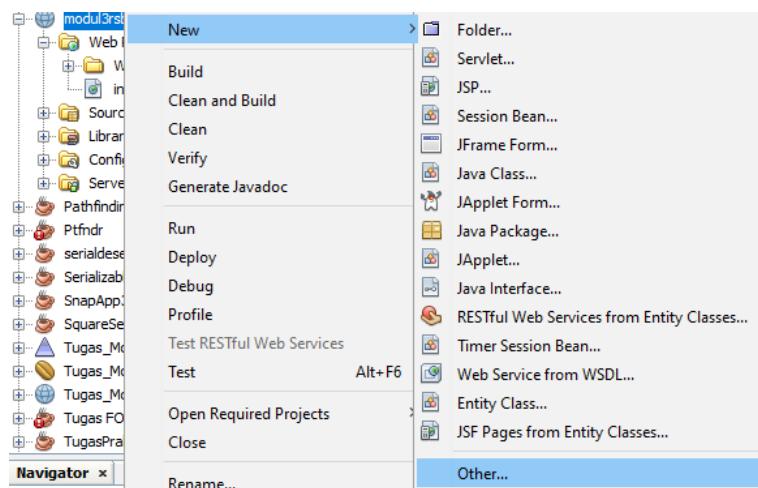
`jdbc:mysql://localhost:3306/ modul3rsbkxx`



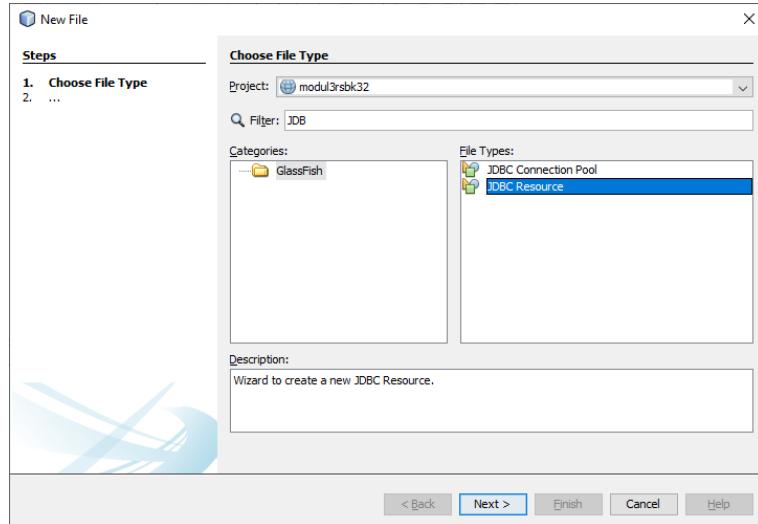
Gambar 4.10 Mengkonfigurasi koneksi

- Buat JDBC Resource. Dengan klik kanan pada Project, *New File*, Other. Lalu pada kategori pilih GlassFish, kemudian pilih JDBC Resource.

JDBC Resource bisa dianggap sebagai alat untuk memilih *database* yang di maksud dari koneksi yang sudah dibuat.

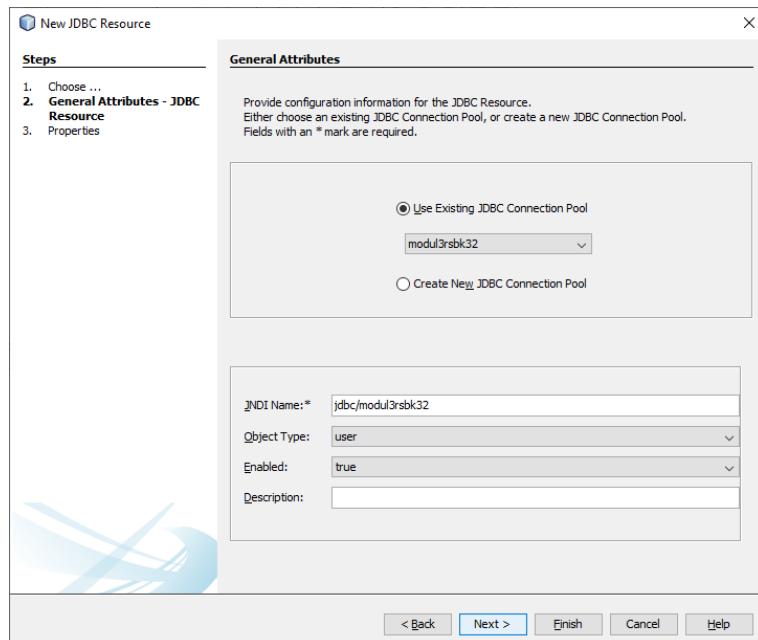


Gambar 4.11 Memilih other untuk membuat file JDBC Resource



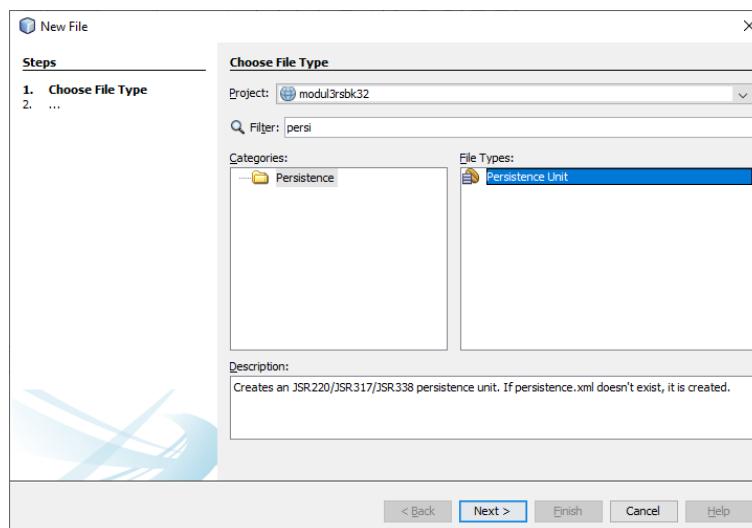
Gambar 4.12 Membuat JDBC Resource baru

10. Pilih ‘Use Existing JDBC Connection Pool’, pilih List JDBC yang baru kita buat tadi. Kemudian isikan JNDI Name seperti yang tadi. Misal modul3rsbkxx. Lalu pilih *Finish*.  
Memilik koneksi yang sudah dibuat. Disini yang dipilih adalah koneksi untuk MySQL yang sudah dibuat di langkah sebelumnya.

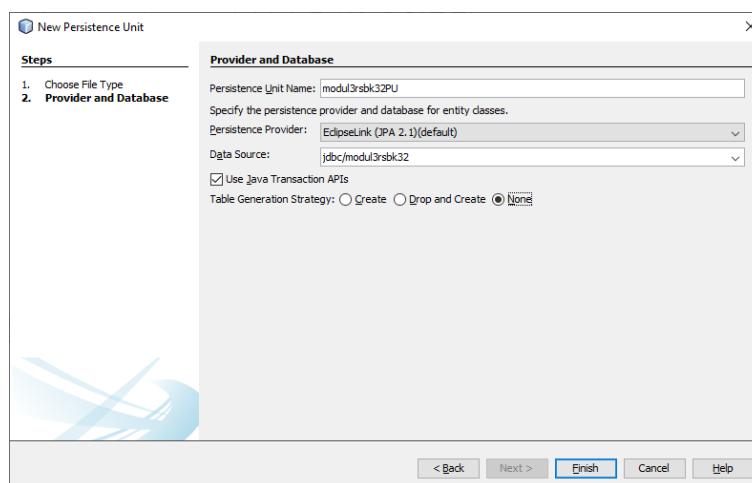


Gambar 4.13 Memilih Connecction pool yang sudah dibuat

11. Buat Persistence Unit dengan klik kanan pada project, *New File*, pilih Persistence Unit (Jika tidak ketemu, pilih other, lalu pada kolom filter ketik persistence unit). Nama Persistence Unit akan otomatis sesuai dengan Project, lalu pilih data resource yang tadi sudah dibuat. Table Generation Strategy pilih ‘None’ Setelah membuat koneksi, presistance unit digunakan untuk memilih *database* yang akan digunakan dan menghubungkannya dengan Glassfish *server*.

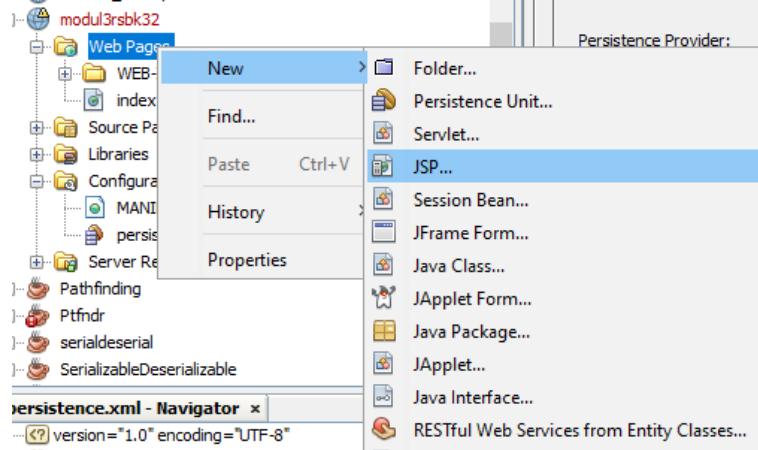


Gambar 4.14 Membuat Persistance Unit baru

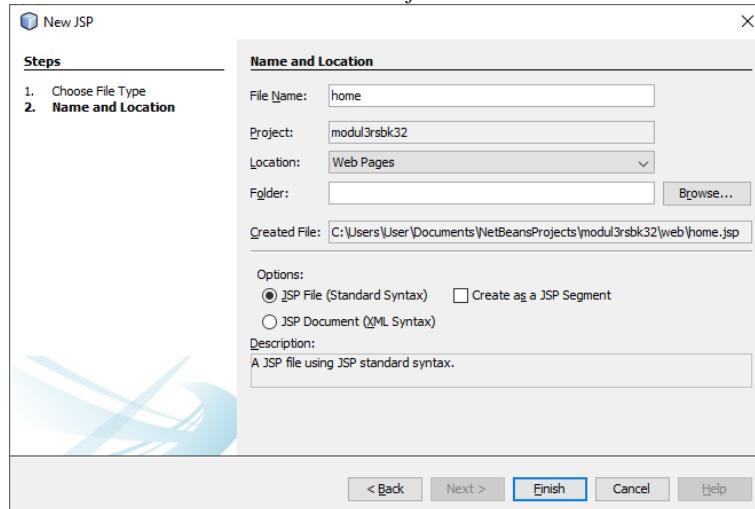


Gambar 4.15 Memilih JDBC resource yang sudah dibuat

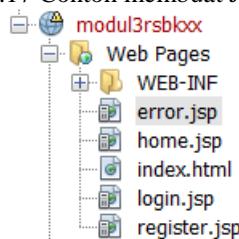
12. Buat 3 JSP *page* pada folder “Web Pages” dengan klik kanan folder lalu *new file*. JSP. Beri nama: *home.jsp*, *register.jsp*, *login.jsp*, dan *error.jsp*. Masukkan source code yang tersedia. JSP bisa di bilang sebagai *file* tampilan yang menggunakan HTML sebagai bahasanya. Seperti pada pemrograman web.



Gambar 4.16 Membuat *file* JSP baru



Gambar 4.17 Contoh membuat JSP *home*



Gambar 4.18 File JSP yang telah dibuat  
**register.jsp**

```
<%--  
    Document      : register  
    Created on   : Sep 22, 2019, 2:48:05 PM  
    Author        : WIN 10  
--%>  
  
<%@page contentType="text/html" pageEncoding="UTF-8"%>  
<!DOCTYPE html>  
<html>  
<head>  
    <meta charset="utf-8">  
    <meta http-equiv="X-UA-Compatible" content="IE=edge">  
    <meta name="viewport" content="width=device-width, initial-  
scale=1">  
  
    <title>Register Data</title>  
    <link rel="stylesheet" href="css/bootstrap.min.css">  
    <script src="js/bootstrap.min.js"></script>  
    <style>  
        .menu {  
            margin-left: -15px;  
            margin-right: 15px;  
        }  
        .daftar{  
            border: 2px solid #e5e5e5;  
            border-radius: 10px;  
            padding: 20px;  
        }  
        .daftar a{  
            margin-top: 2%;  
        }  
        .detail{  
            padding: 10px 0px;  
        }  
        .nav{  
            padding: 0px;  
            border: 1px solid #e5e5e5;  
            border-radius: 5px;  
        }  
        .nav li{  
            border-bottom: 1px solid #e5e5e5;  
            border-radius: 5px;  
        }  
  
    </style>  
</head>  
  
<body>  
    <div class="container">
```

```

<div class="jumbotron row">
    <a href=". /login.jsp" class="btn btn-md btn-success"
style="float:right" />Login</a><br>
    <center><h2><b>Data Mahasiswa</b></h2>
    <h4>Modul RSBK – Kelompok32</h4></center>
</div>
<div class="row content">
    <div class="col-md-12">
        <div class="col-md-4 col-md-offset-4 daftar">
            <p class="form-title">Sign Up</p>
            <form method="POST" action=". /RegistersServlet">
                <div class="form-group">
                    <label>Username</label>
                    <input type="text" class="form-control"
placeholder="Username" name="userName" type="text" autofocus />
                </div>
                <div class="form-group">
                    <label>Password</label>
                    <input type="password" class="form-control"
placeholder="Password" name="password" value="" required />
                </div>
                <input type="submit" name="register"
value="Register" class="btn btn-success" />
            </form>
        </div>
    </div>
</div>
</body>
</html>

```

### *login.jsp*

```

<%-->
    Document : login
    Created on : Sep 22, 2019, 12:45:49 PM
    Author : WIN 10
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,
initial-scale=1">

    <title>Login Data</title>
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <script src="js/bootstrap.min.js"></script>

```

```

<style>
    .menu {
        margin-left: -15px;
        margin-right: 15px;
    }
    .daftar{
        border: 2px solid #e5e5e5;
        border-radius: 10px;
        padding: 20px;
    }
    .daftar a{
        margin-top: 2%;
    }
    .detail{
        padding: 10px 0px;
    }
    .nav{
        padding: 0px;
        border: 1px solid #e5e5e5;
        border-radius: 5px;
    }
    .nav li{
        border-bottom: 1px solid #e5e5e5;
        border-radius: 5px;
    }
</style>
</head>

<body>
    <div class="container">
        <div class="jumbotron row">
            <a href=".register.jsp" class="btn btn-md btn-success" style="float:right" />Register</a><br>
            <center><h2><b>Data Mahasiswa</b></h2>
            <h4>Modul RSBK – Kelompok32</h4></center>
        </div>
        <div class="row content">
            <div class="col-md-12">
                <div class="col-md-4 col-md-offset-4 daftar">
                    <p class="form-title">Sign In</p>
                    <form method="POST" action=".LoginServlet">
                        <div class="form-group">
                            <label>Username</label>
                            <input type="text" class="form-control" placeholder="Username" name="userName" type="text" autofocus />
                        </div>
                        <div class="form-group">
                            <label>Password</label>
                        </div>
                </div>
            </div>
        </div>
    </div>
</body>

```

```

        <input type="password" class="form-control" placeholder="Password" name="password" value="" required />
    </div>
    <input type="submit" name="login" value="Login" class="btn btn-md btn-success" />
</form>
</div>
</div>
</div>
</div>
</body>
</html>
```

### **home.jsp**

```

<%--
 Document      : home
 Created on   : Sep 22, 2019, 12:45:58 PM
 Author       : WIN 10
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib prefix="s" uri="http://java.sun.com/jsp/jstl/core" %>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Home Page</title>
<link rel="stylesheet" href="css/bootstrap.min.css">
<script src="js/bootstrap.min.js"></script>
<style>
    .menu {
        margin-left: -15px;
        margin-right: 15px;
    }
    .daftar{
        border: 2px solid #e5e5e5;
        border-radius: 5px;
        padding: 5px;
    }
    .table th, .table td{
        text-align: center;
    }
    .nav{
        padding: 5px;
        border: 2px solid #e5e5e5;
        border-radius: 5px;
    }
</style>
```

```

.nav li{
    border-bottom: 2px solid #e5e5e5;
    border-radius: 5px;
}
.daftar h3{
    margin-top: 50px;
    margin-bottom: 25px;
}
</style>
</head>

<div class="container">
    <div class="jumbotron row">
        <center><h2><b>Data Mahasiswa</b></h2>
        <p>Modul RSBK - Kelompokxx</p>
        <h5>Selamat Datang,<br/>
<%=session.getAttribute("loginName")%></h6></center>
    </div>
    <div class="row content col-md-8 col-md-offset-2">
        <div class="col-md-3 menu">
            <ul class="nav nav-pills nav-stacked" style="">
                <li><a href="#">Home</a></li>
                <li><a href=".//login.jsp">Logout</a></li>
            </ul>
        </div>
        <div class="col-md-9 daftar">
            <form action=".//StudentServlet" method="POST">
                <table class="table table-bordered">
                    <tr>
                        <td>Student ID</td>
                        <td><input class="form-control" type="text" name="studentId" value="${student.studentId}" /></td>
                    </tr>
                    <tr>
                        <td>First Name</td>
                        <td><input class="form-control" type="text" name="firstname" value="${student.firstName}" /></td>
                    </tr>
                    <tr>
                        <td>Last Name</td>
                        <td><input class="form-control" type="text" name="lastname" value="${student.lastName}" /></td>
                    </tr>
                    <tr>
                        <td colspan="2">
                            <input type="submit" class="btn btn-primary btn-sm" name="action" value="Add" />
                            <input type="submit" class="btn btn-default btn-sm" name="action" value="Edit" />
                            <input type="submit" class="btn btn-danger btn-sm" name="action" value="Delete" />
                        </td>
                    </tr>
                </table>
            </form>
        </div>
    </div>
</div>

```

```

        <input type="submit" class="btn btn-warning
btn-sm" name="action" value="Search" />
    </td>
    </tr>
    </table>
</form>
<h3 align="center">Informasi Data</h3>
<table class="table table-bordered table-hover">
    <thead>
        <tr>
            <th>No. ID</th>
            <th>First Name</th>
            <th>Last Name</th>
        </tr>
    </thead>
    <tbody>
        <s:forEach           items="${allStudents}">
var="stud">
        <tr>
            <td>${stud.studentId}</td>
            <td>${stud.firstName}</td>
            <td>${stud.lastName}</td>
        </tr>
        </s:forEach>
    </tbody>
</table>
</div>
</div>
</div>
</html>

```

### ***error.jsp***

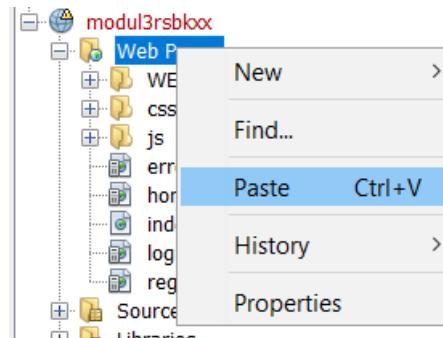
```

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
<title>Error Page</title>
</head>
<body>
<h1>Error - <%=request.getAttribute("error")%> </h1>
</body>
</html>

```

13. Masukkan asset dengan copy dan paste folder css dan js ke web *pages*.

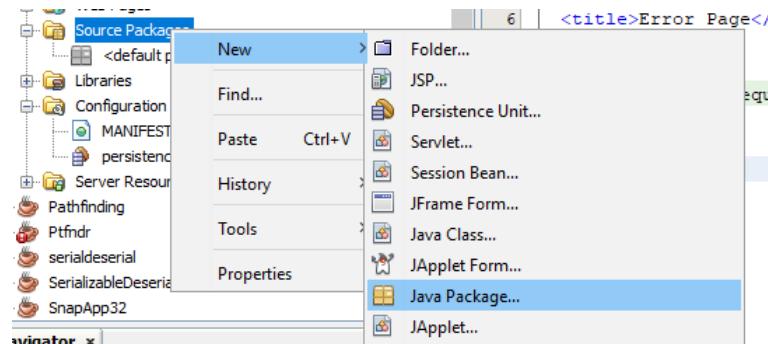
Asset ini berisi css untuk tampilan web.



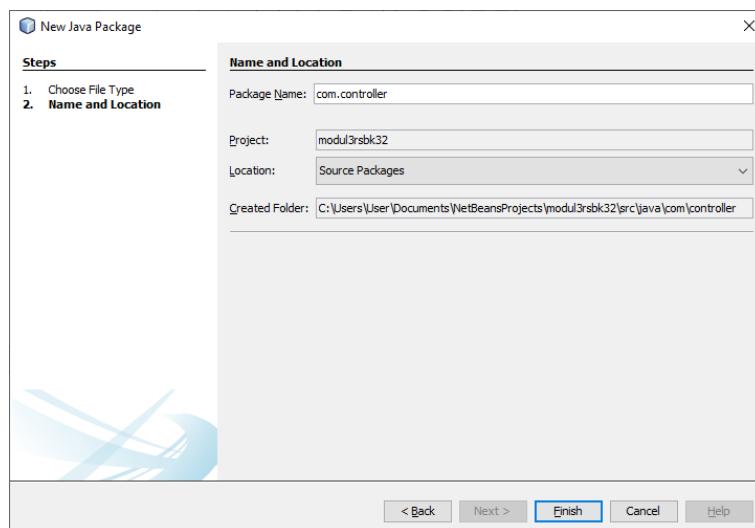
Gambar 4.19 Menambahkan Asset ke project

14. Buat 3 java package pada folder “Source Packages”, beri nama: “com.controller”, “com.dao”, dan “com.model”.

Ini adalah *package* yang digunakan untuk penerapan MVC.



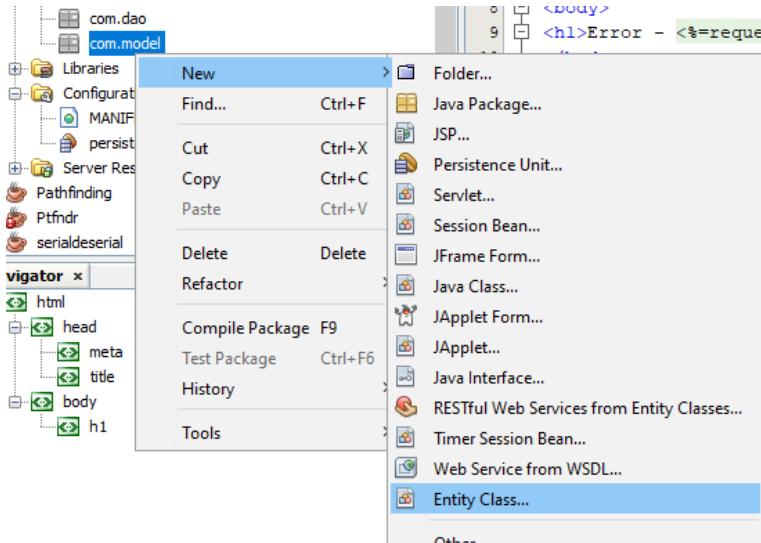
Gambar 4. 20 Membuat Java Package baru



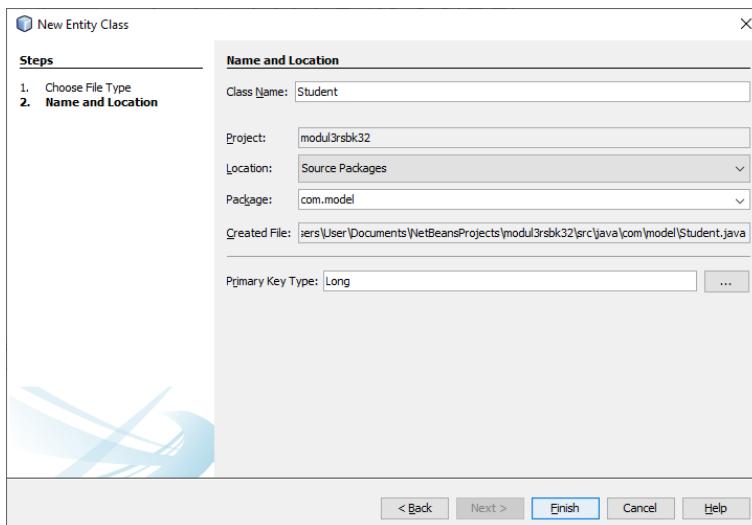
Gambar 4. 21 Contoh membuat Java Package com.controller

15. Buat 2 Entity Class pada package “com.model”, beri nama : “User” dan “Student”. Masukkan source code yang tersedia.

Ini adalah bagian model yang berisi object yang akan diolah pada aplikasi.



Gambar 4. 22 Membuat Entity Class baru



Gambar 4. 23 Contoh membuat Entity Class Student  
**User.java**

```
/*
 * To change this license header, choose License Headers in
 * Project Properties.
 */
```

```
* To change this template file, choose Tools | Templates
* and open the template in the editor.
*/
package com.model;

import java.io.Serializable;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.NamedQueries;
import javax.persistenceNamedQuery;
import javax.persistence.Table;
import javax.persistence.Column;

/**
 *
 * @author USER
 */
@Entity
@Table
@NamedQueries({@NamedQuery(name="User.getAll",query="SELECT e
FROM User e")})
public class User implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int userId;
    @Column
    private String userName;
    @Column
    private String password;

    public User() {
    }

    public User(String userName, String password) {
        this.userName = userName;
        this.password = password;
    }
    public int getUserId() {
        return userId;
    }
    public void setUserId(int userId) {
        this.userId = userId;
    }
    public String getUserName() {
        return userName;
    }
    public void setUserName(String userName) {
        this.userName = userName;
    }
}
```

```

    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
}

```

### **Student.java**

```

/*
 * To change this license header, choose License Headers in
Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.model;

import java.io.Serializable;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.NamedQueries;
import javax.persistenceNamedQuery;
import javax.persistence.Table;
import javax.persistence.Column;

/**
 *
 * @author USER
 */
@Entity
@Table
@NamedQueries({@NamedQuery(name="Student.getAll",query="SELECT
e FROM Student e order by e.studentId")})
public class Student implements Serializable {

    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    private int studentId;
    @Column
    private String firstName;
    @Column
    private String lastName;
    public Student(int studentId, String firstName, String
lastName) {
        this.studentId = studentId;
        this.firstName = firstName;
        this.lastName = lastName;
    }
}

```

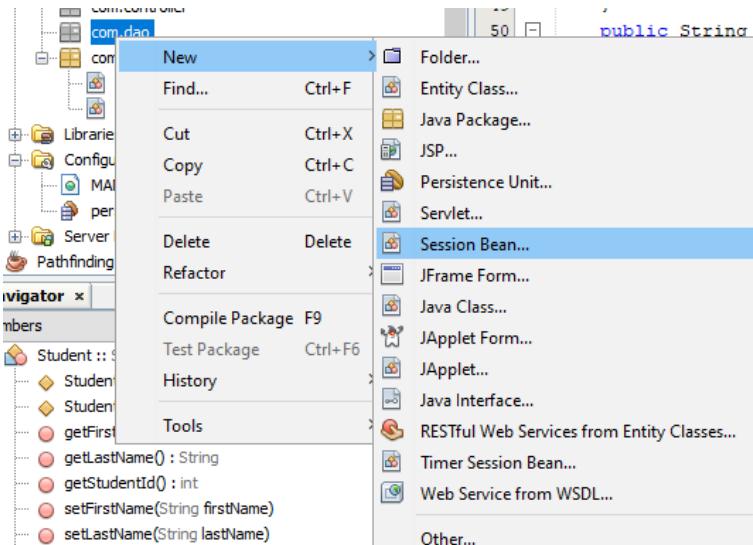
```

public Student() {
}
public void setStudentId(int studentId) {
this.studentId = studentId;
}
public int getStudentId() {
return studentId;
}
public void setFirstName(String firstName) {
this.firstName = firstName;
}
public String getFirstName() {
return firstName;
}
public void setLastName(String lastName) {
this.lastName = lastName;
}
public String getLastName() {
return lastName;
}
}

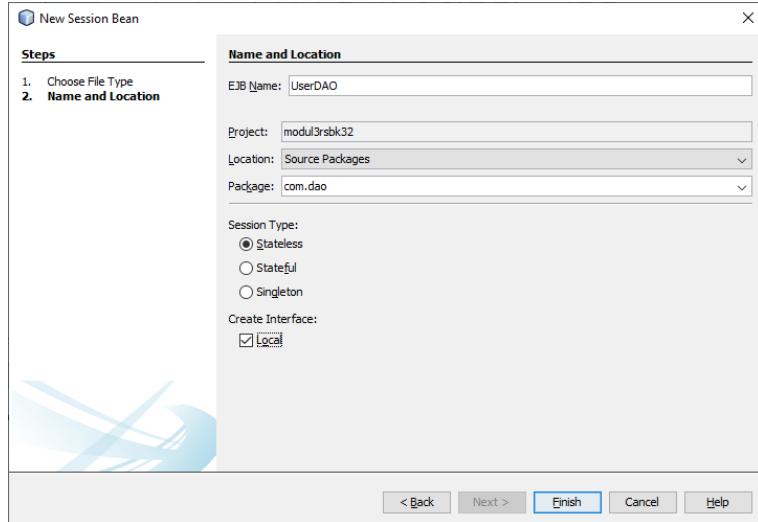
```

16. Buat 2 Session Beans pada *package* “com.dao”, beri nama : “UserDAO” dan “StudentDAO”. Ketika membuat session bean, pilih Session Type ‘**Stateless**’ dan Create Interface ‘**Local**’. Masukkan source code yang tersedia.

Pada *package* DAO berisi sebuah java session bean yang digunakan untuk interface antara model dengan controller nantinya.



Gambar 4. 24 Membuat session bean baru



Gambar 4. 25 Contoh membuat Session Bean *UserDAO*

### ***UserDAO.java***

```
/*
 * To change this license header, choose License Headers in
Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.dao;

import javax.ejb.Stateless;
import com.model.User;
import java.util.List;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

/**
 *
 * @author WIN 10
 */
@Stateless
public class UserDAO implements UserDAOLocal {
    @PersistenceContext
    private EntityManager em;
    @Override
    public boolean credential(String userName, String password)
{
    List<User> s = (List<User>)em.createQuery("select e from
User      e      where      e.userName='"+userName+"'      and
e.password='"+password+"'").getResultList();
}
```

```

        System.out.println("is list empty ?"+s.isEmpty()+" for
the"+userNames+" and "+password);
        if(!s.isEmpty())
        return true;
        else
        return false;
    }
    @Override
    public boolean checkUser(String userName) {
        List<User> s = (List<User>)em.createQuery("select e from
User e where e.userName='"+userNames+"'").getResultList();
        if(s.isEmpty())
        return true;
        else
        return false;
    }
    @Override
    public void addUser(User user){
        em.merge(user);
        em.flush();
    }

    // Add business logic below. (Right-click in editor and
choose
    // "Insert Code > Add Business Method")

}

```

### **UserDAOLocal.java**

```

/*
 * To change this license header, choose License Headers in
Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.dao;

import com.model.User;
import javax.ejb.Local;

/**
 *
 * @author WIN 10
 */
@Local
public interface UserDAOLocal {
    public boolean credential(String userName, String
password);
    public boolean checkUser (String userName);
    void addUser (User user);
}

```

### **StudentDAO.java**

```

/*
 * To change this license header, choose License Headers in
Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.dao;

import com.model.Student;
import java.util.List;
import javax.ejb.Stateless;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

/**
 *
 * @author WIN 10
 */
@Stateless
public class StudentDAO implements StudentDAOLocal {
    @PersistenceContext
    private EntityManager em;
    @Override
    public void addStudent(Student student) {
        em.merge(student);
        em.flush();
    }
    @Override
    public void editStudent(Student student) {
        em.merge(student);
        em.flush();
    }
    @Override
    public void deleteStudent(int studentId) {
        em.remove(getStudent(studentId));
        em.flush();
    }
    @Override
    public Student getStudent(int studentId) {
        em.flush();
        return em.find(Student.class, studentId);
    }
    @Override
    public List<Student> getAllStudents() {
        em.flush();
        return
        em.createNamedQuery("Student.getAll").getResultList();
    }
}

```

```
}
```

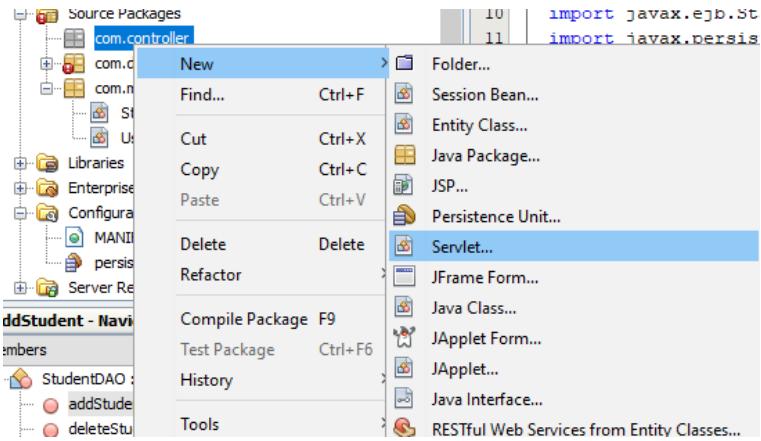
### ***StudentDAOLocal.java***

```
/*
 * To change this license header, choose License Headers in
Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.dao;

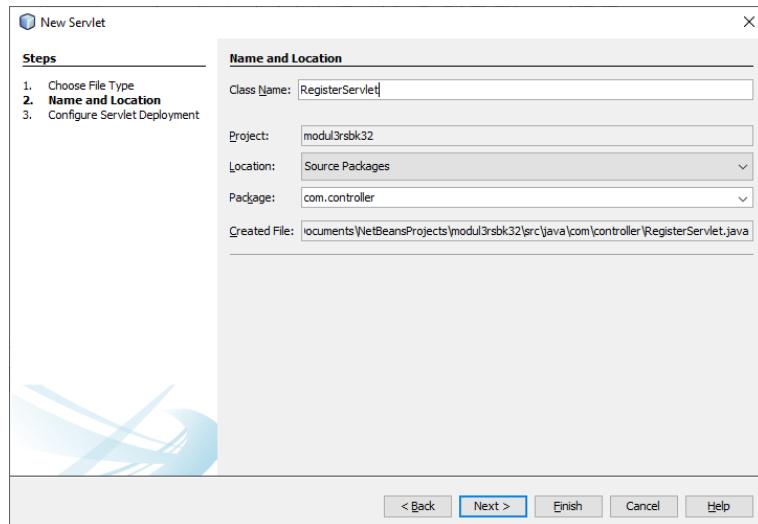
import com.model.Student;
import java.util.List;
import javax.ejb.Local;

/**
 *
 * @author WIN 10
 */
@Local
public interface StudentDAOLocal {
    void addStudent(Student student);
    void editStudent(Student student);
    void deleteStudent(int studentId);
    Student getStudent(int studentId);
    List<Student> getAllStudents();
}
```

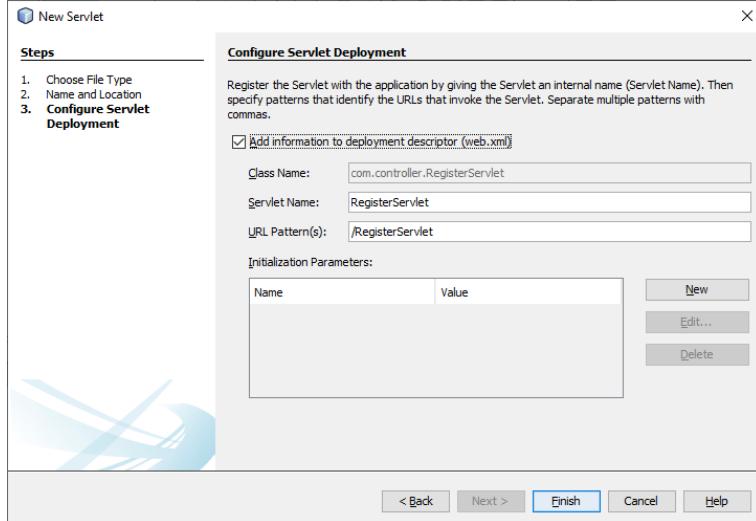
17. Buat 3 Servlet pada package “com.controlletr”, beri nama : “LoginServlet”, “RegistersServlet”, “StudentServlet”. Ketika membuat servlet, centang pada ‘Add information to deployment descriptor (web.xml)’. Kemudian masukkan source code yang tersedia.



Gambar 4. 26 Membuat Servlet baru



Gambar 4. 27 Contoh membuat RegisterServlet



Gambar 4. 28 Mengubah pattern url *RegisterServlet*

### ***LoginServlet.java***

```
/*
 * To change this license header, choose License Headers in
Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.controller;

import java.io.IOException;
import javax.ejb.EJB;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import com.dao.UserDAOLocal;

/**
 *
 * @author WIN 10
 */
@WebServlet(name = "LoginServlet", urlPatterns = {
    "/LoginServlet"})
public class LoginServlet extends HttpServlet {

    @EJB
    private UserDAOLocal userDAO;
```

```

boolean check = false;

/**
 * Processes requests for both HTTP <code>GET</code> and
<code>POST</code>
 * methods.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error
occurs
 * @throws IOException if an I/O error occurs
 */
protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    String userName = request.getParameter("userName");
    String password = request.getParameter("password");
    HttpSession session = request.getSession();
    check = userDAO.credential(userName, password);
    System.out.println("check is"+check+" "+userName);
    if(check)
    {
        session.setAttribute("userName", userName);

request.getRequestDispatcher("./StudentServlet").forward(request, response);
    } else {
        request.setAttribute("error", "Wrong Username or
Password");
    }
}

// <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
/**
 * Handles the HTTP <code>GET</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error
occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
}

```

```

        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request,
    HttpServletResponse response)
        throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     *
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    } // </editor-fold>
}

```

### **RegisterServlet.java**

```

/*
 * To change this license header, choose License Headers in
Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.controller;

import com.model.User;
import java.io.IOException;
import javax.ejb.EJB;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import com.dao.UserDAOLocal;

/**

```

```

/*
 * @author WIN 10
 */
public class RegistersServlet extends HttpServlet {
    @EJB
    private UserDAOLocal UserDAO;
    boolean check = true;
    /**
     * Processes requests for both HTTP <code>GET</code> and
     <code>POST</code> methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
     occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-
8");
        String userName = request.getParameter("userName");
        String password = request.getParameter("password");
        check = UserDAO.checkUser(userName);
        System.out.println("check is"+check+" "+userName);
        if(check){
            User user = new User(userName, password);
            UserDAO.addUser(user);
            request.setAttribute("user", user);

request.getRequestDispatcher("login.jsp").forward(request,
response);
        }else{
            request.setAttribute("error", "Username
already taken");
            request.getRequestDispatcher("error.jsp").forward(request,
response);
        }
    }

// <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
/**
 * Handles the HTTP <code>GET</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error
occurs

```

```

        * @throws IOException if an I/O error occurs
        */
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {
    processRequest(request, response);
}

/**
 * Handles the HTTP <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error
occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
    processRequest(request, response);
}

/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
    return "Short description";
} // </editor-fold>

}

```

### **StudentServlet.java**

```

/*
 * To change this license header, choose License Headers in
Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.controlletr;

import com.dao.StudentDAOLocal;
import com.model.Student;
import java.io.IOException;
import javax.ejb.EJB;
import javax.servlet.ServletException;

```

```

import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 *
 * @author WIN 10
 */
@WebServlet(name = "StudentServlet")
public class StudentServlet extends HttpServlet {
    @EJB
    private StudentDAOLocal studentDao;
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {
        String action = request.getParameter("action");
        String studentIdStr = request.getParameter("studentId");
        int studentId=0;
        if(studentIdStr!=null && !studentIdStr.equals("")){
            studentId=Integer.parseInt(studentIdStr);
        }
        String firstname = request.getParameter("firstname");
        String lastname = request.getParameter("lastname");
        Student student = new Student(studentId, firstname,
lastname);
        if("Add".equalsIgnoreCase(action)){
            studentDao.addStudent(student);
        }else if("Edit".equalsIgnoreCase(action)){
            studentDao.editStudent(student);
        }else if("Delete".equalsIgnoreCase(action)){
            studentDao.deleteStudent(studentId);
        }else if("Search".equalsIgnoreCase(action)){
            student = studentDao.getStudent(studentId);
        }
        request.setAttribute("student", student);
        request.setAttribute("allStudents",
studentDao.getAllStudents());
        request.getRequestDispatcher("home.jsp").forward(request,
response);
    }
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {
        processRequest(request, response);
    }
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {

```

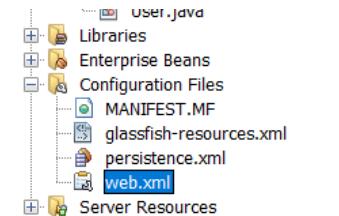
```

        processRequest(request, response);
    }
    @Override
    public String getServletInfo() {
        return "Short description";
        // </editor-fold>
    }
}

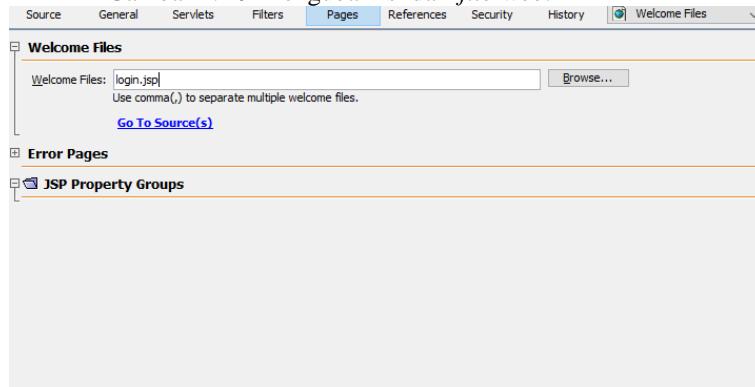
```

18. Buka “web.xml” di folder ‘Configuration Files’. Pada tab ‘Pages’, isikan Welcome Files dengan “login.jsp”.

Ini adalah *file* yang digunakan untuk mengedit tampilan awal.



Gambar 4. 29 Mengubah isi dari *file* web.xml



Gambar 4.30 Mengganti halaman awal web menjadi *login*

19. Lakukan ‘Clean and Build’ (Shift + F11), kemudian ‘Run’ (F6)

#### 4.4 Hasil Percobaan

##### *Package dao*

##### *UserDAO.java*

```

/*
 * To change this license header, choose License Headers in
Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

```

```
package com.dao;

import javax.ejb.Stateless;
import com.model.User;
import java.util.List;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

/**
 *
 * @author WIN 10
 */
@Stateless
public class UserDAO implements UserDAOLocal {
    @PersistenceContext
    private EntityManager em;
    @Override
    public boolean credential(String userName, String password) {
        List<User> s = (List<User>)em.createQuery("select e from User
e where e.userName='"+userName+"' and
e.password='"+password+"'").getResultList();
        System.out.println("is list empty ?"+s.isEmpty()+" for
the"+userName+" and "+password);
        if(!s.isEmpty())
            return true;
        else
            return false;
    }
    @Override
    public boolean checkUser(String userName) {
        List<User> s = (List<User>)em.createQuery("select e from User
e where e.userName='"+userName+"'").getResultList();
        if(s.isEmpty())
            return true;
        else
            return false;
    }
    @Override
    public void addUser(User user) {
        em.merge(user);
        em.flush();
    }

    // Add business logic below. (Right-click in editor and choose
    // "Insert Code > Add Business Method")

}
```

Pada baris code di *UserDAO* berguna untuk penghubung dari *database* ke aplikasi untuk mengolah *database* berupa pengeksekusian query terhadap *user*. Di dalamnya terdapat fungsi pengambilan data dan memasukannya kedalam arraylist model *User*.

### ***UserDAOLocal.java***

```
/*
 * To change this license header, choose License Headers in
Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.dao;

import com.model.User;
import javax.ejb.Local;

/**
 *
 * @author WIN 10
 */
@Local
public interface UserDAOLocal {
    public boolean credential(String userName, String password);
    public boolean checkUser (String userName);
    void addUser (User user);
}
```

Pada baris code di *UserDAOLocal* berguna untuk penghubung dari aplikasi ke *database*. File ini khusus untuk menghubungkan antar fungsi yang ada pada *controller* *user* sehingga dapat diolah dalam pemrograman menggunakan Bahasa java.

### ***StudentDAO.java***

```
/*
 * To change this license header, choose License Headers in
Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.dao;

import com.model.Student;
import java.util.List;
import javax.ejb.Stateless;
import javax.persistence.EntityManager;
```

```

import javax.persistence.PersistenceContext;

/**
 *
 * @author WIN 10
 */
@Stateless
public class StudentDAO implements StudentDAOLocal {
    @PersistenceContext
    private EntityManager em;
    @Override
    public void addStudent(Student student) {
        em.merge(student);
        em.flush();
    }
    @Override
    public void editStudent(Student student) {
        em.merge(student);
        em.flush();
    }
    @Override
    public void deleteStudent(int studentId) {
        em.remove(getStudent(studentId));
        em.flush();
    }
    @Override
    public Student getStudent(int studentId) {
        em.flush();
        return em.find(Student.class, studentId);
    }
    @Override
    public List<Student> getAllStudents() {
        em.flush();
        return
        em.createNamedQuery("Student.getAll").getResultList();
    }
}

```

Pada baris code di *StudentDAO* berguna untuk penghubung dari *database* ke aplikasi untuk mengolah *database* berupa pengeksekusian query terhadap *student*. Di dalamnya terdapat fungsi pengambilan data dan memasukannya ke dalam arraylist model *Student*.

### ***StudentDAOLocal.java***

```

/*

```

```

 * To change this license header, choose License Headers in
Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.dao;

import com.model.Student;
import java.util.List;
import javax.ejb.Local;

/**
 *
 * @author WIN 10
 */
@Local
public interface StudentDAOLocal {
    void addStudent(Student student);
    void editStudent(Student student);
    void deleteStudent(int studentId);
    Student getStudent(int studentId);
    List<Student> getAllStudents();
}

```

Pada baris code di *StudentDAOLocal* berguna untuk penghubung dari aplikasi ke *database*. *File* ini khusus untuk menghubungkan antar fungsi yang ada pada *controller* *student* sehingga dapat diolah dalam pemrograman menggunakan Bahasa java.

### ***Package controller***

#### ***LoginServlet.java***

```

/*
 * To change this license header, choose License Headers in Project
Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.controller;

import java.io.IOException;
import javax.ejb.EJB;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

```

```

import com.dao.UserDAOLocal;

/**
 *
 * @author WIN 10
 */
@WebServlet(name      = "LoginServlet",      urlPatterns      =
{"/LoginServlet"})
public class LoginServlet extends HttpServlet {

    @EJB
    private UserDAOLocal userDAO;
    boolean check = false;

    /**
     * Processes requests for both HTTP <code>GET</code> and
     <code>POST</code>
     * methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        String userName = request.getParameter("userName");
        String password = request.getParameter("password");
        HttpSession session = request.getSession();
        check = userDAO.credential(userName, password);
        System.out.println("check is"+check+" "+userName);
        if(check)
        {
            session.setAttribute("userName", userName);

request.getRequestDispatcher("./StudentServlet").forward(request,
response);
        } else {
            request.setAttribute("error", "Wrong Username or
Password");
        }
        request.getRequestDispatcher("error.jsp").forward(request,
response);
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
    /**

```

```

        * Handles the HTTP <code>GET</code> method.
        *
        * @param request servlet request
        * @param response servlet response
        * @throws ServletException if a servlet-specific error occurs
        * @throws IOException if an I/O error occurs
        */
@Override
protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
    processRequest(request, response);
}

/**
 * Handles the HTTP <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
    processRequest(request, response);
}

/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
    return "Short description";
} // </editor-fold>
}

```

*File* ini berguna untuk menghubungkan pengolahan aplikasi dengan tampilan web. Disini berisi pengkondisian dan pengaturan terhadap data pada model, kemudian menampilkannya pada halaman *login*, ataupun sebaliknya pengolahan data yang dimasukan melalui halaman jsp *login* ke *database* melalui *file interface DAO*.

### **RegistersServlet.java**

```
/*
```

```

 * To change this license header, choose License Headers in
Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.controlletr;

import com.model.User;
import java.io.IOException;
import javax.ejb.EJB;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import com.dao.UserDAOLocal;

/**
 *
 * @author WIN 10
 */
public class RegistersServlet extends HttpServlet {
    @EJB
    private UserDAOLocal UserDAO;
    boolean check = true;
    /**
     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code> methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        String userName = request.getParameter("userName");
        String password = request.getParameter("password");
        check = UserDAO.checkUser(userName);
        System.out.println("check is"+check+" "+userName);
        if(check) {
            User user = new User(userName, password);
            UserDAO.addUser(user);
            request.setAttribute("user", user);

request.getRequestDispatcher("login.jsp").forward(request,
response);
        }else{
            request.setAttribute("error", "Username already
taken");
        }
    }

}

```

```

        request.getRequestDispatcher("error.jsp").forward(request,
response);
    }

}

// <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
/***
 * Handles the HTTP <code>GET</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doGet(HttpServletRequest      request,
HttpServletResponse response)
    throws ServletException, IOException {
    processRequest(request, response);
}

/***
 * Handles the HTTP <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest      request,
HttpServletResponse response)
    throws ServletException, IOException {
    processRequest(request, response);
}

/***
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
    return "Short description";
}// </editor-fold>

}

```

*File* ini berguna untuk menghubungkan pengolahan aplikasi dengan tampilan web. Disini berisi pengkondisian dan pengaturan terhadap data pada model, kemudian menampilkannya pada halaman *register*, ataupun sebaliknya pengolahan data yang dimasukan melalui halaman jsp *register* ke *database* melalui *file interface DAO*.

### **StudentServlet.java**

```

/*
 * To change this license header, choose License Headers in
Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.controlletr;

import com.dao.StudentDAOLocal;
import com.model.Student;
import java.io.IOException;
import javax.ejb.EJB;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 *
 * @author WIN 10
 */
@WebServlet(name = "StudentServlet")
public class StudentServlet extends HttpServlet {
    @EJB
    private StudentDAOLocal studentDao;
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {
        String action = request.getParameter("action");
        String studentIdStr = request.getParameter("studentId");
        int studentId=0;
        if(studentIdStr!=null && !studentIdStr.equals("")){
            studentId=Integer.parseInt(studentIdStr);
        }
        String firstname = request.getParameter("firstname");
        String lastname = request.getParameter("lastname");
        Student student = new Student(studentId, firstname,
lastname);
        if("Add".equalsIgnoreCase(action)){
            studentDao.addStudent(student);
        }else if("Edit".equalsIgnoreCase(action)){
            studentDao.editStudent(student);
        }else if("Delete".equalsIgnoreCase(action)){
            studentDao.deleteStudent(studentId);
        }else if("Search".equalsIgnoreCase(action)){
            student = studentDao.getStudent(studentId);
        }
        request.setAttribute("student", student);
    }
}

```

```

        request.setAttribute("allStudents",
studentDao.getAllStudents());
        request.getRequestDispatcher("home.jsp").forward(request,
response);
    }
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {
    processRequest(request, response);
}
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {
    processRequest(request, response);
}
    @Override
    public String getServletInfo() {
    return "Short description";
// </editor-fold>
}

```

*File* ini berguna untuk menghubungkan pengolahan aplikasi dengan tampilan web. Disini berisi pengkondisian dan pengaturan terhadap data pada model, kemudian menampilkannya pada halaman *home*, ataupun sebaliknya pengolahan data yang dimasukan melalui halaman jsp *home* ke *database* melalui *file interface DAO*.

### **Package model**

#### **User.java**

```

/*
 * To change this license header, choose License Headers in
Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.model;

import java.io.Serializable;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.NamedQueries;

```

```
import javax.persistence.NamedQuery;
import javax.persistence.Table;
import javax.persistence.Column;

/**
 *
 * @author USER
 */
@Entity
@Table
@NamedQueries({@NamedQuery(name="User.getAll",query="SELECT e
FROM User e")})
public class User implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int userId;
    @Column
    private String userName;
    @Column
    private String password;

    public User() {
    }

    public User(String userName, String password) {
        this.userName = userName;
        this.password = password;
    }
    public int getUserId() {
    return userId;
    }
    public void setUserId(int userId) {
    this.userId = userId;
    }
    public String getUserName() {
    return userName;
    }
    public void setUserName(String userName) {
    this.userName = userName;
    }
    public String getPassword() {
    return password;
    }
    public void setPassword(String password) {
    this.password = password;
    }
}
```

*File* ini adalah model *user* yang berisi variabel-variabel pembentuk *user*. Di dalamnya terdapat *setter* dan *getter* serta constructor untuk membangun *object user*. Disinilah penggunaan komponen pada aplikasi ini.

### **Student.java**

```
/*
 * To change this license header, choose License Headers in
Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.model;

import java.io.Serializable;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.Table;
import javax.persistence.Column;

/**
 *
 * @author USER
 */
@Entity
@Table
@NamedQueries({@NamedQuery(name="Student.getAll",query="SELECT e
FROM Student e order by e.studentId")})
public class Student implements Serializable {

    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    private int studentId;
    @Column
    private String firstName;
    @Column
    private String lastName;
    public Student(int studentId, String firstName, String
lastName) {
        this.studentId = studentId;
        this.firstName = firstName;
        this.lastName = lastName;
    }
    public Student() {
}
```

```

public void setStudentId(int studentId) {
    this.studentId = studentId;
}
public int getStudentId() {
    return studentId;
}
public void setFirstName(String firstName) {
    this.firstName = firstName;
}
public String getFirstName() {
    return firstName;
}
public void setLastName(String lastName) {
    this.lastName = lastName;
}
public String getLastName() {
    return lastName;
}
}

```

*File* ini adalah model *student* yang berisi variabel-variabel pembentuk *student*. Didalamnya terdapat *setter* dan *getter* serta constructor untuk membangun *object student*. Disinilah penggunaan komponen pada aplikasi ini.

### **register.jsp**

```

<%-->
    Document      : register
    Created on   : Sep 22, 2019, 2:48:05 PM
    Author        : WIN 10
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1">

    <title>Register Data</title>
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <script src="js/bootstrap.min.js"></script>
    <style>
        .menu {
            margin-left: -15px;
            margin-right: 15px;
        }
    </style>

```

```

        }
    .daftar{
        border: 2px solid #e5e5e5;
        border-radius: 10px;
        padding: 20px;
    }
    .daftar a{
        margin-top: 2%;
    }
    .detail{
        padding: 10px 0px;
    }
    .nav{
        padding: 0px;
        border: 1px solid #e5e5e5;
        border-radius: 5px;
    }
    .nav li{
        border-bottom: 1px solid #e5e5e5;
        border-radius: 5px;
    }

</style>
</head>

<body>
    <div class="container">
        <div class="jumbotron row">
            <a href=".//login.jsp" class="btn btn-md btn-success" style="float:right" />Login</a><br>
            <center><h2><b>Data Mahasiswa</b></h2>
            <h4>Modul RSBK - Kelompokxxx</h4></center>
        </div>
        <div class="row content">
            <div class="col-md-12">
                <div class="col-md-4 col-md-offset-4 daftar">
                    <p class="form-title">Sign Up</p>
                    <form method="POST" action=".//RegistersServlet">
                        <div class="form-group">
                            <label>Username</label>
                            <input type="text" class="form-control" placeholder="Username" name="userName" type="text" autofocus />
                        </div>
                        <div class="form-group">
                            <label>Password</label>
                            <input type="password" class="form-control" placeholder="Password" name="password" value="" required />
                        </div>
                        <input type="submit" name="register" value="Register" class="btn btn-success" />
                    </form>
                </div>
            </div>
        </div>
    </div>
</body>
```

```

        </div>
    </div>
</div>
</body>
</html>
```

*File* jsp ini digunakan untuk menampilkan tampilan web pada *register*.

### **login.jsp**

```

<%--
 Document      : login
 Created on   : Sep 22, 2019, 12:45:49 PM
 Author       : WIN 10
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1">

    <title>Login Data</title>
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <script src="js/bootstrap.min.js"></script>
    <style>
        .menu {
            margin-left: -15px;
            margin-right: 15px;
        }
        .daftar{
            border: 2px solid #e5e5e5;
            border-radius: 10px;
            padding: 20px;
        }
        .daftar a{
            margin-top: 2%;
        }
        .detail{
            padding: 10px 0px;
        }
        .nav{
            padding: 0px;
            border: 1px solid #e5e5e5;
            border-radius: 5px;
        }
        .nav li{
            border-bottom: 1px solid #e5e5e5;
        }
    </style>
</head>
<body>
    <div class="container">
        <div class="row">
            <div class="col-md-6 col-md-offset-3">
                <div class="panel panel-default">
                    <div class="panel-body">
                        <form action="proses_login.jsp" method="post">
                            <div class="form-group">
                                <label for="username">Username</label>
                                <input type="text" class="form-control" id="username" name="username" required="required" placeholder="Masukkan Username" value=""></div>
                            <div class="form-group">
                                <label for="password">Password</label>
                                <input type="password" class="form-control" id="password" name="password" required="required" placeholder="Masukkan Password" value=""></div>
                            <div class="checkbox">
                                <label>Remember Me</label>
                                <input type="checkbox" name="remember_me" checked="checked" value="1"></div>
                            </div>
                            <div class="text-center">
                                <button type="submit" class="btn btn-primary">Submit</button>
                            </div>
                        </form>
                    </div>
                </div>
            </div>
        </div>
    </div>
</body>
```

```

        border-radius: 5px;
    }

</style>
</head>

<body>
    <div class="container">
        <div class="jumbotron row">
            <a href=".register.jsp" class="btn btn-md btn-success" style="float:right" />Register</a><br>
            <center><h2><b>Data Mahasiswa</b></h2>
            <h4>Modul RSBK - Kelompokxxx</h4></center>
        </div>
        <div class="row content">
            <div class="col-md-12">
                <div class="col-md-4 col-md-offset-4 daftar">
                    <p class="form-title">Sign In</p>
                    <form method="POST" action=".LoginServlet">
                        <div class="form-group">
                            <label>Username</label>
                            <input type="text" class="form-control" placeholder="Username" name="userName" type="text" autofocus />
                        </div>
                        <div class="form-group">
                            <label>Password</label>
                            <input type="password" class="form-control" placeholder="Password" name="password" value="" required />
                        </div>
                        <input type="submit" name="login" value="Login" class="btn btn-md btn-success" />
                    </form>
                </div>
            </div>
        </div>
    </body>
</html>

```

*File jsp ini digunakan untuk menampilkan tampilan web pada login.*

### **home.jsp**

```

<%-->
    Document   : home
    Created on : Sep 22, 2019, 12:45:58 PM
    Author     : WIN 10
--%>

```

```

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib prefix="s" uri="http://java.sun.com/jsp/jstl/core" %>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Home Page</title>
<link rel="stylesheet" href="css/bootstrap.min.css">
<script src="js/bootstrap.min.js"></script>
<style>
    .menu {
        margin-left: -15px;
        margin-right: 15px;
    }
    .daftar{
        border: 2px solid #e5e5e5;
        border-radius: 5px;
        padding: 5px;
    }
    .table th, .table td{
        text-align: center;
    }
    .nav{
        padding: 5px;
        border: 2px solid #e5e5e5;
        border-radius: 5px;
    }
    .nav li{
        border-bottom: 2px solid #e5e5e5;
        border-radius: 5px;
    }
    .daftar h3{
        margin-top: 50px;
        margin-bottom: 25px;
    }
</style>
</head>

<div class="container">
    <div class="jumbotron row">
        <center><h2><b>Data Mahasiswa</b></h2>
        <p>Modul RSBK - Kelompokxx</p>
        <h5>Selamat Datang,</h5>
<%=session.getAttribute("loginName")%></h6></center>
    </div>
    <div class="row content col-md-8 col-md-offset-2">
        <div class="col-md-3 menu">
            <ul class="nav nav-pills nav-stacked" style="">
                <li><a href="#">Home</a></li>
                <li><a href=".//login.jsp">Logout</a></li>
            </ul>
        </div>

```

```

        </div>
<div class="col-md-9 daftar">
    <form action=".//StudentServlet" method="POST">
        <table class="table table-bordered">
            <tr>
                <td>Student ID</td>
                <td><input class="form-control" type="text" name="studentId" value="${student.studentId}" /></td>
            </tr>
            <tr>
                <td>First Name</td>
                <td><input class="form-control" type="text" name="firstname" value="${student.firstName}" /></td>
            </tr>
            <tr>
                <td>Last Name</td>
                <td><input class="form-control" type="text" name="lastname" value="${student.lastName}" /></td>
            </tr>
            <tr>
                <td colspan="2">
                    <input type="submit" class="btn btn-primary btn-sm" name="action" value="Add" />
                    <input type="submit" class="btn btn-default btn-sm" name="action" value="Edit" />
                    <input type="submit" class="btn btn-danger btn-sm" name="action" value="Delete" />
                    <input type="submit" class="btn btn-warning btn-sm" name="action" value="Search" />
                </td>
            </tr>
        </table>
    </form>
    <h3 align="center">Informasi Data</h3>
    <table class="table table-bordered table-hover">
        <thead>
            <tr>
                <th>No. ID</th>
                <th>First Name</th>
                <th>Last Name</th>
            </tr>
        </thead>
        <tbody>
            <s:forEach items="${allStudents}" var="stud">
                <tr>
                    <td>${stud.studentId}</td>
                    <td>${stud.firstName}</td>
                    <td>${stud.lastName}</td>
                </tr>
            </s:forEach>
        </tbody>
    </table>
</div>

```

```

        </table>
    </div>
</div>
</div>
</html>

```

File jsp ini digunakan untuk menampilkan tampilan web pada *home*.

### **error.jsp**

```

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Error Page</title>
</head>
<body>
<h1>Error - <%=request.getAttribute("error")%> </h1>
</body>
</html>

```

File jsp ini digunakan untuk menampilkan tampilan web saat *error login*.

The screenshot shows a login interface. At the top right is a green 'Register' button. Below it, the title 'Data Mahasiswa' is displayed above a subtitle 'Modul RSBK - Kelompokxx'. A large rectangular box contains a 'Sign In' form. The form includes fields for 'Username' (with placeholder 'Username') and 'Password' (with placeholder 'Password'), both enclosed in input boxes. Below these fields is a green 'Login' button.

Gambar 4.31 Hasil halaman *Login*

The screenshot shows a registration form titled "Data Mahasiswa" under the heading "Modul RSBK - Kelompokxx". A "Login" button is located in the top right corner. The form itself has a light gray background and contains the following fields:

- A "Sign Up" link.
- A "Username" field containing "rio".
- A "Password" field containing "\*\*\*".
- A "Register" button at the bottom.

Gambar 4.32 Hasil halaman *Register*

The screenshot shows the main interface of the "Data Mahasiswa" application. At the top, it displays the title "Data Mahasiswa" and the subtitle "Modul RSBK - Kelompokxx". Below this, a welcome message "Selamat Datang, rio" is shown. On the left side, there is a sidebar with two buttons: "Home" and "Logout". The main content area contains three input fields for "Student ID", "First Name", and "Last Name", each with its own input box. Below these fields is a row of four buttons: "Add" (blue), "Edit" (light blue), "Delete" (red), and "Search" (orange). Further down, a section titled "Informasi Data" is displayed, featuring a table with columns for "No. ID", "First Name", and "Last Name".

Gambar 4.33 Hasil halaman *Home*

**Data Mahasiswa**  
Modul RSBK - Kelompokxx  
Selamat Datang, rio

|        |  |
|--------|--|
| Home   | <input type="text" value="60"/>  |
| Logout |  |
|        | <input type="text" value="rio"/>   |
|        | <input type="text" value="kisna"/>   |
|        | <input type="button" value="Add"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/> <input type="button" value="Search"/> |

**Informasi Data**

| No. ID | First Name | Last Name |
|--------|------------|-----------|
| 60     | rio        | kisna     |

Gambar 4.34 Hasil percobaan tambah data

**Data Mahasiswa**  
Modul RSBK - Kelompokxx  
Selamat Datang, rio

|        |  |
|--------|--|
| Home   | <input type="text" value="60"/>  |
| Logout | <input type="text" value="ray"/> <input type="button" value="x"/>  |
|        | <input type="text" value="kisna"/>   |
|        | <input type="button" value="Add"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/> <input type="button" value="Search"/> |

**Informasi Data**

| No. ID | First Name | Last Name |
|--------|------------|-----------|
| 60     | rio        | kisna     |

Gambar 4.35 Hasil percobaan *edit* data

**Data Mahasiswa**  
Modul RSBK - Kelompokxx  
Selamat Datang, rio

|   |       |            |    |            |     |           |       |  |  |
|---|-------|------------|----|------------|-----|-----------|-------|--|--|
| Home  |       |            |    |            |     |           |       |  |  |
| Logout  |       |            |    |            |     |           |       |  |  |
| <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 15%;">Student ID</td> <td>60</td> </tr> <tr> <td>First Name</td> <td>ray</td> </tr> <tr> <td>Last Name</td> <td>kisna</td> </tr> <tr> <td style="text-align: right; padding-top: 5px;"> <input type="button" value="Add"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/> <input type="button" value="Search"/> </td> <td></td> </tr> </table> |       | Student ID | 60 | First Name | ray | Last Name | kisna | <input type="button" value="Add"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/> <input type="button" value="Search"/> |  |
| Student ID  | 60    |            |    |            |     |           |       |  |  |
| First Name  | ray   |            |    |            |     |           |       |  |  |
| Last Name   | kisna |            |    |            |     |           |       |  |  |
| <input type="button" value="Add"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/> <input type="button" value="Search"/>  |       |            |    |            |     |           |       |  |  |

**Informasi Data**

| No. ID | First Name | Last Name |
|--------|------------|-----------|
| 60     | ray        | kisna     |

Gambar 4.36 Hasil percobaan search data

**Data Mahasiswa**  
Modul RSBK - Kelompokxx  
Selamat Datang, rio

|   |       |            |    |            |     |           |       |  |  |
|---|-------|------------|----|------------|-----|-----------|-------|--|--|
| Home  |       |            |    |            |     |           |       |  |  |
| Logout  |       |            |    |            |     |           |       |  |  |
| <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 15%;">Student ID</td> <td>60</td> </tr> <tr> <td>First Name</td> <td>ray</td> </tr> <tr> <td>Last Name</td> <td>kisna</td> </tr> <tr> <td style="text-align: right; padding-top: 5px;"> <input type="button" value="Add"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/> <input type="button" value="Search"/> </td> <td></td> </tr> </table> |       | Student ID | 60 | First Name | ray | Last Name | kisna | <input type="button" value="Add"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/> <input type="button" value="Search"/> |  |
| Student ID  | 60    |            |    |            |     |           |       |  |  |
| First Name  | ray   |            |    |            |     |           |       |  |  |
| Last Name   | kisna |            |    |            |     |           |       |  |  |
| <input type="button" value="Add"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/> <input type="button" value="Search"/>  |       |            |    |            |     |           |       |  |  |

**Informasi Data**

| No. ID | First Name | Last Name |
|--------|------------|-----------|
|        |            |           |

Gambar 4.37 Hasil percobaan delete data

## 4.5 Tugas dan Pembahasan

Pada tugas kali ini adalah memperbaiki *error* yang terjadi saat praktikum. Karena saat praktikum kelompok kami tidak mengalami *error*, sehingga kami tidak mengubah aplikasi yang sudah kami buat saat praktikum.

Tugas kedua adalah mempercantik tampilan. Tampilan yang kami buat sebagian besar menggunakan *template*. Pertama kita membuat *file* css dan resource lainnya yang dibutuhkan untuk menghasilkan tampilan yang lebih menarik. Kemudian memasukan *file* html pada jsp tampilan yang diperlukan.

*File login.jsp*

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Login V17</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1">
<!--
=====
=====-->
    <link rel="icon" type="image/png"
href="images/icons/favicon.ico"/>
<!--
=====
=====-->
    <link rel="stylesheet" type="text/css"
href="vendor/bootstrap/css/bootstrap.min.css">
<!--
=====
=====-->
    <link rel="stylesheet" type="text/css" href="fonts/font-
awesome-4.7.0/css/font-awesome.min.css">
<!--
=====
=====-->
    <link rel="stylesheet" type="text/css"
href="fonts/Linearicons-Free-v1.0.0/icon-font.min.css">
<!--
=====
=====-->
    <link rel="stylesheet" type="text/css"
href="vendor/animate/animate.css">
```

```

<!--
=====
=====-->
    <link rel="stylesheet" type="text/css" href="vendor/css-
hamburgers/hamburgers.min.css">
<!--
=====
=====-->
    <link           rel="stylesheet"           type="text/css"
href="vendor/animsition/css/animstion.min.css">
<!--
=====
=====-->
    <link           rel="stylesheet"           type="text/css"
href="vendor/select2/select2.min.css">
<!--
=====
=====-->
    <link           rel="stylesheet"           type="text/css"
href="vendor/daterangepicker/daterangepicker.css">
<!--
=====
=====-->
    <link rel="stylesheet" type="text/css" href="css/util.css">
    <link rel="stylesheet" type="text/css" href="css/main.css">
<!--
=====
=====-->
</head>
<body>

    <div class="limiter">
        <div class="container-login100">
            <div class="wrap-login100">
                <form class="login100-form validate-form"
method="POST" action=".//LoginServlet">
                    <span class="login100-form-title p-b-34">
                        Account Login
                    </span>

                    <div class="wrap-input100 rs1-wrap-
input100 validate-input m-b-20" data-validate="Type user name">
                        <input           id="first-name"
class="input100" type="text" name="userName" placeholder="User
name">
                        <span           class="focus-
input100"></span>
                    </div>
                    <div class="wrap-input100 rs2-wrap-
input100 validate-input m-b-20" data-validate="Type password">

```

```

                <input          class="input100"
type="password" name="password" placeholder="Password">
                <span          class="focus-
input100"></span>
            </div>

            <div  class="container-login100-form-
btn">
                <button    class="login100-form-
btn">
                    Sign in
                </button>
            </div>

            <div class="w-full text-center p-t-27
p-b-239">
                <span class="txt1">
                    Forgot
                </span>

                <a href="#" class="txt2">
                    User name / password?
                </a>
            </div>

            <div class="w-full text-center">
                <a          href=".//register.jsp"
class="txt3">
                    Sign Up
                </a>
            </div>
        </form>

        <div          class="login100-more"
style="background-image: url('images/bg-01.jpg');"></div>
    </div>
</div>

<div id="dropDownSelect1"></div>

<!--
=====
=====-->
    <script src="vendor/jquery/jquery-3.2.1.min.js"></script>
<!--
=====
=====-->
    <script
src="vendor/animsition/js/animsition.min.js"></script>

```

```

<!--
=====
=====-->
    <script src="vendor/bootstrap/js/popper.js"></script>
    <script src="vendor/bootstrap/js/bootstrap.min.js"></script>
<!--
=====
=====-->
    <script src="vendor/select2/select2.min.js"></script>
    <script>
        $(".selection-2").select2({
            minimumResultsForSearch: 20,
            dropdownParent: $('#dropDownSelect1')
        });
    </script>
<!--
=====
=====-->
    <script src="vendor/daterangepicker/moment.min.js"></script>
    <script
src="vendor/daterangepicker/daterangepicker.js"></script>
<!--
=====
=====-->
    <script src="vendor/countdowntime/countdowntime.js"></script>
<!--
=====
=====-->
    <script src="js/main.js"></script>

</body>
</html>

```

### *Register.jsp*

```

<!DOCTYPE html>
<html lang="en">
<head>
    <title>Login V17</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1">
<!--
=====
=====-->
    <link rel="icon" type="image/png"
href="images/icons/favicon.ico"/>

```

```

<!--
=====
=====-->
    <link      rel="stylesheet"          type="text/css"
href="vendor/bootstrap/css/bootstrap.min.css">
<!--
=====
=====-->
    <link  rel="stylesheet" type="text/css" href="fonts/font-
awesome-4.7.0/css/font-awesome.min.css">
<!--
=====
=====-->
    <link      rel="stylesheet"          type="text/css"
href="fonts/Linearicons-Free-v1.0.0/icon-font.min.css">
<!--
=====
=====-->
    <link      rel="stylesheet"          type="text/css"
href="vendor/animate/animate.css">
<!--
=====
=====-->
    <link  rel="stylesheet" type="text/css" href="vendor/css-
hamburgers/hamburgers.min.css">
<!--
=====
=====-->
    <link      rel="stylesheet"          type="text/css"
href="vendor/animsition/css/animsition.min.css">
<!--
=====
=====-->
    <link      rel="stylesheet"          type="text/css"
href="vendor/select2/select2.min.css">
<!--
=====
=====-->
    <link      rel="stylesheet"          type="text/css"
href="vendor/daterangepicker/daterangepicker.css">
<!--
=====
=====-->
    <link      rel="stylesheet"          type="text/css"
href="css/util.css">
        <link      rel="stylesheet"          type="text/css"
href="css/main.css">

```

```

<!--
=====
=====-->
</head>
<body>

    <div class="limiter">
        <div class="container-login100">
            <div class="wrap-login100">
                <form class="login100-form validate-form"
method="POST" action="./RegisterServlet">
                    <span class="login100-form-title p-
b-34">
                        Account Register
                    </span>
                    <span class="txt1">
                        Pastikan anda benar-benar
pengguna baru!
                    </span>
                    <div class="wrap-input100 rs1-wrap-
input100 validate-input m-b-20" data-validate="Type user name">
                        <input id="first-name"
class="input100" type="text" name="userName" placeholder="User
name">
                        <span class="focus-
input100"></span>
                    </div>
                    <div class="wrap-input100 rs2-wrap-
input100 validate-input m-b-20" data-validate="Type password">
                        <input class="input100"
type="password" name="password" placeholder="Password">
                        <span class="focus-
input100"></span>
                    </div>

                    <div class="container-login100-
form-btn">
                        <button class="login100-form-
btn">
                            Sign up
                        </button>
                    </div>

                    <div class="w-full text-center p-t-
27 p-b-239">
                        </div>
                    <div class="w-full text-center">

```

```

                <a href=". / login.jsp"
class="txt3">
                    Sudah Punya Akun?
                </a>
            </div>
        </form>

        <div class="login100-more"
style="background-image: url('images/bg-01.jpg');"></div>
    </div>
</div>

<div id="dropDownSelect1"></div>

<!--
=====
=====-->
<script src="vendor/jquery/jquery-3.2.1.min.js"></script>
<!--
=====
=====-->
<script
src="vendor/animsition/js/animsition.min.js"></script>
<!--
=====
=====-->
<script src="vendor/bootstrap/js/popper.js"></script>
<script
src="vendor/bootstrap/js/bootstrap.min.js"></script>
<!--
=====
=====-->
<script src="vendor/select2/select2.min.js"></script>
<script>
    $(".selection-2").select2({
        minimumResultsForSearch: 20,
        dropdownParent: $('#dropDownSelect1')
    });
</script>
<!--
=====
=====-->
<script
src="vendor/daterangepicker/moment.min.js"></script>
<script
src="vendor/daterangepicker/daterangepicker.js"></script>

```

```
<!--
=====
=====-->
    <script
src="vendor/countdowntime/countdowntime.js"></script>
<!--
=====
=====-->
    <script src="js/main.js"></script>

</body>
</html>
```

### *Home.jsp*

```
<%--
 Document      : home
 Created on   : Sep 22, 2019, 12:45:58 PM
 Author       : WIN 10
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib prefix="s" uri="http://java.sun.com/jsp/jstl/core" %>
<!doctype html>
<html lang="en">
    <head>
        <!-- Required meta tags -->
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width,
initial-scale=1, shrink-to-fit=no">
        <title>Sistem Informasi Mahasiswa TEKKOM</title>
        <!-- Bootstrap CSS -->
        <link
            href="bootstrap/css/bootstrap.min.css" >
            rel="stylesheet"
        <style type="text/css">
            .nav-link {
                color: #fff;
            }
            li a:hover {
                color: #fff;
            }
            .navbar{
                background-color: #4A314D;
                z-index: 100;
            }
            .container{
                margin-top: 20px;
            }
            .jumbotron {
```

```

        background-color: #6B6570;
        margin-left: 5%;
        width: 90%;
    }
.jumbotron hr {
    color: #A8BA9A;
    border: none;
}
.jumbotron p {
    color: #FFF;
    padding-left: 7.5%;
    font-size: 26px;
}
.btn {
    float: right;
    background-color: #6fc94b;
    color: #FFF;
    width: 10%;
    height: 42px;
    margin-left: 2.5%;
}
.btn:hover {
    opacity: 0.8;
}
.table1 {
    margin-left: 5%;
    width: 90%;
}
.table1 td {
    width: 20%;
}
.table1 inputtext {
    width: 80%;
}
</style>

</head>
<body>
    <nav class="navbar navbar-expand-lg">
        <a class="navbar-brand" href="index.php">Mahasiswa
TEKKOM</a>
        <button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-
label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
        </button>

        <div class="collapse navbar-collapse" id="navbarSupportedContent">
            <ul class="navbar-nav mr-auto">

```

```

        <li class="nav-item active">
            <a class="nav-link" href="#">HOME <span
class="sr-only">(current)</span></a>
        </li>

    </ul>
    <ul class="nav navbar-nav navbar-right">
        <li class="nav-item">
            <a href=".about.html" class="nav-link">
<%=session.getAttribute("userName")%></a>
        </li>
        <li class="nav-item">
            <a href=".login.jsp" class="nav-link">
Logout</a>
        </li>
    </ul>
</div>
</nav>

<div class="container">
    <div class="jumbotron">
        <p>Data Mahasiswa</p><hr>
        <p>Modul RSBK - Kelompok32</p><hr>
        <p>Selamat Datang,<br/>
<%=session.getAttribute("userName")%></p>
    </div>

    <div class="daftar">
        <form action=".StudentServlet" method="POST">
            <table class="table table1 table-bordered">
                <tr>
                    <td>Student ID</td>
                    <td><input class="form-control inputtext" type="text" name="studentId" value="${student.studentId}" /></td>
                </tr>
                <tr>
                    <td>First Name</td>
                    <td><input class="form-control inputtext" type="text" name="firstname" value="${student.firstName}" /></td>
                </tr>
                <tr>
                    <td>Last Name</td>
                    <td><input class="form-control inputtext" type="text" name="lastname" value="${student.lastName}" /></td>
                </tr>
                <tr>
                    <td>Alamat</td>
                    <td><input class="form-control inputtext" type="text" name="alamat" value="${student.alamat}" /></td>
                </tr>
                <tr>
                    <td colspan="2" style="text-align: center;">
                        <input type="submit" value="Simpan" />
                    </td>
                </tr>
            </table>
        </form>
    </div>

```

```

                <input type="submit" class="btn btn-sm"
name="action" value="Search" />
                <input type="submit" class="btn btn-sm"
name="action" value="Delete" />
                <input type="submit" class="btn btn-sm"
name="action" value="Edit" />
                <input type="submit" class="btn btn-sm"
name="action" value="Add" />
            </td>
        </tr>
    </table>
</form>
<h3 align="center">Informasi Data</h3>
<table class="table table1 table-bordered table-
hover">
    <thead>
        <tr>
            <th>No. ID</th>
            <th>First Name</th>
            <th>Last Name</th>
            <th>Alamat</th>
        </tr>
    </thead>
    <tbody>
        <s:forEach items="${allStudents}"
var="stud">
            <tr>
                <td>${stud.studentId}</td>
                <td>${stud.firstName}</td>
                <td>${stud.lastName}</td>
                <td>${stud.alamat}</td>
            </tr>
        </s:forEach>
    </tbody>
</table>
</div>
</div>
</html>
```

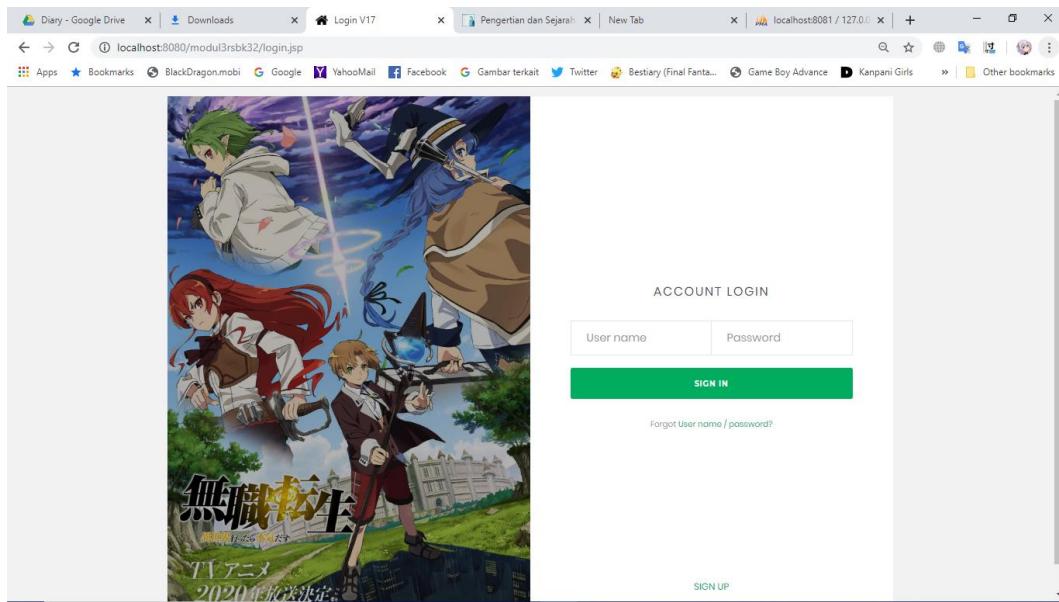
### Error.jsp

```

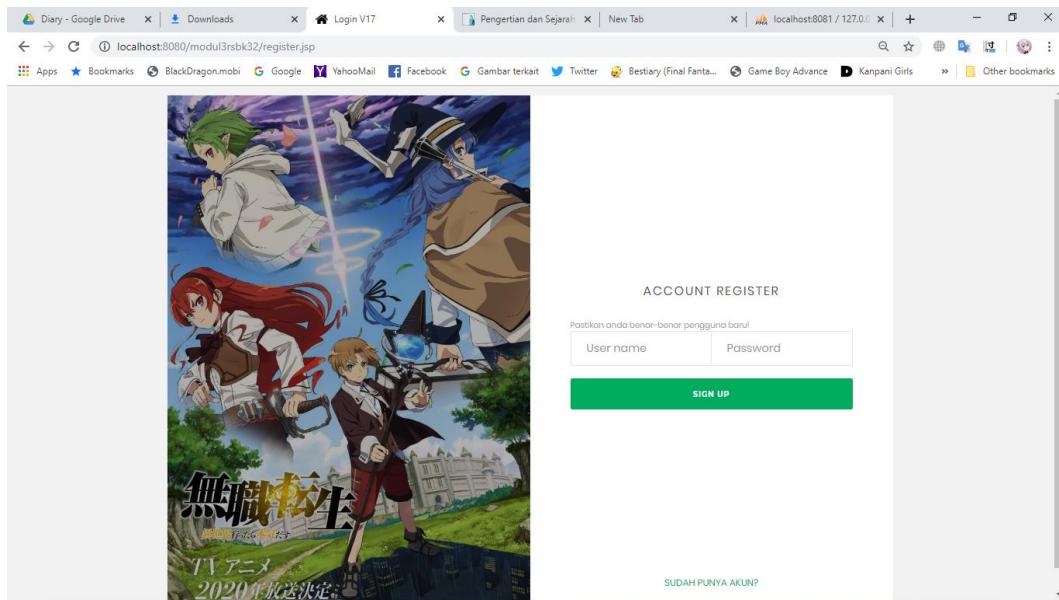
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1">
```

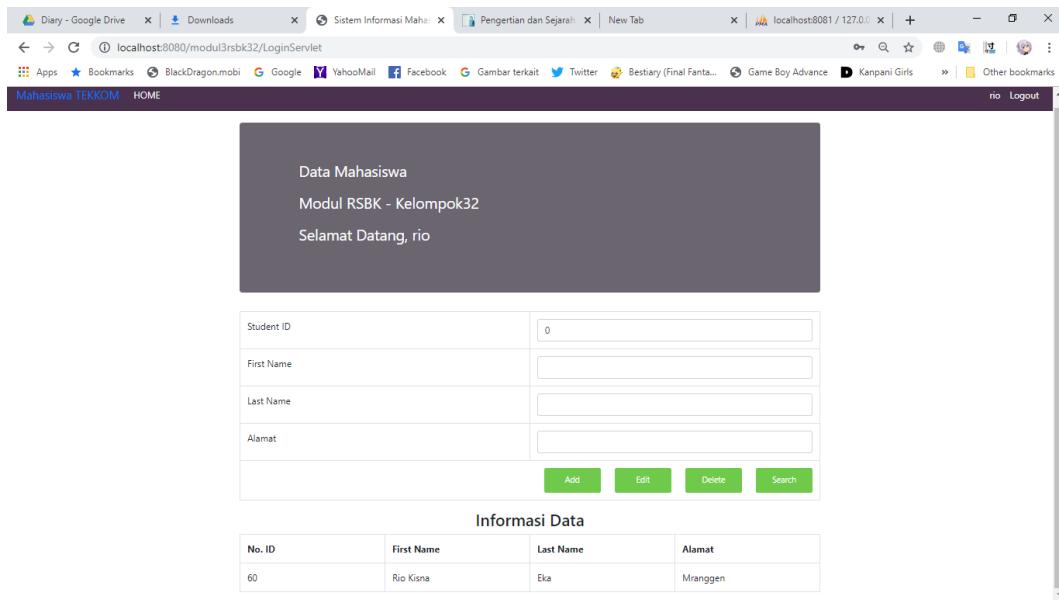
```
<!-- The above 3 meta tags *must* come first in the head; any  
other head content must come *after* these tags -->  
  
<title>404 HTML Tempalte by Colorlib</title>  
  
<!-- Google font -->  
<link  
href="https://fonts.googleapis.com/css?family=Montserrat:400,700,900" rel="stylesheet">  
  
<!-- Custom stlylesheet -->  
<link type="text/css" rel="stylesheet" href="css/style.css" />  
  
<!-- HTML5 shim and Respond.js for IE8 support of HTML5  
elements and media queries -->  
<!-- WARNING: Respond.js doesn't work if you view the page via  
file:// -->  
<!--[if lt IE 9]>  
    <script  
src="https://oss.maxcdn.com/html5shiv/3.7.3/html5shiv.min.js"></script>  
    <script  
src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>  
    <![endif]-->  
  
</head>  
  
<body>  
  
    <div id="notfound">  
        <div class="notfound">  
            <div class="notfound-404">  
                <h1>Oops!</h1>  
            </div>  
            <h2>User tidak terdaftar!</h2>  
            <p>Mohon untuk segera mendaftar kembali.</p>  
            <a href=".//login.jsp">Go To Homepage</a>  
        </div>  
    </div>  
  
</body><!-- This templates was made by Colorlib  
(https://colorlib.com) -->  
  
</html>
```



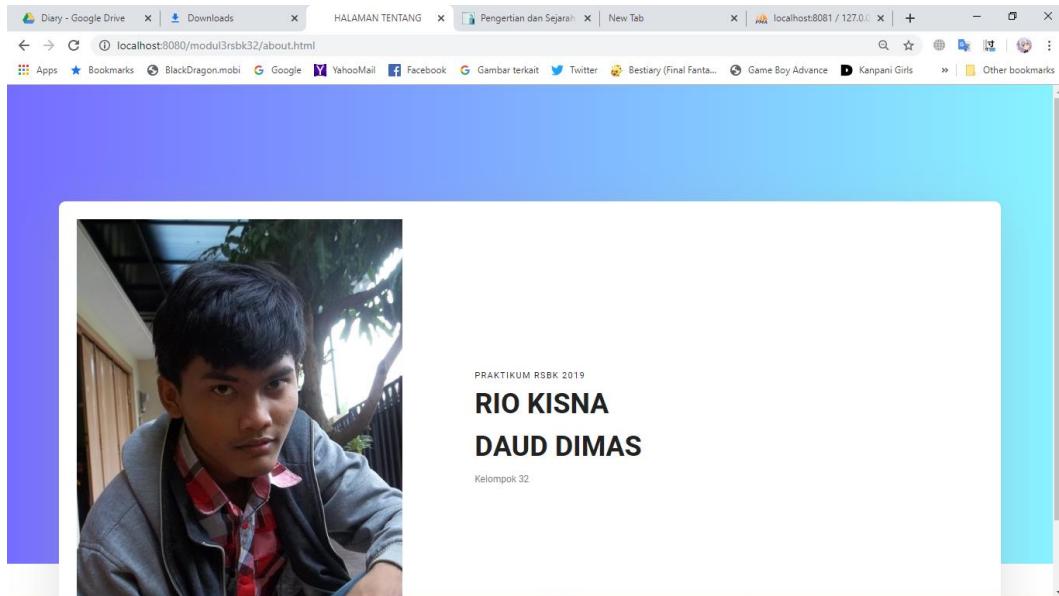
Gambar 4.38 Tampilan *login*



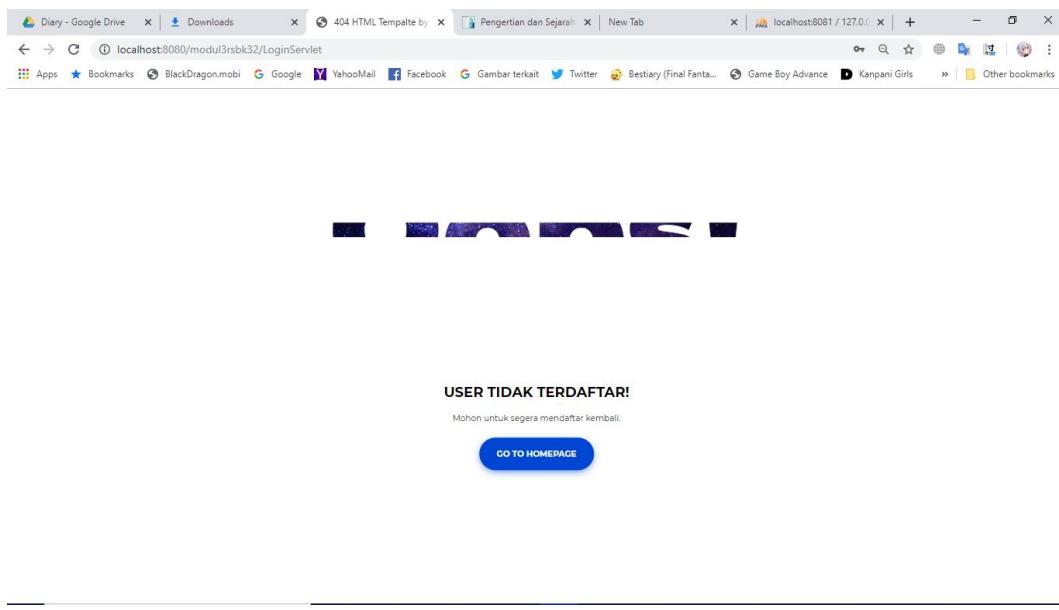
Gambar 4.39 Tampilan *register*



Gambar 4.40 Tampilan Halaman *Home*



Gambar 4.41 Tampilan halaman *About*



Gambar 4.42 Tampilan halaman *error*

**Link GitHub :** <https://github.com/riokisna/Praktikum-RSBK-Kelompok32>

#### 4.6 Kesimpulan

1. Servlet pada praktikum digunakan sebagai *controller* yang menghubungkan antara *view* pada jsp dengan perintah yang terdapat di model.
2. JSP pada praktikum digunakan untuk membuat halaman web yang terdiri dari baris HTML dan fungsi JSP itu sendiri.
3. JPA digunakan untuk menghubungkan aplikasi yang dibuat dengan *database* dan memanipulasi data yang di dalamnya.
4. Entity *Class* berisi atribut dan berfungsi untuk diambil dan diberi nilai pada atribut tersebut.
5. Session Bean berisikan perintah – perintah yang dijalankan di dalam aplikasi, seperti perintah CRUD pada aplikasi, cek *user*, tambah *user*, dan credential *user*.

## **BAB V**

### **JAVA SERVER FACES**

#### **5.1 Tujuan**

1. Praktikan mengenal Framework JSF
2. Praktikan mengetahui penggunaan Java Server Faces.
3. Praktikan mampu membuat aplikasi CRUD sederhana dengan JSF
4. Praktikan dapat menggunakan Apache Tomcat *server* dalam pengembangan web menggunakan Bahasa pemrograman Java.
5. Praktikan dapat mengkoneksikan basis data MySQL dengan Apache Tomcat server.

## 5.2 Dasar Teori

### 5.2.1 NetBeans IDE

NetBeans adalah suatu serambi pengembangan perangkat lunak yang ditulis dalam bahasa pemrograman Java. Serambi Pada NetBeans, pengembangan suatu aplikasi dapat dilakukan dimulai dari setelan perangkat lunak modular bernama modules. Semula, aplikasi NetBeans IDE ini diperuntukkan bagi pengembangan dalam Java. Namun, aplikasi ini juga mendukung program-program pembuatan bahasa lain secara khusus seperti PHP, C/C++ dan HTML5. NetBeans adalah alat lintas serambi serta penerapannya dijalankan pada Microsoft Windows, Mac OS X, Linux, Solaris dan serambi-serambi lainnya yang mendukung JVM yang sepadan.

(Sumber : <https://id.wikipedia.org/wiki/NetBeans> [diakses : 15 November 2019])

### 5.2.2 Java Server Faces

JavaServer Faces (JSF) adalah spesifikasi Java untuk membangun antarmuka pengguna untuk aplikasi web. JSF merupakan bagian dari Java Platform, Enterprise Edition. JSF 2 menggunakan Facelets sebagai sistem templat defaultnya. Teknologi tampilan lainnya, seperti XUL, juga dapat digunakan. Sementara itu, JSF 1.x menggunakan JavaServer Pages (JSP) sebagai sistem templat defaultnya.

Javaserver Faces berdasarkan model perancangan antarmuka pengguna berbasis komponen, menggunakan berkas XML yang disebut templat view atau view Facelets. Permintaan (request) diproses oleh FacesServlet, yang memuat templat view yang sesuai, membangun tree komponen, memproses berbagai event, dan me-render respons (umumnya HTML) kepada klien. State dari komponen UI (dan beberapa objek lain) disimpan pada setiap akhir request (dinamakan stateSaving), dan dikembalikan pada saat pembuatan selanjutnya dari view tersebut. Ada beberapa jenis penyimpanan state, termasuk penyimpanan state client-side dan server-side.

(Sumber : [https://id.wikipedia.org/wiki/JavaServer\\_Faces](https://id.wikipedia.org/wiki/JavaServer_Faces) [diakses : 15 November 2019])

### **5.2.3 Tomcat Server**

Apache Tomcat adalah sebuah web server open source dan servlet container yang dikembangkan oleh Apache Software Foundation (ASF). Tomcat mengimplementasikan Java Servlet dan JavaServer Pages (JSP) dari Oracle dan menyediakan lingkungan server web HTTP “pure Java” untuk menjalankan kode Java. Apache Tomcat mencakup perangkat untuk konfigurasi dan manajemen, tetapi juga dapat dikonfigurasi dengan mengedit file konfigurasi XML.

(Sumber : <https://www.webhozz.com/blog/instalasi-tomcat-di-centos/> [diakses : 15 November 2019])

### **5.2.4 Session Bean**

Session bean adalah EJB yang digunakan untuk mengeksekusi proses. Isi dari Session Bean ini biasanya berupa kata kerja (*transfer, pay, calculate, updateData, dll*). Stateless Session Bean (SLSB) adalah Session Bean yang tidak menyimpan state (keadaan) pada setiap kali eksekusi. Berbeda dengan Statefull Session Bean (SFSB) yang dapat menyimpan state. State ini dapat kita gunakan misalnya untuk menyimpan informasi *user* atau barang-barang yang sudah dibeli (pada kasus online shop).

(Sumber : <https://suhearie.wordpress.com/2008/08/26/java-enterprise-mulai-dari-mana-part-2-netbeans-glassfish/> [diakses : 15 November 2019])

### **5.2.5 MySQL Connector**

Ketika membuat program aplikasi dengan bahasa pemrograman Java yang mengakses ke basis data (database), Anda dapat menggunakan berbagai jenis basis data. Salah satu basis data yang bisa Anda gunakan adalah MySQL. Agar program aplikasi Java dapat terkoneksi ke basis data MySQL, diperlukan JDBC Driver untuk MySQL. JDBC Driver dari MySQL disebut dengan MySQL Connector/J. Sebelum membuat program aplikasi basis data dengan Java, MySQL Connector/J harus sudah

diinstal dengan benar di sistem komputer, bila belum, Anda dapat menggunakan petunjuk instalasi MySQL Connector/J di artikel ini. Bila Anda belum memiliki MySQL JDBC Driver (MySQL Connector/J), Anda dapat mengunduh (download) MySQL JDBC Driver dari website resmi MySQL untuk Connertor/J.

(*Sumber : <https://www.termasmedia.com/pemrograman/java/476-instalasi-mysql-connector-j-koneksi-program-java-ke-database-mysql.html> [diakses : 15 November 2019]*)

### **5.2.6 GitHub**

GitHub adalah layanan penginangan web bersama untuk proyek pengembangan perangkat lunak yang menggunakan sistem pengontrol versi Git dan layanan hosting internet. Hal ini banyak digunakan untuk kode komputer. Ini memberikan kontrol akses dan beberapa fitur kolaborasi seperti pelacakan bug, permintaan fitur, manajemen tugas, dan wiki untuk setiap proyek. GitHub menawarkan paket repositori pribadi dan gratis pada akun yang sama dan digunakan untuk proyek perangkat lunak sumber terbuka. Pada bulan April 2017, GitHub melaporkan bahwa mereka mempunyai lebih dari 20 juta pengguna dan lebih dari 57 juta repositori, menjadikannya layanan terbesar dari kode sumber di dunia.

(*Sumber : <https://id.wikipedia.org/wiki/GitHub> [diakses : 15 November 2019]*)

### **5.2.7 XAMPP**

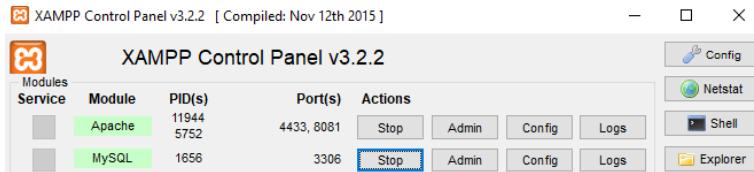
XAMPP adalah perangkat lunak bebas, yang mendukung banyak sistem operasi, merupakan kompilasi dari beberapa program. Fungsinya adalah sebagai server yang berdiri sendiri (localhost), yang terdiri atas program Apache HTTP Server, MySQL database, dan penerjemah bahasa yang ditulis dengan bahasa pemrograman PHP dan Perl. Nama XAMPP merupakan singkatan dari X (empat sistem operasi apapun), Apache, MySQL, PHP dan Perl. Program ini tersedia dalam GNU General Public License dan bebas, merupakan web server yang mudah digunakan yang dapat

melayani tampilan halaman web yang dinamis. Untuk mendapatkannya dapat mendownload langsung dari web resminya.

(*Sumber : <https://id.wikipedia.org/wiki/XAMPP> [diakses : 15 November 2019]*)

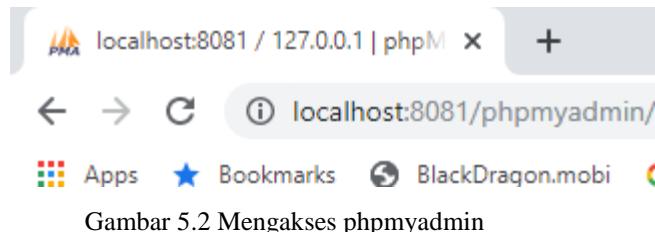
### 5.3 Langkah Kerja

1. Buka XAMPP, lalu Aktifkan Apache dan MySQL



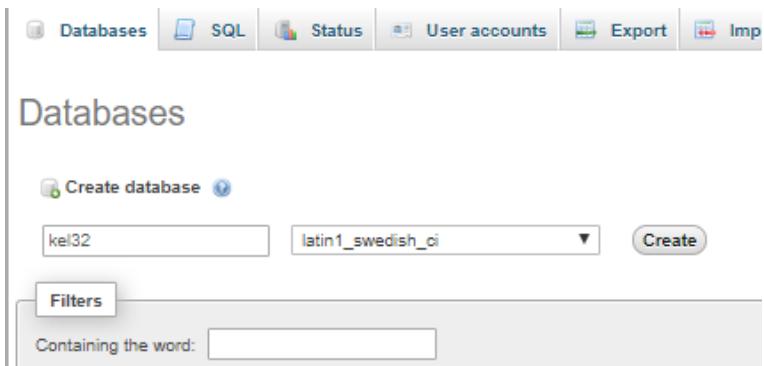
Gambar 5.1 Membuka XAMPP

2. Buka Browser lalu ketikkan : localhost/phpMyAdmin



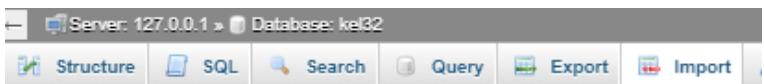
Gambar 5.2 Mengakses phpmyadmin

3. Buat database terlebih dahulu, database tersebut diberi nama kelxx



Gambar 5.3 Membuat Database

4. Kemudian, buka Tab impor



Importing into the database "kel32"

Gambar 5.4 Mengimport ke database baru

5. Lalu, pilih Choose File dan masukkan Database yang sudah disediakan

### Importing into the database "kel32"

File to import:

File may be compressed (gzip, bzip2, zip) or uncompressed.  
A compressed file's name must end in **[format].[compression]**. Example: **.sql.zip**

Browse your computer:  db.sql (Max: 2,048KB)

You may also drag and drop a file on any page.

Character set of the file:

Gambar 5.5 Mengimport file sql

6. Dan kemudian pilih Go

Format-specific options:

SQL compatibility mode:

Do not use AUTO\_INCREMENT for zero values

Gambar 5.6 Mengeksekusi file sql

7. Setelah itu buka tab Hak Akses/ *privilege* dan pilih *add user account*

The screenshot shows the MySQL Workbench interface with the 'Privileges' tab selected. The main pane displays a table titled 'Users having access to "kel32"' with columns: User name, Host name, Type, Privileges, Grant, and Action. There are five rows listed, all with 'ALL PRIVILEGES' and 'Yes' in the 'Grant' column. The 'Action' column contains links for 'Edit privileges' and 'Export'. Below the table are buttons for 'Check all', 'With selected', and 'Export'. At the bottom left of the main panel, there is a red box highlighting the 'Add user account' button.

Gambar 5.7 Membuat user baru terhadap database

8. Setelah masuk ke halaman add user account, ikuti langkah seperti digambar

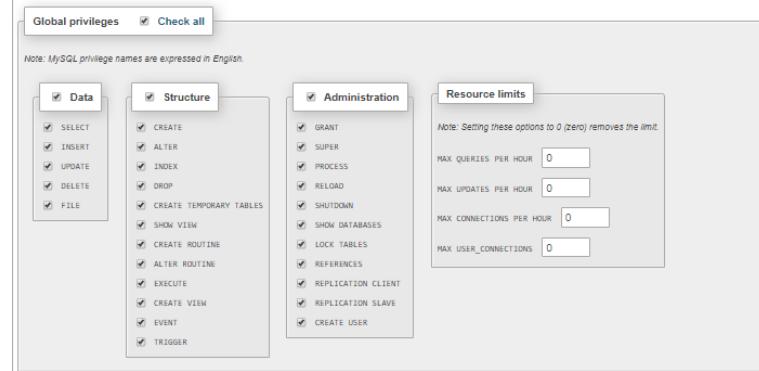
The screenshot shows the 'Add user account' dialog. The 'Login Information' section contains the following fields:  
 - User name: kel32  
 - Host name: Local  
 - Password: \*\*\*\* (Strength: Extremely weak)  
 - Re-type: \*\*\*\*  
 - Authentication Plugin: Native MySQL authentication  
 - Generate password: Generate

Gambar 5.8 Memasukan informasi login

Keterangan:

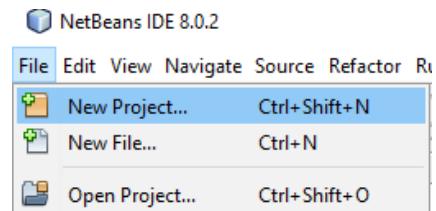
- Nama Pengguna : kelxx
- Nama Pemilik : localhost (form sampingnya harus lokal)
- Kata sandi : kelxx

Setelah itu pilih semua dan kirim



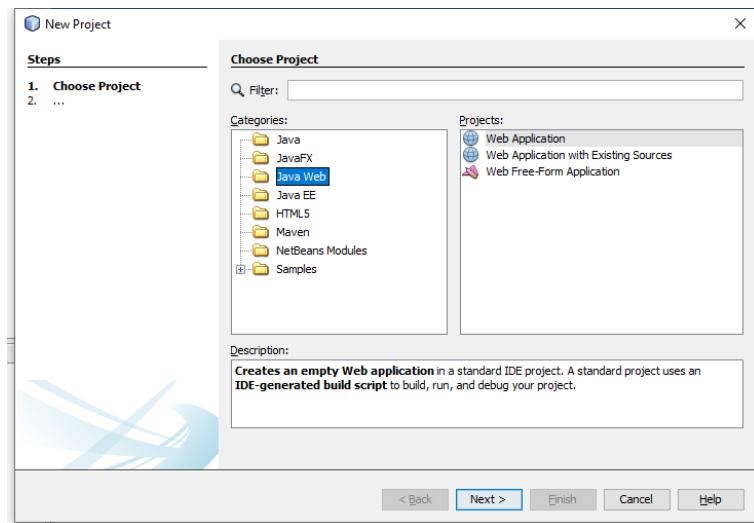
Gambar 5.9 Mengatur hak user

9. Kemudian, buka Netbeans kalian dan pastikan sudah terinstall Tomcat
10. Kemudian pilih File, lalu New Project



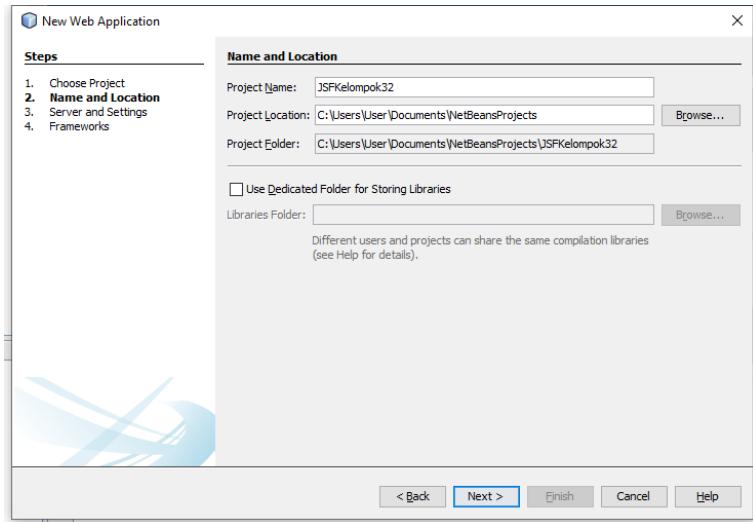
Gambar 5.10 Membuat project baru

11. Lalu pilih Java Web, kemudian pilih Web Application, kemudian Next



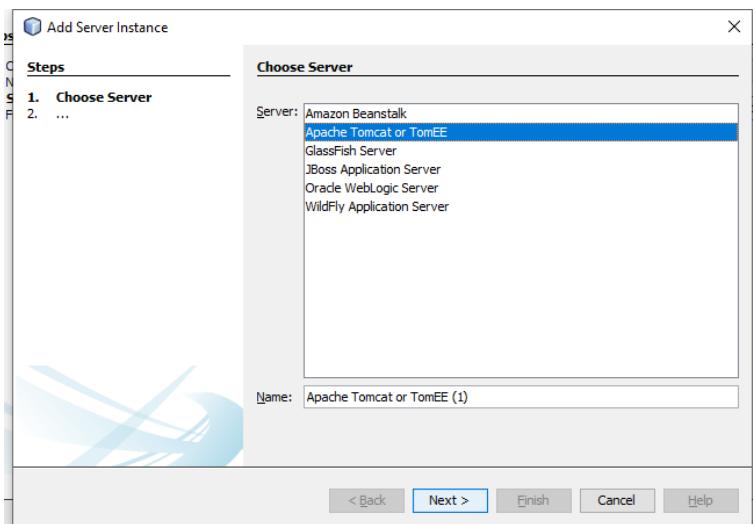
Gambar 5.11 Project Java Web Application

12. Lalu, beri nama ‘JSFKelompokXX’, [ Ganti XX dengan No. Kelompok kalian ], kemudian Next lagi



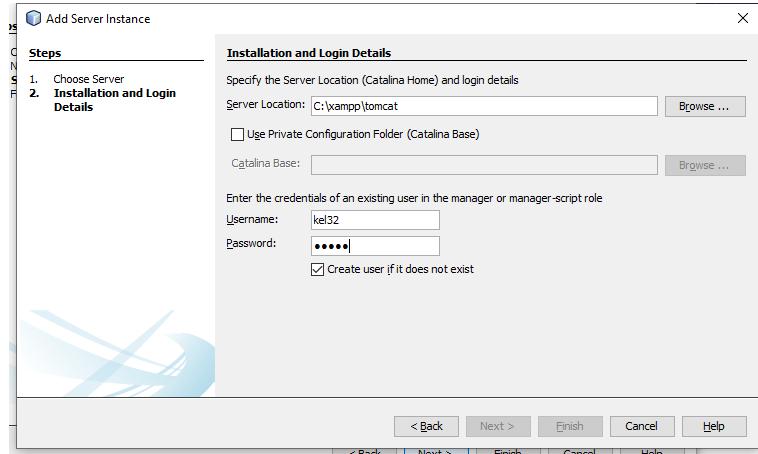
Gambar 5.12 Memberi nama project

13. Kemudian untuk Server, pilih Apache Tomcat or TomEE, dan untuk Java EE Version pilih Java EE 6 Web, jika belum ada tekan tombol add.  
 14. Terus pilih Apache Tomcat or TomEE. next



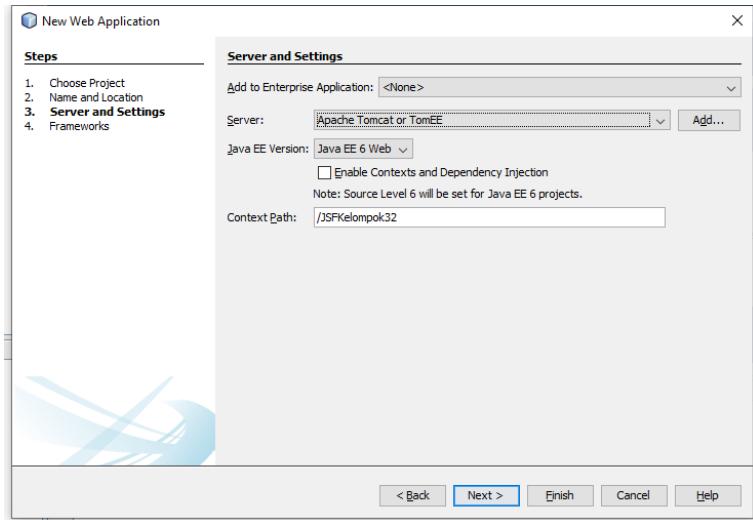
Gambar 5.13 Menambah instance server

15. Maka akan ada form seperti gambar dibawah dan jangan lupa tekan tombol finish



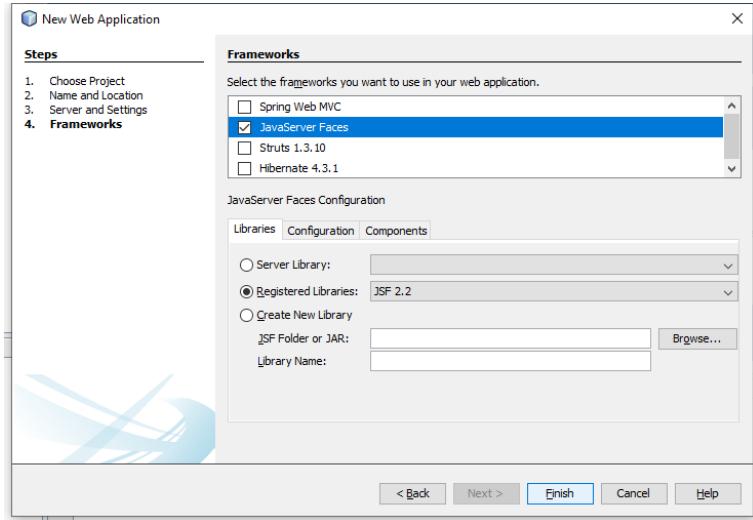
Gambar 5.14 Lokasi instalasi dan detail login server

16. Setelah berhasil menambahkan server Apache Tomcat or TomEE, tekan next



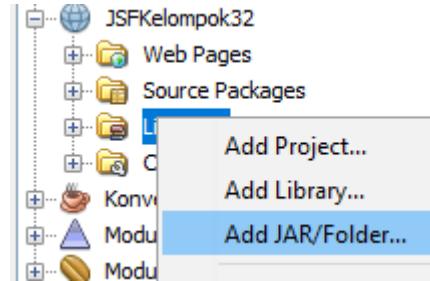
Gambar 5.15 Memilih Apache Tomcat sebagai server

17. Kemudian untuk Framework, centang JavaServer Faces, dan untuk pengaturan ikuti gambar di bawah. Lalu tekan Finish



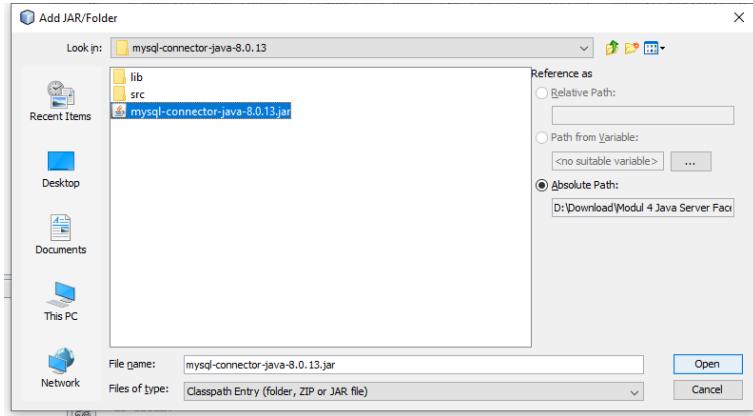
Gambar 5.16 Memilih JSF sebagai *framework*

18. Selanjutnya, tambahkan MySQLConnector dengan cara klik kanan pada Folder Libraries dan pilih Add JAR/Folder



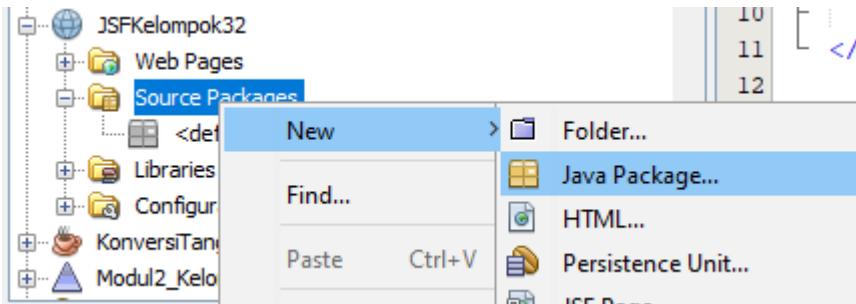
Gambar 5.17 Menambahkan file jar

19. Kemudian, cari lokasi dimana kalian menyimpan MySQL Connectornya, lalu tekan Open



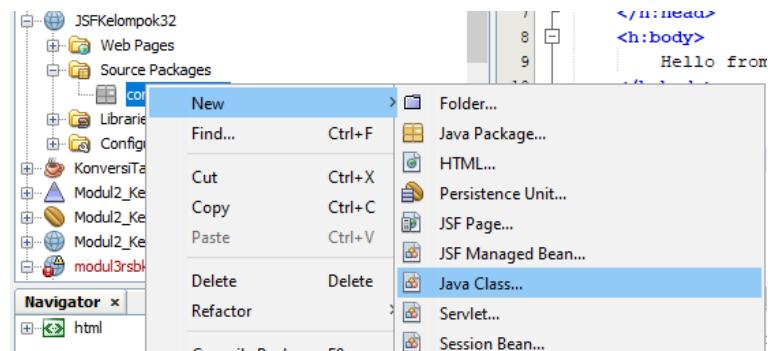
Gambar 5.18 Menambahkan connector

20. Kemudian, pada bagian Source Package, klik kanan dan pilih New, lalu pilih Java Package



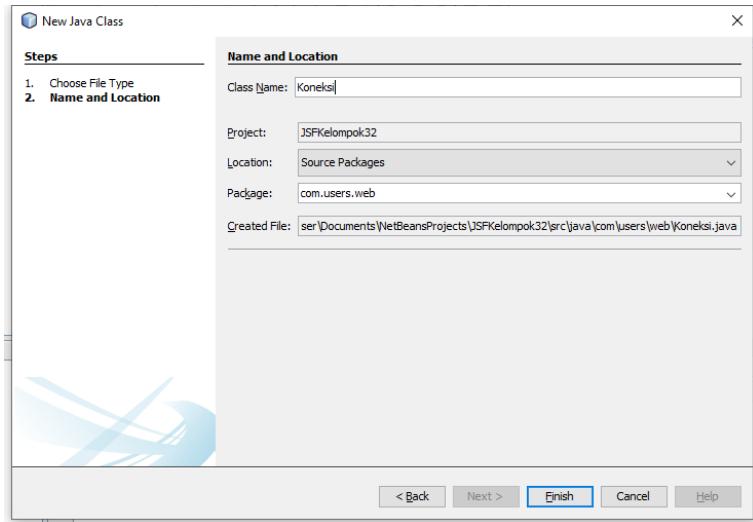
Gambar 5.19 Membuat Java Package baru

21. Lalu beri nama ‘com.users.web’, kemudian klik Finish. Lalu pada Folder com.users.web, klik kanan lalu pilih New dan pilih Java Class



Gambar 5.20 Membuat Java Class baru

22. Kemudian berikan nama ‘Koneksi’, lalu tekan Finish



Gambar 5.21 Membuat class koneksi

23. Lalu, selanjutnya. Salin source code di bawah ke dalam file Koneksi.java. Tepatnya, di dalam perintah public class Koneksi

```

public static void main(String[] args) {
    Koneksi obj_koneksi = new Koneksi();
    System.out.println(obj_koneksi.get_connection());
}

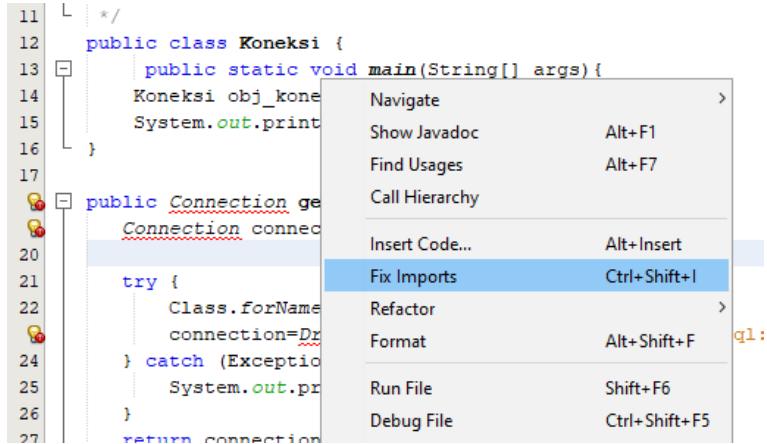
public Connection get_connection() {
    Connection connection = null;

    try {
        Class.forName("com.mysql.cj.jdbc.Driver");

        connection=DriverManager.getConnection("jdbc:mysql://localhost
:3306/rsbk_kelxx", "kelxx", "kelxx");
    } catch (Exception e) {
        System.out.println(e);
    }
    return connection;
}

```

24. Lalu apabila ditemukan Error ketika selesai mengcopy, maka dapat klik kanan pada area kerja, dan pilih Fix Import



Gambar 5.22 Fix import

25. Selanjutnya sesuaikan source code dengan SS-an yang ada diatas dan sesuaikan dengan user hak akses yang sudah di buat sebelumnya

```

    public static void main(String[] args){
        Koneksi obj_koneksi = new Koneksi();
        System.out.println(obj_koneksi.get_connection());
    }

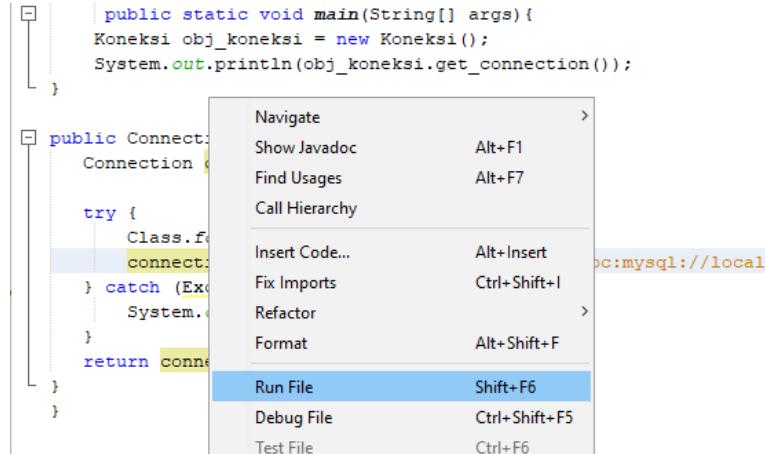
    public Connection get_connection(){
        Connection connection = null;

        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            connection=DriverManager.getConnection("jdbc:mysql://localhost:3306/ke132", "kel32", "kel32");
        } catch (Exception e) {
            System.out.println(e);
        }
        return connection;
    }
}

```

Gambar 5.23 Mengubah informasi login database

26. Kemudian klik kanan pada Area kerja dan pilih Run File untuk memastikan bahwa berhasil koneksi ke Database



Gambar 5.24 Coba run file koneksi

27. Apabila berhasil, maka akan muncul tulisan seperti berikut :

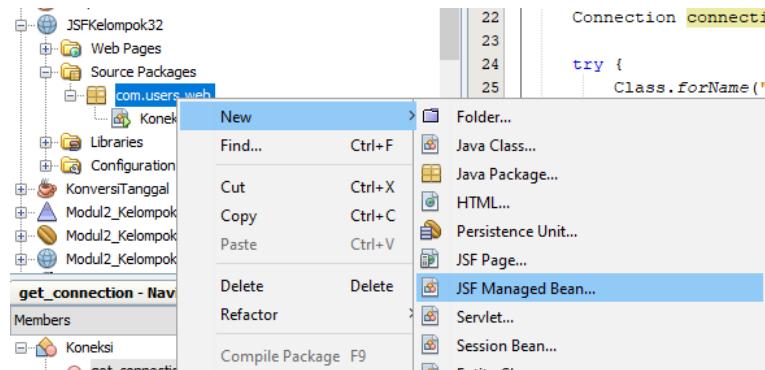
```

run:
com.mysql.cj.jdbc.ConnectionImpl@5315b42e
BUILD SUCCESSFUL (total time: 1 second)

```

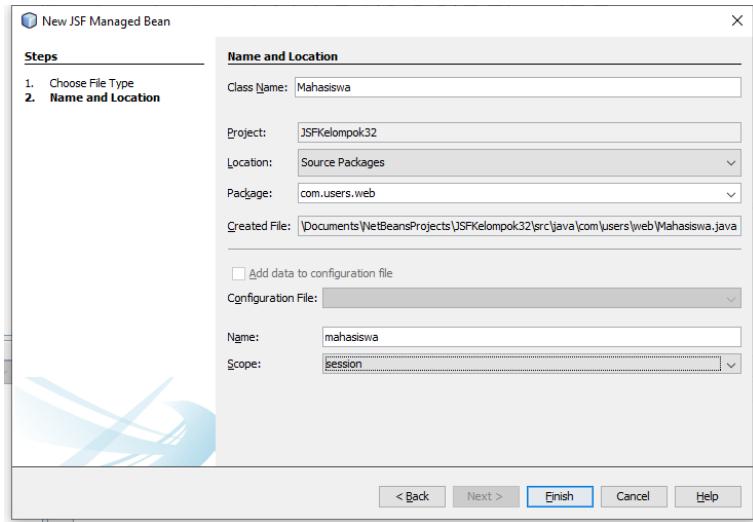
Gambar 5.25 Koneksi berhasil dibuat

28. Selanjutnya, klik kanan lagi pada ;com.users.web;. Lalu pilih New, dan pilih JSF Managed Bean. Jika pilihan tidak tersedia klik pilihan “Other”



Gambar 5.26 Membuat JSF Managed Bean baru

29. Lalu beri nama Mahasiswa, dan pada bagian Scope. Ubah menjadi Session. Kemudian klik Finish



Gambar 5.27 Membuat JSF Managed Bean Mahasiswa

30. Selanjutnya, pada Area Kerja ‘Mahasiswa.java’ yang terletak dibawah tulisan ‘package com.users.web’, isi dengan source code di bawah ini.

```

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.Map;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.RequestScoped;
import javax.faces.context.FacesContext;

/**
 *
 * @author Hakushuu
 */
@ManagedBean
@RequestScoped
public class Mahasiswa {

    /**
     * Creates a new instance of Mahasiswa
     */

    private String NIM;
    public void setNIM(String NIM) {
        this.NIM = NIM;
    }
}

```

```

public String getNIM() {
    return NIM;
}

private String NAMA;
public void setNAMA(String NAMA) {
    this.NAMA = NAMA;
}
public String getNAMA() {
    return NAMA;
}

private String PENJURUSAN;
public void setPENJURUSAN(String PENJURUSAN) {
    this.PENJURUSAN = PENJURUSAN;
}
public String getPENJURUSAN() {
    return PENJURUSAN;
}

private Map<String, Object> sessionMap =
FacesContext.getCurrentInstance().getExternalContext().getSessionMap();

public String Edit_Mahasiswa(){
    FacesContext fc = FacesContext.getCurrentInstance();
    Map<String, String> params =
fc.getExternalContext().getRequestParameterMap();
    String Field_NIM = params.get("action");
    try {
        Koneksi obj_koneksi = new Koneksi();
        Connection connection = obj_koneksi.get_connection();
        Statement st = connection.createStatement();
        ResultSet rs = st.executeQuery("select * from
mahasiswa where NIM='"+Field_NIM+"'");
        Mahasiswa obj_Mahasiswa = new Mahasiswa();
        rs.next();
        obj_Mahasiswa.setNIM(rs.getString("NIM"));
        obj_Mahasiswa.setNAMA(rs.getString("Nama"));

        obj_Mahasiswa.setPENJURUSAN(rs.getString("Penjurusan"));
        sessionMap.put("EditMahasiswa", obj_Mahasiswa);
    } catch (Exception e) {
        System.out.println(e);
    }
    return "/Edit.xhtml?faces-redirect=true";
}

public String Delete_Mahasiswa(){
    FacesContext fc = FacesContext.getCurrentInstance();
    Map<String, String> params =
fc.getExternalContext().getRequestParameterMap();

```

```

String Field_NIM = params.get("action");
try {
    Koneksi obj_koneksi = new Koneksi();
    Connection connection = obj_koneksi.get_connection();
    PreparedStatement ps =
connection.prepareStatement("delete from mahasiswa where
NIM=?");
    ps.setString(1, Field_NIM);
    System.out.println(ps);
    ps.executeUpdate();
} catch (Exception e) {
    System.out.println(e);
}
return "/index.xhtml?faces-redirect=true";
}

public String Update_Mahasiswa(){
FacesContext fc = FacesContext.getCurrentInstance();
Map<String, String> params =
fc.getExternalContext().getRequestParameterMap();
String Update_NIM= params.get("Update_NIM");

try {
    Koneksi obj_koneksi = new Koneksi();
    Connection connection = obj_koneksi.get_connection();
    PreparedStatement ps =
connection.prepareStatement("update mahasiswa set NIM=?, Nama=?, Penjurusan=? where NIM=?");
    ps.setString(1, NIM);
    ps.setString(2, NAMA);
    ps.setString(3, PENJURUSAN);
    ps.setString(4, Update_NIM);
    System.out.println(ps);
    ps.executeUpdate();
} catch (Exception e) {
    System.out.println(e);
}
return "/index.xhtml?faces-redirect=true";
}

public ArrayList getGet_all_mahasiswa() throws Exception{
ArrayList list_of_mahasiswa=new ArrayList();
Connection connection=null;
try {
    Koneksi obj_koneksi = new Koneksi();
    connection = obj_koneksi.get_connection();
    Statement st = connection.createStatement();
    ResultSet rs = st.executeQuery("Select * from
mahasiswa");
    while(rs.next()){
        Mahasiswa obj_Mahasiswa = new Mahasiswa();
}
}
}

```

```

        obj_Mahasiswa.setNIM(rs.getString("NIM"));
        obj_Mahasiswa.setNAMA(rs.getString("Nama"));

obj_Mahasiswa.setPENJURUSAN(rs.getString("Penjurusan"));
    list_of_mahasiswa.add(obj_Mahasiswa);
}
} catch (Exception e) {
    System.out.println(e);
} finally{
    if(connection!=null){
        connection.close();
    }
}
return list_of_mahasiswa;
}

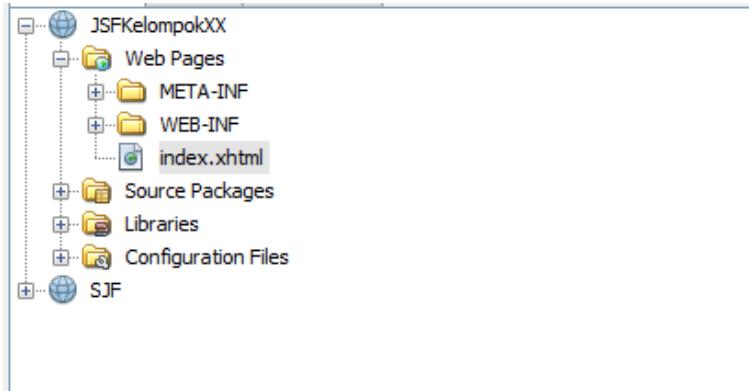
public String Tambah_Mahasiswa(){
try {
    Connection connection=null;
    Koneksi obj_koneksi = new Koneksi();
    connection = obj_koneksi.get_connection();
    PreparedStatement
ps=connection.prepareStatement("insert into mahasiswa(NIM,
Nama,
Penjurusan)
value ('"+NIM+"','"+Nama+"','"+PENJURUSAN+"')");
    ps.executeUpdate();
} catch (Exception e) {
    System.out.println(e);
}
return "/index.xhtml?faces-redirect=true";
}

public Mahasiswa() {}

}

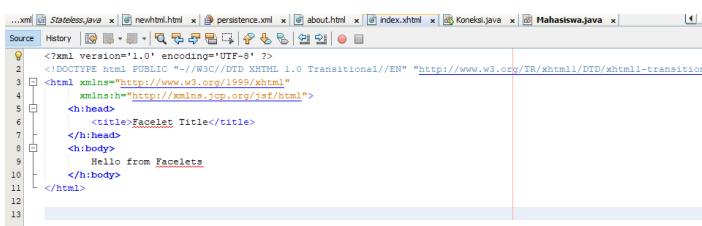
```

31. Selanjutnya, Expand bagian Web Pages, lalu kalian akan melihat file ‘index.xhtml’. Kemudian klik 2 kali pada File tersebut



Gambar 5.28 File index.xhtml

32. Lalu akan tampil, tampilan seperti berikut



Gambar 5.29 isi file index.xhtml

33. Lalu salin source code di bawah ke dalam file ‘index.xhtml’ tersebut

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://xmlns.jcp.org/jsf/html">
    <head>
        <title>JSF RSBK</title>
    </head>
    <h:body>

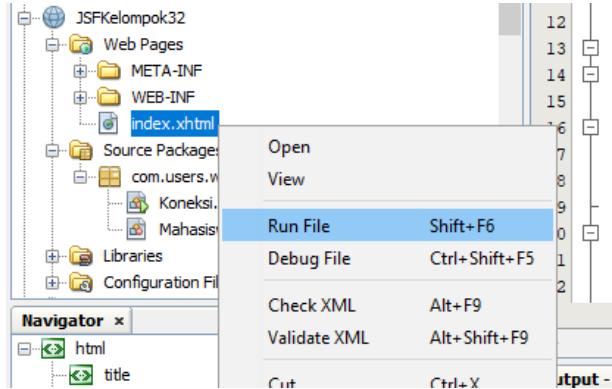
        <center>
            <h2>Daftar Mahasiswa</h2>
            <h:form>
                <h:dataTable
value="#{mahasiswa.get_all_mahasiswa}"                      var="maha"
width="800px">
                    <h:outputText value="#{mahasiswa.NAMA}" />
                    <h:column>
                        <f:facet name="header">NIM</f:facet>
                        <h:outputText
value="#{maha.NIM}"></h:outputText>
                    </h:column>
```

```

        <h:column>
            <f:facet name="header">Nama</f:facet>
            <h:outputText
value="#{maha.NAMA}"></h:outputText>
        </h:column>
        <h:column>
            <f:facet name="header">Penjurusan</f:facet>
            <h:outputText
value="#{maha.PENJURUSAN}"></h:outputText>
        </h:column>
        <h:column>
            <f:facet name="header">Edit</f:facet>
            <h:commandButton
action="#{mahasiswa.Edit_Mahasiswa}">
                <f:param
value="#{maha.NIM}" />
                </h:commandButton>
            </h:column>
            <h:column>
                <f:facet name="header">Delete</f:facet>
                <h:commandButton
action="#{mahasiswa.Delete_Mahasiswa}">
                <f:param
value="#{maha.NIM}" />
                </h:commandButton>
            </h:column>
        </h:dataTable>
        <br></br>
    </h:form>
    <h:form>
        <h:form>
            Tambahkan Data Mahasiswa &nbsp;
            <h:commandButton
action="Tambah.xhtml"></h:commandButton>
        </h:form>
    </h:form>
</center>
</h:body>
</html>

```

34. Kemudian klik kanan pada file ‘Index.xhtml’ lalu pilih Run File

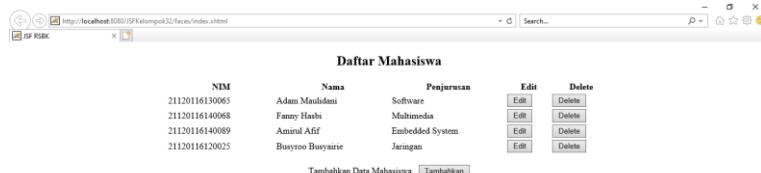


Gambar 5.30 Mencoba run file index.xhtml

Jika terjadi suatu kesalahan / error “port is used” gunakan perintah

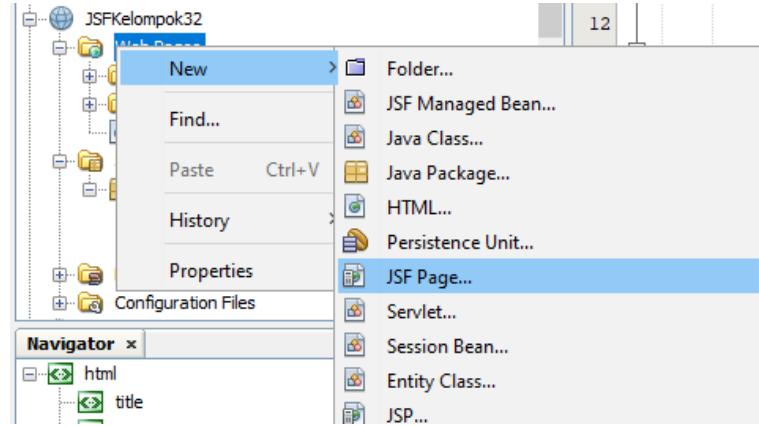
```
Run cmd as administrator
netstat -ano | findstr :8080
taskkill /PID 3748 /F
# 3748 adalah process id
```

35. Maka, akan muncul tampilan sebagai berikut pada browser kalian :



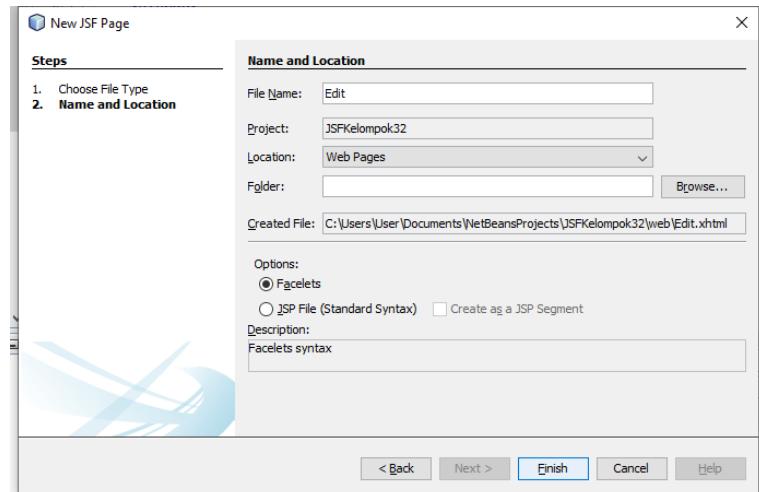
Gambar 5.31 Hasil run file index.xhtml

36. Pada tampilan tersebut, button Edit dan Tambahkan, belum dapat digunakan, maka kita lanjut ke langkah selanjutnya
37. Kembali lagi ke Netbeans kalian, lalu klik kanan pada Folder Web Pages, lalu pilih New, dan pilih JSF Page

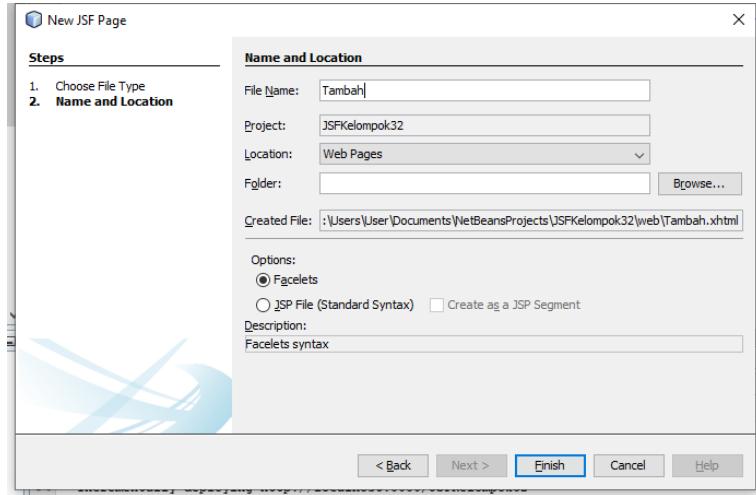


Gambar 5.32 Membuat JSF Page baru

38. Buat 2 buah File JSF Page, dengan nama ‘Edit’ dan ‘Tambah’



Gambar 5.33 Membuat JSF Page Edit



Gambar 5.34 Membuat JSF Page Tambah

39. Kemudian salin source code di bawah ke dalam file ‘Edit.xhtml’.

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
xmlns:f="http://java.sun.com/jsf/core"
xmlns:h="http://xmlns.jcp.org/jsf/html">
<h:head>
<title>JSF RSBK</title>
</h:head>
<h:body>
<center>
<h:form>
<h1>Edit Data Mahasiswa</h1>

<table>
<tr>
<td>NIM :</td>
<td><h:inputText value="#{EditMahasiswa.NIM}"></h:inputText></td>
</tr>
<tr>
<td>Nama Mahasiswa :</td>
<td><h:inputText value="#{EditMahasiswa.NAMA}"></h:inputText></td>
</tr>
<tr>
<td>Penjurusan :</td>
<td><h:inputText value="#{EditMahasiswa.PENJURUSAN}"></h:inputText></td>
</tr>
</table>
<br><br>
```

```

<h:commandButton value="Update"
action="#{EditMahasiswa.Update_Mahasiswa}">
    <f:param name="Update_NIM" value="#{EditMahasiswa.NIM}" /></h:commandButton>
    <br><br>
    <br><br>
    </h:form>
    <h:form>
        <h:commandButton value="Kembali"
action="index.xhtml"></h:commandButton>
    </h:form>

    </center>
</h:body>
</html>

```

40. Dan juga salin source code berikut ke dalam file ‘Tambah.xhtml’.

```

<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html">
    <h:head>
        <title>JSF RSBK</title>
    </h:head>
    <h:body>
        <center>
            <h2>Tambah Data Mahasiswa</h2>

            <h:form>
                <table>
                    <tr>
                        <td>NIM :</td>
                        <td><h:inputText value="#{mahasiswa.NIM}"></h:inputText></td>
                    </tr>
                    <tr>
                        <td>Nama Mahasiswa :</td>
                        <td><h:inputText value="#{mahasiswa.NAMA}"></h:inputText></td>
                    </tr>
                    <tr>
                        <td>Penjurusan :</td>
                        <td><h:inputText value="#{mahasiswa.PENJURUSAN}"></h:inputText></td>
                    </tr>
                </table>
                <br><br>
                <h:commandButton value="Tambahkan"
action="#{mahasiswa.Tambah_Mahasiswa}"></h:commandButton>

```

```
<br></br>
<br></br>
</h:form>
    <h:form>
        <h:commandButton value="Kembali"
action="index.xhtml"></h:commandButton>
    </h:form>
</center>

</h:body>
</html>
```

41. Kemudian, coba kalian lakukan lagi Run File ‘index.xhtml’ . Dan, coba tekan button Edit atau Tambahkan. Maka button tersebut kini dapat digunakan

## 5.4 Hasil dan Analisa



The screenshot shows a table titled "Daftar Mahasiswa" with the following data:

| NIM            | Nama              | Penjurusan      | Edit                 | Delete                 |
|----------------|-------------------|-----------------|----------------------|------------------------|
| 21120116130065 | Adam Maulidani    | Software        | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116140068 | Fanny Haibi       | Multimedia      | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116140089 | Amral Afif        | Embedded System | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120025 | Basyroo Buuyaarie | Jaringan        | <a href="#">Edit</a> | <a href="#">Delete</a> |

Tambahkan Data Mahasiswa [\[Tambahkan\]](#)

Gambar 5.35 Hasil praktikum halaman index

Hasil praktikum pada halaman index akan seperti itu. Sebuah tabel di tampilkan berdasarkan data yang diambil dari basis data melalui fungsi `get_all_mahasiswa()`. Kemudian juga ada button Edit, delete, dan tambahkan data. Ketika button edit atau delete di tekan, maka akan mengambil dan mengirim sebuah parameter NIM, untuk di eksekusi lebih lanjut pada fungsi lainnya.



The screenshot shows a form titled "Tambah Data Mahasiswa" with the following fields:

- NIM :
- Nama Mahasiswa :
- Penjurusan :

Tombol: [\[Tambahkan\]](#) [\[Kembali\]](#)

Gambar 5.36 Hasil praktikum tambah data

Pada halaman tambah data, terdapat textBox yang dapat diisi dengan nilai untuk memasukan informasi. Informasi tersebut nantinya akan di baca dan di kirimkan ke fungsi Tambah\_mahasiswa() untuk di eksekusi menggunakan query ke database.

| NIM  | Nama                | Pengurusan      | Edit                 | Delete                 |
|--|---------------------|-----------------|----------------------|------------------------|
| 21120116130065                                       | Adam Maulidani      | Software        | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116140068                                       | Fanny Habsi         | Multimedya      | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116140089                                       | Amirul Afif         | Embedded System | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120025                                       | Basyroo Basyarie    | Jaringan        | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116130060                                       | Rio Kenna Eka Putra | Software        | <a href="#">Edit</a> | <a href="#">Delete</a> |
| Tambahkan Data Mahasiswa <a href="#">[Tambahkan]</a> |                     |                 |                      |                        |

Gambar 5.37 Hasil praktikum berhasil menambah data

Ketika data ditambahkan, maka akan terdapat data baru sesuai data yang ditambahkan sebelumnya.

| Edit Data Mahasiswa     |  |
|-------------------------|--|
| NIM :                   | <input type="text" value="21120116130060"/>      |
| Nama Mahasiswa :        | <input type="text" value="Rio Kenna Eka Putra"/> |
| Pengurusan :            | <input type="text" value="Software"/>            |
| <a href="#">Update</a>  |  |
| <a href="#">Kembali</a> |  |

Gambar 5.38 Hasil praktikum edit mahasiswa

Untuk halaman edit data diakses dengan menekan button edit pada halaman index. Pada button tersebut, Facelet akan menyimpan nilai pada parameter berupa objek mahasiswa. Nim tersebut akan di gunakan untuk mengakses data pada tabel

mahasiswa kemudian menulikannya ke textBox yang sudah di buat. Fungsi yang di eksekusi ketika membuka halaman ini adalah fungsi Edit\_Mahasiswa().

NIM :

Nama Mahasiswa :

Penjurusan :

Gambar 5.39 Hasil praktikum halaman edit mahasiswa

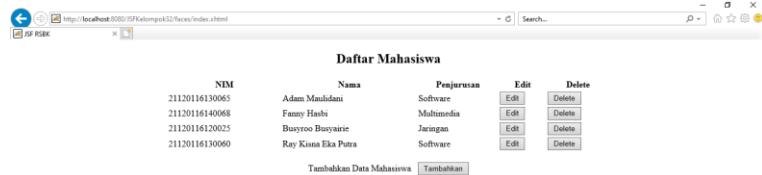
Ketika button update di tekan, maka program akan menjalankan fungsi Update\_Mahasiswa(). Fungsi tersebut berguna untuk mengupdate pada basis data sesuai dengan data yang di ubah. Kemudian dengan menggunakan nilai NIM yang juga disimpan sebagai parameter, maka data sesuai dengan NIM lama akan diubah dengan data yang baru.

| NIM            | Nama                | Penjurusan      | Edit                                | Delete                                |
|----------------|---------------------|-----------------|-------------------------------------|---------------------------------------|
| 21120116130065 | Adam Maulidani      | Software        | <input type="button" value="Edit"/> | <input type="button" value="Delete"/> |
| 21120116140068 | Fanny Hashi         | Multimedia      | <input type="button" value="Edit"/> | <input type="button" value="Delete"/> |
| 21120116140089 | Amirul Arif         | Embedded System | <input type="button" value="Edit"/> | <input type="button" value="Delete"/> |
| 21120116120025 | Busyro Buayarie     | Jaringan        | <input type="button" value="Edit"/> | <input type="button" value="Delete"/> |
| 21120116130060 | Ray Kisna Eka Putra | Software        | <input type="button" value="Edit"/> | <input type="button" value="Delete"/> |

Tambahkan Data Mahasiswa

Gambar 5.40 Hasil praktikum edit mahasiswa berhasil

Terlihat bahwa pada nama berubah.



The screenshot shows a web browser window with the URL <http://localhost:8000/SFKelompok22/faces/index.xhtml>. The page title is "Daftar Mahasiswa". A table lists five students with columns for NIM, Name, Major, Edit, and Delete. The "Edit" and "Delete" buttons are shown as hyperlinks. At the bottom left is a "Tambahkan Data Mahasiswa" button, and at the bottom right is a "Tombol Batal" button.

| NIM            | Nama               | Penjurusan | Edit                 | Delete                 |
|----------------|--------------------|------------|----------------------|------------------------|
| 21120116130065 | Adam Maulidani     | Software   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116140068 | Fanny Habsi        | Multimedia | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120025 | Busyro Busyurie    | Jaringan   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116130060 | Ray Kina Eka Putra | Software   | <a href="#">Edit</a> | <a href="#">Delete</a> |

Gambar 5.41 Hasil praktikum delete mahasiswa berhasil

Kemudian untuk fungsi delete, sama dengan fungsi edit dengan mengambil parameter NIM, kemudian menjalankan fungsi Delete\_Mahasiswa() untuk mengeksekusi query delete berdasarkan parameter NIM yang tersimpan. Sehingga data tersebut akan langsung terhapus

## 5.5 Tugas dan Pembahasan

Tugas diminta untuk menambahkan tabel baru yaitu tabel penjurusan, setelah itu menghubungkannya dengan tabel mahasiswa. Ketika melakukan tambah data atau edit data dan salah memasukan penjurusan, maka akan menampilkan pesan error.

Pertama kali yang dilakukan adalah menambahkan tabel baru, yaitu tabel penjurusan. Berikut ini adalah tabel penjurusan yang kami buat

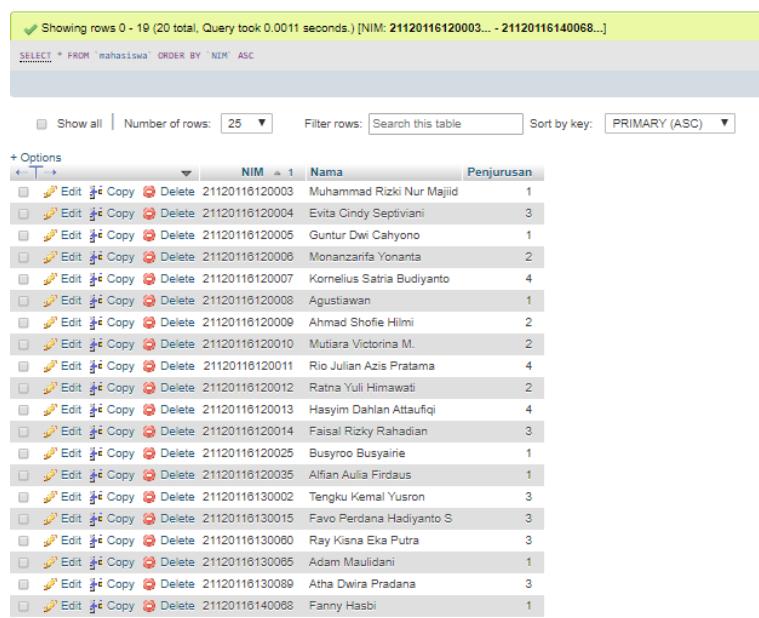
The screenshot shows the MySQL Workbench interface with the 'Browse' tab selected. At the top, there are tabs for 'Browse', 'Structure', 'SQL', and 'Search'. Below the tabs, a green status bar displays: 'Showing rows 0 - 3 (4 total, Query took 0.0010 sec)'. The SQL query shown is 'SELECT \* FROM `penjurusan`'. The main area shows a table with the following data:

|                          | id | penjurusan |
|--------------------------|----|------------|
| <input type="checkbox"/> | 1  | Software   |
| <input type="checkbox"/> | 2  | Jaringan   |
| <input type="checkbox"/> | 3  | Multimedia |
| <input type="checkbox"/> | 4  | Embedded   |

Below the table, there are buttons for 'Show all', 'Number of rows: 25', and 'Filter'. On the left side of the table, there are 'Edit', 'Copy', and 'Delete' buttons for each row, along with a 'T' icon for sorting.

Gambar 5.42 Tabel penjurusan

Karena primary key yang di gunakan pada tabel tersebut adalah ID, untuk membuatnya dapat dihubungkan dengan tabel mahasiswa, perlu dilakukan pengubahan struktur tabel mahasiswa dimana penjurusan pada tabel itu dijadikan sebagai id nya.



The screenshot shows a MySQL Workbench interface with the following details:

- Query Bar:** Shows the query: "SELECT \* FROM 'mahasiswa' ORDER BY 'NIM' ASC".
- Table View:** The 'mahasiswa' table has columns: NIM, Nama, and Penjurusan.
- Data Rows:** There are 20 rows of data, each with an edit, copy, and delete icon. The data includes:
 

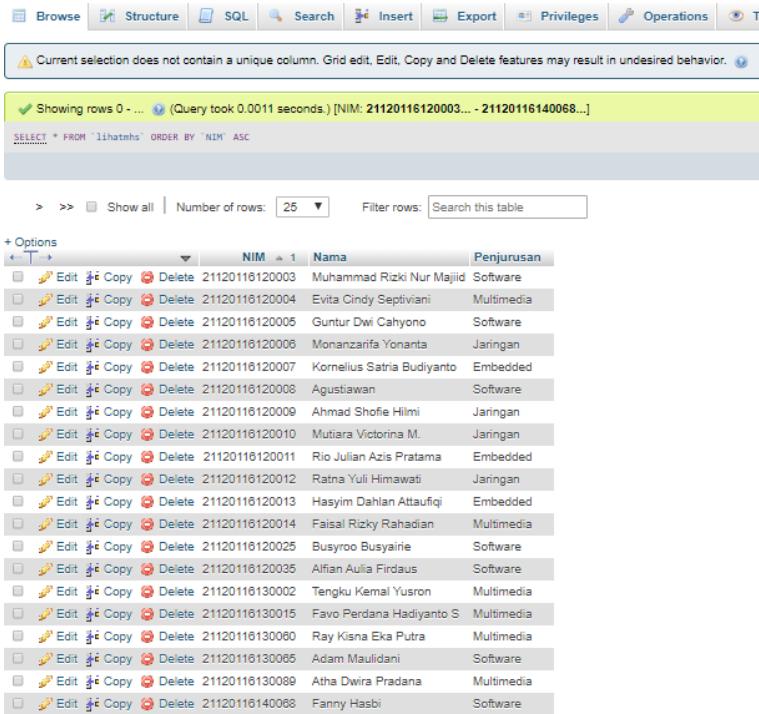
| NIM            | Nama                       | Penjurusan |
|----------------|----------------------------|------------|
| 21120116120003 | Muhammad Rizki Nur Majid   | 1          |
| 21120116120004 | Evita Cindy Septiviani     | 3          |
| 21120116120005 | Guntur Dwi Cahyono         | 1          |
| 21120116120006 | Monanzanifa Yonanta        | 2          |
| 21120116120007 | Kornelius Satria Budiyanto | 4          |
| 21120116120008 | Agustiawan                 | 1          |
| 21120116120009 | Ahmad Shofie Hilmi         | 2          |
| 21120116120010 | Mutiara Victoria M.        | 2          |
| 21120116120011 | Rio Julian Azis Pratama    | 4          |
| 21120116120012 | Ratna Yuli Himawati        | 2          |
| 21120116120013 | Hasyim Dahlan Attaufiq     | 4          |
| 21120116120014 | Faisal Rizky Rahadian      | 3          |
| 21120116120025 | Busyroo Busyairie          | 1          |
| 21120116120035 | Alian Aulia Firdaus        | 1          |
| 21120116130002 | Tengku Kemal Yusron        | 3          |
| 21120116130015 | Favo Perdana Hadiyanto S   | 3          |
| 21120116130060 | Ray Kisna Eka Putra        | 3          |
| 21120116130065 | Adam Maulidani             | 1          |
| 21120116130089 | Atha Dwira Pradana         | 3          |
| 21120116140088 | Fanny Hasbi                | 1          |

Gambar 5.43 Tabel mahasiswa baru  
Setelah itu melakukan constraint antara tabel mahasiswa dengan tabel penjurusan



Gambar 5.44 Constraint foreign key

Setelah di constraint agar data yang ditampilkan pada tabel di index.xhtml tetap seperti semula, perlu dibuat sebuah view untuk menghasilkan tampilan hasil join kedua tabel, kemudian mengubah semua select mahasiswa menjadi select view tersebut. Berikut ini adalah view yang dimaksud



|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | NIM            | Nama                      | Penjurusan |
|--|--------------------------|-------------------------------------|-------------------------------------|---------------------------------------|----------------|---------------------------|------------|
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116120003 | Muhammad Rizki Nur Majid  | Software   |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116120004 | Evita Cindy Septiviani    | Multimedia |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116120005 | Guntur Dwi Cahyono        | Software   |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116120006 | Monanzarifa Yonanta       | Jaringan   |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116120007 | Kornelius Satia Budiyanto | Embedded   |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116120008 | Agustiawan                | Software   |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116120009 | Ahmad Shofie Hilmi        | Jaringan   |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116120010 | Mutiara Victorina M.      | Jaringan   |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116120011 | Rio Julian Azis Pratama   | Embedded   |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116120012 | Ratna Yuli Hinawati       | Jaringan   |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116120013 | Hasyim Dahlan Attaufiqi   | Embedded   |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116120014 | Faisal Rizky Rahadian     | Multimedia |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116120025 | Busyroo Busyairie         | Software   |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116120035 | Alfian Aulia Firdaus      | Software   |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116130002 | Tengku Kemal Yusron       | Multimedia |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116130015 | Favo Perdana Hadiyanto S  | Multimedia |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 2112011613008  | Ray Kisona Eka Putra      | Multimedia |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116130065 | Adam Maulidani            | Software   |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116130089 | Atha Dwira Pradana        | Multimedia |
|  | <input type="checkbox"/> | <input type="button" value="Edit"/> | <input type="button" value="Copy"/> | <input type="button" value="Delete"/> | 21120116140068 | Fanny Hasbi               | Software   |

Gambar 5.45 tabel view

Basis data telah di siapkan, selanjutnya adalah memberi pengkondisian baru pada source code Mahasiswa.java untuk proses penambahan data dan pengubahan data. Pengkondisian tersebut digunakan untuk mengecek apakan penjurusan yang di maksud sudah ada pada tabel penjurusan. Ketika ada maka data berhasil ditambahkan, namun ketika tidak ada maka data tidak jadi ditambahkan dan akan menampilkan pesan error.

#### Source code pada fungsi tambah Mahasiswa

```
public String Tambah_Mahasiswa () {
    try {
        Connection connection=null;
        Koneksi obj_koneksi = new Koneksi();
        connection = obj_koneksi.get_connection();
        Statement st = connection.createStatement();
        ResultSet rs = st.executeQuery("Select * from
penjurusan where penjurusan = '"+PENJURUSAN+"'");
        String id=null;
        while(rs.next()){
            id = rs.getString("id");
        }
    }
}
```

```

        }
        if (id!=null){
            PreparedStatement
ps=connection.prepareStatement("insert into mahasiswa(NIM, Nama,
Penjurusan) value('"+NIM+"','"+NAMA+"','"+id+"')");
            ps.executeUpdate();
        }
        else {
            FacesMessage message = new FacesMessage("Ayolah.. gak
ada penjurusan macam itu");
            FacesContext context = =
FacesContext.getCurrentInstance();
            context.addMessage(myButton.getClientId(context),
message);
            return null;
        }
    } catch (Exception e) {
        System.out.println(e);
    }
    return "/index.xhtml?faces-redirect=true";
}

```

### Source code pada fungsi Update\_Mahasiswa()

```

public String Update_Mahasiswa(){
    FacesContext fc = FacesContext.getCurrentInstance();
    Map<String, String> params = =
fc.getExternalContext().getRequestParameterMap();
    String Update_NIM= params.get("Update_NIM");
    try {
        Connection connection=null;
        Koneksi obj_koneksi = new Koneksi();
        connection = obj_koneksi.get_connection();
        Statement st = connection.createStatement();
        ResultSet rs = st.executeQuery("Select * from
penjurusan where penjurusan = '"+PENJURUSAN+"' ");
        String id2=null;
        while(rs.next()){
            id2 = rs.getString("id");
        }
        if (id2!=null){
            PreparedStatement ps = =
connection.prepareStatement("update mahasiswa set NIM=?, Nama=?,
Penjurusan=? where NIM=?");
            ps.setString(1, NIM);
            ps.setString(2, NAMA);
            ps.setString(3, id2);
            ps.setString(4, Update_NIM);
            ps.executeUpdate();
            System.out.println(id2);
        }
    }
}

```

```

        }
        else {
//            FacesMessage message2 = new FacesMessage("Ayolah..
gak ada penjurusan macam itu");
//            FacesContext context2 =
FacesContext.getCurrentInstance();
//            context2.addMessage(myButton2.getClientId(context2),
message2);
            return "/ErrorEdit.xhtml?faces-redirect=true";
        }
    } catch (Exception e) {
        System.out.println(e);
    }

    return "/index.xhtml?faces-redirect=true";
}

```

| Daftar Mahasiswa |                            |            |                      |                        |
|------------------|----------------------------|------------|----------------------|------------------------|
| NIM              | Nama                       | Penjurusan | Edit                 | Delete                 |
| 21120116120003   | Muhammad Rizki Nur Mquid   | Software   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120004   | Eviya Cindy Septiviani     | Multimedia | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120005   | Guntur Dwi Cahyono         | Software   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120006   | Monazariza Yonanta         | Jaringan   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120007   | Kornelius Satria Budiyanto | Embedded   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120008   | Agustiawan                 | Software   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120009   | Ahmad Shofie Hilmi         | Jaringan   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120010   | Minara Victoria M.         | Jaringan   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120011   | Ruo Julian Azri Pratama    | Embedded   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120012   | Rama Yuli Hinawati         | Jaringan   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120013   | Hayyim Dahlan Attaufiqi    | Embedded   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120014   | Faisal Rudy Rahadian       | Multimedia | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120025   | Busyro Busyurie            | Software   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120035   | Alfian Aulia Firdaus       | Software   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116130008   | Tengku Kemal Yusron        | Multimedia | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116130015   | Favo Perdana Hadjyanu S    | Multimedia | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116130060   | Ray Kiana Eka Putra        | Multimedia | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116130065   | Adam Maulidani             | Software   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116130089   | Atika Dewina Pradesta      | Multimedia | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116140068   | Fanny Hasti                | Software   | <a href="#">Edit</a> | <a href="#">Delete</a> |

Gambar 5.46 Hasil tugas halaman index

Berikut ini adalah 20 lebih daftar mahasiswa yang sudah di tambahkan dengan fitur tambah sesuai tugas.

The screenshot shows a web page titled "Tambah Data Mahasiswa". It has three input fields: "NIM" (21120116120000), "Nama Mahasiswa" (Gholib), and "Penjurusan" (Jarlegan). Below the form are two buttons: "Tambahkan" and "Kembali". A blue link at the bottom left says "Ayolah, gak ada penjurusan macam itu".

Gambar 5.47 Hasil tugas pesan tambah mahasiswa

Tampilan diatas adalah peringatan ketika menambahkan mahasiswa dengan penjurusan yang salah, maka akan keluar peringatan seperti gambar di atas.

The screenshot shows a web page titled "Tambah Data Mahasiswa". It has three input fields: "NIM" (21120116120000), "Nama Mahasiswa" (Gholib), and "Penjurusan" (Jarlegan). Below the form are two buttons: "Tambahkan" and "Kembali". A green success message at the bottom left says "Berhasil menambahkan Mahasiswa".

Gambar 5.48 Hasil tugas tambah mahasiswa berhasil

Namun jika menambahkan dengan penjurusan yang benar, maka data akan langsung tertambah ke basis data.

The screenshot shows a table titled "Daftar Mahasiswa" with columns: NIM, Nama, Penjurusan, Edit, and Delete. The table lists 30 student entries. At the bottom right of the table, there is a green success message: "Berhasil menambahkan Mahasiswa".

| NIM            | Nama                       | Penjurusan | Edit                 | Delete                 |
|----------------|----------------------------|------------|----------------------|------------------------|
| 21120116120000 | Gholib                     | Jaringan   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120003 | Muhammad Rizki Nur Majeed  | Software   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120004 | Eviita Cindy Septiviani    | Multimedia | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120005 | Guntur Dwi Cahyono         | Software   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120006 | Moenanatha Yonata          | Jaringan   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120007 | Kornelius Satrio Budhyanto | Embedded   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120008 | Agustianus                 | Software   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120009 | Ahmad Shofie Hilmi         | Jaringan   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120010 | Mutara Victoria M.         | Jaringan   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120011 | Ric Julian Azis Pramana    | Embedded   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120012 | Ratu Yuli Himawati         | Jaringan   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120013 | Hayyim Dahlan Attenuqi     | Embedded   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120014 | Faisal Rizky Rahadian      | Multimedia | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120025 | Busyono Buayarie           | Software   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116120035 | Alfin Aulia Firdaus        | Software   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116130002 | Tengku Kemal Yusron        | Multimedia | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116130015 | Favo Perdana Hadayanto S   | Multimedia | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116130060 | Ray Kiana Eka Putra        | Multimedia | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116130065 | Adam Maulidani             | Software   | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116130089 | Atta Dwara Pradana         | Multimedia | <a href="#">Edit</a> | <a href="#">Delete</a> |
| 21120116140068 | Fasya Hashi                | Software   | <a href="#">Edit</a> | <a href="#">Delete</a> |

Gambar 5.49 Hasil tugas data tertambah

Data dengan nama Ghoib sudah di tambahkan dan akan langsung di alihkan ke halaman index, sedangkan jika salah maka akan tetap di halaman tambah.

Gambar 5.50 Hasil tugas pesan error edit mahasiswa  
Ini adalah screenshot ketika mengubah data mahasiswa dengan penjurusan yang salah, maka akan di tampilkan pesan dengan hint di bagian bawah.

| Daftar Mahasiswa |                            |             |                      |                        |  |
|------------------|----------------------------|-------------|----------------------|------------------------|--|
| NIM              | Nama                       | Penjurusan  | Edit                 | Delete                 |  |
| 21120116120000   | Ghoib                      | Multimedia  | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116120003   | Muhammad Rizki Nur Majeed  | Software    | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116120004   | Evina Cindy Septiani       | Multimedia  | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116120005   | Guntur Dwi Cahyono         | Software    | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116120006   | Monazarifa Yonanta         | Jaringan    | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116120007   | Kornelius Satria Budhyanto | Embedded    | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116120008   | Aguastawan                 | Software    | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116120009   | Aldmad Shofie Holmi        | Jaringan    | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116120010   | Mutara Victoria M.         | Jaringan    | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116120011   | Rio Julian Azis Pratama    | Embedded    | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116120012   | Rama Yuli Himewati         | Jaringan    | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116120013   | Havim Dahlan Attaufiqi     | Embedded    | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116120014   | Faisal Ricky Rahadian      | Multimedia  | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116120025   | Basyroo Basyane            | Software    | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116120035   | Alfian Andia Firdaus       | Software    | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116130002   | Tengku Kemal Yuson         | Multimedia  | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116130015   | Favo Perdana Hadiyanto S   | Multimedias | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116130060   | Ray Kista Eka Putra        | Multimedia  | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116130065   | Adam Maulidani             | Software    | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116130089   | Atha Dwira Pradana         | Multimedia  | <a href="#">Edit</a> | <a href="#">Delete</a> |  |
| 21120116140068   | Fanny Hashi                | Software    | <a href="#">Edit</a> | <a href="#">Delete</a> |  |

Gambar 5.51 Hasil tugas edit data berhasil

Jika data di edit dengan data yang benar, maka data dapat di ubah dan dialihkan ke halaman index.

## 5.6 Kesimpulan

1. *Connector MySQL* digunakan untuk membangun suatu koneksi antara program dengan *database MySQL*.
2. Untuk menambahkan *connector MySQL* dilakukan dengan cara menambahkannya pada *library* program.
3. Pada JSF xhtml untuk menampilkan *table* dilakukan dengan cara `<dataTable></dataTable>`.
4. Bootstrap CSS dapat diimplementasikan pada program dengan cara menambahkannya pada *folder resource* kemudian untuk memanggilnya dengan cara `<h:openStyleSheet>`.
5. File JSF *Managed Bean* berguna sebagai *back-end* yang mana pada *file* ini terdapat *method* untuk menjalankan *query* CRUD.
6. Untuk menampilkan suatu *join* dapat dilakukan dengan cara membuat *method* dengan *query join*.
7. Fungsi dari `required = true` agar input teks tidak boleh dibiarkan kosong.
8. Untuk menambahkan pesan peringatan ketika *input* teks kosong yaitu dapat dilakukan dengan fungsi `requiredMessage = " "`.

## **BAB VI**

### **PENUTUP**

#### **6.1 Kesimpulan**

Dari praktikum kali ini dapat disimpulkan bahwa:

1. Java Bean adalah komponen dalam java yang bersifat reusable dan memiliki kelebihan JavaBean, dapat dimanipulasi secara visual menggunakan builder tool dan memungkinkan pengguna untuk membangun aplikasi secara mudah.
2. Enterprise Java Beans (EJB) adalah komponen server-side yang menyimpan business logic dan dapat diakses secara remote.
3. Untuk mengolah ataupun pengatur data relational dalam platform JavaStandard Edition dan Java Enterprise Edition digunakan Java Persistence API.
4. JSF terbagi menjadi komponen model, view, dan controller dimana komponen controller bekerja di front end sedangkan komponen lain bekerja di back end.

## 6.2 Saran

1. Kerjasama kelompok yang baik dan ketelitian dalam melaksanakan percobaan maupun tugas praktikum sangat diperlukan dalam praktikum.
2. Asisten sebaiknya terus memantau praktikan agar praktikum berjalan dengan lancar sesuai dengan prosedur yang semestinya.
3. Perlunya pembaruan modul untuk meningkatkan kemampuan praktikan.
4. Penambahan penggunaan bahasa pemrograman lain selain Java mungkin akan lebih baik.

## **DAFTAR PUSTAKA**

- [1] <http://lea.si.fti.unand.ac.id/2013/11/javabeans/>
- [2] <https://perbedaan.budisma.net/perbedaan.awt-dan.swing.html>
- [3] [https://en.wikipedia.org/wiki/Java\\_Platform,\\_Enterprise\\_Edition](https://en.wikipedia.org/wiki/Java_Platform,_Enterprise_Edition)
- [4] <https://medium.com/@wafaakamilahmaulanihermawan/Statefull-vs-Stateless-application-6d600eddefe3>
- [5] <https://www.mahirkoding.com/pengenalan-jsp/>
- [6] <https://aeroyid.wordpress.com/2012/07/16/javamemperkenalkan-java-servlet-dan-membuat-helloworld/>
- [7] <https://dartoblog.wordpress.com/2012/08/01/pengenalan-jpa-java-persistence-api/>
- [8] <https://suhearie.wordpress.com/2008/08/26/java-enterprise-mulai-dari-mana-part-2-netbeans-glassfish/>
- [9] <https://upyes.wordpress.com/2013/02/06/pengertian-dan-sejarah-mysql/>
- [10] <https://id.wikipedia.org/wiki/NetBeans> [diakses : 15 November 2019]
- [11] [https://id.wikipedia.org/wiki/JavaServer\\_Faces](https://id.wikipedia.org/wiki/JavaServer_Faces) [diakses : 15 November 2019]
- [12] <https://www.webhozz.com/blog/installasi-tomcat-di-centos/> [diakses : 15 November 2019]
- [13] <https://suhearie.wordpress.com/2008/08/26/java-enterprise-mulai-dari-mana-part-2-netbeans-glassfish/> [diakses : 15 November 2019]
- [14] <https://www.termasmedia.com/pemrograman/java/476-instalasi-mysql-connector-j-koneksi-program-java-ke-database-mysql.html> [diakses : 15 November 2019]
- [15] <https://id.wikipedia.org/wiki/GitHub> [diakses : 15 November 2019]
- [16] <https://id.wikipedia.org/wiki/XAMPP> [diakses : 15 November 2019]

## LAMPIRAN 1 (LEMBAR ASSISTENSI)

### Laboratorium Rekayasa Perangkat Lunak

Teknik Komputer Universitas Diponegoro

D204 Gedung Kuliah Bersama Fakultas Teknik  
Universitas Diponegoro Semarang



#### LEMBAR ASISTENSI

PRAKTIKUM REKAYASA SOFTWARE BERBASIS KOMPONEN 2019

Judul Praktikum : Java Bean  
Tanggal Praktikum : 26 September 2019  
Nama Asisten : 1. Guntur Dwi Cahyono (21120116120005)  
Nama Praktikan : Rio Kisna Eka P (21120116130060)  
Kelompok : 32



| No. | Hari/Tanggal/Jam | Keterangan | TTD 1 | TTD 2 |
|-----|------------------|------------|-------|-------|
| 1.  |                  | Rev 1      | b     |       |
| 2.  |                  | AC         | ↓     |       |
| 3.  |                  | fe         | ↓     |       |
| 4.  |                  |            |       |       |

Mengetahui,  
Koordinator Praktikum

Guntur Dwi Cahyono  
21120116120005



Scanned with  
CamScanner

## Laboratorium Rekayasa Perangkat Lunak

Teknik Komputer Universitas Diponegoro

D204 Gedung Kuliah Bersama Fakultas Teknik  
Universitas Diponegoro Semarang



### LEMBAR ASISTENSI

#### PRAKTIKUM REKAYASA SOFTWARE BERBASIS KOMPONEN 2019

Judul Praktikum : EJB Session Bean  
Tanggal Praktikum : 3 Oktober 2019  
Nama Asisten : 1. Jeremy Karisma M (21120116140078)  
                  2. Muhammad Ikhsan (21120116120033)  
Nama Praktikan : Rio Kisna Eka P (21120116130060)  
Kelompok : 32



| No. | Hari/Tanggal/Jam | Keterangan | TTD 1 | TTD 2 |
|-----|------------------|------------|-------|-------|
| 1.  |                  | Rev 1      |       |       |
| 2.  |                  | ACC        |       |       |
| 3.  |                  |            |       |       |
| 4.  |                  |            |       |       |

Mengetahui,

Koordinator Praktikum

Guntur Dwi Cahyono  
21120116120005



Scanned with  
CamScanner

## Laboratorium Rekayasa Perangkat Lunak

Teknik Komputer Universitas Diponegoro

D204 Gedung Kuliah Bersama Fakultas Teknik  
Universitas Diponegoro Semarang



### LEMBAR ASISTENSI

#### PRAKTIKUM REKAYASA SOFTWARE BERBASIS KOMPONEN 2019

Judul Praktikum : CRUD Menggunakan EJB, JPA dan MySQL  
Tanggal Praktikum : 24 Oktober 2019  
Nama Asisten : 1. Agyan Atma Villantya (21120116140071)  
                  2. Yogie Mesya Tama (21120116130061)  
Nama Praktikan : Rio Kisna Eka P (21120116130060)  
Kelompok : 32



| No. | Hari/Tanggal/Jam | Keterangan | TTD 1 | TTD 2 |
|-----|------------------|------------|-------|-------|
| 1.  |                  | ACC        |       |       |
| 2.  |                  |            |       |       |
| 3.  |                  |            |       |       |
| 4.  |                  |            |       |       |

Mengetahui,

Koordinator Praktikum

Guntur Dwi Cahvono  
21120116120005



Scanned with  
CamScanner

## Laboratorium Rekayasa Perangkat Lunak

Teknik Komputer Universitas Diponegoro

D204 Gedung Kuliah Bersama Fakultas Teknik  
Universitas Diponegoro Semarang



### LEMBAR ASISTENSI

#### PRAKTIKUM REKAYASA SOFTWARE BERBASIS KOMPONEN 2019

Judul Praktikum : Java Server Faces  
Tanggal Praktikum : 31 Oktober 2019  
Nama Asisten : 1. Fanny Hasbi (21120116140068)  
                  2. Adam Maulidani (21120116130065)  
Nama Praktikan : Rio Kisna Eka P (21120116130060)  
Kelompok : 32



| No. | Har/Tanggal/Jam | Keterangan | TTD 1 | TTD 2 |
|-----|-----------------|------------|-------|-------|
| 1.  |                 |            |       |       |
| 2.  |                 |            |       |       |
| 3.  |                 |            |       |       |
| 4.  |                 |            |       |       |

Mengetahui,  
Koordinator Praktikum

Guntur Dwi Cahyono  
21120116120005



Scanned with  
CamScanner