BAB III

EJB SESSION BEAN

* 1. Tujuan

1. Praktikan mengetahui apa itu *session bean*
2. Praktikan mengenal GlashFish Server pada NetBeans
3. Praktikan mengetahui perbedaan *stateless* dan *stateful*
4. Praktikan dapat mengetahui kegunaan Java Servlet
   1. Alat dan Bahan
      1. Laptop



Gambar 3.1. Laptop

Laptop adalah [komputer](http://id.wikipedia.org/wiki/Komputer" \t "_blank" \o "Komputer) portabel (kecil dan dapat dibawa ke mana-mana dengan mudah) yang terintegrasi pada sebuah casing. Pada Praktikum Modul 1 ini, laptop yang digunakan adalah laptop yang memenuhi *minimum system requirements* untuk menjalankan NetBeans IDE, yaitu Processor Intel Core-I5, RAM 6 GB, dan sistem operasi Windows 10.

* + 1. Netbeans IDE



Gambar 3.2. NetBeans

Netbeans adalah sebuah aplikasi Integrated Development Environment (IDE) yang berbasiskan Java dari Sun Microsystems yang berjalan di atas swing. Swing merupakan sebuah teknologi Java untuk pengembangan aplikasi dekstop yang dapat berjalan pada berbagai macam platform seperti windows, linux, Mac OS X dan Solaris. Sebuah IDE merupakan lingkup pemrograman yang di integrasikan ke dalam suatu aplikasi perangkat lunak yang menyediakan Graphic User Interface (GUI), suatu kode editor atau text, suatu compiler dan suatu debugger.

Netbeans juga dapat digunakan progammer untuk menulis, meng-compile, mencari kesalahan dan menyebarkan program netbeans yang ditulis dalam bahasa pemrograman java namun selain itu dapat juga mendukung bahasa pemrograman lainnya dan program ini pun bebas untuk digunakan dan untuk membuat professional dekstop, enterprise, web, and mobile applications dengan Java language, C/C++, dan bahkan dynamic languages seperti PHP, JavaScript, Groovy, dan Ruby.

* + 1. Java EE SDK



Gambar 3.3. Java EE

Java EE SDK adalah *integrated development kit* gratis yang digunakan untuk membangun, menguji, dan menggunakan aplikasi Java EE.

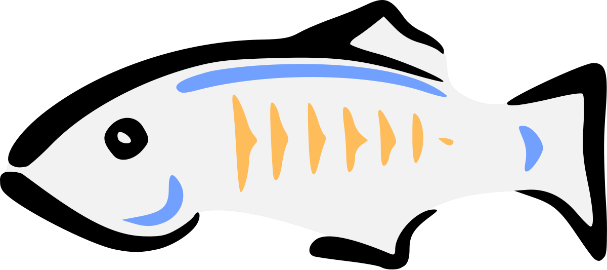
* + 1. Java SDK 7



Gambar 3.4. Java SDK 7

SDK ini biasanya adalah kumpulan dari *tools* yang dibutuhkan untuk membuat serta menjalankan program. Jadi di dalam SDK itu ada JDK-nya, ada JRE-nya serta mungkin IDE-nya juga.

* + 1. Glassfish Server



Gambar 3.5. Glassfish Server

Web server ini merupakan teman satu IDE dengan Apache Tomcat bila kamu menggunakan Netbeans. Glassfish mengklaim dirinya sebagai web server pertama untuk aplikasi Java EE 7 di dunia. Tentu saja web server ini disediakan untuk keluarga Java. Beberapa fitur inti dari Glassfish yang mesti kamu ketahui adalah sebagai berikut:

* Bagian inti yang ringan dan ekstensibel
* Memiliki *web container*
* Halaman administrasi untuk konfigurasi dan manajamen yang mudah digunakan
* Mendukung *high availability clustering*
* Mendukung *load balancing*
  1. Dasar Teori
     1. Java Enterprise Editional (Java EE)

Java EE atau Java Enterprise Edition merupakan paket terlengkap dari Java yaitu bisa memeiliki *library* yang sangat lengkap, biasanya paket ini digunakkan untuk membuat program dengan tingkat *enterprise*. Seperti Java Beans, Servlets, JSP dan lain - lain. Tapi di dalam paket ini tetap memasukkan pustaka standard yang ada dalam paket Java SE.

*(Sumber:* [*https://www.oracle.com/java/technologies/javaee-8-sdk-downloads.html*](https://www.oracle.com/java/technologies/javaee-8-sdk-downloads.html)*)*

* + 1. Component Stateful

Stateful adalah suatu model aplikasi yang **menggunakan penyimpanan (Session, dsb)**, artinya **setiap transaksi yang dilakukan akan diingat, dan akan berpengaruh kepada transaksi yang akan dilakukan selanjutnya**. Cara kerja stateful adalah menyimpan data anda di server, seperti *username* dan *password*, lalu menggunakannya untuk terhubung kepada client. *Stateful application* akan memberikan respon sesuai dengan permintaan *client* sampai *client* tersebut *logout*. Jadi *stateful application* merupakan **aplikasi dependen** (tidak independen), yaitu respon yang diberikan server **tergantung kepada aksi *user /client***.

*(Sumber:* [*https://medium.com/@MHMMDNURHUDA/apa-itu-stateful-dan-stateless-application-penjelasan-dan-contohnya-6a585b1a0a94*](https://medium.com/@MHMMDNURHUDA/apa-itu-stateful-dan-stateless-application-penjelasan-dan-contohnya-6a585b1a0a94)*)*

* + 1. Component Stateless

Stateless adalah suatu model aplikasi yang selalu memberikan **respon sama terhadap setiap client** yang mengakses web ataupun aplikasi tertentu. Pada stateless tidak ada penyimpanan ataupun perubahan setiap kita mengakses aplikasi nya. Jadi, setiap kita membuka sebuah aplikasi, respon pada stateless aplication akan selalu sama, jadi jika kita membuka aplikasi tersebut, seolah olah itu adalah pertama kali kita membuka aplikasi tersebut, padahal sudah sering (Misalnya) itu dikarenakan stateless selalu memberikan respon yang sama terhadap setiap request. jadi stateless itu merupakan **model aplikasi yang independen,** artinya bisa berdiri sendiri tanpa ketergantungan ke web ataupun aplikasi lain.

*(Sumber:* [*https://medium.com/@MHMMDNURHUDA/apa-itu-stateful-dan-stateless-application-penjelasan-dan-contohnya-6a585b1a0a94*](https://medium.com/@MHMMDNURHUDA/apa-itu-stateful-dan-stateless-application-penjelasan-dan-contohnya-6a585b1a0a94)*)*

* + 1. Java Server Page (JSP)

JSP adalah bahasa *scripting* untuk *web programming* yang *bersifat server side* seperti halnya PHP dan ASP. JSP dapat berupa gabungan antara baris html dan fungsi-fungsi dari JSP itu sendiri. Berbeda dengan Servlet yang harus dikompilasi oleh user menjadi *class* sebelum dijalankan, JSP tidak perlu dikompilasi oleh user tapi server yang akan melakukan tugas tersebut. Teknologi JSP menyediakan cara yang lebih mudah dan cepat dalam membuat halaman-halaman web yang menampilkan isi secara dinamik dan bekerja dengan berbagai macam *web server, application server, browser* dan *development tool.*

*(Sumber:* [*https://ptwfighter.wordpress.com/2011/09/28/servlet-jsp-dan-web-browser/*](https://ptwfighter.wordpress.com/2011/09/28/servlet-jsp-dan-web-browser/)*)*

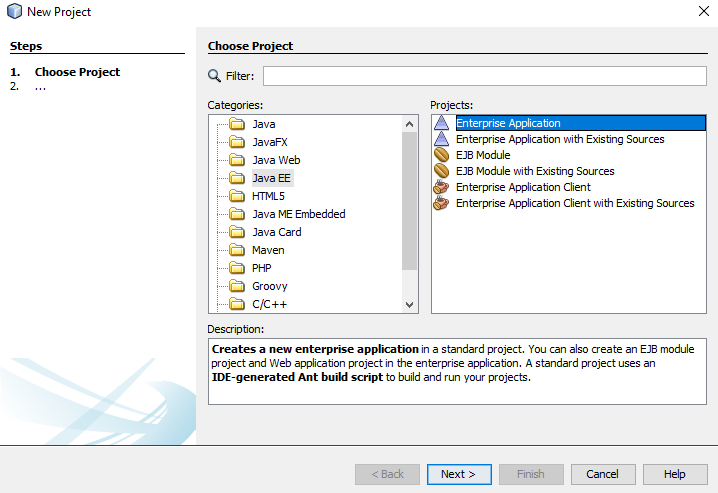
* + 1. Java Servlet

Servlet adalah teknologi Java untuk aplikasi web berupa *class* yang digunakan untuk menerima *request* dan memberi respon melalui protokol http (html, xml, file dan sebagainya). Pada dasarnya Servlet merupakan *file java class* yang telah dikompilasi dan dijalankan oleh *servlet container* atau *application server*. Istilah *application server* digunakan apabila *software server* dapat menjalankan servlet, JSP serta teknologi J2EE utama seperti EJB (Enterprise Java Bean). Contoh Application Servlet adalah BEA Web Logic, IBM Websphere, Jboss, dsb. *Servlet container* biasanya juga merupakan JSP container, seperti Apache Tomcat, Macromedia Jrun, Resin.

*(Sumber:* [*https://ptwfighter.wordpress.com/2011/09/28/servlet-jsp-dan-web-browser/*](https://ptwfighter.wordpress.com/2011/09/28/servlet-jsp-dan-web-browser/)*)*

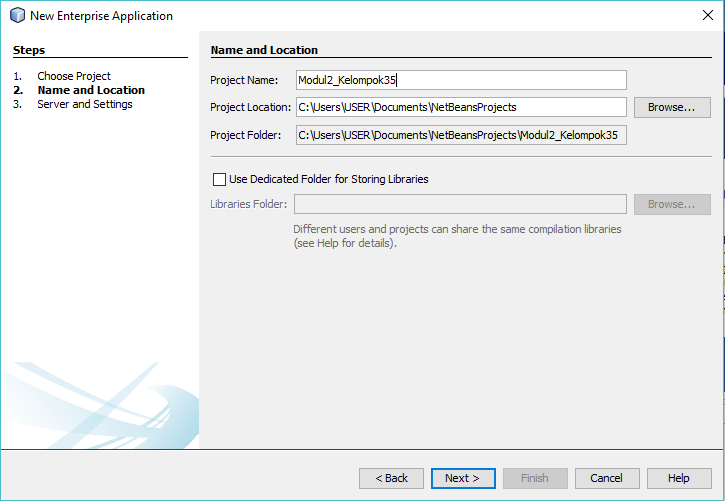
* 1. Langkah Kerja

1. Buka aplikasi NETBEANS IDE.
2. Buat project baru pilih Java EE > Enterprise Application lalu klik next.



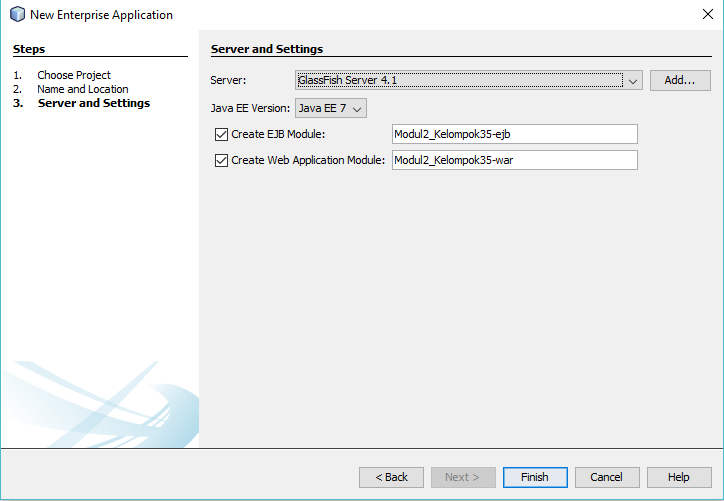
Gambar 3.6. Membuat project baru

1. Pilih location project lalu klik next



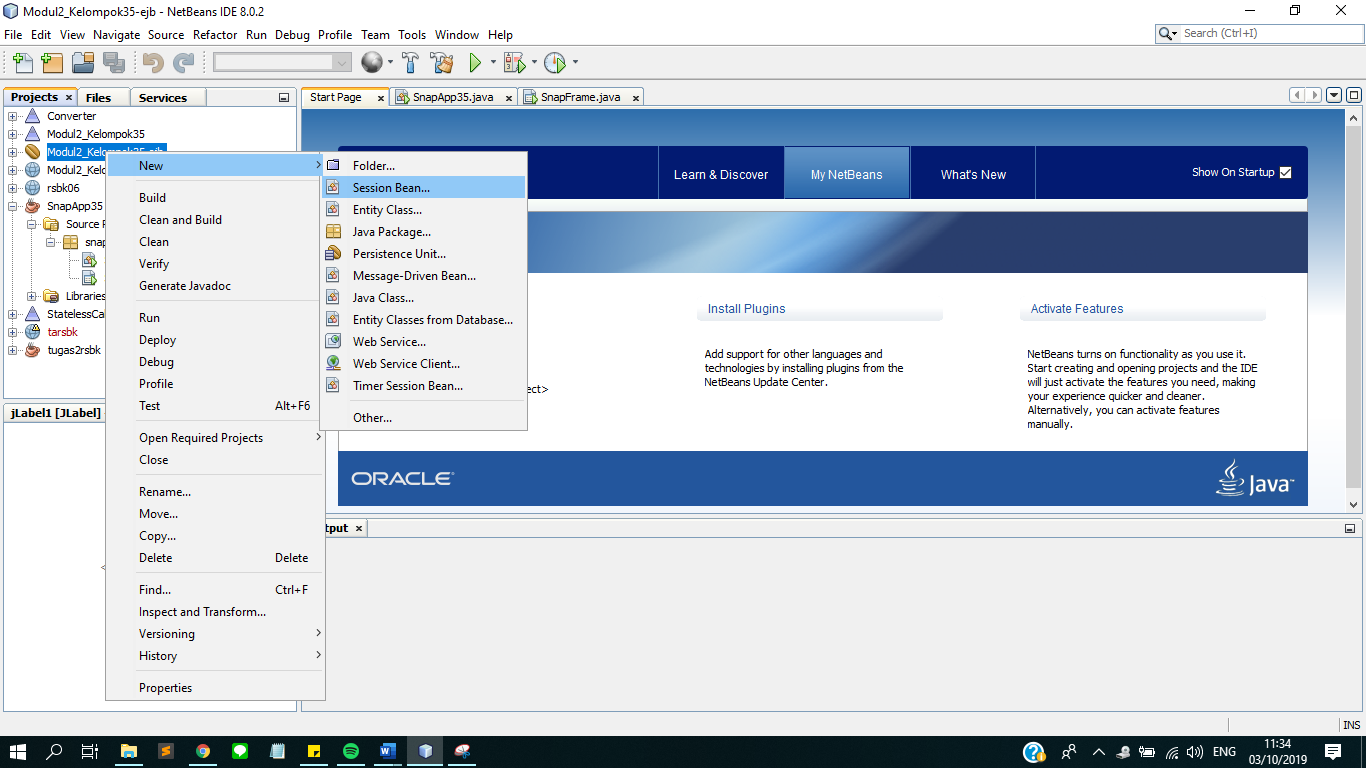
Gambar 3.7. Memilih project location

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



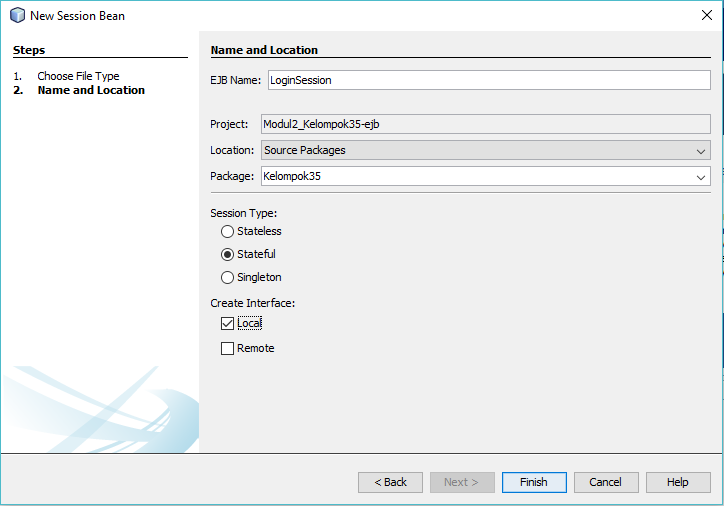
Gambar 3.8. Memilih server GlassFish Server

1. Buat file Session Been baru dengan cara klik kanan project EJB pilih New>Session Bean.



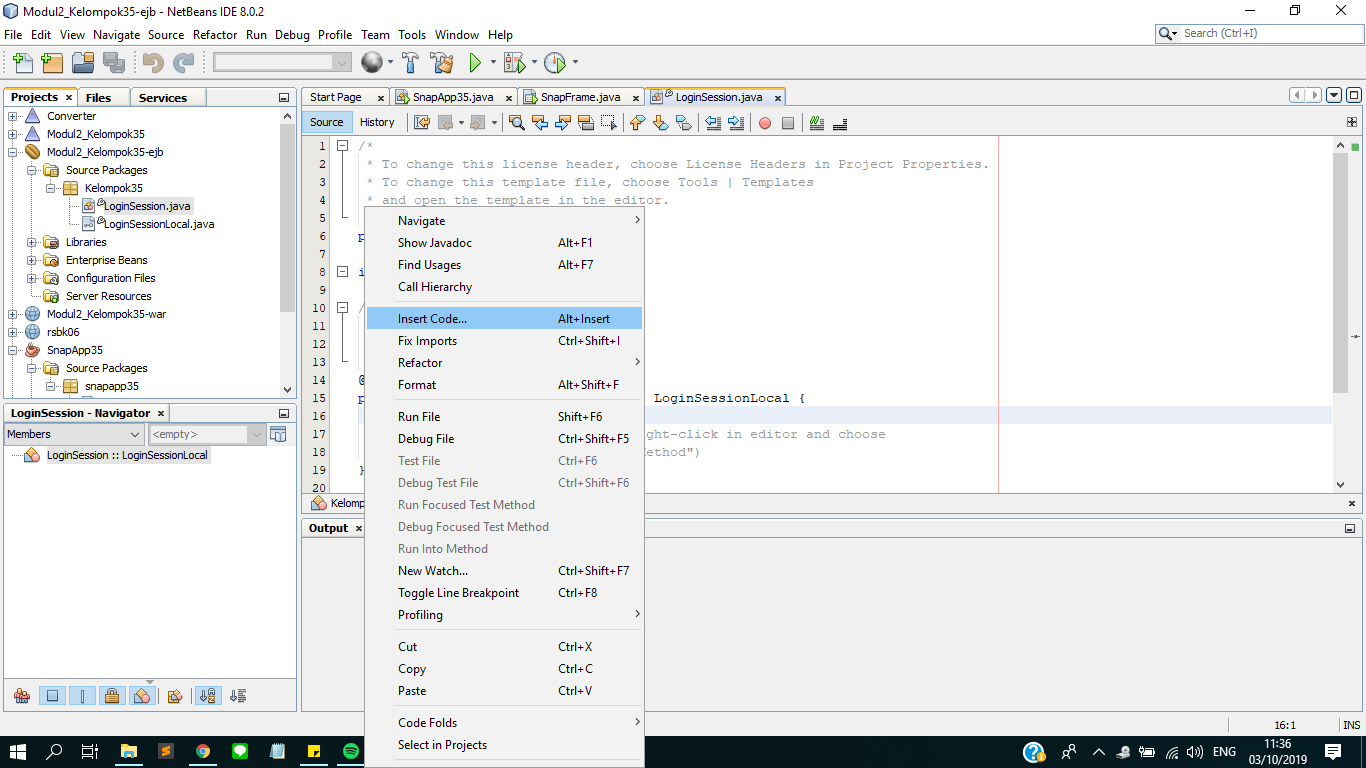
Gambar 3.9. Membuat file Session Bean

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



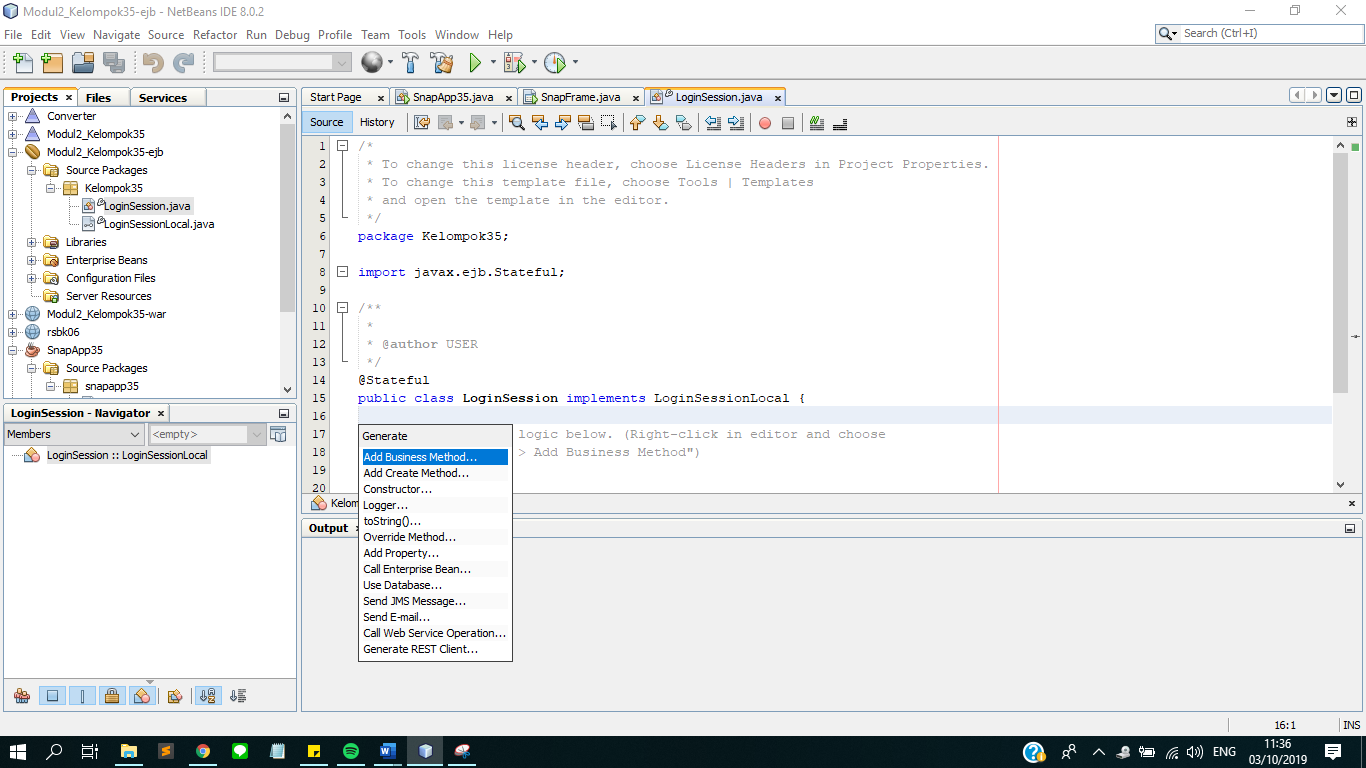
Gambar 3.10. Memberi nama Session Bean

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



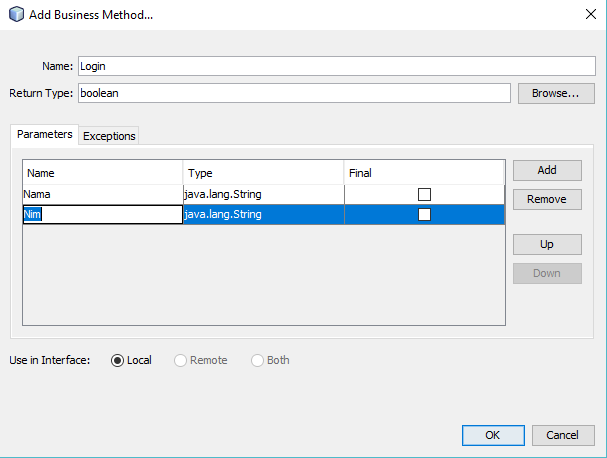
Gambar 3.11. Memilih insert code

1. Pilih **Add Business Methode.**



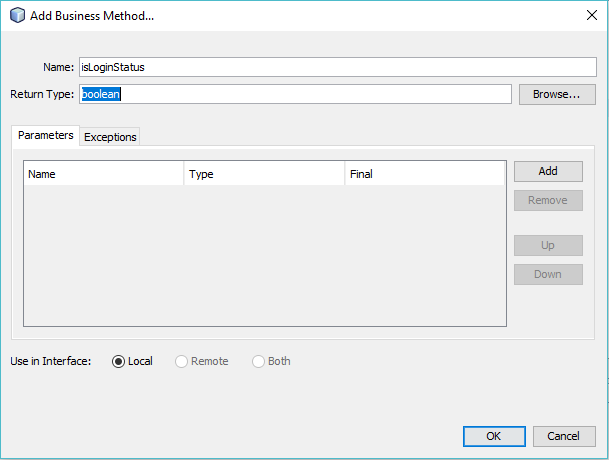
Gambar 3.12. Mimilih Add Business Method

1. Masukan nama Login dengan **Return Type** boolean. Tambahkan 2 parameter dengan nama **Nama** dan **Nim**. Pastikan parameter sesuai dengan yang terdapat pada Gambar.

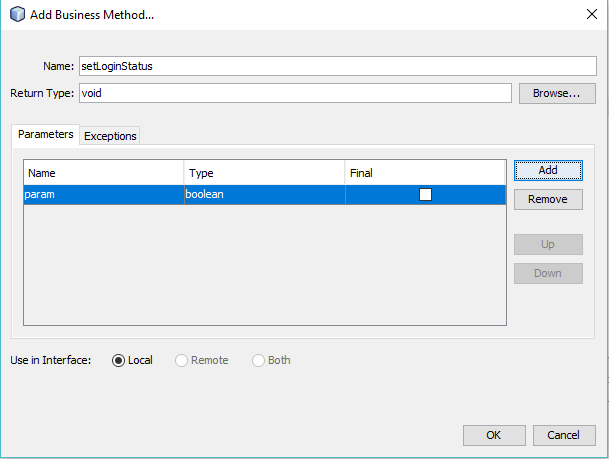


Gambar 3.13. Membuat method Login

1. Lakukan hal yang sama untuk method **isLoginStatus** dan **setLoginStatus**



Gambar 3.14. Membuat method isiLoginStatus

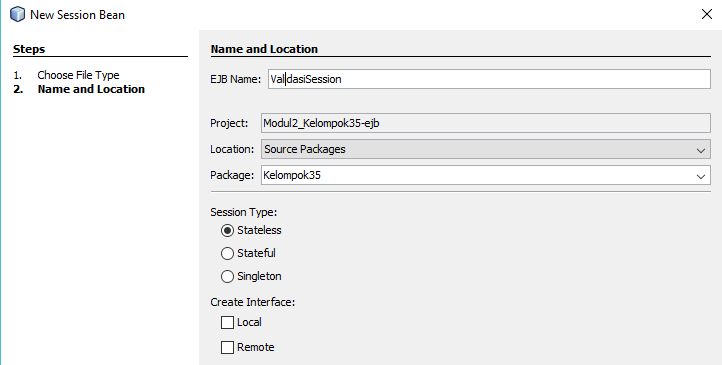


Gambar 4.15. Membuat method setLoginStatus

1. Hapus source code yanga ada didalam class LoginSession lalu tambahkan source code berikut.

|  |
| --- |
| 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 loginStatus) {  this.loginStatus = loginStatus;  } |

1. Ikuti langkah 5 untuk membuat Session Bean baru
2. Pada Isikan ValidasiSession pada **EJB Name**. Pada **Session Type** pilih Stateless.



Gambar 3.16. Membuat session bean Validasi Session

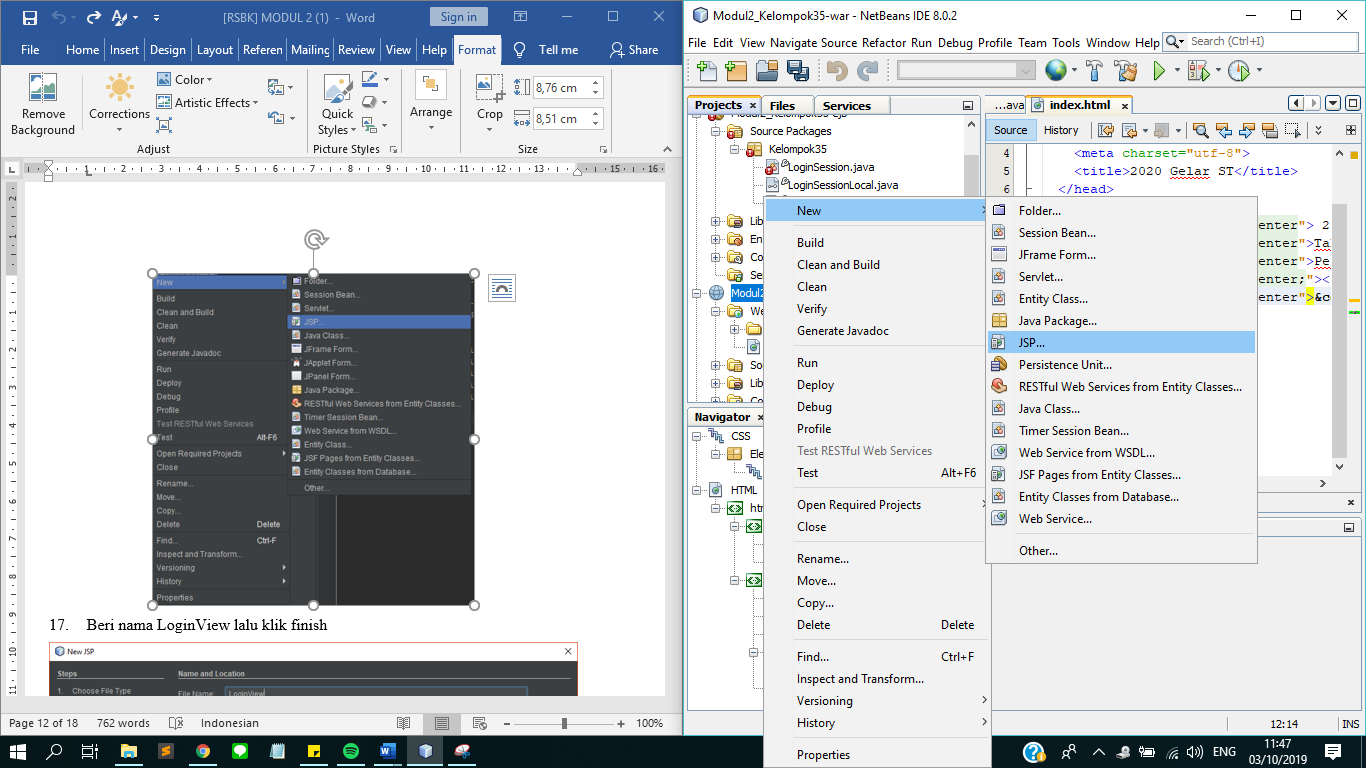
1. Tambakan source code berikut didalam class **ValidasiSession**.

|  |
| --- |
| 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;  } |

1. Buka file index.html di directory **Modul2\_Kelompok35-war>Web Page** dan timpa source code yang sudah ada dengan source code berikut.

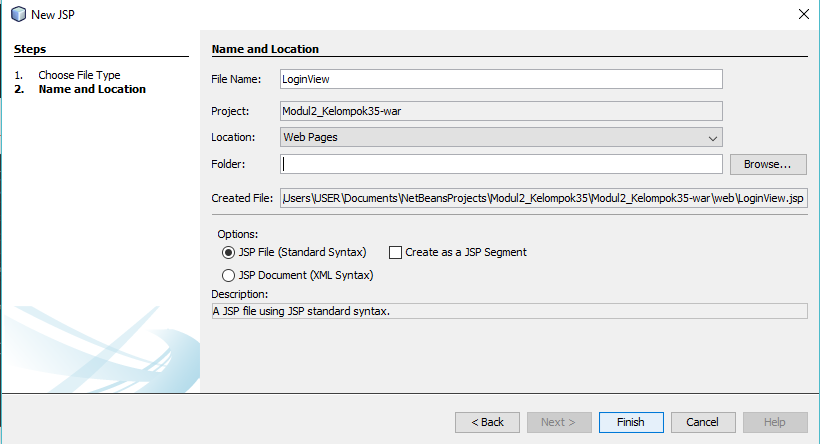
|  |
| --- |
| <!DOCTYPE html>  <html lang="en" dir="ltr">  <head>  <meta charset="utf-8">  <title>2020 Gelar ST</title>  </head>  <body>  <h1 style="text-align:center"> 2020 GELAR ST</h1>  <h3 style="text-align:center">Tahun depan kita wisuda</h3>  <h3 style="text-align:center">Pelajari program berikut 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> |

1. Buat file JSP dengan cara klik kanan pada project war pilih **New>JSP**



Gambar 3.17. Membuat file JSP

1. Beri nama LoginView lalu klik finish

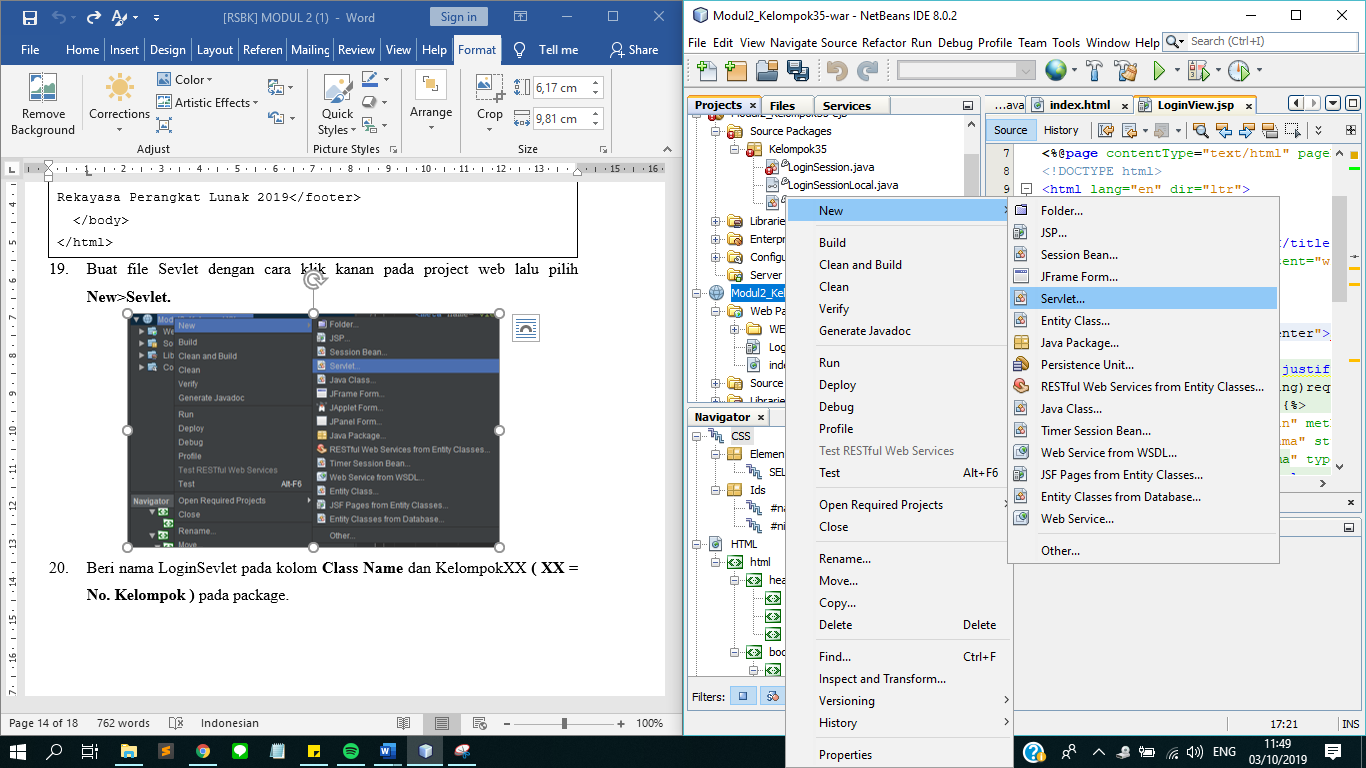


Gambar 3.18. Memberi nama JSP dengan LoginView

1. Paste source code berikut ke dalam file JSP yang dibuat.

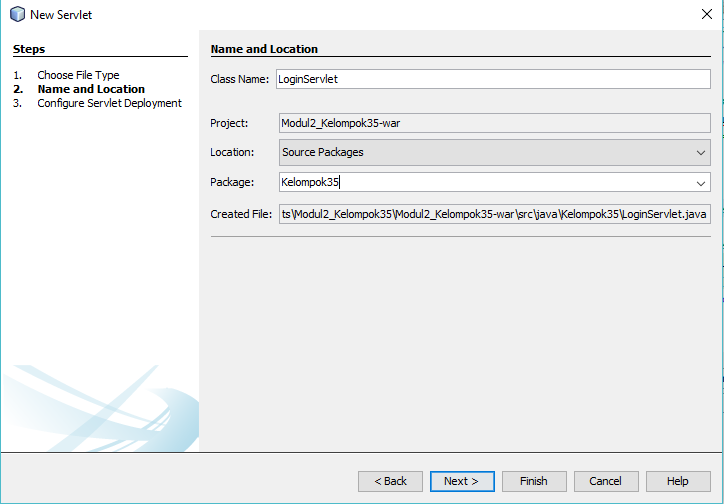
|  |
| --- |
| <%@page contentType="text/html" pageEncoding="UTF-8"%>  <!DOCTYPE html>  <html lang="en" dir="ltr">  <head>  <meta charset="utf-8">  <title>Tahun Depan Wisuda</title>  <meta name="viewport" content="width=device-width, initial-scale=1.0">  </head>  <body>  <header>  <h1 style="text-align:center">SEMANGAT, 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> |

1. Buat file Sevlet dengan cara klik kanan pada project web lalu pilih **New>Sevlet.**



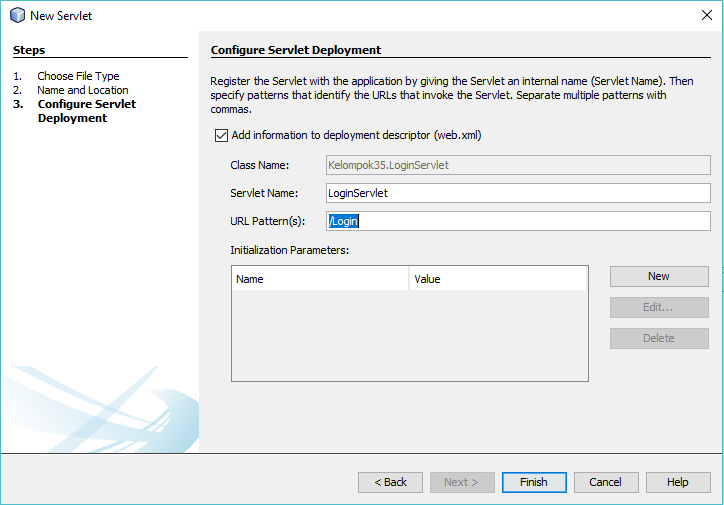
Gambar 3.19. Membuat file servlet

1. Beri nama LoginSevlet pada kolom **Class Name** dan KelompokXX **( XX = No. Kelompok )** padapackage.



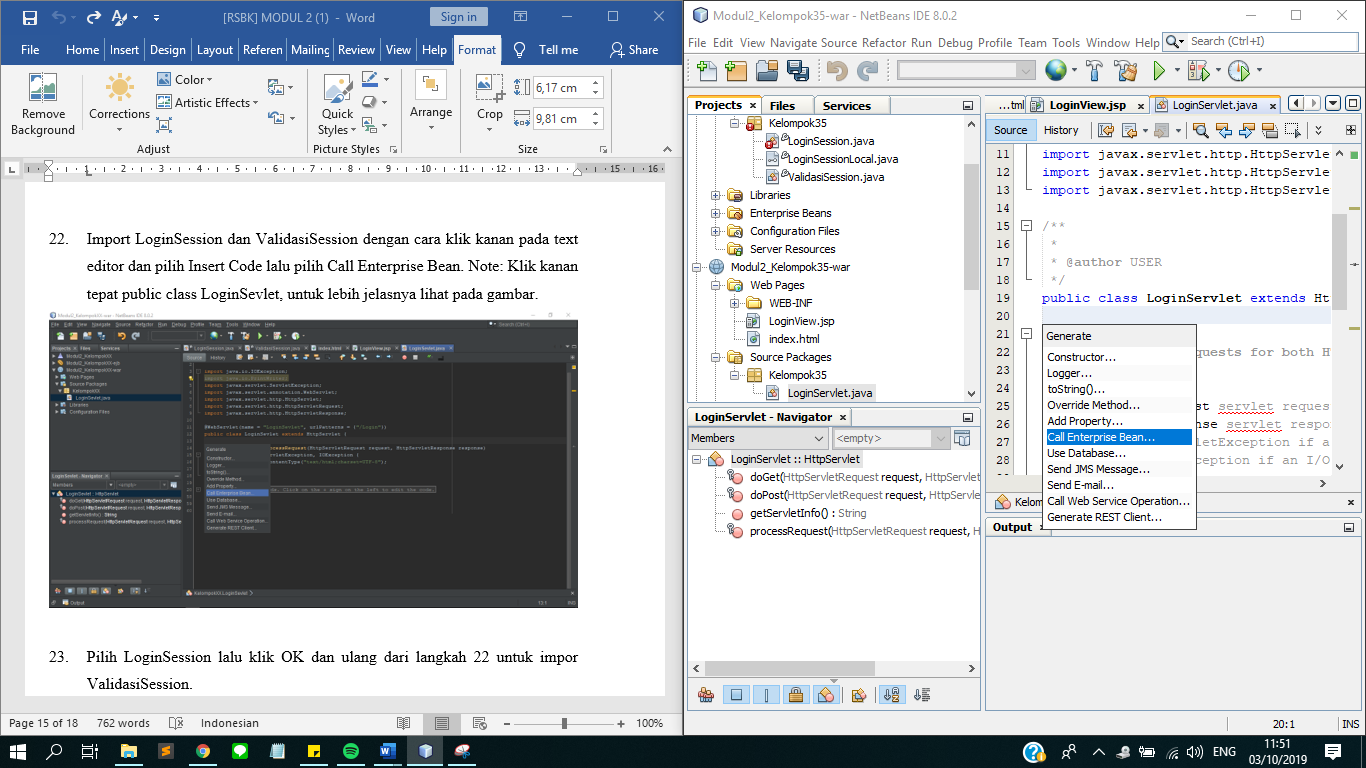
Gambar 3.20. Memberi nama servlet dengan LoginServlet

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



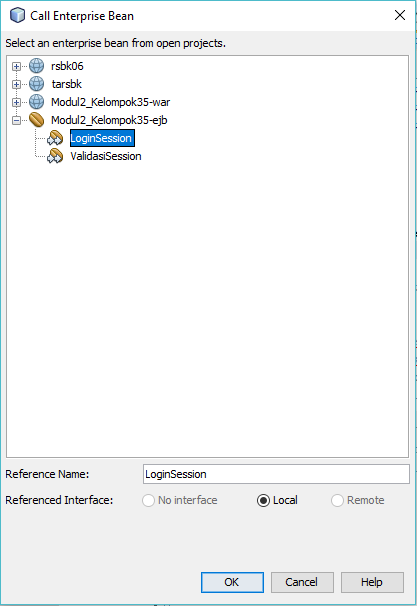
Gambar 3.21. Mengkonfigurasi deployment servlet

1. Import LoginSession dan ValidasiSession dengan cara klik kanan pada text editor dan pilih Insert Code lalu pilih Call Enterprise Bean. Note: Klik kanan tepat public class LoginSevlet, untuk lebih jelasnya lihat pada gambar.



Gambar 3.22. Memilih call enterprise bean

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



Gambar 3.23. Meng-import LoginSession

1. Hapus source code pada method **processRequest,** lalutambahkan source code berikut.

|  |
| --- |
| validasiSession = new ValidasiSession();    request.setAttribute("status", "false");  RequestDispatcher rd = getServletContext().getRequestDispatcher("/LoginView.jsp");  rd.forward(request, response); |

1. Tambahkan source code berikut didalam method **doGet**.

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

1. Tambahkan source code berikut didalam method **doPost**.

|  |
| --- |
| 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);  } |

1. Jalankan project enterprise dengan cara klik kanan project enterprise lalu klik pilih run.
   1. Hasil Percobaan

Pada Praktikum Modul 2, praktikan diminta untuk membuat sebuah aplikasi yang bisa mencari data mahasiswa dengan memasukan Nama Mahasiswa dan NIM Mahasiswa. Pertama-tama yang dibuat adalah LoginSession.java. Di dalamnya terdapat 2 parameter, yaitu Nama dan Nim. Kemudian Nama diisi dengan “Tendi N” dan Nim diisi dengan “21120116140039”. Berikut source code lengkapnya:

package Kelompok35;

import javax.ejb.Stateful;

@Stateful

public class LoginSession implements LoginSessionLocal {

private final String [] Nama = {"Tendi N"};

private final String [] Nim = {"21120116140039"};

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 loginStatus) {

this.loginStatus = loginStatus;

}

}

Selanjutnya yaitu membuat ValidasiSession.java yang digunakan untuk melakukan validasi terhadap inputan Nama dan Nim yang dimasukan. Berikut source codenya:

package Kelompok35;

import javax.ejb.Stateless;

import javax.ejb.LocalBean;

@Stateless

@LocalBean

public class ValidasiSession {

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;

}

// Add business logic below. (Right-click in editor and choose

// "Insert Code > Add Business Method")

}

Selanjutnya yaitu membuat index.html yang merupakan tampilan awal ketika kita *running* aplikasi. Terdapat link <<Klik Disini>> yang jika ditekan akan beralih ke halaman pencarian data mahasiswa.

<!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>

Selanjutnya yaitu LoginView merupakan halaman yang digunakan untuk melakukan pencarian dan menampilkan data mahasiswa. Untuk melakukan pencarian mahasiswa, input yang dimasukan adalah Nama Mahasiswa dan NIM Mahasiswa. Terdapat sebuah form yang terdiri dari label, textfield, dan button. Label digunakan untuk menampilkan teks, button digunakan sebagai tombol untuk menjalankan proses, sedangkan textfield untuk memasukan nilai yang akan diproses. Setelah button ditekan, jika Nama Mahasiswa atau NIM Mahasiswa salah, maka akan ditampilkan pesan “Mahasiswa tidak terdaftar” atau “Inputan salah” dengan tulisan berwarna merah. Jika data yang diisikan benar akan beralih ke tampilan berikutnya yaitu halaman yang akan menampilkan data mahasiswa yang dicari.

<%--

Document : LoginView

Created on : Oct 3, 2019, 11:48:13 AM

Author : USER

--%>

<%@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>

<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>

Terakhir yaitu LoginServlet.java. Servlet berfokus pada memproses informasi. Dalam proses ini, pertama-tama diambil Nama dan Nim yang diisikan, kemudian akan divalidasi. Jika Nama dan Nim valid maka akan dilanjutkan ke proses berikutnya yaitu mengecek apakah Nama dan Nim sama dengan yang ada di parameter. Jika Nama salah maka akan ditampilkan pesan “Mahasiswa tidak terdaftar”. Jika Nama dan Nim salah maka akan ditampilkan pesan “Inputan salah”. Jika data yang diisikan benar akan beralih ke tampilan berikutnya yaitu halaman yang akan menampilkan bahwa data mahasiswa yang dimasukan benar.

/\*

\* 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 Kelompok35;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.ejb.EJB;

import javax.naming.Context;

import javax.naming.InitialContext;

import javax.naming.NamingException;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/\*\*

\*

\* @author USER

\*/

public class LoginServlet extends HttpServlet {

@EJB

private ValidasiSession validasiSession;

LoginSessionLocal loginSession = lookupLoginSessionLocal();

/\*\*

\* 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 {

validasiSession = new ValidasiSession();

request.setAttribute("status", "false");

RequestDispatcher rd = getServletContext().getRequestDispatcher("/LoginView.jsp");

rd.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 {

if("Keluar".equals(request.getParameter("keluar"))){

loginSession.setLoginStatus(false);

request.setAttribute("nimEr", "");

request.setAttribute("namaEr", "");

}

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 {

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);

}

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>

private LoginSessionLocal lookupLoginSessionLocal() {

try {

Context c = new InitialContext();

return (LoginSessionLocal) c.lookup("java:global/Modul2\_Kelompok35/Modul2\_Kelompok35-ejb/LoginSession!Kelompok35.LoginSessionLocal");

} catch (NamingException ne) {

Logger.getLogger(getClass().getName()).log(Level.SEVERE, "exception caught", ne);

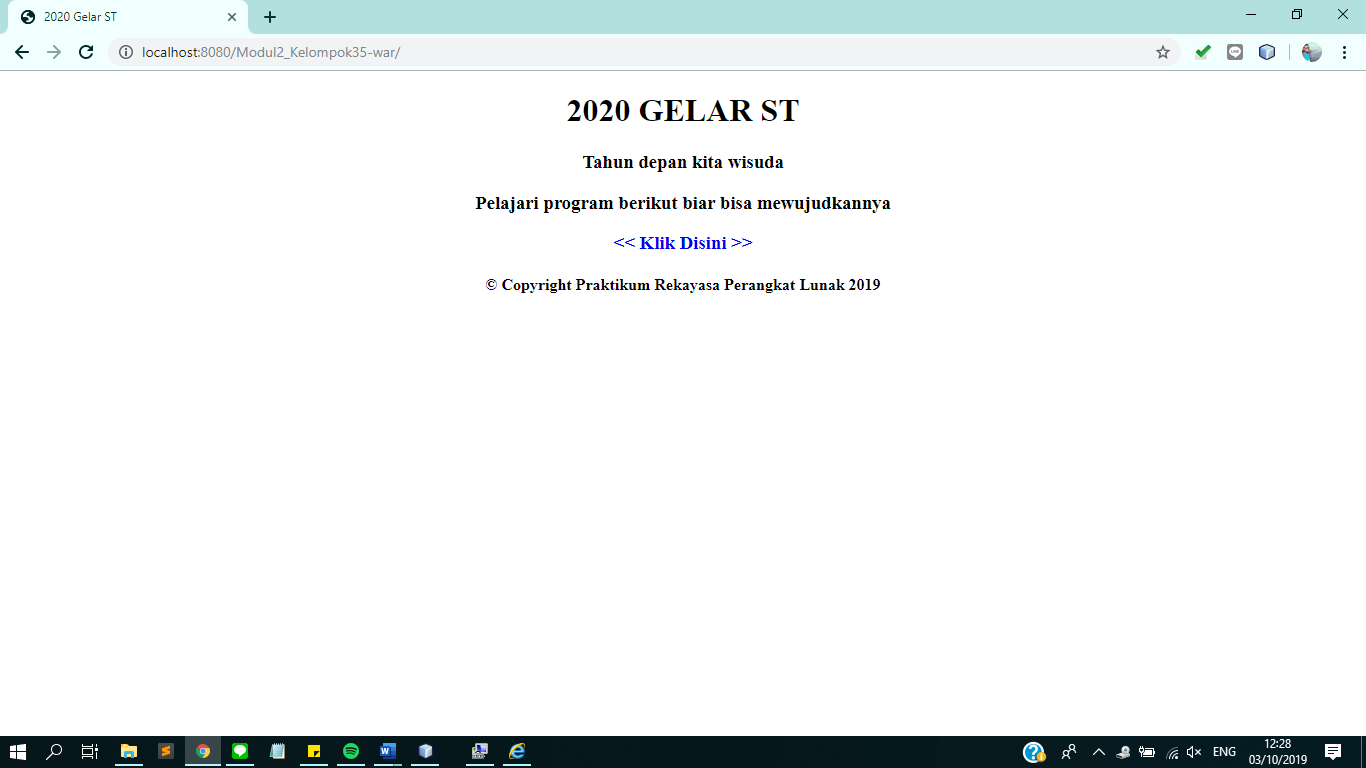
throw new RuntimeException(ne);

}

}

}

Berikut adalah tampilan saat *running* aplikasi. Pada saat pertama kali aplikasi dibuka tampilannya adalah seperti gambar di bawah. Kemudian untuk melanjutkan, kita pilih <<Klik Disini>>.



Gambar 3.24. Tampilan awal aplikasi

Setelah itu maka akan ditampilkan halaman pencarian. Kita diminta untuk memasukan nama dan nim, kemudian tekan tombol cari.



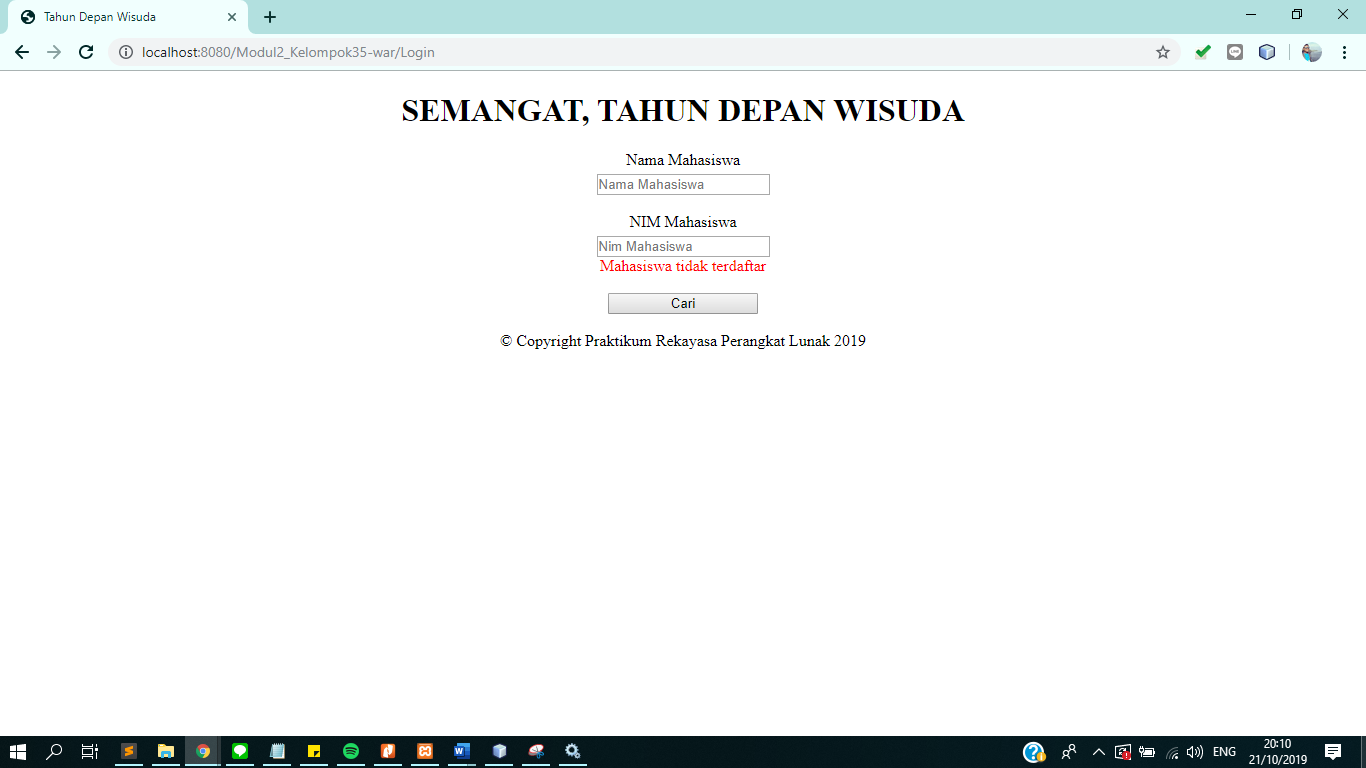
Gambar 3.25. Tampilan aplikasi untuk pencarian mahasiswa

Jika data yang dimasukan sesuai, maka akan ditampilkan halaman ini. Untuk memulai pencarian kembali, tekan tombol Kembali.



Gambar 26. Tampilan aplikasi ketika data mahasiswa ditemukan

Ketika kita memasukan Nama Mahasiswa dan NIM Mahasiswa yang salah, maka tampilannya akan seperti gambar di bawah ini.



Gambar 3.27. Tampilan aplikasi ketika data mahasiswa tidak ditemukan

* 1. Tugas dan Pembahasan

3.6.1 Tugas

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> dataMahasiswa = new ArrayList<Mahasiswa>();  public void cekMahasiswa(String nama,String nim){  dataMahasiswa.add(new Mahasiswa(nama,nim));  }    public cariMahasiswa() {  cekMahasiswa("Rio Kisna Eka Putra","21120116130060");  cekMahasiswa("Alfian Aulia Firdaus","21120116130035");  cekMahasiswa("Favo Perdana HS","21120116120015");  cekMahasiswa("Farrell Denando","21120116120022");  cekMahasiswa("Tendi Nugeraha","21120116140039");  }  private String checkNama(String param){  for (int i = 0; i < dataMahasiswa.size(); i++){  if(param.equals(dataMahasiswa.get(i).nama)){  return "Nama Praktikan : " +dataMahasiswa.get(i).nama+" ("+dataMahasiswa.get(i).nim+")";  }  }  return null;  }  private String checkNIM(String param){  for (int i = 0; i < dataMahasiswa.size(); i++) {  if(param.equals(dataMahasiswa.get(i).nim)){  return "Nama Praktikan : " +dataMahasiswa.get(i).nama+" (" +dataMahasiswa.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 "Kosong";  } |

*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 constractor 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: #5DADE2">  <header>  <h1 style="text-align:center">DATA PENCARIAN MAHASISWA KELOMPOK 35</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;">  <label for="nama" style="margin-bottom:5px; font-size: 23px">Cari Mahasiswa</label>  <input id="nama" type="text" name="param" placeholder="Masukan Nama / NIM" style="width:50%; height: 50px; font-size: 20px; text-align: center">  <span style="color:#641E16; font-size: 25px; margin-top: 15px;">${show}</span>  <br>  <input type="submit" name="submit" value="Cari" style="width:50%; height: 50px; font-size: 20px; background-color: #FFC300 ">  </form>  </main><br>  <main style="display:flex;justify-content:center;flex-direction: column;">  <form action="search" method="get" style="display:flex;justify-content:center;flex-direction: column;align-items: center;">  <label for="nama" style="margin-bottom:5px; font-size: 23px">Nama Mahasiswa</label>  <input id="nama" type="text" name="nama" placeholder="Masukkan Nama" style="width:50%; height: 50px; font-size: 20px; text-align: center">  <br>  <label for="nim" style="margin-bottom:5px; font-size: 23px">NIM Mahasiswa</label>  <input id="nim" type="text" name="nim" placeholder="Masukkan Nama" style="width:50%; height: 50px; font-size: 20px; text-align: center">  <br>  <input type="submit" name="submit" value="Tambah" style="width:50%; height: 50px; font-size: 20px; background-color: #FFC300">  </form>  </main><br>  <footer style="text-align:center; font-size: 18px">&copy; Copyright Praktikum Rekayasa Perangkat Lunak 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 di lakukan 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

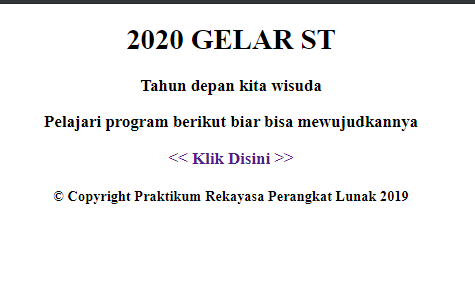
|  |
| --- |
| cariMahasiswa.cekMahasiswa(request.getParameter("nama"), request.getParameter("nim"));  processRequest(request, response); |

Fungsi doGet di eksekusi ketika action get search pada search.jsp di jalankan. Isinya berupa pemanggilan fungsi cekMahasiswa dari *class* cariMahasiswa dengan parameter nama dan nim 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”.



Gambar 2.28 index.html tugas



Gambar 2.29 Halaman pencarian



Gambar 2.30 Mencari mahasiswa yang terdaftar



Gambar 2.31 Mencoba mencari mahasiswa yang belum terdaftar



Gambar 2.32 Menambahkan mahasiswa baru



Gambar 2.33 Mahasiswa baru telah di tambahkan

Link Github : https://github.com/perdanafavo/PraktikumRSBKkel35

* 1. Kesimpulan
     + 1. *Session bean* merepresentasikan proses/task yang dilakukan atas nama *client*, yang sifatnya hanya berasosiasi dengan *client* tertentu, di-*create* dan di-*destroy* oleh *client* lain, dan akan hilang setelah sistem *shutdown*.
       2. Jika menggunakan *stateless session bean*, maka ketika di-*refresh* pada *browser* tidak dapat menampilkan data tersebut lagi.
       3. *Glasshfish* berfungsi untuk *webserver*, yang mana *webserver* merupakan sebuah perangkat dimana aplikasi kita dapat menerima *request* dan mengirimkan *response* dalam protokol HTTP.
       4. Fungsi *servlet* adalah untuk membuat *web* dinamis bagi *user*.
       5. Penggunaan *stateless bean* dalam praktikum dimaksudkan agar siapa saja dapat mengakses aplikasi tersebut karena *stateless bean* dapat digunakan *multiuser.*
       6. Pada *index.html* bisa menambahkan kode *css* untuk mempercantik tampilan aplikasi yang dibuat.