

## **BAB III**

### **EJB SESSION BEANS**

#### **3.1 Tujuan**

1. Praktikan dapat mengetahui dan memahami perihal EJB Session Bean.
2. Praktikan dapat mengenal penggunaan server pada Java NetBeans misal glashfish server.
3. Praktikan dapat mengetahui serta membedakan antara komponen stateful dengan komponen stateless pada Java NetBeans.
4. Praktikan dapat menerapkan penggunaan komponen stateful dan komponen stateless pada sebuah program yang dibangun dengan Java NetBeans.
5. Praktikan mampu bekerja dengan menggunakan Netbeans dan membuat *project* JAVA sederhana dengan konsep EJB Session Bean.

## 3.2 Alat dan Bahan

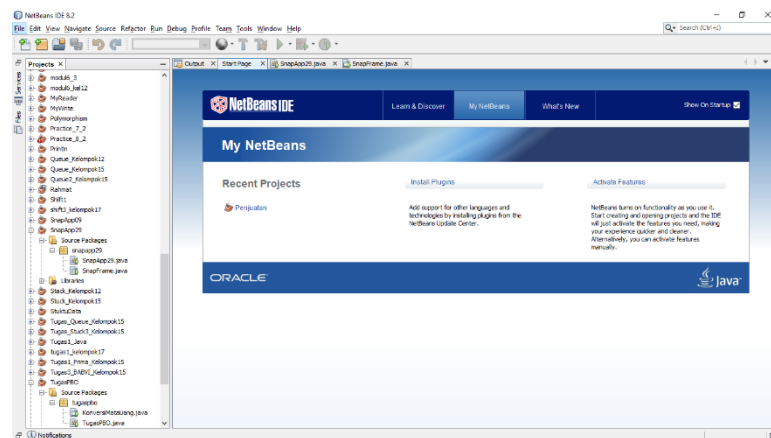
### 3.2.1 Laptop



Gambar 3.1 Laptop

Laptop adalah perangkat *end-device* yang bekerja seperti komputer dengan ukuran yang lebih kecil. Pada praktikum ini laptop digunakan untuk membuat program Java Beans.

### 3.2.2 Netbeans IDE



Gambar 3.2 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.

### 3.2.3 Java EE SDK



Gambar 3.3 *Java Enterprise Edition*

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

### 3.2.4 Java SDK 7



Gambar 3.4 Java SDK 7

SDK (*Software Development Kit*) adalah kumpulan dari tools yang dibutuhkan untuk membuat serta menjalankan program. Dimana pada SDK ini di dalamnya terdapat tools JDK (*Java Development Kit*), ada JRE (*Java Runtime Environment*) serta terdapat IDE juga. Pada praktikum ini kita menggunakan Java SDK versi 7.

### 3.2.5 Glassfish Server



Gambar 3.5 Glassfish Server

Glassfish adalah implementasi referensi Java EE dan karenanya mendukung Enterprise JavaBeans, JPA, Java Server Faces, JMS, RMI, Java Server Pages, servlets, dll. Hal ini memungkinkan pengembang untuk membuat aplikasi perusahaan yang portabel dan dapat diukur, serta berintegrasi dengan teknologi lainnya. Glassfish mengklaim dirinya sebagai web server pertama untuk aplikasi Java EE 7 di dunia. Sehingga web server ini disediakan untuk pemrograman Java.

### 3.3 Dasar Teori

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

J2EE adalah singkatan dari Java 2 Enterprise Edition, dan sejak versi 5 berubah menjadi Java EE atau cukup Java Enterprise Edition. Java EE adalah sebuah spesifikasi, bukan berupa product berwujud nyata. Ada banyak vendor software atau komunitas opensource membuat software berupa application server yang mengikuti standard Java EE sehingga disebut Java EE compliant application server.



Gambar 3.6 *Java Enterprise Edition*

Application server ini memungkinkan untuk membuat aplikasi berskala enterprise dengan lebih mudah karena application server sudah menyediakan berbagai fasilitas yang siap untuk digunakan sebagai pendukung aplikasi. Full Java EE compliant application server menyediakan berbagai service seperti web container, messaging, web service, mail, directory service, database connectivity, distributed transaction, remoting, persistence, dan lain lain.

Contoh application server adalah Glassfish, Oracle AS, JBoss, IBM Websphere, JRun, JOnAS, dll. Apache Tomcat adalah salah satu webserver/webcontainer untuk aplikasi web Java yang cukup terkenal, tetapi bukan termasuk Java EE application server karena tidak menyediakan semua service yang ada di spesifikasi Java EE.

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

### 3.3.2 Component Stateful

*Stateful* adalah disaat informasi yang diberikan sebelumnya disimpan dan mempengaruhi konten/informasi/data yang akan diberikan setelahnya. *Stateful component* mendeskripsikan *component* yang mempunyai *property state*. *Programming pattern* menggunakan *stateful* dan *stateless component* artinya, kita melewati atau mempassing *state* dari *stateful component* ke *stateless component*.

(Sumber: <https://medium.com/@dickymaudie/perbedaan-antara-stateful-dan-stateless-974325bfce4b>)

### 3.3.3 Component Stateless

*Stateless* adalah di saat informasi tidak disimpan sehingga tampilan web akan sama saja kalau dilihat oleh anda berulang kali atau oleh orang lain. Di java selalu berhubungan dengan objek dan *component*, dan untuk mengakses dan menyimpan alur data pada *component* kita menggunakan *props* dan *state*. *Stateless component* di istilahkan untuk *component* yang tidak mempunyai *property state*.

(Sumber: <https://medium.com/@dickymaudie/perbedaan-antara-stateful-dan-stateless-974325bfce4b>)

### 3.3.4 Java Server Page (JSP)

JSP adalah suatu teknologi web berbasis bahasa pemrograman Java dan berjalan di Platform Java, serta merupakan bagian teknologi J2EE (Java 2 Enterprise Edition). JSP sangat sesuai dan tangguh untuk menangani presentasi di web. Sedangkan J2EE merupakan *platform* Java untuk pengembangan sistem aplikasi enterprise dengan dukungan API (Application Programming Inteface) yang lengkap dan portabilitas serta memberikan sarana untuk membuat suatu aplikasi yang memisahkan antara *business logic* (sistem), presentasi dan data.

JSP (Java Server Page) merupakan bagian dari J2EE dan khususnya merupakan komponen web dari aplikasi J2EE secara keseluruhan. JSP juga memerlukan JVM (Java Virtual Machine) supaya dapat berjalan, yang berarti juga mengisyaratkan keharusan menginstal Java Virtual Machine di server, dimana JSP akan dijalankan. Selain JVM, JSP juga memerlukan server yang disebut dengan Web Container. Teknologi JSP menyediakan cara yang lebih mudah dan cepat untuk membuat

halaman-halaman web yang menampilkan isi secara dinamik. Teknologi JSP di desain untuk membuat lebih mudah dan cepat dalam membuat aplikasi berbasis web yang bekerja dengan berbagai macam web server, application server, browser dan development tool. Java Server Pages (JSP) adalah bahasa scripting untuk web programming yang bersifat server side seperti halnya PHP dan ASP.

(Sumber: <https://aminawm.wordpress.com/pengertian-jsp-java-server-pages/>)

### 3.3.5 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.

Kelebihan servlet:

1. Performance Servlet baik karena tidak ada proses pembuatan berulang untuk tiap *request* dari client. Jadi tiap request ditangani oleh proses servlet container (apache tomcat), di mana servlet tidak dibuat dan dihapus berulang-ulang tetapi tetap tersimpan pada memori untuk menangani request selanjutnya.
2. Servlet memiliki kemampuan yang lengkap, antara lain penanganan request ke request, penanganan cookie dan session, akses database dengan JDBC, caching serta library yang lengkap untuk pembuatan aplikasi web.
3. Servlet memiliki fasilitas *security* yang baik dan merupakan bagian dari teknologi Java yang sudah dari asalnya didesain dengan *security* yang baik.
4. Teknologi Java Servlet portabel karena dapat dijalankan di berbagai servlet container, application server, maupun sistem operasi.

5. Proses development yang lebih cepat. Dengan menggunakan Servlet dapat menggunakan library java yang lengkap maupun menggunakan komponen yang sudah ada.
6. Karena servlet merupakan teknologi java yang memiliki penanganan memori yang baik serta memiliki garbage collection sehingga aplikasi web menjadi aplikasi yang tangguh dan stabil.

#### Proses Kerja Servlet

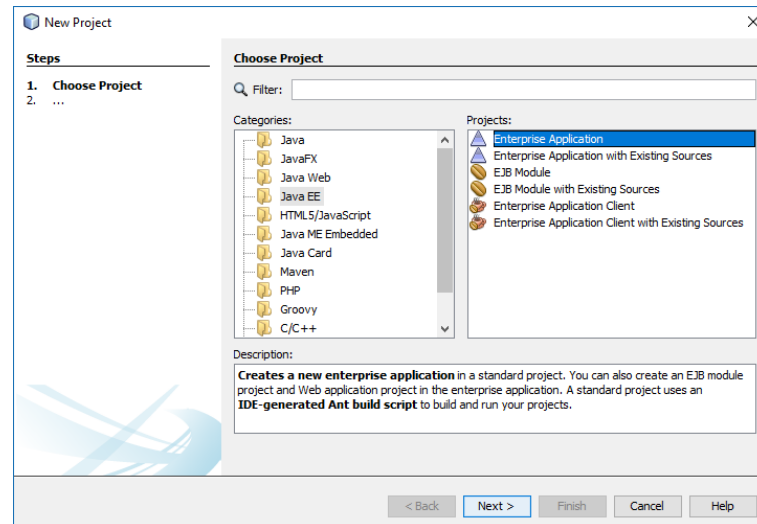
1. Servlet di load ke JVM oleh Servlet container apabila terjadi *request* pertama kali oleh client.
2. Proses penanganan *request* dijalankan sebagai *thread* dari web server atau servlet container. Setelah di load maka servlet tetap ada di memori untuk menangani *request* berikutnya.
3. Tiap kali menangani request, servlet container membandingkan *timestamp* dari servlet dalam memori dengan file class java servlet. Apabila *timestamp* file java servlet ada yang lebih baru maka secara otomatis servlet container akan me load servlet yang baru dari class servlet.

(Sumber: <https://ptwfighter.wordpress.com/2011/09/28/servlet-jsp-dan-web-browser/>)



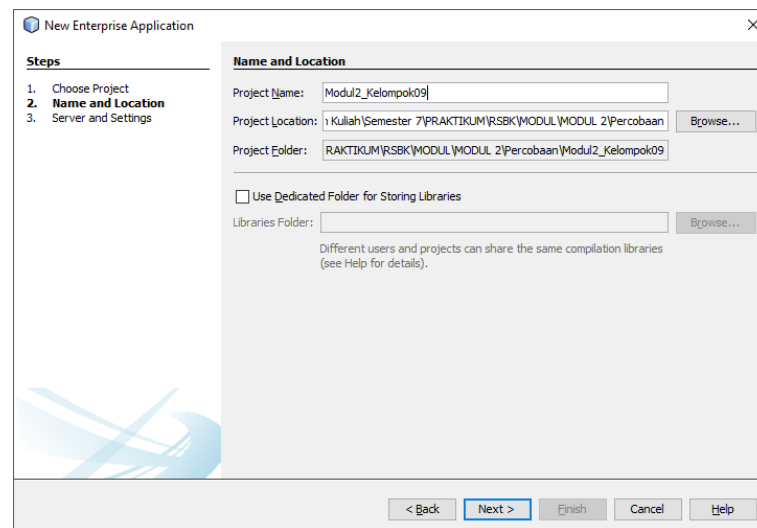
### 3.4 Langkah Kerja

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



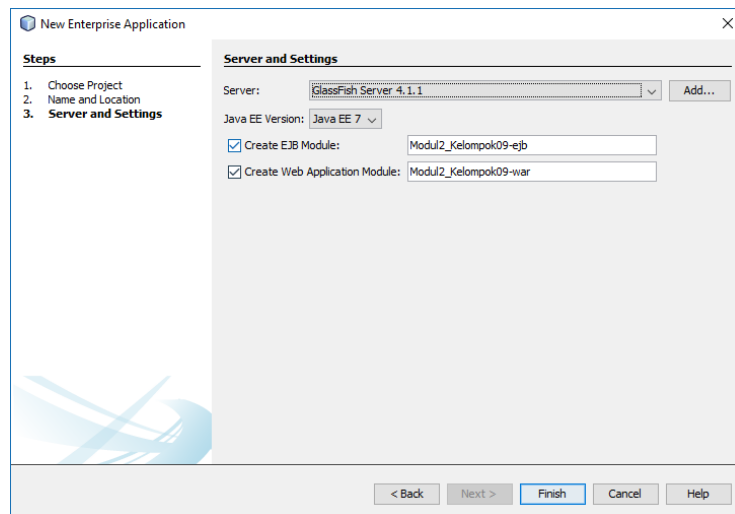
Gambar 3.7 Tampilan New Project Enterprise Application

3. Pilih location project lalu klik next



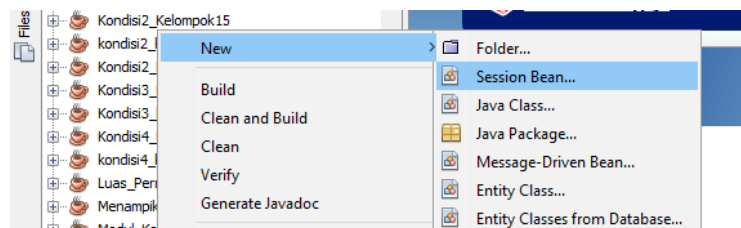
Gambar 3.8 Halaman New Enterprose Application

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



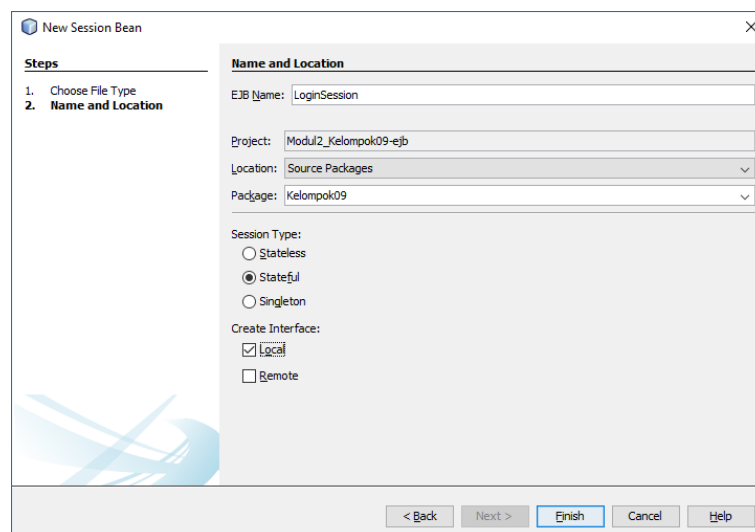
Gambar 3.9 Memilih *Server* dan *Setting*

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



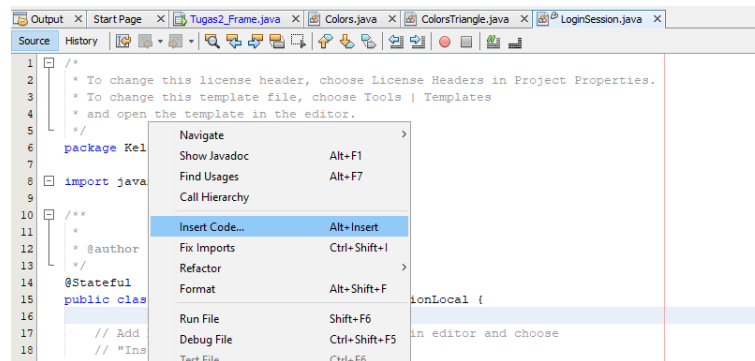
Gambar 3.10 *Create Session Bean* pada Project EJB

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



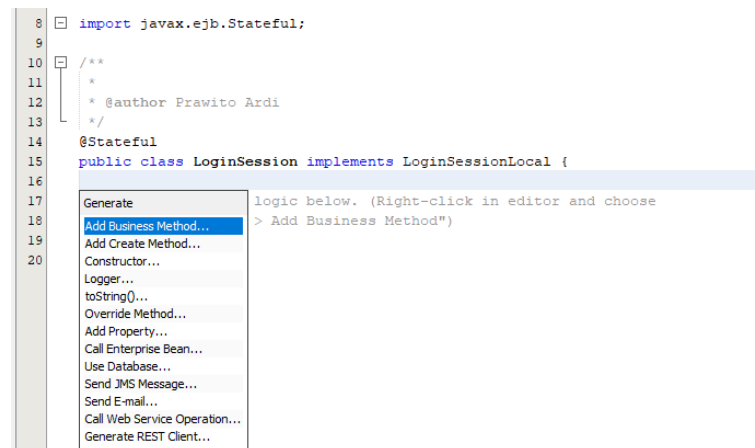
Gambar 3.11 Mengatur Name and Location pada New Session Bean

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



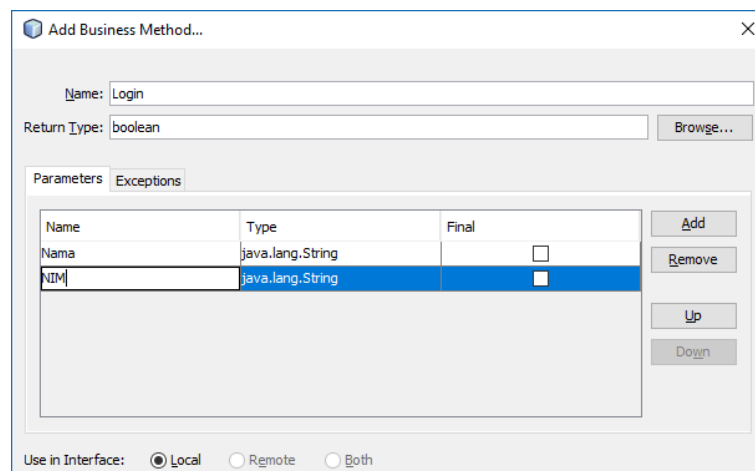
Gambar 3.12 Insert Code pada Text Editor LoginSession

8. Pilih **Add Business Methode.**



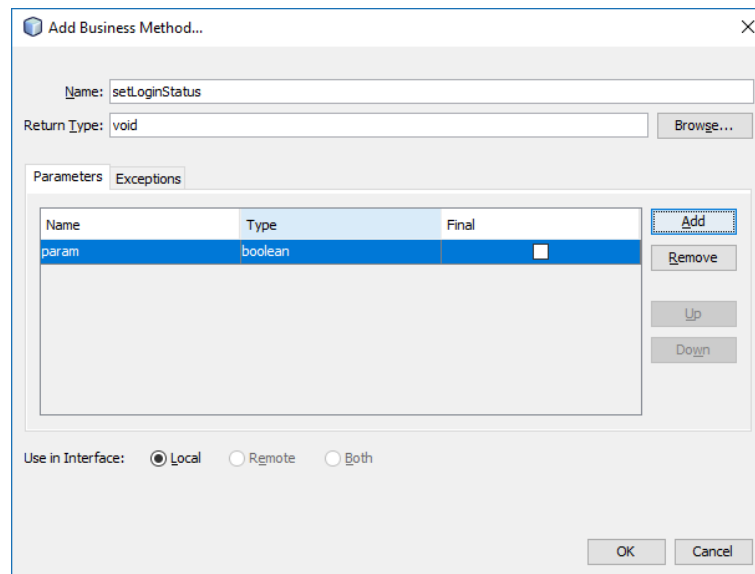
Gambar 3.13 Add Business Methode pada Text Editor

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

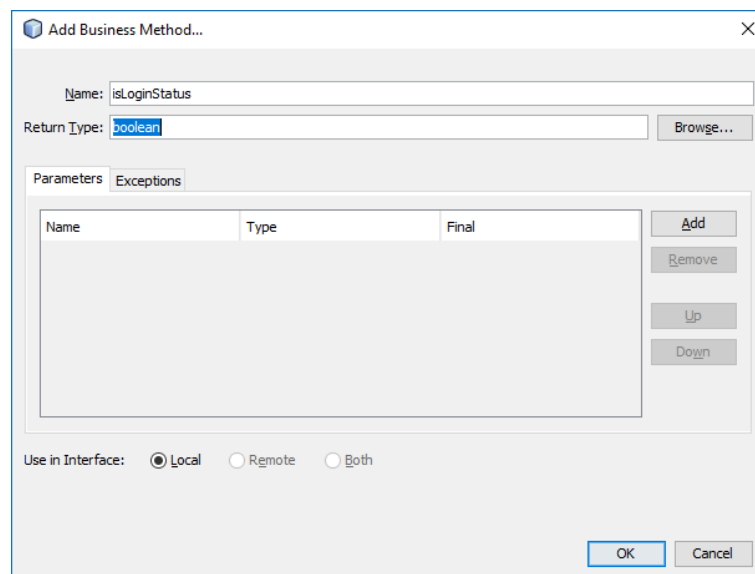


Gambar 3. 14 Membuat Method Login dengan Parameter NIM dan Nama

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



Gambar 3.15 Membuat Method setLoginStatus dengan *return type* void



Gambar 3.16 Membuat Method isLoginStatus dengan return type boolean

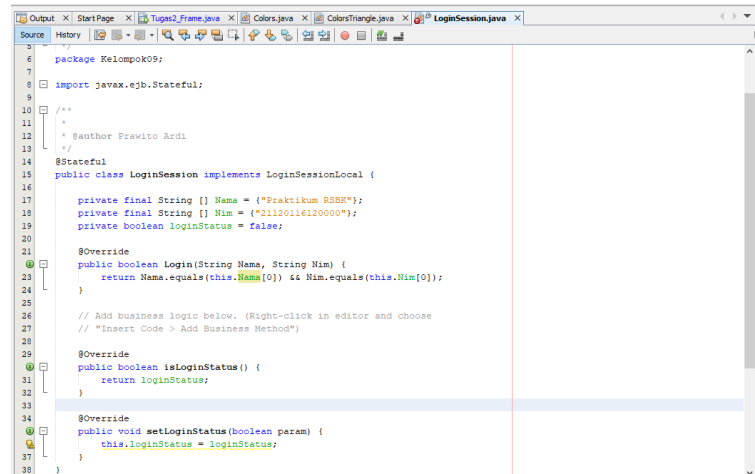
11. Tambahkan source code berikut didalam class LoginSession

```
private final String [] Nama = {"Prawito"};
private final String [] Nim = {"21120116120019"};
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() {
```

```

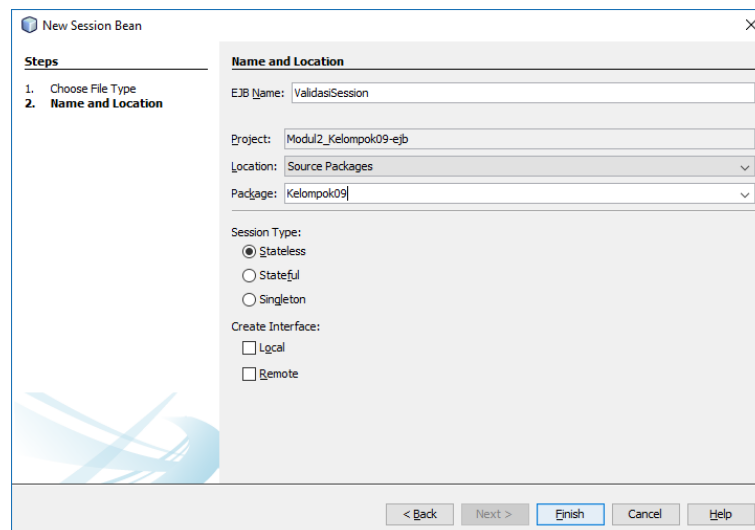
        return loginStatus;
    }
    @Override
    public void setLoginStatus(boolean loginStatus) {
        this.loginStatus = loginStatus;
    }
}

```



Gambar 3.17 Hasil Insert *Source Code*

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



Gambar 3.18 Membuat Session Bean dengan EJB name validasiSession

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

```

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

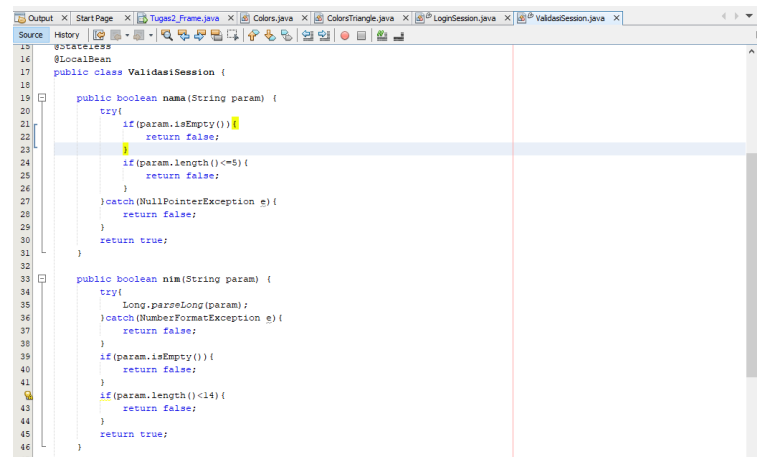
```

```

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

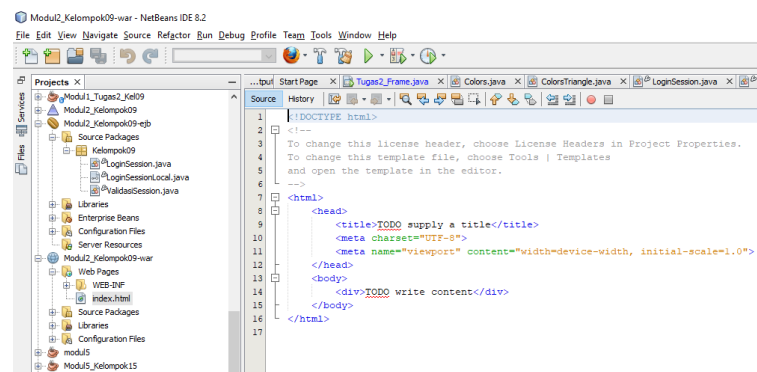
public boolean nim(String param) {
    try{
        Long.parseLong(param);
    } catch (NumberFormatException e) {
        return false;
    }
    if(param.isEmpty()){
        return false;
    }
    if(param.length()<14){
        return false;
    }
    return true;
}

```



Gambar 3.19 Hasil Insert Source Code

15. Buka file index.html di directory **Modul2\_Kelompok09-war>Web Page** dan timpa source code yang sudah ada dengan source code berikut.

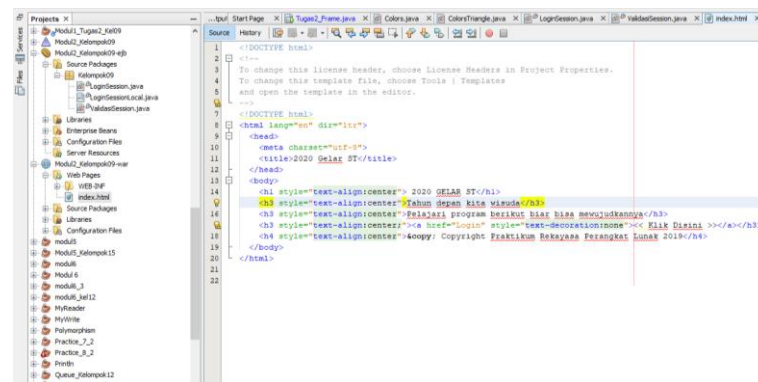


Gambar 3. 20 File Index in Modul2\_Kelompok09-war sebelumnya

```

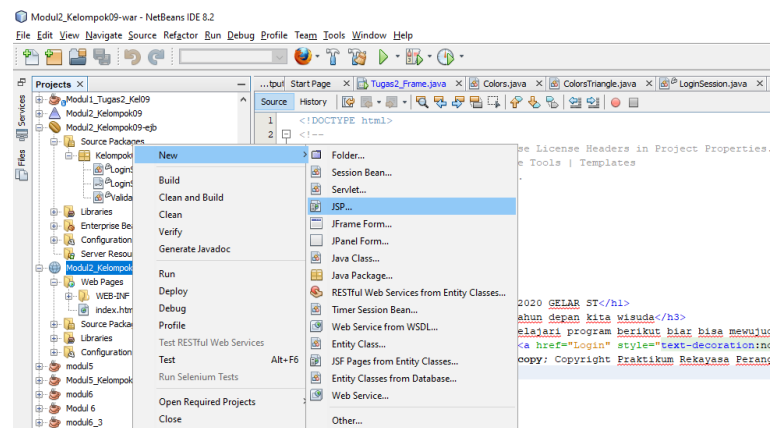
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>2020 Gelar ST</title>
  </head>
  <body>
    <h1 style="text-align:center"> 2020 GELAR ST</h1>
    <h3 style="text-align:center">Tahun depan kita wisuda</h3>
    <h3 style="text-align:center">Pelajari program berikut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

```



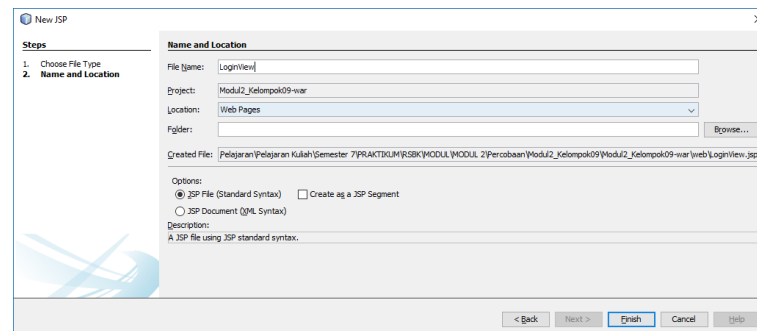
Gambar 3.21 File Index in Modul2\_Kelompok09-war sesudahnya

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



Gambar 3.22 Langkah membuat New JSP

17. Beri nama LoginView lalu klik finish



Gambar 3.23 Mengatur File Name pada New JSP

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

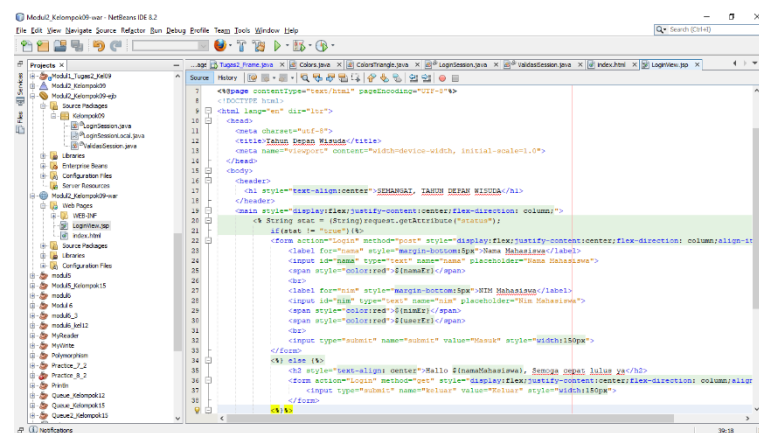
```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Tahun Depan Wisuda</title>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <header>
      <h1 style="text-align:center">SEMANGAT, 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="Masuk" style="width:150px">
        </form>
      <%> else {%>
```



```

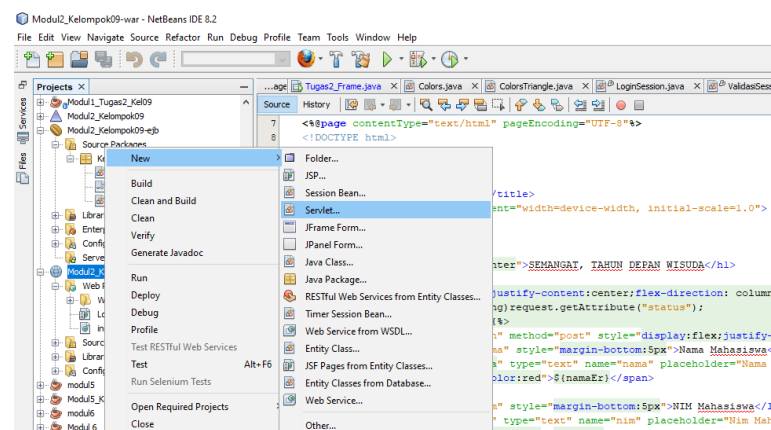
        <h2 style="text-align: center">Hallo
        ${namaMahasiswa}, Semoga cepat lulus ya</h2>
        <form action="Login" method="get"
        style="display: flex; justify-content: center; flex-direction:
        column; align-items: center;">
            <input type="submit" name="keluar"
            value="Keluar" style="width: 150px">
        </form>
        <%}%>
    </main><br>
    <footer style="text-align: center">&copy; Copyright
    Praktikum Rekayasa Perangkat Lunak 2019</footer>
</body>
</html>

```



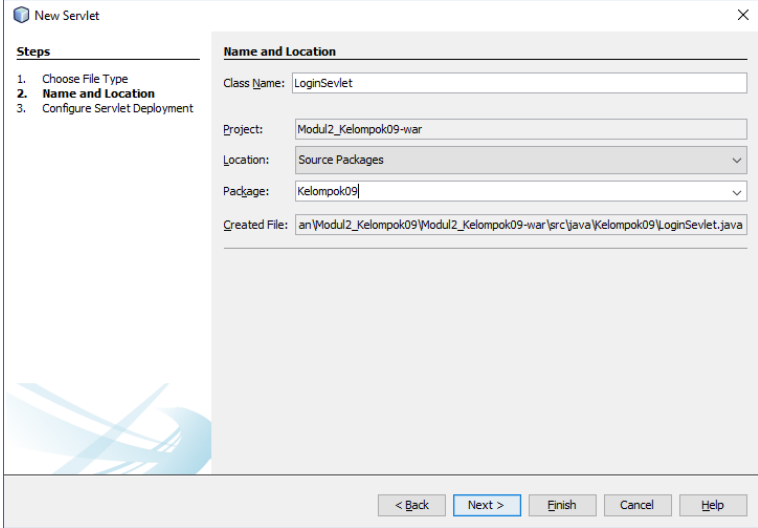
Gambar 3.24 File LoginView in Modul2\_Kelompok09-war

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



Gambar 3.25 Membuat New Servlet pada Modul2\_Kelompok09-war

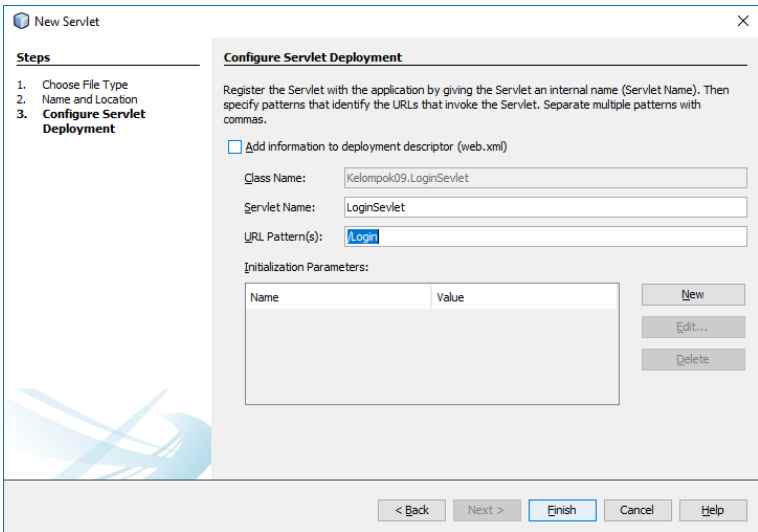
20. Beri nama LoginServlet pada kolom **Class Name** dan Kelompok09 pada package.



The screenshot shows the 'New Servlet' dialog box with the 'Name and Location' tab selected. The 'Steps' list on the left indicates the current step is '2. Name and Location'. The 'Class Name' field is set to 'LoginServlet'. The 'Project' field is set to 'Modul2\_Kelompok09-war'. The 'Location' dropdown is set to 'Source Packages'. The 'Package' dropdown is set to 'Kelompok09'. The 'Created File' path is shown as 'an\Modul2\_Kelompok09\Modul2\_Kelompok09-war\src\java\Kelompok09\LoginServlet.java'. At the bottom, there are buttons for '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'.

Gambar 3.26 Mengatur Class Name dan Package pada file New Servlet

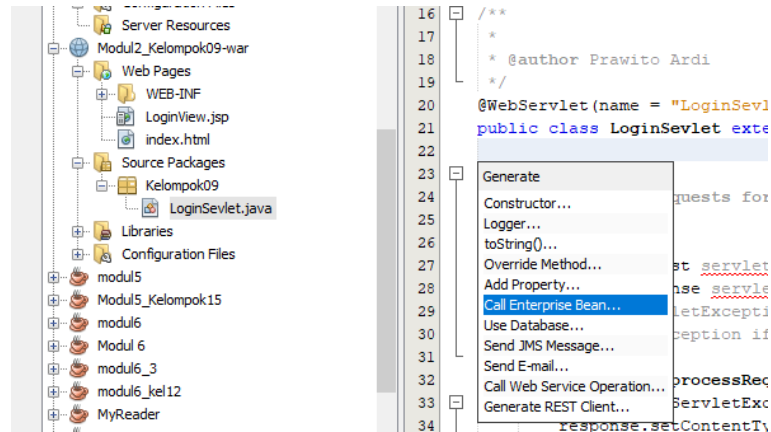
21. Pastikan **Add information** di uncheck dan ganti URL Pattern(s) menjadi /Login



The screenshot shows the 'New Servlet' dialog box with the 'Configure Servlet Deployment' tab selected. The 'Steps' list on the left indicates the current step is '3. Configure Servlet Deployment'. The 'Add information to deployment descriptor (web.xml)' checkbox is unchecked. The 'Class Name' field is set to 'Kelompok09.LoginServlet'. The 'Servlet Name' field is set to 'LoginServlet'. The 'URL Pattern(s)' field is set to '/Login'. Below these fields is a table for 'Initialization Parameters' with columns 'Name' and 'Value'. To the right of the table are buttons for 'New', 'Edit...', and 'Delete'. At the bottom, there are buttons for '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'.

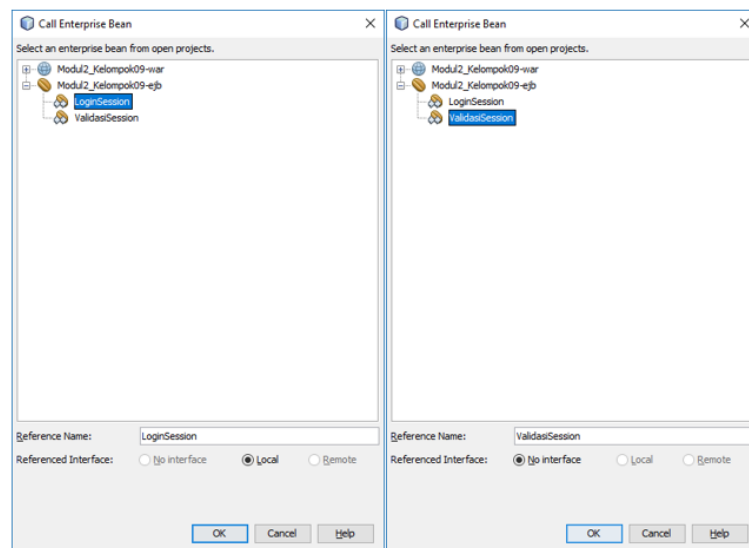
Gambar 3.27 Mengatur URL Pattern

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



Gambar 3.28 Klik kanan pada Text Editor - Insert Code dan pilih Call Enterprise Bean

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



Gambar 3.29 Import file LoginSession dan ValidasiSession

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

```
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**.

```
validasiSession = new ValidasiSession();

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

26. 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");
    }

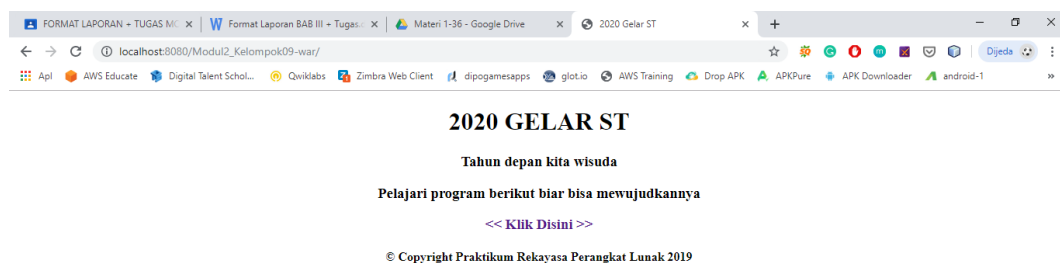
    if (loginSession.isLoginStatus()) {
        request.setAttribute("status", "true");
        RequestDispatcher rd =
getServletContext().getRequestDispatcher("/LoginView.jsp");
        rd.forward(request, response);
    } else {
        request.setAttribute("status", "false");
        RequestDispatcher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

27. Jalankan project enterprise dengan cara klik kanan project enterprise lalu klik pilih run.

### 3.5 Hasil Percobaan

Pada praktikum kali ini kita akan membuat program sederhana menggunakan java yang di hubungkan dengan web server dengan menggunakan Glassfish server. Pada percobaan ini kita juga menggunakan konsep komponen statefull dan stateless. Penggunaan session bean pada pemrograman ini digunakan untuk mendeklarasikan suatu session dengan beberapa parameter yang ada di dalamnya dan pada percobaan ini membuat session bean Login, isLoggedIn dan set Login Status. Penggunaannya adalah ketika kita mencoba masuk pada program dimana ketika men input parameter nama dan nim akan di cek terlebih dahulu.

Kemudian membuat session bean baru yaitu ValidasiSession, session yang berisi beberapa validasi dan kondisi inputan berupa nama misal harus memiliki karakter lebih dari 5 dan nim yang memiliki atribut karakter tidak boleh lebih dari 14 karakter. Selanjutnya web page yang nantinya digunakan untuk menampilkan halaman dengan isi file html dan css sederhana pada. Terdapat button “Klik Disini” yang akan menuju href Login (halaman view login yang dibuat pada Web Page) dan berikut adalah tampilan dari file index.html



Gambar 3.30 Halaman tampilan file index.html pada web

Selanjutnya terdapat penggunaan file JSP (Java Enterprise Edition) dengan membuat sebuah tampilan view web yaitu Login View. Pada file ini juga berisi dokumen html dan css sederhana dan dengan beberapa pemanggilan id atau kondisi inputan yang di deklarasikan pada file servlet. Terdapat 2 kolom inputan dengan masing-masing placeholder NIM Mahasiswa dan Nama Mahasiswa serta 1 button

Masuk dengan 1 button keluar di halaman selanjutnya ketika kondisi sudah berhasil masuk.

Terdapat penggunaan Servelt yang berisi source dari validasiSession dan beberapa source lain, dan pada Servelt ini menggunakan konsep komponen stateless dan statefull yang digunakan untuk file ini yaitu berupa file LoginSession, validasiSession, serta setLoginSession. Terdapat beberapa pengkondisian pada pendeklarasian beberapa variabel dan parameter. Misal pada method loginSession terdapat `loginSession.Login(nama, nim)` { `loginSession.setLoginStatus(true)` }, dimana jika kondisi nama dan nim dan yang di inputan benar atau dalam kondisi true maka akan bisa masuk dan terdapat beberapa kondisi lainnya. Dan berikut adalah



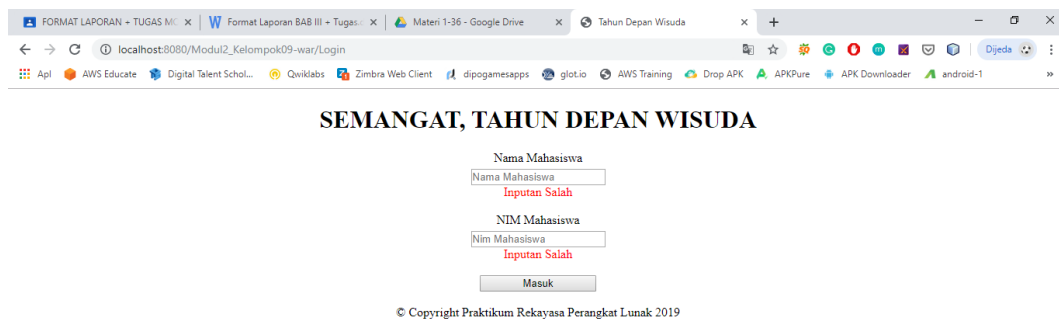
Gambar 3.31 Tampilan Awal LoginView



Gambar 3.32 Tampilan Inputan yang benar pada LoginView



Gambar 3.33 Tampilan dengan kondisi inputan Nama atau NIM salah



Gambar 3.34 Tampilan dengan kondisi inputan Nama dan NIM salah



Gambar 3.35 Tampilan dengan kondisi inputan Nama dan NIM benar

## 3.6 Tugas dan Pembahasan

### 3.6.1 Tugas

Pada tugas EJB Session Beans ini, membuat sebuah program sederhana java perihal penerapan atau penggunaan session bean, komponen stateful dan stateless maupun java servlet. Program sederhana ini akan menampilkan sebuah halaman dengan fungsi untuk mencari list nama dan nim mahasiswa yang kemudian akan di cek dalam fungsi apakah nama dan nim mahasiswa ini terdapat pada array list atau tidak. Dan satu fungsi lainnya yaitu menambahkan fungsi tambah mahasiswa dengan form nama dan nim yang kemudian data ini akan tersimpan sementara ketika halaman belum di *clean and build* ulang.

Percobaan sederhana ini terdapat beberapa file program yaitu file session bean cariMahasiswa, dimana pada file ini mendeklarasikan fungsi array list dengan variable nama dan nim dengan tipe data String. Kemudian terdapat beberapa method checkNama dan checkNim dengan fungsi for (perulangan) terhadap nilai input nama nim mahasiswa serta kondisi get data nama/nim dalam array list yang kemudian akan ditampilkan pada fungsi return yaitu `return "Nama Mahasiswa: "+mahasiswa.get(i).nama+" (" +mahasiswa.get(i).nim+" ) alhamdulillah anak ini lulus !!";`. Dan ini berlaku juga terhadap method checkNim yang dimana variabel nim menjadi objek get dan pengecekan kondisi. Serta terdapat method search dengan variable param serta pengkondisian dari pengecekan nama dan nim dalam fungsi search nantinya. Dan berikut adalah source dari file session beans cariMahasiswa dan cariMahasiswaLocal.

```
package Kelompok09;

import javax.ejb.Stateful;
import java.util.ArrayList;

@Stateful
public class cariMahasiswa implements cariMahasiswaLocal {
    ArrayList<Mahasiswa> mahasiswa = new ArrayList<Mahasiswa>();

    @Override
    public void isiData(String nama, String nim){
        mahasiswa.add(new Mahasiswa (nama,nim));
    }

    public cariMahasiswa(){
        isiData("Prawito", "21120116120019");
    }
}
```



```

isiData("Busyroo Busyairie ", "21120116140000");
isiData("SEPTI NURNA ALFIANI", "21120116120019");
isiData("DEMARA RAMADHANI ", "21120116140000");
isiData("MONANZARIFA YONANTA", "21120116120019");
isiData("RATNA YULI HIMAWATI ", "21120116140000");
isiData("MUTIARA VICTORINA M. ", "21120116140000");
}

private String checkNama(String param){
    for (int i = 0; i< mahasiswa.size(); i++) {
        if (param.equals(mahasiswa.get(i).nama)) {
            return "Nama Mahasiswa :
"+mahasiswa.get(i).nama+" ("+mahasiswa.get(i).nim+")
alhamdullilah anak ini lulus !!";
        }
    }
    return null;
}

private String checkNim(String param){
    for (int i = 0; i< mahasiswa.size(); i++) {
        if (param.equals(mahasiswa.get(i).nim)) {
            return "Nama Mahasiswa :
"+mahasiswa.get(i).nama+" ("+mahasiswa.get(i).nim+")
alhamdullilah anak ini lulus !!";
        }
    }
    return null;
}

@Override
public String search(String param){
    if (checkNama(param) != null){
        return checkNama(param);
    }
    else if (checkNim(param) != null) {
        return checkNim(param);
    }
    else {
        return "No";
    }
}
};
}

```

```

package Kelompok09;

import javax.ejb.Local;

/**
 *
 * @author Prawito
 */
@Local
public interface cariMahasiswaLocal {

```

```
String search(String param);  
void isiData(String nama, String nim);  
}
```

Selanjutnya terdapat file class mahasiswa yang menjelaskan fungsi setter dan getter dari parameter atau variable nama dan nim. Dan berikut adalah source code filenya.

```
package Kelompok09;  
  
/**  
 *  
 * @author Prawito  
 */  
public class Mahasiswa {  
    String nama,nim;  
    public Mahasiswa(String nama, String nim) {  
        this.nama = nama;  
        this.nim = nim;  
    }  
    public String getNama() {  
        return nama;  
    }  
    public String getNim() {  
        return nim;  
    }  
    public void setNama(String nama) {  
        this.nama = nama;  
    }  
    public void setNim(String nim) {  
        this.nim = nim;  
    }  
}
```

Setelah komponen session bean dibuat, terdapat file lain dalam program sederhana ini yaitu index.html yang berfungsi sebagai file dengan output/menampilkan halaman web, kemudian file search.jsp merupakan file untuk menampilkan halaman web yang akan menampilkan beberapa fungsi form input dan dengan fungsi button masing-masing yang nantinya akan di deklarasikan pada file servlet. Pada file search.jsp ini juga menggunakan beberapa fungsi html dan css yang digunakan untuk memperindah fungsi tampilan pada halaman web. Berikut adalah file index.html

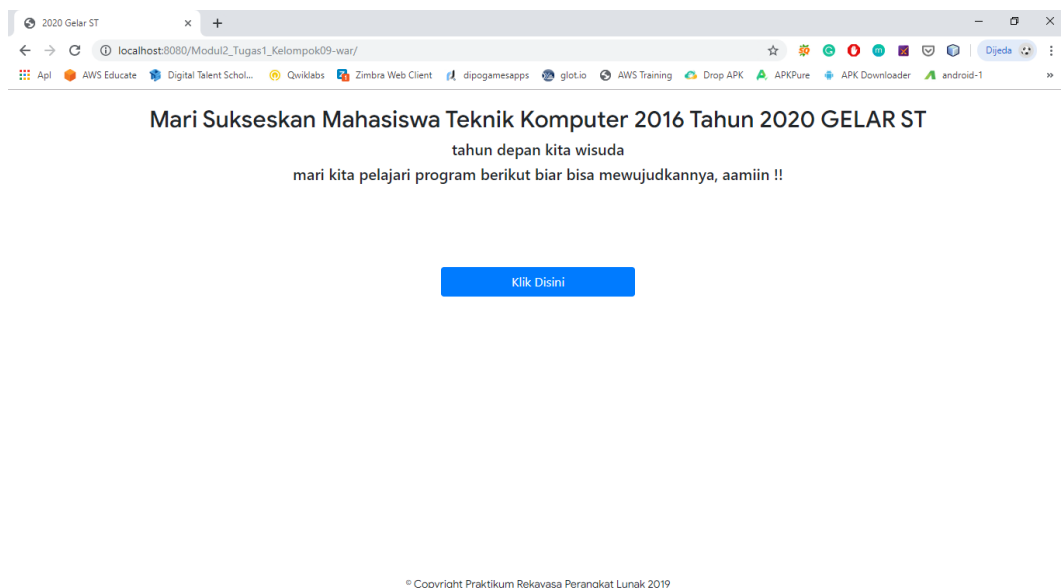
```
<html lang="en" dir="ltr">  
  <head>  
    <link          rel="stylesheet"          type="text/css"  
href="//fonts.googleapis.com/css?family=Open+Sans" />  
    <link          rel="stylesheet"  
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/boo  
tstrap.min.css"          integrity="sha384-
```

```

ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T
" crossorigin="anonymous">
  <meta charset="utf-8">
  <title>2020 Gelar ST</title>
</head>
<body>
  <h1 style="text-align:center; margin-top: 20px; font-size:
30px; font-family: Google Sans;"> Mari Sukseskan Mahasiswa Teknik
Komputer 2016 Tahun 2020 GELAR ST</h1>
  <h5 style="text-align:center">tahun depan kita wisuda</h5>
  <h5 style="text-align:center">mari kita pelajari program
berikut biar bisa mewujudkannya, aamiin !!</h5>
  <h3 style="text-align:center;"><a href="search" class="btn
btn-primary" style="text-decoration:none; width: 250px;margin-
top: 100px;">Klik Disini</a></h3>
  <footer style="text-align:center; margin-top:360px; font-
family: Google Sans;font-size:14px; "> <center> &copy; Copyright
Praktikum Rekayasa Perangkat Lunak 2019</center></footer>
</body>
</html>

```

Berisi file html dan sedikit fungsi css yang akan menghasilkan output tampilan mulai dari title “2020 Gelar ST”, kemudian text dengan header 1 “Mari Sukseskan Mahasiswa Teknik Komputer 2016 Tahun 2020 GELAR ST” dengan beberapa style yang digunakan untuk memperindah tampilan, hingga tombol “Klik Disini” yang akan mengarah ke halaman selanjutnya yang merupakan penggunaan fungsi href. Dan css class button btn btn-primary yang diambil dari bootstrap. Hasil output tampilan.



Gambar 3.36 Hasil Output Tampilan file index.html

Selanjutnya adalah file search.jsp yang juga merupakan file yang memiliki output tampilan pada web namun dengan fungsi dan kondisi tentunya, misal fungsi atau tambah data mahasiswa. Sama dengan file index.html, pada file search.jsp ini akan mendeklarasikan mengenai beberapa form action mulai dari Cari dan Tambah, pengecakan input Nama atau NIM serta form input nama dan nim untuk menambahkan data baru pada array list yang bersifat sementara (ketika di clean and build data akan hilang). Pada form pertama ini memiliki id “nama” dengan placeholder “jangan lupa masukkan nama atau nim ya !” yang diikuti dengan button cari sehingga ketika kita sudah memberikan nama/nim pada form serta memberikan fungsi button cari maka akan menampilkan text sesuai dengan inputan yang diberikan yang di cek pada array list apabila kondisi true dan menampilkan text “Mahasiswa Tidak Ditemukan” apabila kondisinya false.

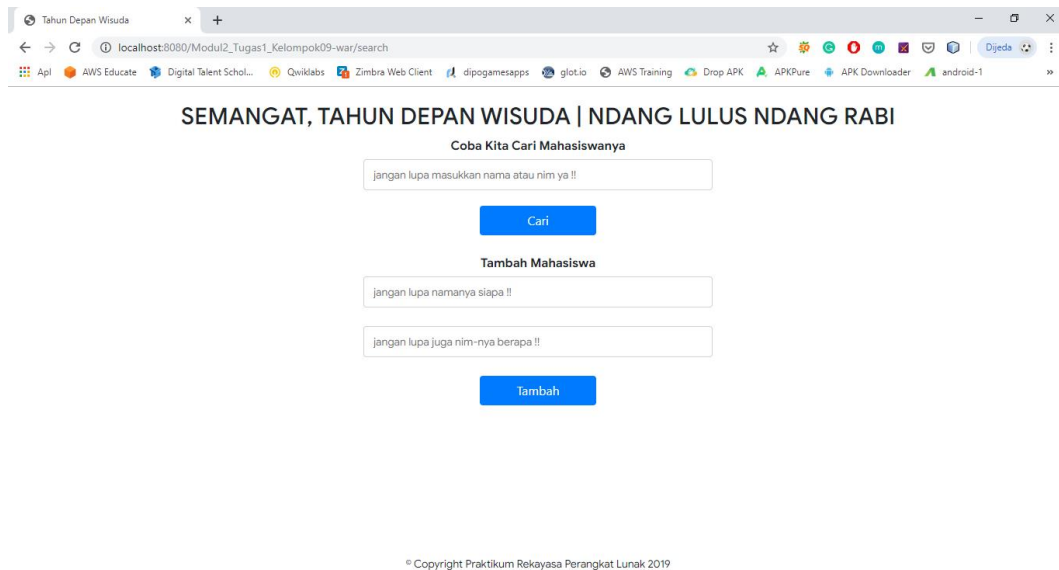
Kemudian terdapat form action berupa button Tambah dan form input Nama dan Nim, dimana pada fungsi ini kita menginputkan nama dan nim yang nantinya akan disimpan sementara pada fungsi post pada file servlet. Dan sifat array list atau data yang diinputkan bersifat sementara dan apabila di refresh data yang tersebut menghilang karena ini pengaruh penggunaan dari komponen java stateless. Dan berikut adalah source dan hasil output dari file search.jsp.

```
<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>
    <link          rel="stylesheet"          type="text/css"
href="//fonts.googleapis.com/css?family=Open+Sans" />
    <link          rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/boo
tstrap.min.css"          integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxt2MZw1T
" crossorigin="anonymous">
    <header>
      <h1 style="text-align:center; margin-top: 20px; font-size:
30px; font-family: Google Sans;">SEMANGAT, TAHUN DEPAN WISUDA |
NDANG LULUS NDANG RABI</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-
family:  Google  Sans;font-weight:  bold;">Coba  Kita  Cari
Mahasiswanya</label>
            <input  id="nama"  type="text"  name="param"
placeholder="jangan lupa masukkan nama atau nim ya !!"
style="font-family:  Google  Sans;  width:  450px;height:40px;
display:block;padding:.375rem  .75rem;font-size:14px;  line-
height:1.5; color:  #495057; background-color:#fff; background-
clip: padding-box; border:1px solid #ced4da; border-radius:
.25rem;  ">
            <span style="color:red; font-family: Google Sans;
font-weight:  bold;  margin-top:  10px;  margin-bottom:
10px;">${show}</span>
            <input type="submit" name="submit" value="Cari"
class="btn btn-primary" style="width:150px">
        </form>
        <br>
        <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-
family: Google Sans;font-weight: bold;">Tambah Mahasiswa</label>
            <input  id="nama2"  type="text"  name="nama"
placeholder="jangan lupa namanya siapa !!" style="font-family:
Google  Sans;  width:  450px;height:40px;
display:block;padding:.375rem  .75rem;font-size:14px;  line-
height:1.5; color:  #495057; background-color:#fff; background-
clip: padding-box; border:1px solid #ced4da; border-radius:
.25rem;">
            <br>
            <input  id="nim"  type="text"  name="nim"
placeholder="jangan lupa juga nim-nya berapa !!" style="font-
family:  Google  Sans;  width:  450px;height:40px;
display:block;padding:.375rem  .75rem;font-size:14px;  line-
height:1.5; color:  #495057; background-color:#fff; background-
clip: padding-box; border:1px solid #ced4da; border-radius:
.25rem;">
            <br>
            <input type="submit" name="submit" value="Tambah"
class="btn btn-primary" style="width:150px">
        </form>
    </main><br>
    <footer style="text-align:center; margin-top:170px; font-
family: Google Sans;font-size:14px; "> <center> &copy; Copyright
Praktikum Rekayasa Perangkat Lunak 2019</center></footer>
    </body>
</html>

```



Gambar 3.37 Hasil Output Tampilan file search.jsp

Dan terakhir adalah file `cariServlet` yang berisi pendeklarasian dari beberapa fungsi pada tampilan halaman web mulai dari fungsi tambah, cari, pengkondisian ketika inputan diberikan pada fungsi cara, dan penyimpanan data sementara dari form input nama dan nim. Pertama kita mengambil session beans `cariMahasiswa` yang telah dibuat untuk digunakan pada file servlet ini. Kemudian fungsi atau method `doPost` yang mendeklarasikan beberapa kondisi seperti syntax `if (print == "No") { request.setAttribute("show", "Mahasiswa tidak terdaftar"); }` yang dimana apabila parameter yang kita berikan tidak terdapat pada array list maka akan menampilkan kalimat “Mahasiswa Tidak Terdaftar”, fungsi method ini berhubungan dengan file session yang ada pada `cariMahasiswa`, serta akan menampilkan kalimat sesuai dengan variabel param yang inputkan dan dicek pada array list ternyata true (data ada pada array list) dan akan menampilkan kalimat yang sesuai dengan pendeklarasian pada file session beans `cariMahasiswa` sebelumnya yaitu `"Nama Mahasiswa : "+mahasiswa.get(i).nama+" (" +mahasiswa.get(i).nim+" ) alhamdullilah anak ini lulus !!";`.

Selanjutnya adalah method `doGet` yang mendeklarasikan fungsi tambah data pada halaman web dimana variabel nama dan nim menyimpan sementara data input yang diberikan yang kemudian akan disimpan sementara pada session beans `cariMahasiswa` sehingga ketika halaman belum di refresh maka data yang

ditambahkan sebelumnya ini tersimpan yang kemudian saat data ini di inputkan pada form cari maka akan menampilkan kalimat “Nama Mahasiswa : “Data yang diinputkan (“Nim yang diinputkan) . Dan apabila halaman di refresh data ini akan hilang. Dan berikut adalah source code file servlet.

```
package kelompok09;

import Kelompok09.cariMahasiswaLocal;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet(name = "searchServlet", urlPatterns = {"/search"})
public class searchServlet extends HttpServlet {

    cariMahasiswaLocal cariMahasiswa1 =
lookupcariMahasiswaLocal();
    cariMahasiswaLocal cariMahasiswa =
lookupcariMahasiswaLocal();

    /**
     * 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");
        RequestDispatcher rd =
getServletContext().getRequestDispatcher("/search.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 {
        String nama = request.getParameter("nama");
        String nim = request.getParameter("nim");
        cariMahasiswa.isiData(nama, nim);
        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 param = request.getParameter("param");
        String print = cariMahasiswa.search(param);

        if (print == "No") {
            request.setAttribute("show", "Mahasiswa tidak
terdaftar");
        }
        else {
            request.setAttribute("show", print);
        }
        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 cariMahasiswaLocal lookupcariMahasiswaLocal() {
        try {
            Context c = new InitialContext();
            return (cariMahasiswaLocal)
c.lookup("java:global/Modul2_Tugas1_Kelompok09/Modul2_Tugas1_Kelompok09-ejb/cariMahasiswa!Kelompok09.cariMahasiswaLocal");
        } catch (NamingException ne) {

```



```

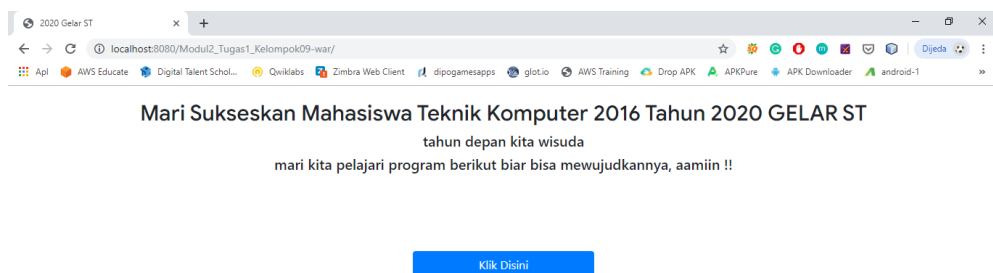
        Logger.getLogger(getClass().getName()).log(Level.SEVERE,
        "exception caught", ne);
            throw new RuntimeException(ne);
        }
    }
}

```

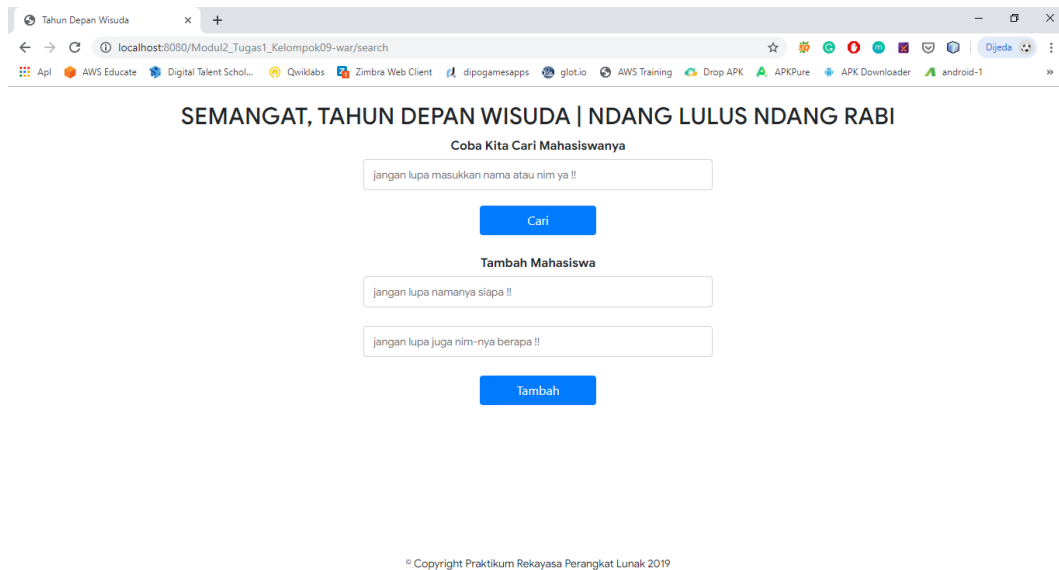
Kemudian pada program sederhana ini membuat fungsi tambah dan cari dalam satu url, karena ketika memberikan input pada form tambah, data yang masuk akan tersimpan sementara yang kemudian apabila mencoba mencari pada fungsi cari maka fungsi ini dalam kondisi true, dan apabila membuat fungsi tambah dan cari berbeda url maka data ini akan hilang ketika di refresh atau di build dan clean ulang.

Pada program sederhana ini kita membuat 1 method public yaitu Search dan 2 method private checkNama dan checkNim. Kemudian penggunaan 2 method private didasarkan pada variabel data yang kita gunakan yaitu Nama dan NIM, jadi ketika input yang diberikan pada form berupa nama maka method checkNama akan menjalankan fungsinya, dan ketika input yang diberikan berupa NIM maka method checkNim lah yang akan di jalankan. Kemudian satu method public yaitu Search ini mendeklarasikan pengkondisian dari method private checkNama dan checkNim yang kita gunakan, jadi berdasarkan input yang diberikan method ini lah yang memberikan perintah apabila inputan itu Nama maka dijalankan method nama dan apabila inputan nim maka mengeksekusi method checkNim.

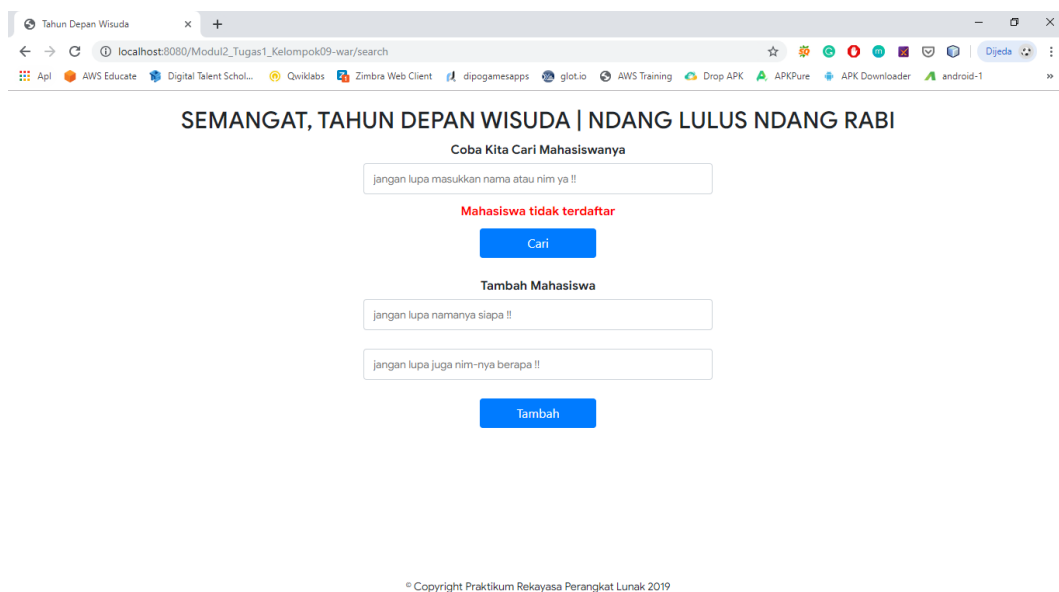
Dan berikut hasil output program:



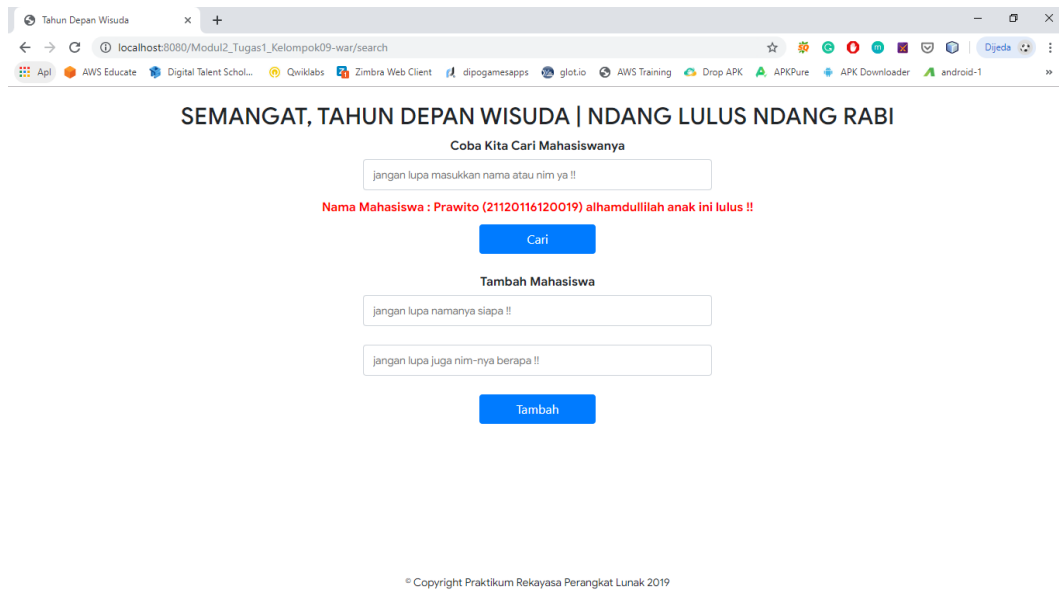
Gambar 3.38 Tampilan halaman awal program



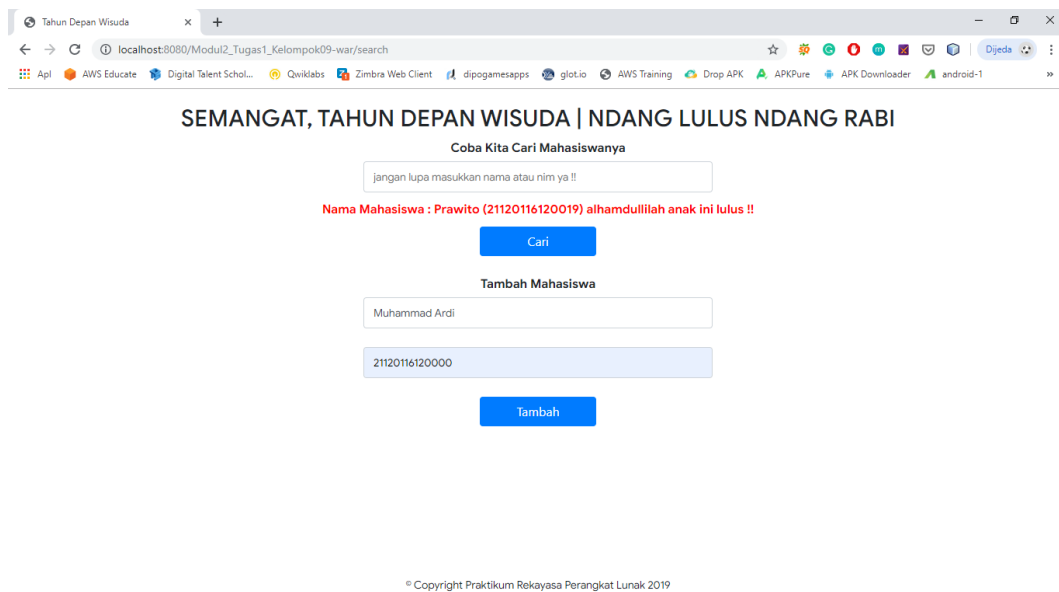
Gambar 3.39 Tampilan halaman fungsi search dan tambah data mahasiswa



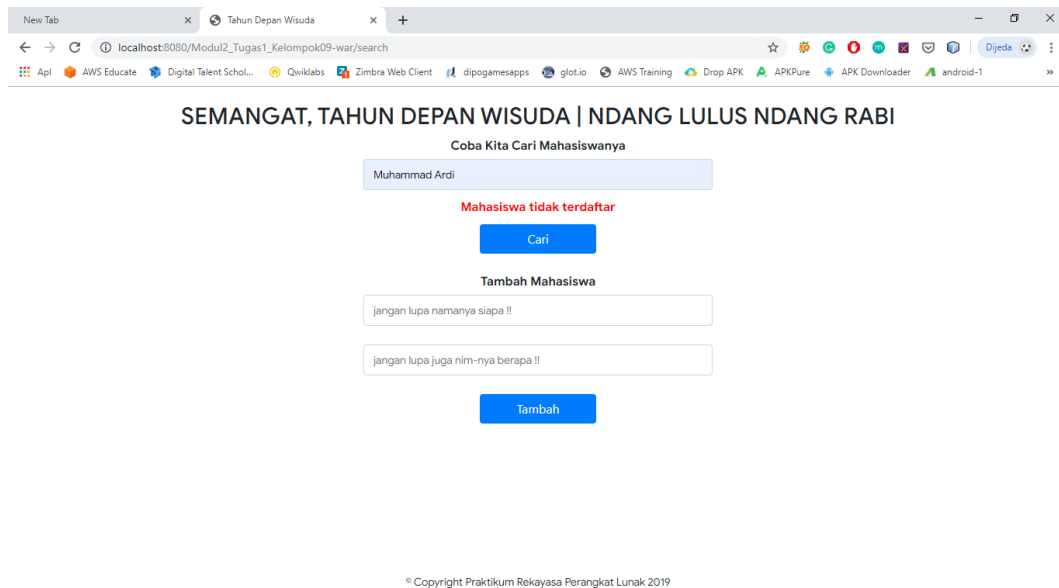
Gambar 3.40 Hasil Tampilan kondisi data yang di inputkan tidak ada dalam *array list*



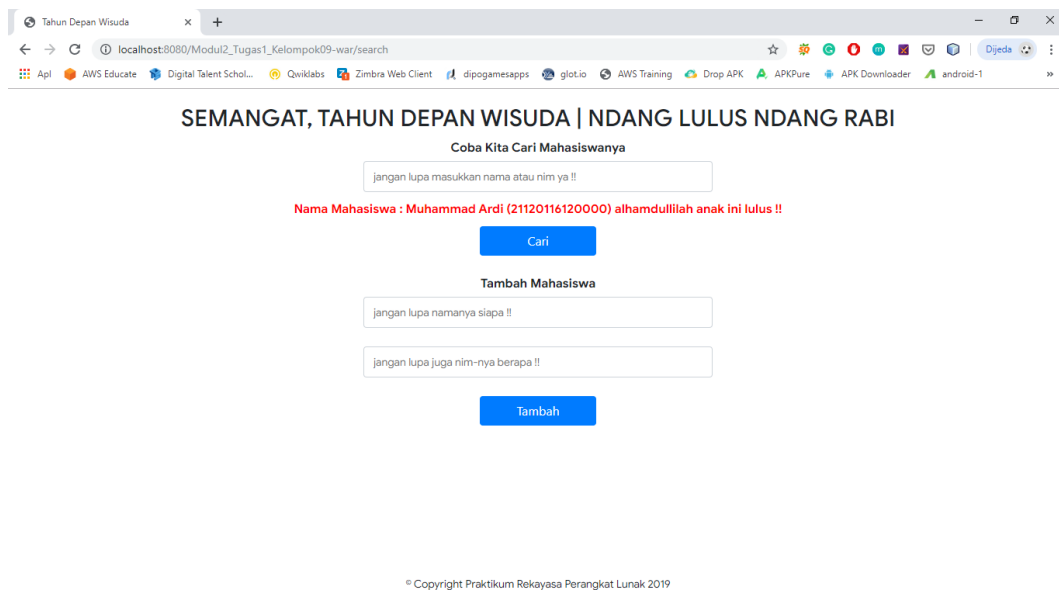
Gambar 3.41 Hasil Tampilan kondisi data yang di inputkan ada dalam array list



Gambar 3.42 Tampilan kondisi dimana mencoba untuk menambahkan data



Gambar 3.43 Kondisi dimana program di clean and build ulang, jadi data tidak tersimpan dalam *array list*



Gambar 3.44 Kondisi dimana program belum di clean and build ulang, jadi data terdapat dalam *array list*

Link Github : <https://github.com/Prawito-17/PraktikumRSBKKel08.git>

Nama File : MODUL 2 EJB SESSION BEAN

### 3.7 Kesimpulan

1. *Session bean* mempunyai 3 tipe yaitu *stateless*, *stateful*, dan *singleton session bean*.
2. *Glassfish* berfungsi untuk *webserver*, yang mana *webserver* merupakan sebuah perangkat dimana aplikasi kita dapat menerima *request* dan mengirimkan *response* dalam protokol HTTP.
3. Jika menggunakan *stateless session bean*, maka ketika di-refresh pada *browser* tidak dapat menampilkan data tersebut lagi. Dan ini terjadi ketika kita menambahkan data mahasiswa untuk nantinya di cek pada fungsi pencarian.
4. *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*.
5. Fungsi *servlet* adalah untuk membuat *web* dinamis bagi *user*.
6. Penggunaan *setter* dan *getter* digunakan dalam pembuatan *array list* dan *ArrayList* merupakan *array* dinamis yang bisa digunakan jika kita membutuhkan *array* dengan batas maksimum yang tak terbatas.