

COMP6051 - Web Programming (Final Exam)

Name : Mike Christ Heru

NIM : 2001572080

Class : LA05

I. Esai (50%)

1. [10%] Jelaskan apa yang dimaksud dengan *Session tracking* dan bagaimana cara mengatasinya. Berikan contoh dalam bentuk skrip JSP.

Jawaban :

Session Tracking merupakan salah satu fitur dalam JSP untuk mempertahankan sesi dari *Web Client* dan *Web Server*. Dengan kata lain, *Session Tracking* akan menyimpan data dari pengguna dan apabila user melakukan permintaan (*request*) kepada server, maka server akan memperlakukan *request* tersebut sebagai *request* baru. *Session Tracking* yang paling umum digunakan adalah dengan menggunakan *Cookies*.

Berikut ini adalah skrip JSP mengenai *Session tracking*, dengan *cookies*.

```
<body>
<!--Main Page / Landing Page for any kind of user (Admin, User, Guest)-->
<%
    String username = "";
    String role = "";
    Integer id = 0;
    Integer code = -1;
    Integer save = 0; // code for Remember Me
    try{
        code = Integer.parseInt(request.getParameter("code"));
        save = Integer.parseInt(request.getParameter("save"));
    }catch(Exception e){
    }
    try{
        username = (String) session.getAttribute("username");
        id = (Integer)session.getAttribute("id");
        role = (String) session.getAttribute("role");
        // Set Cookie when login/register success
        Cookie name = new Cookie("username",username);
        Cookie userID = new Cookie("id",String.valueOf(id));
        if(save <= 0){
            name.setMaxAge(60*60);
            userID.setMaxAge(60*60);
        }
        else{
            name.setMaxAge(24*60*60);
            userID.setMaxAge(24*60*60);
        }
        response.addCookie(name);
        response.addCookie(userID);
    }catch(Exception e){
    }
}
```

Untuk skrip JSP tersebut melakukan *handle* terhadap *login user* pada halaman *index.jsp* dan apabila didapati bahwa terdapat *session* yang berisikan *username*, *id*, *role*, maka cookie yang berisikan *username* dan *id* dari user dan *response.addCookie* akan melakukan penyimpanan Cookie.

2. [10%] Berikan contoh penggunaan *URL rewriting* dan *Hidden field*. Jelaskan kegunaan implementasi kedua teknik tersebut serta kekurangannya.

Jawaban :

URL rewriting dan *Hidden field* merupakan salah satu teknik untuk melakukan *session management* dalam aplikasi web.

Hidden field sendiri memiliki kelebihan yaitu dapat meningkatkan *website security* karena dapat menyimpan dan melakukan *submit security tokens* atau rahasia tertentu. Namun di sisi lain, hanya informasi yang sifatnya tekstual yang bisa digunakan dan juga perlu adanya *form* tambahan untuk tiap pagenya (tentu ini akan menambah kompleksitas dari *halaman web*).

Contoh *hidden field* adalah pada gambar di bawah ini.

```
//File: index.html
<HTML>
  <HEAD>
    <TITLE>Submitting Hidden Fields</TITLE>
  </HEAD>
  <BODY>
    <H1>Submitting Hidden Fields</H1>
    <FORM ACTION="formAction.jsp" METHOD="POST">
      <INPUT TYPE="HIDDEN" NAME="HIDDEN" VALUE="Hello from JSP!">
      <INPUT TYPE="SUBMIT" VALUE="Submit">
    </FORM>
  </BODY>
</HTML>

////////////////////////////////////
//File: formAction.jsp
<HTML>
  <HEAD>
    <TITLE>Reading Hidden Controls</TITLE>
  </HEAD>
  <BODY>
    <H1>Reading Hidden Controls</H1>
    The hidden text is:
    <% out.println(request.getParameter("HIDDEN")); %>
  </BODY>
</HTML>
```

Gambar di atas menjelaskan bahwa *hidden field* akan menyimpan suatu informasi di dalam dirinya dan dapat diakses oleh page lain dengan menggunakan *form*

sehingga nantinya passing data-data (parameter dan nilai) akan lebih rapi dan tidak dapat dilihat secara langsung oleh pengguna (dibandingkan dengan *URL rewriting*).

Sedangkan *URL rewriting* sendiri merupakan salah satu fitur untuk *session management* yang mana memberikan *append token* atau *identifier* ke dalam URL yang sudah ada. Parameter yang diberikan akan dicantumkan ke dalam URL dan biasanya dipisahkan dengan tanda “?” pada url diikuti dengan nama parameter beserta nilai parameter tersebut.

Contoh *URL rewriting* cukup banyak dalam kehidupan sehari-hari. Misalnya saja *link* di bawah dari *website* Google untuk mencari kata kunci “binus” dari halaman <https://www.google.com> :

https://www.google.com/search?safe=strict&sxsrf=ALeKk03BT0zNJ69CaQ-_yMofwPeak14HPg%3A1593705829468&source=hp&ei=ZQX-XoTBGsjez7sPxOycYA&q=binus&btnK=Google+Search#btnK=Google%20Search

Contoh tersebut menyamakan parameter keyword menjadi “q” dengan nilai “binus”. Tiap parameter dan nilainya (dianggap 1 pasang) akan dipisahkan dengan tanda “&” (ampersand).

3. [10%] Jelaskan apa yang dimaksud dengan *JavaBean*. Berikan contoh penggunaan JavaBean di JSP untuk menampilkan data nama, telepon, dan email yang diinput user melalui *form*.

Jawaban :

JavaBeans merupakan kelas yang mengenkapsulasi beberapa objek menjadi sebuah objek yang bisa disebut sebagai beans. Hal ini cukup membantu dalam melakukan akses terhadap suatu objek dari berbagai lokasi. Terdapat beberapa elemen penting dalam JavaBeans seperti Constructor, Setter/Getter, dan lain sebagainya.

Contoh penggunaan JavaBean di JSP adalah pada dua gambar di bawah ini.

a. Pembuatan Class

```
package com.myexample;

public class StudentsBean implements java.io.Serializable {
    private String name = null;
    private String phone = null;
    private String email = null;

    public StudentsBean() {
    }
    public String getName(){
        return name;
    }
    public String getPhone(){
        return phone;
    }
    public String getEmail(){
        return email;
    }
    public void setName(String name){
        this.name = name;
    }
    public void setPhone(String phone){
        this.phone = phone;
    }
    public void setEmail(String email){
        this.email = email;
    }
}
```

b. Form sederhana dengan implementasi JavaBeans

```
<html>
<head>
  <title>get and set properties Example</title>
</head>

<body>
  <jsp:useBean id = "students" class = "com.myexample.StudentsBean">
    <jsp:setProperty name = "students" property = "name" value = "Rudy"/>
    <jsp:setProperty name = "students" property = "phone" value = "08123456789"/>
    <jsp:setProperty name = "students" property = "email" value = "rudy.001@binus.ac.id"/>
  </jsp:useBean>

  <p>Student Name:
    <jsp:getProperty name = "students" property = "name"/>
  </p>

  <p>Student Phone:
    <jsp:getProperty name = "students" property = "phone"/>
  </p>

  <p>Student Email:
    <jsp:getProperty name = "students" property = "email"/>
  </p>

</body>
</html>
```

4. [10%] Jelaskan apa yang dimaksud dengan JSTL dan bagaimana cara menggunakannya. Berikan contoh penggunaan *Core tag* dan *SQL tag*.

Jawaban :

JSTL atau *JSP Standard Tag Library* merupakan representasi dari suatu set yang berisikan *tags* yang berguna untuk menyederhanakan pengembangan dari JSP. Ada beberapa hal umum yang bisa dilakukan khususnya untuk spesialisasi pekerjaan seperti *data processing*, *database access*, *internationalization*, dan sebagainya.

Untuk penggunaan JSTL diperlukan adanya *taglib* yang direktif yang menjelaskan *JSTL core library*. Untuk pemakaian *libraries* apapun, perlu untuk *include* `<taglib>` directive di bagian atas tiap JSP.

Contoh penggunaan *core tag* adalah sebagai berikut :

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<html>
  <head>
    <title><c:if> Tag Example</c:if></title>
  </head>
  <body>
    <c:set var="salary" scope="session" value="${2000*2}"/>
    <c:if test="${salary > 2000}">
      <p>My salary is: <c:out value="${salary}"/></p>
    </c:if>
  </body>
</html>
```

Contoh penggunaan *SQL tag* adalah sebagai berikut :

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
    <title>&lt;sql:query&gt; Demo</title>
  </head>
  <body>
    <h1>&lt;sql:query&gt; Demo</h1>
    <sql:setDataSource var="myDS" driver="com.mysql.jdbc.Driver"
      url="jdbc:mysql://localhost:3306/test"
      user="root" password="" />

    <sql:query dataSource="${myDS}" var="citizens">
      SELECT * from citizens;
    </sql:query>
    <table border="1">
      <c:forEach var="row" items="${citizens.rows}">
        <tr>
          <td><c:out value="${row.ssn}" /></td>
          <td><c:out value="${row.first_name}" /></td>
          <td><c:out value="${row.last_name}" /></td>
          <td><c:out value="${row.address}" /></td>
          <td><c:out value="${row.telephone}" /></td>
        </tr>
      </c:forEach>
    </table>
  </body>
</html>
```


5. [10%] Jelaskan apa yang dimaksud dengan IP address, Web server, DNS, dan SSL.

Jawaban :

IP Address atau *Internet Protocol Address* merupakan label numerik yang dipasangkan pada suatu perangkat yang sudah terhubung dengan suatu jaringan komputer yang memakan protokol internet dalam berkomunikasi. Fungsi utama dari *IP Address* adalah sebagai *host* dan juga *location address*.

Web Server adalah *software* yang berfungsi sebagai penerima permintaan yang dikirimkan melalui *browser* dan kemudian akan ditampilkan pada pengguna sesuai dengan permintaan yang dikirimkan kepada *server*. Atau dengan kata lain, *web server* sebagai pusat kontrol yang fungsinya sebagai bagian yang memproses permintaan yang telah diterima dari *browser*.

DNS atau *Domain Name System* dapat dianalogikan sebagai buku telepon dalam dunia internet. Dalam dunia sehari-hari, pengguna seringkali melakukan akses secara daring dengan menggunakan nama domain seperti *google.com*, *facebook.com*, dan lain sebagainya. Namun, *web browsers* berinteraksi dengan menggunakan *IP Address*, sehingga harus ada yang menerjemahkan alamat domain dengan *IP Address*. Disinilah hadirnya *DNS* untuk menerjemahkan alamat domain yang diberikan pengguna. Tentunya ini memudahkan pengguna (dalam hal ini manusia) untuk tidak mengingat banyak angka-angka yang mendeskripsikan alamat *IP*.

SSL atau *Secure Socket Layer* merupakan suatu standar dalam keamanan teknologi yang menggunakan suatu *encrypted link* antara *server* dan *client* (biasanya *web server* dan *browser* atau *mail server* dengan *mail client*). Dengan adanya *SSL* ini memungkinkan adanya transmisi data-data sensitif seperti kata sandi, nomor pin, nomor kartu kredit, dan sebagainya. Dengan kata lain, *SSL* merupakan *security protocol* yang mengamankan banyak pengguna dalam berselancar di Internet terutama saat melakukan transaksi informasi.

II. Kasus (50%)

Project Aplikasi Web

Buatlah sebuah aplikasi web berbasis JSP dan MySQL untuk mengelola Perpustakaan Publik. Perpustakaan Publik ini dikelola tiga orang staf untuk menangani peminjaman buku maupun pengelolaan buku. Seseorang dapat meminjam buku jika sudah mendaftar menjadi anggota. Pendaftaran sebagai anggota juga dapat dilakukan melalui aplikasi ini.

Secara garis besar fitur aplikasi meliputi:

1. Fitur untuk Staf

Untuk dapat menggunakan fitur-fitur khusus Staf, seorang Staf harus melakukan *login* terlebih dahulu. Staf dapat melihat data Anggota, data Buku serta melakukan pengelolaannya (*insert, update, delete*), melayani transaksi peminjaman dan pengembalian buku, serta melihat denda yang masuk.

2. Fitur untuk Anggota

Untuk dapat menjadi Anggota, *user* harus mendaftar dulu menggunakan *form* pendaftaran dari aplikasi. Kemudian Anggota dapat masuk dengan *login* terlebih dahulu sesuai dengan *email* dan *password* yang sudah didaftarkan. Setelah *login*, selain dapat mengubah data pribadinya, seorang Anggota dapat pula melakukan pencarian buku dan melihat riwayat peminjaman buku.

Syarat dan Proses Peminjaman Buku :

1. Anggota hanya dapat meminjam 1 buah buku. Anggota tidak boleh meminjam buku jika ada buku yang belum dikembalikan.
2. Peminjaman 1 buah buku maksimal 7 hari. Keterlambatan pengembalian akan dikenakan denda sebesar 50,000 Rupiah.
3. Anggota membawa buku yang ingin dipinjam kepada Staf untuk diproses. Staf akan memeriksa status Anggota. Jika Anggota tidak ada penunggakan pengembalian buku maka transaksi peminjaman dilanjutkan. Jika Anggota ada tunggakan pengembalian buku, transaksi peminjaman tidak dilanjutkan.

Bobot penilaian :

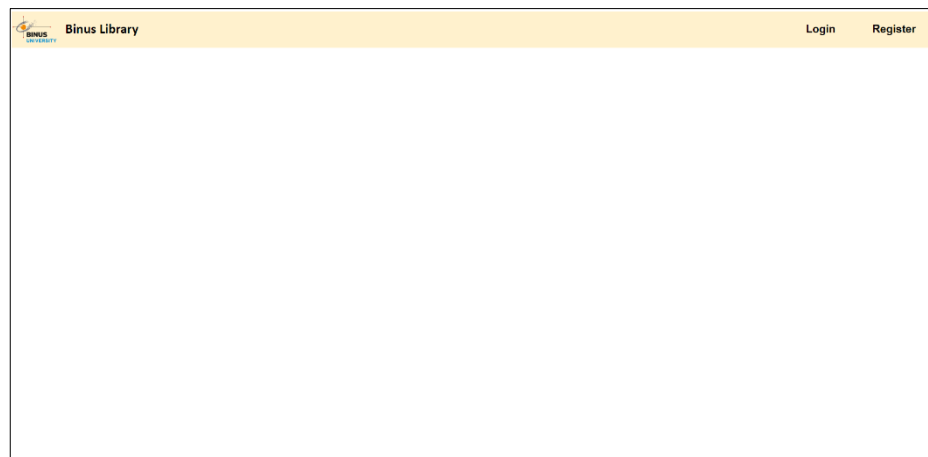
1. Keseluruhan fitur untuk Staf: *Login*, lihat data Anggota, kelola data Buku (*insert*, *update*, *delete*), fitur peminjaman dan pengembalian buku, serta melihat rincian denda yang masuk. [30%]
2. Keseluruhan fitur untuk Anggota: *Register*, *Login*, ubah data pribadi, cari buku, dan melihat riwayat peminjaman. [20%]

Buatlah aplikasi sesuai persyaratan di atas. Gunakan validasi data yang sesuai. Tambahkan asumsi Anda sendiri jika diperlukan (desain UI, nama *database*, nama tabel, tipe data, jumlah tabel, nama variabel, dan lainnya).

Hasil Dokumentasi :

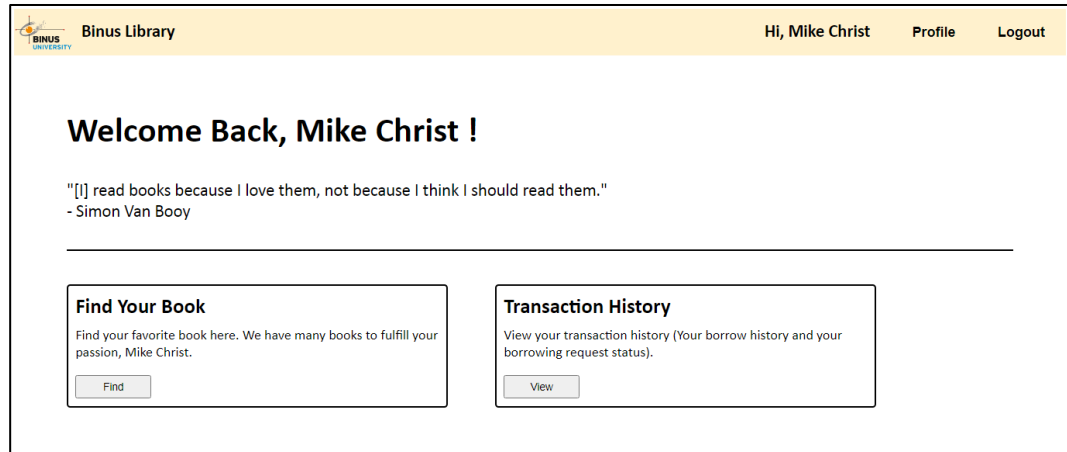
1. **Home Page** (*Guest* / G, *User* / U dan *Staff* / S) (index.jsp)

Halaman ini akan menampilkan tampilan utama (*Landing Page*). Jika pengguna merupakan *guest*, tampilan hanya menunjukkan *Header* yang berisikan logo, judul web, tombol *Login* dan *Register* (lihat gambar 2.1).



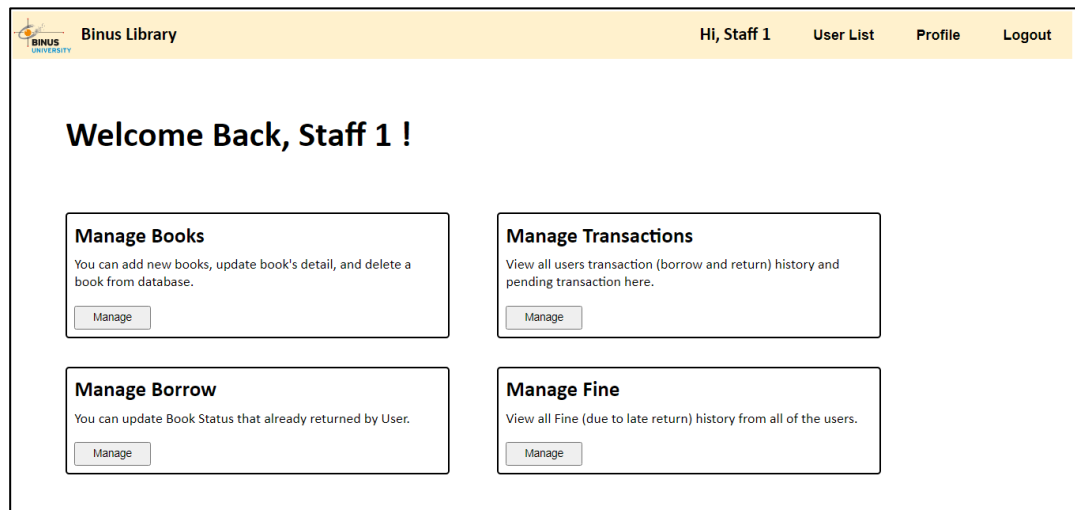
Gambar 2.1. Home Page (index.jsp) untuk role guest

Jika pengguna memiliki *role user*, tampilan index.jsp akan berubah menjadi seperti gambar 2.2. *user* akan diberikan akses ke halaman profil, pencarian buku, lihat histori transaksi, dan juga *logout*.



Gambar 2.2. Home Page (index.jsp) untuk role user

Jika pengguna memiliki *role staff*, tampilan index.jsp akan berubah menjadi seperti gambar 2.3. *staff* akan diberikan akses untuk *manage books*, *manage* transaksi, *manage* peminjaman, dan *manage* denda. Selain itu *staff* dapat melihat list pengguna, melakukan pembaruan data profil dirinya, dan *logout*.

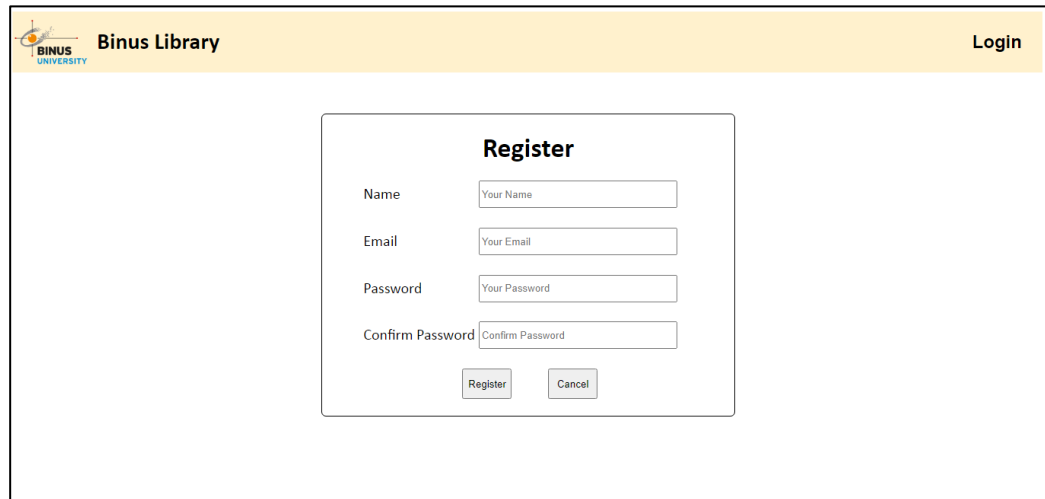


Gambar 2.3. Home Page (index.jsp) untuk role staff

2. Register Page (G) (register.jsp)

Halaman ini akan melayani pengguna yang ingin melakukan pendaftaran sebagai anggota perpustakaan (*user*). *Guest* akan diminta data-data pendaftaran berupa nama, alamat *email*, dan password beserta konfirmasi password. Apabila *email* pernah digunakan sebelumnya, notifikasi permintaan penggantian email baru akan

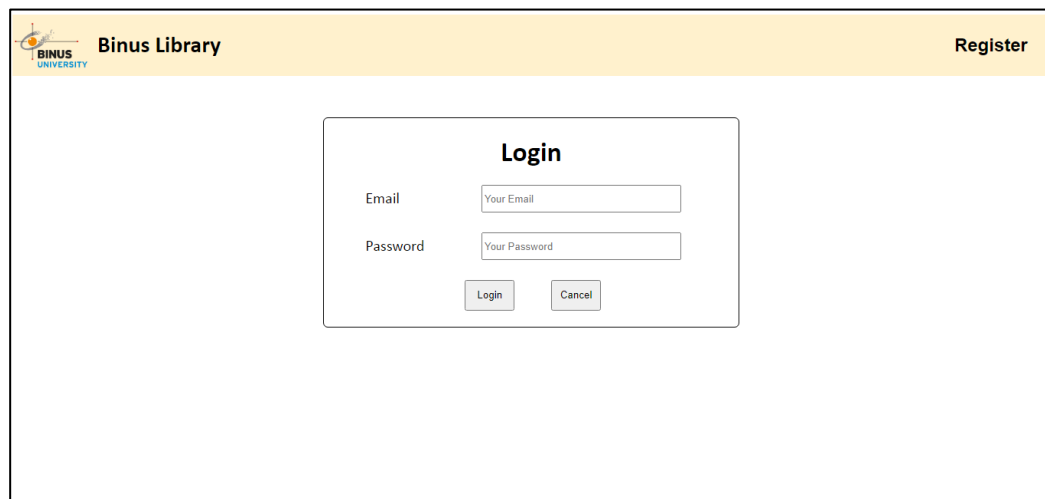
muncul. Selain itu, apabila konfirmasi password tidak sesuai juga akan memunculkan notifikasi. Tampilan halaman dapat dilihat pada gambar 2.3.



Gambar 2.4. Register Page (register.jsp)

3. *Login Page* (G) (login.jsp)

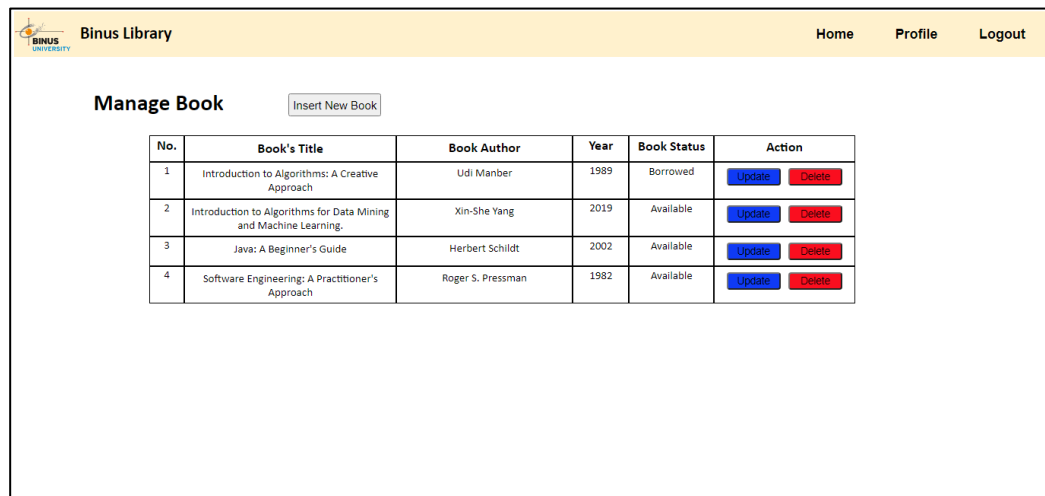
Pada halaman ini, pengguna dengan akses *user* dan *staff* dapat segera melakukan *login* dengan *email* dan kata sandi yang sudah didaftarkan sebelumnya. Untuk alamat *email* dan kata sandi dapat dilihat di lampiran pada *file* readme.txt atau dapat langsung akses dari phpmyadmin. Pengguna akan divalidasi setelah menekan tombol *Login* dan apabila sudah sesuai dengan data pengguna, maka akan diijinkan masuk sesuai dengan *role* dari pengguna. Berikut tampilan *login page*.



Gambar 2.5. Login Page (login.jsp)

4. **Manage Book** (S, U) (manageBook.jsp)


Pada halaman ini, jika *role* merupakan *staff*, maka halaman akan menampilkan beberapa fitur. Pertama, tampilan dari seluruh buku akan muncul sebagai konten utama, dilengkapi dengan 2 buah tombol di kolom *Action* yang mengarahkan *staff* untuk melakukan *Update* ataupun *Delete* pada buku tertentu. Kedua, *staff* juga diberikan akses untuk menambah buku baru dengan menekan tombol *Insert New Book* di sebelah tulisan *Manage Book*. Tampilan *Manage Book* dari *role staff* seperti pada gambar 2.6.



No.	Book's Title	Book Author	Year	Book Status	Action
1	Introduction to Algorithms: A Creative Approach	Udi Manber	1989	Borrowed	Update Delete
2	Introduction to Algorithms for Data Mining and Machine Learning.	Xin-She Yang	2019	Available	Update Delete
3	Java: A Beginner's Guide	Herbert Schildt	2002	Available	Update Delete
4	Software Engineering: A Practitioner's Approach	Roger S. Pressman	1982	Available	Update Delete

Gambar 2.6. Manage Book Page (manageBook.jsp) untuk staff

Untuk *role user*, ada sedikit pembatasan akses pada halaman ini. *User* hanya diijinkan untuk melihat *list* buku yang ada di perpustakaan dan dapat meminjam buku tersebut. Peminjaman hanya boleh dilakukan terhadap 1 buah buku. Apabila buku belum dikembalikan atau buku sudah dikembalikan namun belum membayar denda keterlambatan, maka *user* tidak dapat melakukan peminjaman (ditandai dengan tombol *Borrow* tidak dapat ditekan). Apabila buku tidak ada, *user* juga tidak dapat menekan tombol *Borrow*. Lebih lengkapnya, tampilan untuk *user* dapat dilihat pada gambar 2.7.



Binus Library

Home

Profile

Logout


Manage Book

No.	Book's Title	Book Author	Year	Book Status	Action
1	Introduction to Algorithms: A Creative Approach	Udi Manber	1989	Available	<button>Borrow</button>
2	Introduction to Algorithms for Data Mining and Machine Learning.	Xin-She Yang	2019	Available	<button>Borrow</button>
3	Software Engineering: A Practitioner's Approach	Roger S. Pressman	1982	Borrowed	<button>Borrow</button>

Gambar 2.7. Manage Book Page (manageBook.jsp) untuk user

5. Manage Transaction (S, U) (manageTransaction.jsp)

Halaman ini ditujukan bagi *Staff* maupun *User* namun dengan akses dan tampilan sedikit berbeda. Jika *role* merupakan seorang *staff*, maka halaman akan berisikan akses terhadap *transaction list* beserta *pending transaction* yang mana membutuhkan *approval* dari *staff* (hanya muncul saat ada *pending transaction*). Apabila tombol *approve* pada salah satu baris di *pending transaction* ditekan, maka *user* tersebut dapat meminjam buku tersebut dan *pending transaction* tersebut akan langsung masuk ke *transaction list* di bawah. Tampilan *manage transaction* bagi *staff* adalah seperti gambar 2.8 di bawah ini.



Binus Library

Home

Profile

Logout

Pending Transaction

No.	Transaction Date	User Name	Book Name	Action
1	06/07/2020	Edward	Software Engineering: A Practitioner's Approach	<button>Approve</button>
2	06/07/2020	Mike Christ	Introduction to Algorithms for Data Mining and Machine Learning.	<button>Approve</button>

Transaction

No.	Transaction Date	User Name	Book Name	Book Status
1	05/07/2020	Andreas	Introduction to Algorithms: A Creative Approach	Borrowed
2	05/07/2020	Andreas	Software Engineering: A Practitioner's Approach	Returned
3	15/06/2020	Edward	Introduction to Algorithms for Data Mining and Machine Learning.	Returned
4	20/06/2020	Mike Christ	Introduction to Algorithms: A Creative Approach	Returned

Gambar 2.8. Manage Transaction (manageTransaction.jsp) untuk staff

Untuk seorang *user*, halaman ini hanya menunjukkan *transaction list* dari transaksi peminjaman yang pernah dilakukan. Tampilan halaman adalah sebagai berikut seperti pada gambar 2.9.

BINUS UNIVERSITY		Binus Library	Home	Profile	Logout
Transaction					
No.	Transaction Date	User Name	Book Name	Book Status	
1	06/07/2020	Mike Christ	Introduction to Algorithms for Data Mining and Machine Learning.	Pending	
2	06/07/2020	Mike Christ	Introduction to Algorithms for Data Mining and Machine Learning.	Returned	
3	20/06/2020	Mike Christ	Introduction to Algorithms: A Creative Approach	Returned	

Gambar 2.9. Manage Transaction (manageTransaction.jsp) untuk user

6. Manage Borrow (S) (manageBorrow.jsp)

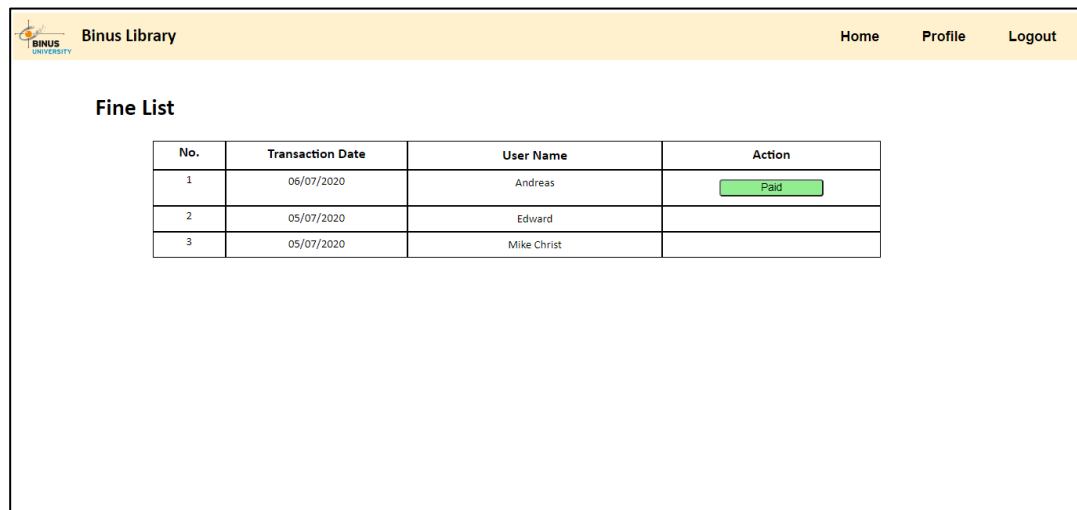
Halaman ini hanya untuk pengguna dengan akses *staff* dikarenakan halaman ini digunakan untuk konfirmasi pengembalian buku. Apabila *user* sudah melakukan pengembalian buku, maka *staff* dapat melakukan konfirmasi pengembalian buku dengan cara menekan tombol *Returned* di baris yang berisikan *user* yang mengembalikan buku. Apabila pengembalian dilakukan lebih dari 7 hari setelah peminjaman, maka *user* akan dikenakan denda sebesar Rp 50.000,- (lima puluh ribu rupiah). Saat terlambat, buku akan diperbarui statusnya menjadi sudah dikembalikan, namun pengguna masih belum dapat meminjam buku selama denda belum dibayar. Tampilan halaman seperti pada gambar 2.10.

BINUS UNIVERSITY		Binus Library	Home	Profile	Logout
Borrow List					
No.	Transaction Date	User Name	Book Name	Action	
1	06/07/2020	Edward	Software Engineering: A Practitioner's Approach	Returned	
2	06/07/2020	Mike Christ	Introduction to Algorithms for Data Mining and Machine Learning.	Returned	
3	05/07/2020	Andreas	Introduction to Algorithms: A Creative Approach	Returned	

Gambar 2.10. Manage Borrow (manageBorrow.jsp)

7. *Manage Fine* (S) (manageFine.jsp)

Halaman ini akan menampilkan *list* denda dari *user* baik untuk yang sudah membayar denda ataupun yang masih belum membayar. Apabila *user* belum membayar, maka akses *user* untuk meminjam buku akan dibekukan sementara sampai denda dibayar oleh *user*. Saat pembayaran denda sudah diterima, *staff* dapat menekan tombol *Paid* pada *user* yang membayar, dan seketika itu juga akses *user* untuk meminjam buku dibuka kembali. Tampilan akan seperti gambar 2.11.



No.	Transaction Date	User Name	Action
1	06/07/2020	Andreas	<input type="button" value="Paid"/>
2	05/07/2020	Edward	
3	05/07/2020	Mike Christ	

Gambar 2.11. *Manage Fine* (manageFine.jsp)

8. *Insert Book* (S) (insertBook.jsp)

Seorang *staff* dapat menambahkan buku seandainya ada buku baru yang datang ke perpustakaan. *Staff* dapat mengakses penambahan buku baru melalui *manageBook.jsp* seperti yang sudah dijelaskan pada bagian 4 di atas. Halaman akan berisikan *form* yang harus dilengkapi dan berisi data-data dari buku baru tersebut seperti nama buku, penulis/pengarang buku, dan tahun cetak buku. Semua kolom pengisian harus dan wajib diisi oleh *staff* karena sebagai syarat pasti dari buku sebelum dimasukkan ke dalam *database*. Tampilan halaman *Insert Book* akan seperti yang ditunjukkan oleh gambar 2.12.

The screenshot shows a web application interface for a library. At the top, there is a yellow header bar with the 'BINUS UNIVERSITY' logo on the left, the text 'Binus Library' in the center, and navigation links 'Home', 'Profile', and 'Logout' on the right. Below the header, the main content area is white. In the center, there is a white rectangular box with a thin black border titled 'Insert New Book'. Inside this box, there are three input fields: 'Book Name', 'Book Author', and 'Book Year'. Each field has a placeholder text matching its label. Below the input fields are two buttons: 'Insert' and 'Cancel'.

Gambar 2.12. *Insert Book* (insertBook.jsp)

9. *Update Book* (S) (updateBook.jsp)

Tidak beda jauh dengan halaman *Insert Book*, halaman ini digunakan untuk pembaruan data buku (semisal ada salah pengetikan, kurang lengkap, dan sebagainya). *Staff* diijinkan mengubah data buku, namun semua kolom tetap harus terisi, karena adanya validasi semua kolom yang tidak memperbolehkan satu pun kolom kosong. Setelah menekan tombol *Update*, maka data buku akan langsung diperbarui pada *database*. Tampilan halaman ditunjukkan oleh gambar 2.13.

The screenshot shows the same web application interface as before. The header bar is identical. The main content area is white. In the center, there is a white rectangular box with a thin black border titled 'Update Book'. Inside this box, there are three input fields: 'Book Name', 'Book Author', and 'Book Year'. The 'Book Name' field contains the text 'Introduction to Algorithms: A Creative Appr'. The 'Book Author' field contains the text 'Udi Manber'. The 'Book Year' field contains the text '1989'. Below the input fields are two buttons: 'Update' and 'Cancel'.

Gambar 2.13. *Update Book* (updateBook.jsp)


10. *Delete Book* (S)

Seorang *staff* juga dapat melakukan penghapusan buku seandainya dibutuhkan.

Jika ingin melakukan penghapusan buku, dapat langsung melakukan akses melalui `manageBook.jsp` seperti ditampilkan oleh gambar 2.7 di atas dan menekan tombol *Delete* pada baris di mana terdapat buku yang ingin dihapus dari *database*. Berikut tampilan `manageBook.jsp` setelah menekan tombol *Delete* pada gambar 2.15 (dibandingkan dengan gambar 2.7).



Gambar 2.14. Notifikasi *delete* berhasil



Binus Library

HomeProfileLogout

Manage Book

Insert New Book

No.	Book's Title	Book Author	Year	Book Status	Action
1	Introduction to Algorithms: A Creative Approach	Udi Manber	1989	Borrowed	<button>Update</button> <button>Delete</button>
2	Introduction to Algorithms for Data Mining and Machine Learning.	Xin-She Yang	2019	Available	<button>Update</button> <button>Delete</button>
3	Software Engineering: A Practitioner's Approach	Roger S. Pressman	1982	Available	<button>Update</button> <button>Delete</button>

Gambar 2.15. *Delete Book* (`deleteBook.jsp`)

11. *Profile* (S, U) (`profile.jsp`)

Halaman profil akan berisikan data-data pengguna baik *user* ataupun *staff* tergantung pada *role* yang sedang *login*. Data yang ditampilkan hanyalah nama dan *email*. Jika ingin melakukan penggantian kata sandi, dapat dilakukan juga dengan mengisi form tersebut. Apapun penggantian data, kata sandi dan konfirmasinya juga harus diisi. Tampilan halaman ditunjukkan pada gambar 2.16.

Binus Library Home Profile Logout

Profile

Name:

Email:

Password:

Confirm Password:

Gambar 2.16. Profile Page (profile.jsp)

12. *User List* (S) (userList.jsp)

Halaman ini berisikan *list* dari *user* yang terdaftar pada *database* perpustakaan. *Staff* dapat melihat nama dan alamat *email* dari semua *user* yang tercatat dalam sistem perpustakaan. Lebih lanjutnya, tampilan dapat dilihat pada gambar 2.17 di bawah.

Binus Library Home Profile Logout

No.	Name	Email	Role
1	Staff 1	staff1@binus.ac.id	Staff
2	Staff 2	staff2@binus.ac.id	Staff
3	Staff 3	staff3@binus.ac.id	Staff
4	Edward	edward@binus.ac.id	User
5	Mike Christ	mike.heru@binus.ac.id	User
6	Andreas	andreas@binus.ac.id	User

Gambar 2.17. User List Page (userList.jsp)