HCMUTE

LẬP TRÌNH WEB

Cookie trong Java Servlet

Khoa Công nghệ Thông tin
Đại học Sư phạm Kỹ thuật TP.HCM

ThS. Nguyễn Hữu Trung

Xử lý Cookie trong Servlet

- **Cookie** là các tập tin văn bản được lưu trữ trên client. Mục đích của cookie là để theo dõi các thông tin khác nhau. Ví dụ trường hợp remember login.
- Có ba bước liên quan đến xác định người dùng cũ quay trở lại hệ thống:
  - Tập lệnh của máy chủ gửi một tập hợp các cookie đến trình duyệt. Ví dụ tên, tuổi, hoặc số nhận dạng vv.
  - Trình duyệt lưu trữ thông tin này trên máy local để sử dụng trong tương lai.
  - Trong lần truy cập tiếp theo, trình duyệt gửi yêu cầu tới web server, nó sẽ gửi những thông tin cookie tới máy chủ và máy chủ sử dụng thông tin đó để xác định người dùng.

Ví dụ về một Cookie

- Cookie thường được đặt trong HTTP Header
  - HTTP/1.1 200 OK
  - Date: Fri, 13 Oct 2020 21:03:38 GMT
  - Server: Apache/1.3.9 (UNIX) PHP/4.0b3
  - Set-Cookie: name = abc; expires = Friday, 13-Oct-2020 22:03:38 GMT; path = /; domain = iotstar.vn
  - Connection: close
  - Content-Type: text/html

Một servlet sau đó sẽ có quyền truy cập cookie thông qua yêu cầu của phương thức request.getCookies() trả về một mảng các đối tượng Cookie.

Các phương thức xử lý cookie trong servlet

No.	Phương thức & Mô tả
1	public void setDomain(String pattern): thiết lập tên miền mà cookie áp dụng, ví dụ như iotstar.vn.
2	public String getDomain(): lấy tên miền mà cookie áp dụng, ví dụ như iotstar.vn.
3	public void setMaxAge(int expiry): Phương thức này đặt khoảng thời gian(tính bằng giây) trước khi cookie hết hạn. Nếu bạn không đặt điều này, cookie sẽ chỉ kéo dài cho phiên hiện tại.

Các phương thức xử lý cookie trong servlet

No	Phương thức & Mô tả
4	public getMaxAge int(): Phương thức này trả về độ tuổi tối đa của cookie, được chỉ định bằng giây, Theo mặc định, -1 cho biết cookie sẽ tồn tại cho đến khi trình duyệt tắt máy.
5	public String getName(): Phương thức này trả về tên của cookie. Không thể thay đổi tên sau khi tạo.
6	public void setValue(String newValue): Phương thức này đặt giá trị kết hợp với cookie

Các phương thức xử lý cookie trong servlet

No.	Phương thức & Mô tả
7	public getValue String() : Phương thức này lấy giá trị kết hợp với cookie.
8	public void setPath(String uri): Phương thức này đặt đường dẫn đến cookie này được áp dụng. Nếu bạn không chỉ định đường dẫn, cookie sẽ được trả về cho tất cả các URL trong cùng thư mục với trang hiện tại cũng như tất cả các thư mục con.
9	public getPath String(): Phương thức này là đường dẫn đến cookie này được áp dụng.

Các phương thức xử lý cookie trong servlet

No.	Phương thức & Mô tả
10	public void setSecure(boolean flag): Phương thức này thiết lập giá trị boolean cho biết liệu cookie chỉ nên được gửi qua các kết nối được mật mã(tức là SSL).
11	public void setComment(String purpose): Phương thức này xác định một comment mô tả mục đích của một cookie. comment này hữu ích nếu trình duyệt trình bày cookie cho người dùng.
12	public String getComment(): Phương thức này trả về comment mô tả mục đích của cookie này, hoặc không hợp lệ nếu cookie không có comment.

Tạo Cookie trong Servlet

- Việc tạo cookie trong servlet bao gồm 3 bước sau:
  - **1. Tạo một đối tượng Cookie**: gọi constructor Cookie với các tham số tên và giá trị của cookie, cả hai đều có kiểu là String.

```
Cookie cookie = new Cookie("key","value");
```

  - Hãy ghi nhớ, tên và giá trị không nên chứa khoảng trắng hoặc bất kỳ ký tự nào sau đây: [ ] ( ) = , " / ? @ : ;

Tạo Cookie trong Servlet

- Việc tạo cookie trong servlet bao gồm 3 bước sau:
  - **2. Thiết định thời gian tồn tại cho Cookie**: sử dụng phương thức setMaxAge() để xác định cookie được sống trong thời gian bao lâu (tính bằng giây). Sau đây sẽ thiết lập một cookie sống trong 24 giờ.

    ```
    cookie.setMaxAge(60 * 60 * 24);
    ```

  - **3. Đính kèm cookie vào HTTP response header**: sử dụng phương thức response.addCookie() để thêm các cookie và HTTP response header như sau:

    ```
    response.addCookie(cookie);
    ```

Ví dụ: xử lý cookie trong servlet

- Tạo Cookie : CreateCookie.java

```
//Nhận dữ liệu từ FORM
String ten = request.getParameter("ten");
String holot = request.getParameter("holot");

// Create cookies for first and last names.
 Cookie firstName = new Cookie("ten",ten);
 Cookie lastName = new Cookie("holot",holot);

 // Set expiry date after 24 Hrs for both the cookies.
 firstName.setMaxAge(60 * 60 * 24);
 lastName.setMaxAge(60 * 60 * 24);
        // Add both the cookies in the response header.
        response.addCookie(firstName);
        response.addCookie(lastName);

        PrintWriter out = response.getWriter();
        out.println("<b>First Name</b>: " + firstName.getValue()  + " -
<b>Last Name</b>: " + lastName.getValue());
```

```
<form action="createcookie"
method="GET">
Tên: <input type="text"
name="ten"> <br />
 Họ lót: <input type="text"
name="holot" />
          <input type="submit"
value="Submit" />
</form>
```

Index.html

Ví dụ: xử lý cookie trong servlet

Đọc Cookie: ReadCookie.java

```
 Cookie cookie = null;
 Cookie[] cookies = null;
 // Get an array of Cookies associated with this domain
cookies = request.getCookies();
// Set response content type
response.setContentType("text/html");
PrintWriter out = response.getWriter();

if (cookies != null) {
    out.println("<h2> Found Cookies Name and Value</h2>");
    for (int i = 0; i < cookies.length; i++) {
        cookie = cookies[i];
        out.print("Name : " + cookie.getName() + ",  ");
        out.print("Value: " + cookie.getValue() + " <br/>");
    }
} else {
    out.println("<h2>No cookies founds</h2>");
}
out.close();
```

Ví dụ: xử lý cookie trong servlet

- Xóa Cookie: DeleteCookie.java
  - Để xóa cookie rất đơn giản. Nếu bạn muốn xóa một cookie thì bạn chỉ cần làm theo ba bước sau:
    - Đọc một cookie hiện có và lưu trữ nó trong đối tượng Cookie.
    - Đặt tuổi cookie về **0** bằng cách sử dụng phương thức **setMaxAge()** để xóa cookie.
    - Add cookie trở lại response header.

Ví dụ: xử lý cookie trong servlet

- Xóa Cookie: DeleteCookie.java

```
Cookie cookie = null;
Cookie[] cookies = null;

// Get an array of Cookies associated with this domain
cookies = request.getCookies();
// Set response content type
response.setContentType("text/html");
PrintWriter out = response.getWriter();
if (cookies != null) {
    out.println("<h2> Cookies Name and Value</h2>");
    for (int i = 0; i < cookies.length; i++) {
        cookie = cookies[i];
        if ((cookie.getName()).compareTo("ten") == 0) {
            // delete cookie
            cookie.setMaxAge(0);
            response.addCookie(cookie);
          out.print("Deleted cookie : " + cookie.getName() + "<br/>");
        }
        out.print("Name : " + cookie.getName() + ",  ");
        out.print("Value: " + cookie.getValue() + " <br/>");
    }
}
out.close();
```

HCMUTE

LẬP TRÌNH WEB

Servlets & Jsp

Login bằng Cookie

Khoa Công nghệ Thông tin
Đại học Sư phạm Kỹ thuật TP.HCM

ThS. Nguyễn Hữu Trung

Bước 1: Tạo trang Login.html

- Login.html

```
<form action=“login" method=“Post">
Name: <input type="text" name="username "> <br />
 Pass: <input type="password" name="password" />
          <input type="submit" value=“Login" />
</form>
```

Bước 2: Tạo Class LoginServlet.java


```
@Override
protected void doPost(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
resp.setContentType("text/html");
//lấy dữ liệu từ tham số của form
String user = req.getParameter("username");
String pass = req.getParameter("password");
if(user.equals("trung") && pass.equals("123"))
{
//khởi tạo cookie
Cookie cookie = new Cookie("username", user);
//thiết lập thời gian tồn tại 30s của cookie
cookie.setMaxAge(30);
//thêm cookie vào response
resp.addCookie(cookie);
//chuyển sang trang HelloServlet
resp.sendRedirect("/HelloServlet/hello");
}else {
//chuyển sang trang LoginServlet
resp.sendRedirect("/HelloServlet/login");
}}
```

Bước 3: Tạo trang HelloServlet.java

- @WebServlet(urlPatterns= {"/hello","/xin-chao"})

```
@Override
protected void doGet(HttpServletRequest req, HttpServletResponse
resp) throws ServletException, IOException {
resp.setContentType("text/html");
PrintWriter printWriter = resp.getWriter();
String name="";
//Nhận cookie
Cookie[] cookie = req.getCookies();
for (Cookie c: cookie) {
        if(c.getName().equals("username")) {
                name = c.getValue();}}
        if(name.equals("")){
                //chuyển sang trang LoginServlet
                resp.sendRedirect("/HelloServlet/login");
        }
//hiển thị lên trang bằng đối tượng PrintWriter()
printWriter.println("Xin chao " + name);
}
```

HCMUTE

LẬP TRÌNH WEB

UTEx

HttpSession trong Java Servlet

Khoa Công nghệ Thông tin
Đại học Sư phạm Kỹ thuật TP.HCM

ThS. Nguyễn Hữu Trung

Session trong Servlet

- **HTTP** là một giao thức "stateless" nghĩa là mỗi lần client truy xuất một trang Web, client sẽ mở một kết nối riêng đến máy chủ Web và máy chủ sẽ tự động **không giữ lại bất kỳ thông tin nào** về yêu cầu của client trước đó.
- Có ba cách sau để thực hiện theo dõi phiên (**session tracking**) giữa client và máy chủ web:
  - Sử dụng Cookies
  - Sử dụng Hidden Fields của HTML From
  - Sử dụng URL Rewriting
- Ngoài ra Servlet cung cấp HttpSession để xác định người dùng ghé thăm trang web và lưu trữ thông tin về người dùng đó

Đối tượng HttpSession

- Tạo đối tượng HttpSession bằng cách gọi phương thức getSession() của HttpServletRequest, như sau:
  - HttpSession session = request.getSession();
  - Bạn cần gọi request.getSession() trước khi gửi bất kỳ nội dung nào đến client. Dưới đây là tóm tắt các phương thức quan trọng hiện có của đối tượng HttpSession.

Đối tượng HttpSession

No	Phương thức & Mô tả
1	**public Object getAttribute(String name)**: trả về đối tượng bị ràng buộc với tên được chỉ định trong session này, hoặc null nếu không có đối tượng nào bị ràng buộc dưới tên.
2	**public Enumeration getAttributeNames():** trả về một Enumeration of String các đối tượng có chứa tên của tất cả các đối tượng ràng buộc vào session này.

Đối tượng HttpSession

No	Phương thức & Mô tả
3	**public long getCreationTime():** trả về thời gian khi session này được tạo ra, được đo bằng mili giây kể từ nửa đêm ngày 1 tháng 1 năm 1970 GMT.
4	**public String getId():** trả về một chuỗi chứa mã định danh duy nhất được gán cho session này.
5	**public long getLastAccessedTime():** trả về thời gian truy cập cuối cùng của session, theo định dạng mili giây kể từ nửa đêm ngày 1 tháng 1 năm 1970 GMT

Đối tượng HttpSession

No	Phương thức & Mô tả
6	**public int getMaxInactiveInterval():** trả về khoảng thời gian tối đa (giây), rằng vùng chứa servlet sẽ giữ session mở giữa các lần truy cập của client.
7	**public void invalidate():** làm mất hiệu lực session này và unbinds bất kỳ đối tượng liên quan đến nó.
8	**public boolean isNew():** trả về true nếu client chưa biết về session hoặc nếu client không tham gia session làm việc.

Đối tượng HttpSession

No	Phương thức & Mô tả
9	**public void removeAttribute(String name):** loại bỏ các đối tượng bị ràng buộc với tên quy định từ session này.
10	**public void setAttribute(String name, Object value):** liên kết một đối tượng đến session này, sử dụng tên được chỉ định.
11	**public void setMaxInactiveInterval(int interval):** chỉ định thời gian, tính bằng giây, giữa các yêu cầu của client trước khi bộ chứa servlet sẽ làm mất hiệu lực session này.

Ví dụ về Session trong Servlet

- CreateSession.java

```
@Override
protected void doGet(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
    //khởi tạo session
    HttpSession s = req.getSession();
    //Gán dữ liệu vào session
    s.setAttribute("ten", "Nguyễn Hữu Trung");
    s.setAttribute("tuoi", new Integer(40));
    //thiết lập thời gian tồn tại session
    s.setMaxInactiveInterval(30);
    //hiển thị thông báo lên web
    resp.setContentType("text/html");
    resp.setCharacterEncoding("UTF-8");
    PrintWriter out = resp.getWriter();
    out.println("Xin chào bạn session đã được tạo");
    out.close();
}
```

Ví dụ về Session trong Servlet

- ShowSession.java

```
@Override
protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, IOException {
// hiển thị session lên web
resp.setContentType("text/html");
resp.setCharacterEncoding("UTF-8");
PrintWriter out = resp.getWriter();
String ten = "";
// khởi tạo session
HttpSession s = req.getSession();
Object obj= s.getAttribute("ten"); // truy xuất dữ liệu từ session
//kiểm tra đối tượng Object có null không
if(obj != null) {
ten = String.valueOf(obj); //ép kiểu về String
}else {
resp.sendRedirect("/HelloServlet/creatsession"); //nếu null thì chuyển về trang tạo
session
}
int tuoi = (Integer)s.getAttribute("tuoi");//ép kiểu
//Hiển thị session lên web
out.println("Xin chào bạn: " + ten + " tuổi: " + tuoi);
out.close();
}
```

Xóa Session Data

- Bạn có thể xóa session data với các tùy chọn như sau:
  - **Xoá một thuộc tính cụ thể**: bạn có thể gọi phương thức public void removeAttribute(String name) để xóa giá trị kết hợp với một khoá cụ thể.
  - **Xóa toàn bộ sesssion**: bạn có thể gọi phương thức public void invalidate() để xóa toàn bộ sesssion.
  - **Cài đặt thời gian chờ cho session**: bạn có thể gọi phương thức public void setMaxInactiveInterval(int interval) để đặt thời gian chờ cho một session riêng.
  - **Đăng xuất user**: các máy chủ hỗ trợ servlet 2.4, bạn đăng xuất khách hàng ra khỏi Web server và làm mất hiệu lực tất cả session của tất cả người dùng.
  - **Cấu hình web.xml**: nếu bạn đang sử dụng Tomcat (30 phút), ngoài các phương pháp đã đề cập ở trên, bạn có thể thiết định thời gian session trong tệp web.xml như sau:

```
<session-config>
    <session-timeout>15 phút (900 giây)</session-timeout>
</session-config>
```

HCMUTE

LẬP TRÌNH WEB

Login bằng Session

Khoa Công nghệ Thông tin
Đại học Sư phạm Kỹ thuật TP.HCM

ThS. Nguyễn Hữu Trung

Session

- Session thiết lập trên Server
- Session giúp duy trì trạng thái của dữ liệu của người dùng
- Session có thời gian sống xác định
- Servlet quản lý bằng Session Management
- Session lưu thông tin dạng key/value trên server

Cách hoạt động

- Cookie
  - Set ID vào cookie để nhận dạng
- Hidden Field trong Form
  - Set ID vào một trường ẩn trong Form
  - <input type="hidden" name ="sessionid", value="123">
- URL Rewriting
  - http://iotstar.vn/home;sessionid=123

HttpSession

- Là một interface trong Servlet giúp lưu các thông tin người dùng servlet khác nhau
- HttpSession session = request.getSession();

```
//Thiết lập Session
HttpSession session = req.getSession();
session.setAttribute("uname", "Nguyễn Hữu Trung");
session.setMaxInactiveInterval(10); //Thiết lập thời gian tồn tại Session
//Lấy thông tin Session
String uname="";
Object obj = session.getAttribute("uname");
if (obj != null) {
uname = String.valueOf(obj);
//hiển thị lên trang bằng đối tượng PrintWriter()
printWriter.println(uname);
}else {
//chuyển sang trang LoginServlet
resp.sendRedirect("/HelloServlet/login");
}
```

Login bằng session

Login.html

```
<form action="login" method="post">
UserName:<input type="text" name="username"><br/>
Password:<input type="password" name="password"><br/>
<input type="submit" value="login">
</form>
```

LoginServlet.java

```
String username = request.getParameter("username");
String password = request.getParameter("password");

if (username.equals("trungnh")&& password.equals("123")) {
out.print("Chào mừng bạn, " + username);
HttpSession session = request.getSession();
session.setAttribute("name", username);
} else {
out.print("Tài khoản hoặc mật khẩu không chính xác");
request.getRequestDispatcher("Login.html").include(request
, response);
}
```

Logout.java


```
//Hủy Session
HttpSession session = request.getSession();
session.invalidate();
request.getRequestDispatcher("Login.html").include(request,
response);
```

Profile.java

```
if(session!=null){
String name=(String)session.getAttribute("name");
out.print("Chào bạn, "+name+" đến với trang quản lý tài khoản");
}
else {
out.print("Xin vui lòng đăng nhập");
resp.sendRedirect("/HelloServlet/Login.html");
}
```

HCMUTE

LẬP TRÌNH WEB

Example Login và Register trong Servlet

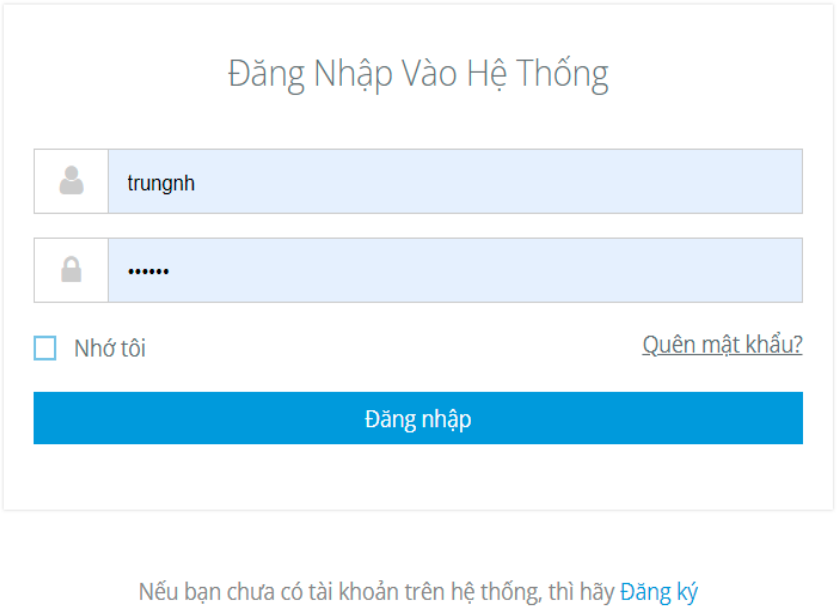
Khoa Công nghệ Thông tin
Đại học Sư phạm Kỹ thuật TP.HCM

ThS. Nguyễn Hữu Trung

Chức năng đăng nhập tài khoản

HCMUTE

- Bước 1: Thiết kế view: login.jsp




```
<form action="login" method="post">
<h2>Tạo tài khoản mới</h2>
<c:if test="${alert !=null}">
        <h3 class="alert alert-
        danger">${alert}</h3>
</c:if>
<section>
<label class="input login-input">
<div class="input-group">
<span class="input-group-addon"><i
class="fa fa-user"></i></span>
<input type="text" placeholder="Tài
khoản" name="username"
class="form-control">
</div>
</label>
</section>
```


LT WEB

Bước 2: Tạo Model: User.java

```
@SuppressWarnings("serial")
public class User implements Serializable {
private int id;
private String email;
private String userName;
private String fullName;
private String passWord;
private String avatar;
private int roleid;
private String phone;
private Date createdDate;

//Tạo constructor, getters/setters
```

Chức năng đăng nhập tài khoản

- Bước 3: Định nghĩa hàm trong interface service

```
public interface UserService {

User login(String username, String password);
User get(String username);
}
```

- Bước 4: Viết hàm thực thi trong service implement

```
public class UserServiceImpl implements UserService {
UserDao userDao = new UserDaoImpl();

//viết các hàm đã định nghĩa bên interface service tại đây

}
```

Chức năng đăng nhập tài khoản

- Bước 4: Viết hàm thực thi trong service implement

```
@Override
public User login(String username, String password) {
User user = this.findByUserName(username);
if (user != null && password.equals(user.getPassWord())) {
return user;
}

return null;
}

 @Override
 public User findByUserName(String username) {
 return userDao.findByUserName(username);
 }
```

Chức năng đăng nhập tài khoản

- Bước 5: Định nghĩa hàm trong interface DAO

```
public interface UserDao {
    User get(String username);
}
```

- Bước 6: Viết hàm thực thi trong DAO Implement

```
public class UserDaoImpl implements UserDao {
    public Connection conn = null;
    public PreparedStatement ps = null;
    public ResultSet rs = null;

        //Viết hàm xử lý DAO tại đây
}
```

Chức năng đăng nhập tài khoản

HCMUTE

- Bước 6: Viết hàm thực thi trong DAO Implement

```
@Override
public User findByUserName(String username) {
String sql = "SELECT * FROM [User] WHERE username = ? ";
try {
conn = new DBConnection().getConnection();
ps = conn.prepareStatement(sql);
ps.setString(1, username);
rs = ps.executeQuery();
while (rs.next()) {
User user = new User();
user.setId(rs.getInt("id"));
user.setEmail(rs.getString("email"));
user.setUserName(rs.getString("username"));
user.setFullName(rs.getString("fullname"));
user.setPassWord(rs.getString("password"));
user.setAvatar(rs.getString("avatar"));
user.setRoleid(Integer.parseInt(rs.getString("roleid")));
user.setPhone(rs.getString("phone"));
user.setCreatedDate(rs.getDate("createdDate"));
return user;}
} catch (Exception e) {e.printStackTrace();}
return null;}
```

- Bước 7: Viết Controller : LoginController.java

```
@SuppressWarnings("serial")
@WebServlet(urlPatterns = "/login")
public class LoginController extends HttpServlet {
@Override
protected void doGet(HttpServletRequest req, HttpServletResponse
resp) throws ServletException, IOException {

HttpSession session = req.getSession(false);
if (session != null && session.getAttribute("account") != null) {
        resp.sendRedirect(req.getContextPath()+ "/waiting");
return;
}
```

Bước 7: Viết Controller : LoginController.java

```
// Check cookie
Cookie[] cookies = req.getCookies();
if (cookies != null) {
for (Cookie cookie : cookies) {
if (cookie.getName().equals("username")) {
session = req.getSession(true);
session.setAttribute("username", cookie.getValue());
resp.sendRedirect(req.getContextPath()+ "/waiting");
return;
}
}
}

req.getRequestDispatcher("views/login.jsp").forward(req, resp);
}
```

Bước 7: Viết Controller : LoginController.java

```
@Override
protected void doPost(HttpServletRequest req, HttpServletResponse
resp) throws ServletException, IOException {
  resp.setContentType("text/html");
  resp.setCharacterEncoding("UTF-8");
  req.setCharacterEncoding("UTF-8");
String username = req.getParameter("username");
        String password = req.getParameter("password");
        boolean isRememberMe = false;
        String remember = req.getParameter("remember");

        if("on".equals(remember)){
            isRememberMe = true;
        }
        String alertMsg="";
```

- Bước 7: Viết Controller : LoginController.java

```
if(username.isEmpty() || password.isEmpty()){
            alertMsg = "Tài khoản hoặc mật khẩu không được rỗng";
            req.setAttribute("alert", alertMsg);

req.getRequestDispatcher("/views/login.jsp").forward(req, resp);
            return;
        }

        UserService service = new UserServiceImpl();

        User user = service.login(username, password);
```

- Bước 7: Viết Controller : LoginController.java

```
if(user!=null){
            HttpSession session = req.getSession(true);
            session.setAttribute("account", user);

            if(isRememberMe){
                saveRemeberMe(resp, username);
            }

            resp.sendRedirect(req.getContextPath()+"/waiting");
        }else{
            alertMsg = "Tài khoản hoặc mật khẩu không đúng";
            req.setAttribute("alert", alertMsg);
            req.getRequestDispatcher("/views/login.jsp").forward(req,
resp);
        }
    }
```

- Bước 7: Viết Controller : LoginController.java

```
 private void saveRemeberMe(HttpServletResponse response, String
username){
        Cookie cookie = new Cookie(Constant.COOKIE_REMEMBER,
username);
        cookie.setMaxAge(30*60);
        response.addCookie(cookie);
    }
```

```
public static final String SESSION_USERNAME = "username";
public static final String COOKIE_REMEMBER = "username";
```

Chức năng đăng nhập tài khoản

Bước 9: Viết Controller : WaitingController.java

```
@SuppressWarnings("serial")
@WebServlet(urlPatterns="/waiting")
public class WaitingController extends HttpServlet {
@Override
protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, IOException {
HttpSession session= req.getSession();
if(session != null && session.getAttribute("account") != null) {
    User u=(User) session.getAttribute("account");
    req.setAttribute("username", u.getUserName());
    if(u.getRoleid()==1) {
        resp.sendRedirect(req.getContextPath()+"/admin/home");
    }else if(u.getRoleid()==2) {
        resp.sendRedirect(req.getContextPath()+"/manager/home");
    }else {
        resp.sendRedirect(req.getContextPath()+"/home");
    }
}else {
        resp.sendRedirect(req.getContextPath()+"/login");
}}}
```

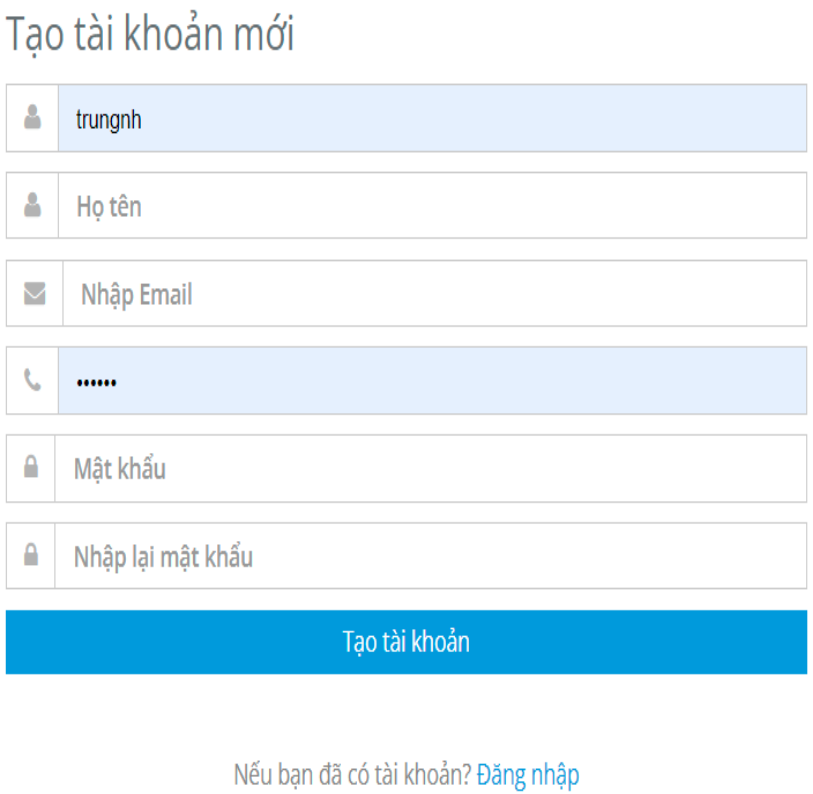
Chức năng đăng nhập tài khoản

- Bước 10: Điều chỉnh và gọi Session trên trang topbar.jsp

```
<c:choose>
<c:when test="${sessionScope.account == null}">
    <div class="col-sm-6">
    <ul class="list-inline right-topbar pull-right">
    <li><a href="${pageContext.request.contextPath }/login">Đăng nhập</a>
    | <a href="${pageContext.request.contextPath }/register">Đăng ký</a></li>
    <li><i class="search fa fa-search search-button"></i></li>
    </ul>
    </div>
</c:when>
<c:otherwise>
    <div class="col-sm-6">
    <ul class="list-inline right-topbar pull-right">
    <li><a href="${pageContext.request.contextPath
    }/member/myaccount">${sessionScope.account.fullName}</a> | <a
    href="${pageContext.request.contextPath }/logout">Đăng Xuất</a></li>
    <li><i class="search fa fa-search search-button"></i></li>
    </ul>
    </div>
</c:otherwise></c:choose>
```

Chức năng đăng ký tài khoản

Bước 1: Thiết kế view: register.jsp




```
<form action="register" method="post">
<h2>Tạo tài khoản mới</h2>
<c:if test="${alert !=null}">
        <h3 class="alert alert-
        danger">${alert}</h3>
</c:if>
<section>
<label class="input login-input">
<div class="input-group">
<span class="input-group-addon"><i
class="fa fa-user"></i></span>
<input type="text" placeholder="Tài
khoản" name="username"
class="form-control">
</div>
</label>
</section>
```

Bước 2: Tạo Model: User.java

```
@SuppressWarnings("serial")
public class User implements Serializable {
private int id;
private String email;
private String userName;
private String fullName;
private String passWord;
private String avatar;
private int roleid;
private String phone;
private Date createdDate;

//Tạo constructor, getters/setters
```

- Bước 3: Định nghĩa hàm trong interface service

```
public interface UserService {

void insert(User user);

boolean register(String email, String password, String username,
String fullname, String phone);

boolean checkExistEmail(String email);

boolean checkExistUsername(String username);

boolean checkExistPhone(String phone);

}
```

Chức năng đăng ký tài khoản

- Bước 4: Viết hàm thực thi trong service implement

```
public class UserServiceImpl implements UserService {
UserDao userDao = new UserDaoImpl();

//viết các hàm đã định nghĩa bên interface service tại đây

}
```

Chức năng đăng ký tài khoản

- Bước 4: Viết hàm thực thi trong service implement

```
@Override
public boolean register(String username, String password,
String email, String fullname, String phone ) {
if (userDao.checkExistUsername(username)) {
return false;
}
long millis=System.currentTimeMillis();
java.sql.Date date=new java.sql.Date(millis);
userDao.insert(new User(email, username, fullname,password,
null,5,phone,date));
return true;
}
```

Chức năng đăng ký tài khoản

- Bước 4: Viết hàm thực thi trong service implement

```
public boolean checkExistEmail(String email) {
return userDao.checkExistEmail(email);
}
public boolean checkExistUsername(String username) {
return userDao.checkExistUsername(username);
}
@Override
public boolean checkExistPhone(String phone) {
return userDao.checkExistPhone(phone);
}
@Override
public void insert(User user){
userDao.insert(user);
}
```

Chức năng đăng ký tài khoản

- Bước 5: Định nghĩa hàm trong interface DAO

```
public interface UserDao {

void insert(User user);

boolean checkExistEmail(String email);

boolean checkExistUsername(String username);

boolean checkExistPhone(String phone);

}
```

Chức năng đăng ký tài khoản

HCMUTE

- Bước 6: Viết hàm thực thi trong DAO Implement

```
public class UserDaoImpl implements UserDao {
    public Connection conn = null;
    public PreparedStatement ps = null;
    public ResultSet rs = null;

        //Viết hàm xử lý DAO tại đây
}
```

- Bước 6: Viết hàm thực thi trong DAO Implement

```
@Override
public void insert(User user) {
String sql = "INSERT INTO [User](email, username, fullname, password,
avatar, roleid,phone, createddate) VALUES (?,?,?,?,?,?,?,?)";
try {
    conn = new DBConnection().getConnection();
    ps = conn.prepareStatement(sql);
    ps.setString(1, user.getEmail());
    ps.setString(2, user.getUserName());
    ps.setString(3, user.getFullName());
    ps.setString(4, user.getPassWord());
    ps.setString(5, user.getAvatar());
    ps.setInt(6,user.getRoleid());
    ps.setString(7,user.getPhone());
    ps.setDate(8, user.getCreatedDate());
    ps.executeUpdate();
} catch (Exception e) {e.printStackTrace();}
}
```

Chức năng đăng ký tài khoản

- Bước 6: Viết hàm thực thi trong DAO Implement

```
@Override
public boolean checkExistEmail(String email) {
boolean duplicate = false;
String query = "select * from [user] where email = ?";
try {
    conn = new DBConnection().getConnection();
    ps = conn.prepareStatement(query);
    ps.setString(1, email);
    rs = ps.executeQuery();
    if (rs.next()) {
    duplicate = true;
    }
    ps.close();
    conn.close();
} catch (Exception ex) {}
return duplicate;
}
```

Chức năng đăng ký tài khoản

HCMUTE

- Bước 6: Viết hàm thực thi trong DAO Implement

```
@Override
public boolean checkExistUsername(String username) {
boolean duplicate = false;
String query = "select * from [User] where username = ?";

try {
    conn = new DBConnection().getConnection();
    ps = conn.prepareStatement(query);
    ps.setString(1, username);
    rs = ps.executeQuery();
    if (rs.next()) {
    duplicate = true;
    }
    ps.close();
    conn.close();
} catch (Exception ex) {}
return duplicate;
}
```

Chức năng đăng ký tài khoản

- Bước 7: Viết Controller : RegisterController.java

```
@SuppressWarnings("serial")
@WebServlet(urlPatterns = "/register")
public class RegisterController extends HttpServlet {
@Override
    protected void doGet(HttpServletRequest req,
    HttpServletResponse resp) throws ServletException,
    IOException {
        HttpSession session = req.getSession(false);
        if (session != null && session.getAttribute("username")
        != null) {
        resp.sendRedirect(req.getContextPath() + "/admin");
        return;
    }
```

Chức năng đăng ký tài khoản

- Bước 7: Viết Controller : RegisterController.java

```
// Check cookie
Cookie[] cookies = req.getCookies();
if (cookies != null) {
 for (Cookie cookie : cookies) {
  if (cookie.getName().equals("username")) {
    session = req.getSession(true);
    session.setAttribute("username", cookie.getValue());
    resp.sendRedirect(req.getContextPath() + "/admin");
    return;
  }
 }
}
req.getRequestDispatcher(Constant.Path.REGISTER).forward(req,
resp);
}
```

Chức năng đăng ký tài khoản

HCMUTE

- Bước 7: Viết Controller : RegisterController.java

```
@SuppressWarnings("static-access")
@Override
protected void doPost(HttpServletRequest req, HttpServletResponse
resp) throws ServletException, IOException {
    resp.setCharacterEncoding("UTF-8");
    req.setCharacterEncoding("UTF-8");
    String username = req.getParameter("username");
    String password = req.getParameter("password");
    String email = req.getParameter("email");
    String fullname = req.getParameter("fullname");
    String phone = req.getParameter("phone");

    UserService service = new UserServiceImpl();
    String alertMsg = "";
```

Chức năng đăng ký tài khoản

- Bước 7: Viết Controller : RegisterController.java

```
if (service.checkExistEmail(email)) {
    alertMsg = "Email đã tồn tại!";
    req.setAttribute("alert", alertMsg);
    req.getRequestDispatcher(Constant.Path.REGISTER).forward(req,
    resp);
    return;
}
if (service.checkExistUsername(username)) {
    alertMsg = "Tài khoản đã tồn tại!";
    req.setAttribute("alert", alertMsg);
    req.getRequestDispatcher(Constant.Path.REGISTER).forward(req,
    resp);
    return;
}
```

Chức năng đăng ký tài khoản

- Bước 7: Viết Controller : RegisterController.java

```
boolean isSuccess = service.register(username, password, email,
fullname, phone);
if (isSuccess) {
//SendMail sm = new SendMail();
//sm.sendMail(email, "Shopping.iotstar.vn", "Welcome to Shopping.
Please Login to use service. Thanks !");
        req.setAttribute("alert", alertMsg);
        resp.sendRedirect(req.getContextPath() + "/login");
} else {
        alertMsg = "System error!";
        req.setAttribute("alert", alertMsg);
        req.getRequestDispatcher(Constant.Path.REGISTER).forward(req
, resp);
}
}
}
```

```
public static final String REGISTER = "/views/register.jsp";
```

DEMO

Thông tin liên lạc

- Nguyễn Hữu Trung
- 0908617108
- trungnh@hcmute.edu.vn
- utex.hcmute.edu.vn