


| | | | | |
|--|--|----------|-------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 1/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF

1 Quản lý xác thực

1.1 Thông tin định danh

1.1.1 Tên đăng nhập phải là duy nhất và chỉ bao gồm các ký tự chữ cái, số, gạch chân, dấu chấm.

```
public static boolean isExistUsername(String username) {
    SessionFactory sessionFactory = HibernateUtil.getSessionFactory();
    Session session = sessionFactory.openSession();
    Query createQuery = session.createQuery("from Customer where name = :username");
    createQuery.setParameter("username", username);
    List list = createQuery.list();
    session.close();
    return list.size() > 0;
}


public static boolean isValidUsername(String username) {
    if (username == null || username.equals("")) {
        return false;
    }
    return username.matches("^[a-zA-Z0-9_]{6,30}$");
}
```

1.1.2 Thiết lập chính sách mật khẩu mạnh:

- Mật khẩu có độ dài tối thiểu là 8 ký tự.
- Chứa chữ cái, chữ số và ký tự đặc biệt.
- Thiết lập blacklist các mật khẩu yếu (VD: 123456A@, 123456a@,...).

```
public static String generatePswd(int minlen, int maxlen)
{
    String CHARACTERS =
    "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789!@#$%^&*_
    =+~/-";

    Random rand = new Random();
    char[] arrCharacter = CHARACTERS.toCharArray();
    int passwdlen = rand.nextInt((maxlen - minlen) + 1) + minlen;
    int characters_len = arrCharacter.length;
    char[] passwd = new char[passwdlen];
    int j = 0;
    do
    {
        for(int i = 0; i < passwdlen; ++i)
```

| | | | | |
|--|--|----------|--------------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 2/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

```

    {
        j = rand.nextInt(characters_len);
        passwd[i] = arrCharacter[j];
    }
}while(isStrongPassword(String.valueOf(passwd)));
return String.valueOf(passwd);
}
public static boolean isStrongPassword(String password) {
    List<String> blacklistStr = new ArrayList<>();
    blacklistStr.add("123456");
    if (password == null) {
        return false;
    }
    return password.matches("(?=.{8,}) (?!.*[0-9]) (?!.*[a-z]) (?!.*[A-Z]) (?!.*[!@#$%^&* _+=-./]) .*$")
        && blacklistStr.contains(password.toLowerCase());
}

```

Thiết lập thời gian sống cho mật khẩu là 90 ngày và không trùng mật khẩu đang sử dụng.

```

public static boolean isExpirePassword(long customerId) {
    SessionFactory sessionFactory = HibernateUtil.getSessionFactory();
    Session session = sessionFactory.openSession();
    Query createQuery = session.createQuery("select lastPasswordChange
from Customer where customerId = :customerId");
    createQuery.setParameter("customerId", customerId);
    Date uniqueResult = (Date) createQuery.uniqueResult();
    session.close();
    Date today = Calendar.getInstance().getTime();
    long diffInMillies = today.getTime() - uniqueResult.getTime();
    long diffInDays = TimeUnit.DAYS.convert(diffInMillies,
TimeUnit.MILLISECONDS);
    return diffInDays > 90;
}

```


1.1.3 Đối với chức năng reset/ quên mật khẩu:

- Đường dẫn reset/quên mật khẩu được gửi qua email phải bị mất hiệu lực sau lần truy cập đầu tiên hoặc sau 8 giờ nếu không được truy cập.
- Nếu chức năng reset/quên mật khẩu thực hiện gửi mật khẩu qua email thì mật khẩu phải được sinh ngẫu nhiên và phải tuân theo chính sách mật khẩu mạnh tại mục 2.

```

public static boolean isExpirePasswordChange(long customerId) {
    SessionFactory sessionFactory = HibernateUtil.getSessionFactory();
    Session session = sessionFactory.openSession();
    // expiredPasswordChange = today.getTime() + 8*60*60*1000 tại thời
    // điểm người dùng sử dụng chức năng reset/ quên mật khẩu.

```

| | | | | |
|--|--|----------|--------------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 3/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

```

    Query createQuery = session.createQuery("select
expiredPasswordChange from Customer where customerId = :customerId");
    createQuery.setParameter("customerId", customerId);
    Date uniqueResult = (Date) createQuery.uniqueResult();
    session.close();
    Date today = Calendar.getInstance().getTime();
    long diff = today.getTime() - uniqueResult.getTime();
    return diff > 0;
}

public static boolean isValidToken(long customerId, String token) {
    SessionFactory sessionFactory = HibernateUtil.getSessionFactory();
    Session session = sessionFactory.openSession();
    // tokenPasswordChange được sinh ngẫu nhiên, độ dài tối thiểu là
    128 bit tại thời điểm người dùng sử dụng chức năng reset/ quên mật khẩu
    Query createQuery = session.createQuery("select
tokenPasswordChange from Customer where customerId = :customerId");
    createQuery.setParameter("customerId", customerId);
    String uniqueResult = (String) createQuery.uniqueResult();
    boolean isValid = token.equals(uniqueResult);
    session.beginTransaction();
    if (isValid || isExpirePasswordChange(customerId)) {
        Query query = session.createQuery("update Customer set
tokenPasswordChange = NULL where customerId = :customerId");
        query.setParameter("customerId",
customerId).executeUpdate();
    }
    session.getTransaction().commit();
    session.close();
    return isValid;
}


```

- 1.1.4 Chỉ lưu dạng mã hash của mật khẩu trong DB (khuyến nghị thuật toán hash là SHA-256), thêm chuỗi salt ngẫu nhiên vào mật khẩu trước khi thực hiện hash.

```

public static String hashPassword(String password) throws
NoSuchAlgorithmException {
    String salt = generatePswd(8,16);
    new SecureRandom().nextBytes(salt);
    MessageDigest digest = MessageDigest.getInstance("SHA-256");
    String input = new String(salt) + password;
    byte[] hash = digest.digest(input.getBytes());
    return Base64.encodeBase64String(hash) + "&salt:" + salt;
}

```

| | | | | |
|--|--|----------|--------------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 4/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

1.2 Xử lý xác thực


- 1.2.1 Trả về thông báo chung cho trường hợp người dùng đăng ký thông tin định danh (username, email,...) đã tồn tại tại chức năng đăng ký, hoặc gửi sai thông tin định danh tại các chức năng đăng nhập, reset/quên mật khẩu, đổi địa chỉ email,...
- 1.2.2 Bật cơ chế bảo vệ bằng Captcha hoặc các hình thức tương đương khi đăng nhập sai quá 5 lần liên tiếp. Cần triển khai cơ chế này tại các chức năng quan trọng khác của ứng dụng.
- Sử dụng Captcha an toàn theo Chỉ thị sử dụng Captcha an toàn Tập đoàn đã ban hành.
 - Thực hiện kiểm tra tính hợp lệ của Captcha trước khi thực hiện chức năng được request.
- 1.2.3 Chỉ gửi thông tin định danh qua phương thức POST, khuyến nghị cấu hình sử dụng HTTPS cho ứng dụng để tăng tính bảo mật.

2 Quản lý phiên đăng nhập

- 2.1.1 Session phải được quản lý bởi server, sinh ngẫu nhiên và độ dài tối thiểu là 128 bit (16 ký tự).
- 2.1.2 Ví dụ sử dụng sessionId được sinh bởi Tomcat server, giá trị sessionId đã đảm bảo được sinh ngẫu nhiên và độ dài mặc định là 128 bit.
- 2.1.3 Session phải được thiết lập thời gian timeout, giá trị timeout cần cân bằng giữa nhu cầu thương mại và yếu tố bảo mật.

Ví dụ trên tomcat 7.0, thiết lập cấu hình timeout tại **tomcat_path/conf/web.xml**:

```
<!-- ===== Default Session Configuration ===== -->
<!-- You can set the default session timeout (in minutes) for all newly -->
<!-- created sessions by modifying the value below. -->
<session-config>
<session-timeout>30</session-timeout>
</session-config>
```

| | | | | |
|--|--|----------|-------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 5/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

2.1.4 Tạo mới session sau khi đăng nhập thành công.

2.1.5 Xóa giá trị sessionid và các dữ liệu gắn với session đó khi người dùng đăng xuất. Tại class quản lý phiên của người dùng:

```
@ManagedBean
@SessionScoped
public class UserManager {
    private User current;
    public void logout() throws IOException {
        ExternalContext ec =
        FacesContext.getCurrentInstance().getExternalContext();
        ec.invalidateSession();
        ec.redirect(ec.getRequestContextPath() + "/home.xhtml");
    }
}
```

2.1.6 Thiết lập thuộc tính “Secure” đối với các ứng dụng sử dụng HTTPS và “HTTP-Only” cho trường session cookie.

– Cấu hình thuộc tính “Secure” trong file *tomcat_path/conf/web.xml*:


```
<Connector port="8443" protocol="org.apache.coyote.http11.Http11Protocol"
    maxThreads="150" SSLEnabled="true" scheme="https" secure="true"
    clientAuth="false" sslProtocol="TLS" />
```

2.1.7 Cấu hình tính năng chống CSRF của JSF 2 trong file faces-config.xml như sau cho tất cả các trang.xhtml có các chức năng/nghiệp vụ quan trọng. Khuyến khích cấu hình cho toàn bộ các trang.xhtml của ứng dụng.

```
<?xml version = '1.0' encoding='UTF-8'?>
<face-config version="2.2"
    xmlns="http://xmlns.jcp.org/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
    http://xmlns.jcp.org/xml/ns/javaee/web-faceconfig 2.2.xsd">
    <protected-views>
    <url-pattern>csrf_protected_page.xhtml</url-pattern>
    </protected-views>
</face-config>
```

3 Phân quyền

3.1.1 Kiểm tra phân quyền dựa trên các đối tượng được lưu tại server (VD: tham số lưu trên session server, ma trận phân quyền lưu trên DB,...).

| | | | | |
|--|--|----------|--------------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 6/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

- 3.1.2 Phân quyền tối thiểu, chỉ đáp ứng đủ chức năng và tài nguyên cho người dùng/ứng dụng.
- 3.1.3 Sử dụng cơ chế quyền dữ liệu cho người dùng - người dùng chỉ được phép tác động (đọc, xóa, sửa) tới dữ liệu thuộc quyền sở hữu của mình hoặc được phép tác động.
- 3.1.4 Phía giao diện người dùng: Chỉ hiển thị các thành phần giao diện, đường dẫn, hàm,... tương ứng với quyền của người dùng.
- 3.1.5 Phía server: Kiểm tra quyền tác động của người dùng/ứng dụng trên các hàm và tài nguyên tương ứng trước khi thực hiện bất cứ tác vụ nào tới hệ thống.
- 3.1.6 Phải có tính năng xóa phiên làm việc hiện tại của người dùng hoặc các cơ chế tương đương đối với các trường hợp quyền người dùng bị thay đổi hoặc bị disable bởi người dùng có thẩm quyền.
- 3.1.7 Không đặt trang quản trị public internet, trong trường hợp bắt buộc phải đặt public cần giới hạn các IP được phép truy cập hoặc sử dụng cơ chế xác thực đa nhân tố (multiple authentications).

4 Tương tác với backend


4.1 SQL

- 4.1.1 Sử dụng mô hình truy vấn prepared statement (parameterized query) hoặc các hình thức tương đương.

```
SessionFactory sessionFactory = HibernateUtil.getSessionFactory();
Session session = sessionFactory.openSession();
Query createQuery = session.createQuery("from Customer where name =
:username");
createQuery.setParameter("username", username);
List list = createQuery.list();
session.close();
```

- 4.1.2 Trong 1 số trường hợp không sử dụng được các mô hình ở trên, cần thiết lập danh sách whitelist các đầu vào mong muốn

```
private static final String[] orderByWhitelist = new String[]{"name",
"address", "age"};
public static boolean isInOrderByWhitelist(String input){
    return Arrays.asList(orderByWhitelist).contains(input);
```

| | | | | |
|--|--|----------|--------------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 7/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

}

4.2 Tương tác với OS

4.2.1 Chỉ sử dụng đối tượng ProcessBuilder để thực hiện tạo lệnh cần truyền tới hệ điều hành, các tham số lấy từ client được đưa vào trong ArrayList và lệnh cần gọi phải nằm ở đầu danh sách (vị trí số 0). Trong một số trường hợp yêu cầu giới hạn đầu vào (do nghiệp vụ) thì cần thực hiện kiểm tra lại các tham số sao cho phù hợp với nghiệp vụ.


Ví dụ:

```
public static boolean pingServer(List<String> args)
{
    String command = "ping"
    try
    {
        args.add(0,command);
        ProcessBuilder pb = new ProcessBuilder(args);
        BufferedReader reader = new BufferedReader(new
InputStreamReader(pb.start().getInputStream()));
        StringBuilder builder = new StringBuilder();
        String line;
        while((line = reader.readLine()) != null)
        {
            builder.append(line);
            builder.append(System.getProperty("line.separator"));
        }
        System.out.println(builder.toString());
        return true;
    } catch (IOException ex) {
        Logger.getLogger(Vpn.class.getName()).log(Level.SEVERE,
null, ex);
        System.out.println("Execute command " + args.get(0) + "
faile");
        return false;
    }
}
```

4.3 Tương tác với file

4.3.1 Cần thực hiện validate các file được upload thông qua một class được implement giao diện Validator

```
import java.io.InputStream;
import java.util.Scanner;
import javax.faces.application.FacesMessage;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
```

| | | | | |
|--|--|----------|--------------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 8/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

```
import javax.faces.validator.*;
import javax.servlet.http.Part;

@FacesValidator(value="FileValidator")
public class FileValidator implements Validator{
    public void validateFile(FacesContext ctx,
                            UIComponent comp,
                            Object value) {
        List<FacesMessage> msgs = new ArrayList<FacesMessage>();
        Part file = (Part)value;
        if (file.getSize() > 1024) {
            msgs.add(new FacesMessage("file too big"));
        }
        if (!"text/plain".equals(file.getContentType())) {
            msgs.add(new FacesMessage("not a text file"));
        }
        if (!msgs.isEmpty()) {
            throw new ValidatorException(msgs);
        }
    }
}
```

4.3.2 Đối với các file được upload lên server.

Lưu trữ file bên ngoài thư mục web của ứng dụng.


Lập danh sách các phần mở rộng của các file được phép upload (whitelist).

Thực hiện đổi tên file (tên file được sinh ngẫu nhiên, phần mở rộng được lấy từ whitelist).

Ví dụ:

Implement của `Validator`, kiểm tra định dạng của file, độ lớn cho phép của file, ...:

```
public void validateFile(FacesContext ctx,
                        UIComponent comp,
                        Object value) {
    List<FacesMessage> msgs = new ArrayList<FacesMessage>();
    Part file = (Part)value;
    if (file.getSize() > 20971520) { //20 MB
        msgs.add(new FacesMessage("file too big"));
    }
    if (!"text/plain".equals(file.getContentType())) {
        msgs.add(new FacesMessage("not a text file"));
    }
    if (!msgs.isEmpty()) {
        throw new ValidatorException(msgs);
    }
}
```


| | | | | |
|--|--|----------|--------------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 9/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

Implement trên file .xhtml:

```
<h:form id="form" enctype="multipart/form-data">
<h:inputFile id="file"
value="#{bean.file}" validator="#{bean.validateFile}" />
<h:commandButton value="Upload" action="#{bean.upload}" />
</h:form>
```

- 4.3.3 Với các trường hợp không bắt buộc thì không lưu file upload trong thư mục web, bỏ quyền thực thi trên thư mục upload.
- 4.3.4 Khi cần tham chiếu tới các file tồn tại trên hệ thống cần thiết lập danh sách whitelist đầu vào mong muốn hoặc gán các giá trị định danh tương ứng cho các file thay vì truyền tên file như trong ví dụ 4.3.1.
- 4.3.5 Không trả về đường dẫn tuyệt đối của file.
- 4.3.6 Tất cả dữ liệu, tài nguyên hệ thống (báo cáo, file upload, file cấu hình...) không được lưu trong thư mục web hoặc trong thư mục cho phép truy cập trực tiếp không qua xác thực.


4.4 Mã hóa dữ liệu nhạy cảm

- 4.4.1 Thực hiện mã hóa các dữ liệu nhạy cảm trước khi lưu trữ (thông tin người dùng/khách hàng như: Số tài khoản, mã PIN, mật khẩu truy cập hệ thống/server khác, khóa mật mãm ...)
- Sử dụng các thuật toán mã hóa mạnh với độ dài khóa tối thiểu 256 bit như: AES, PGP, 3DES.
- Sử dụng hàm hash SHA1 để sinh khóa mã hóa cho thuật toán mã hóa từ mật khẩu.

Dưới đây là ví dụ về sinh khóa/mã hóa và giải mã sử dụng AES

```
public class Encryptor{
private static Logger LOGGER = LoggerFactory.getLogger(Encryptor.class);
private SecretKey secretKey;

public Encryptor(String passwordEncryption) {
    SecretKeyFactory factory;
    try {
        factory = SecretKeyFactory.getInstance("PBKDF2WithHmacSHA1");
        KeySpec spec = new
        PBEKeySpec(passwordEncryption.toCharArray(),
        passwordEncryption.getBytes(), 65536, 128);
```

| | | | | |
|--|--|----------|--------------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 10/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

```


        SecretKey derivedKey = factory.generateSecret(spec);
        this.secretKey = new SecretKeySpec(derivedKey.getEncoded(),
"AES");
    } catch (NoSuchAlgorithmException | InvalidKeySpecException e) {
        LOGGER.error("Khởi tạo mã hóa không thành công", e);
        throw new RuntimeException("Khởi tạo mã hóa không thành
cong", e);
    }
}
public String encrypt(String message) {
    try {
        Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
        cipher.init(Cipher.ENCRYPT_MODE, secret);
        AlgorithmParameters params = cipher.getParameters();
        byte[] iv =
params.getParameterSpec(IvParameterSpec.class).getIV();
        byte[] ciphertext = cipher.doFinal(message.getBytes("UTF-
8"));

        byte[] result = new byte[iv.length + ciphertext.length];
        System.arraycopy(iv, 0, result, 0, iv.length);
        System.arraycopy(ciphertext, 0, result, iv.length,
            ciphertext.length);
        return Base64.encodeBase64String(result);

    } catch (NoSuchAlgorithmException | NoSuchPaddingException |
InvalidKeyException | InvalidParameterSpecException |
IllegalBlockSizeException | BadPaddingException |
UnsupportedEncodingException e) {
        LOGGER.error("Mã hóa không thành công", e);
        throw new RuntimeException("Mã hóa không thành công", e);
    }
}
public String decrypt(String message) {
    byte[] result = Base64.decodeBase64(message);
    byte[] iv = Arrays.copyOfRange(result, 0, 16);
    byte[] m = Arrays.copyOfRange(result, 16, result.length);
    try {
        Cipher decryptor =
Cipher.getInstance("AES/CBC/PKCS5Padding");
        decryptor
            .init(Cipher.DECRYPT_MODE, secret, new
IvParameterSpec(iv));
        return new String(decryptor.doFinal(m), "UTF-8");

    } catch (NoSuchAlgorithmException | NoSuchPaddingException |
InvalidKeyException | InvalidAlgorithmParameterException |
UnsupportedEncodingException | IllegalBlockSizeException |
BadPaddingException e) {
        LOGGER.error("Giải mã không thành công", e);
        throw new RuntimeException("Giải mã không thành công", e);
    }
}
}

```

| | | | | |
|--|--|----------|--------------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 11/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

4.5 Tương tác với webservice/API của ứng dụng web khác

4.5.1 Khi tạo thực hiện tạo các tham số trên URL, tham số trong HTTP Body của các request GET/POST tới các ứng dụng/dịch vụ web khác phía sau thì các tham số nhận được từ phía người dùng cần phải được thực hiện encode.

Đối với JSF2 có thể sử dụng encode qua Filter.

Ví dụ:

```
public class EncodingFilter implements Filter {

    private String encoding = "utf-8";

    public void doFilter(ServletRequest request,

        ServletResponse response, FilterChain filterChain) throws IOException,
        ServletException {
        request.setCharacterEncoding(encoding);
        filterChain.doFilter(request, response);
    }

    public void init(FilterConfig filterConfig) throws ServletException {
        String encodingParam = filterConfig.getInitParameter("encoding");
        if (encodingParam != null) {
            encoding = encodingParam;
        }
    }

    public void destroy() {
        // nothing todo
    }
}
```

4.6 NoSQL


4.6.1 Các hệ NoSQL cần phải được cấu hình đảm bảo các yêu cầu: Xác thực, phân quyền, tắt các dịch vụ dư thừa, không sử dụng.

4.6.2 Phụ thuộc vào hệ NoSQL sử dụng, sử dụng các api hỗ trợ truy vấn an toàn, thực hiện ép kiểu/encode dữ liệu trước khi đưa vào tương tác với các hệ NoSQL. Ví dụ với MongoDB:

- Với tìm kiếm dữ liệu:

Sử dụng filter trong 4.4.1 như filter default của ứng dụng, như vậy tất cả các param khi truyền tới đã được encode theo định dạng mong muốn. Sau đó thực hiện thao tác truy vấn database bình thường.

```
DBCollection table = db.getCollection("user");
```

| | | | | |
|--|--|----------|--------------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 12/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

```

BasicDBObject searchQuery = new BasicDBObject();
searchQuery.put("name", "xxx");

DBCursor cursor = table.find(searchQuery);

while (cursor.hasNext()) {
    System.out.println(cursor.next());
}

```

4.7 Xpath

- 4.7.1 Thiết lập danh sách whitelist các ký tự đầu vào mong muốn, đầu vào nên là tập hợp của chữ cái, chữ số (`regex = "^[a-zA-Z0-9]*$"`)
- 4.7.2 Lập danh sách blacklist các ký tự đặc biệt (() = '[] : , * / và dấu cách), loại bỏ các đầu vào có chứa các ký tự nằm trong blacklist.

4.8 Tương tác XML


- 4.8.1 Disable tính năng external entity resolve và remote doctype retrieval của đối tượng xml parser khi đọc dữ liệu xml từ phía người dùng gửi lên.

- Đối với các đối tượng như **JAXP** như **DocumentBuilderFactory** hoặc **SAXParserFactory** thiết lập các 4 thuộc tính sau trước khi xử lý XML như ví dụ:

```

FEATURE = "http://apache.org/xml/features/disallow-doctype-decl";
FEATURE = "http://xml.org/sax/features/external-general-entities";
FEATURE = "http://xml.org/sax/features/external-parameter-entities";
public void processXML()
{
    DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
    try {
        String FEATURE = "http://apache.org/xml/features/disallow-
doctype-decl";
        dbf.setFeature(FEATURE, true);
        // If you can't completely disable DTDs, then at least do
the following:
        // Xerces 1 - http://xerces.apache.org/xerces-
j/features.html#external-general-entities
        // Xerces 2 - http://xerces.apache.org/xerces2-
j/features.html#external-general-entities
        FEATURE = "http://xml.org/sax/features/external-general-
entities";
        dbf.setFeature(FEATURE, false);
        // Xerces 1 - http://xerces.apache.org/xerces-
j/features.html#external-parameter-entities

```

| | | | | |
|--|--|----------|--------------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 13/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

```
// Xerces 2 - http://xerces.apache.org/xerces2-j/features.html#external-parameter-entities
FEATURE = "http://xml.org/sax/features/external-parameter-entities";

dbf.setFeature(FEATURE, false);
// and these as well, per Timothy Morgan's 2014 paper: "XML Schema, DTD, and Entity Attacks" (see reference below)
dbf.setIncludeAware(false);
dbf.setExpandEntityReferences(false);
.....
catch (ParserConfigurationException | SAXException | IOException
e) {
    // This should catch a failed setFeature feature
    logger.info("ParserConfigurationException was thrown. The
feature '" +
FEATURE +
"' is probably not supported by your XML
processor.");
}
```

- Đối với **StAX Parser** như **XMLInputFactory** thì thiết lập thuộc tính **SUPPORT_DTD** là false như sau:

```
xmlInputFactory.setProperty(XMLInputFactory.SUPPORT_DTD, false);
xmlInputFactory.setProperty(XMLInputFactory.IS_SUPPORTING_EXTERNAL_ENTITIES, false);
```

- Đối với **JDOM Parser** thì thiết lập thuộc tính **setExpandEntities** là false như:


```
SAXBuilder builder = new SAXBuilder();
builder.setExpandEntities(false); //Retain Entities
builder.setValidation(false);
```

4.8.2 Đưa các đoạn dữ liệu nhận được từ người dùng vào các thẻ **<![CDATA[]]>** khi tạo dữ liệu xml.

4.8.3 Sử dụng **XPath** để thao tác với dữ liệu XML.

Ví dụ:

```
$xml = <?xml version="1.0" encoding="UTF-8"?>
<people>
  <person>
    <firstname>Lars</firstname>
    <lastname>Vogel</lastname>
    <city>Heidelberg</city>
  </person>
  <person>
    <firstname>Jim</firstname>
    <lastname>Knopf</lastname>
    <city>Heidelberg</city>
```

| | | | | |
|--|--|----------|--------------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 14/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

```

</person>
<person>
  <firstname>Lars</firstname>
  <lastname>Strangelastname</lastname>
  <city>London</city>
</person>
<person>
  <firstname>Landerman</firstname>
  <lastname>Petrelli</lastname>
  <city>Somewhere</city>
</person>
<person>
  <firstname>Lars</firstname>
  <lastname>Tim</lastname>
  <city>SomewhereElse</city>
</person>
</people>

```

```

public class QueryXML {
    public void query() throws ParserConfigurationException, SAXException,
        IOException, XPathExpressionException {
        // standard for reading an XML file
        DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
        factory.setNamespaceAware(true);
        DocumentBuilder builder;
        Document doc = null;
        XPathExpression expr = null;
        builder = factory.newDocumentBuilder();
        doc = builder.parse("person.xml");

        // create an XPathFactory
        XPathFactory xFactory = XPathFactory.newInstance();


        // create an XPath object
        XPath xpath = xFactory.newXPath();

        // compile the XPath expression
        expr = xpath.compile("//person[firstname='Lars']/lastname/text()");
        // run the query and get a nodeset
        Object result = expr.evaluate(doc, XPathConstants.NODESET);

        // cast the result to a DOM NodeList
        NodeList nodes = (NodeList) result;
        for (int i=0; i<nodes.getLength();i++){
            System.out.println(nodes.item(i).getNodeValue());
        }

        // new XPath expression to get the number of people with name Lars
        expr = xpath.compile("count(//person[firstname='Lars'])");
        // run the query and get the number of nodes
    }
}

```

| | | | | |
|--|--|----------|--------------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 15/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

```

Double number = (Double) expr.evaluate(doc, XPathConstants.NUMBER);
System.out.println("Number of objects " + number);

// do we have more than 2 people with name Lars?
expr = xpath.compile("count(//person[firstname='Lars']) >2");
// run the query and get the number of nodes
Boolean check = (Boolean) expr.evaluate(doc,
XPathConstants.BOOLEAN);
System.out.println(check);
}

public static void main(String[] args) throws
XPathExpressionException, ParserConfigurationException, SAXException,
IOException {
    QueryXML process = new QueryXML();
    process.query();
}
}

```

4.9 LDAP

- 4.9.1 Thiết lập danh sách whitelist các ký tự đầu vào mong muốn, đầu vào nên là tập hợp của chữ cái, chữ số.
- 4.9.2 Lập danh sách blacklist các ký tự đặc biệt (() ; , * | & = và nullbyte), loại bỏ các đầu vào có chứa các ký tự nằm trong blacklist.


5 Kiểm soát dữ liệu đầu vào

- 5.1.1 Việc kiểm tra dữ liệu đầu vào phải được thực hiện trên server và tiến hành trên tất cả các nguồn dữ liệu có tương tác với người dùng (tham số lấy từ GET/POST request, HTTP Headers, dữ liệu từ DB, từ file upload,...).
- 5.1.2 Xác định rõ chuẩn định dạng bảng mã sử dụng (Unicode – UTF8/UTF16 hay ASCII...) của dữ liệu đầu vào, thực hiện validate dữ liệu sau khi đã decode đầu vào về 1 định dạng chuẩn và nhất quán. Việc này cần được thực hiện cấu hình bộ lọc filter trong file web.xml của ứng dụng như sau:

```

<filter>
  <filter-name>CharsetFilter</filter-name>
  <filter-class>security.baseline.CharsetFilter</filter-class>
  <init-param>
    <param-name>requestEncoding</param-name>
    <param-value>UTF-8</param-value>
  </init-param>
</filter>

```

| | | | | |
|--|--|----------|-------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 16/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

```

<filter-mapping>
  <filter-name>CharsetFilter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>

```

- Xây dựng class CharsetFilter để thực hiện thiết lập encoding

```

public class CharsetFilter implements Filter {
    private String encoding;
    @Override
    public void init(FilterConfig config) throws ServletException {
        encoding = config.getInitParameter("requestEncoding");

        if (null == encoding) {
            encoding = "UTF-8";
        }
    }
    @Override
    public void doFilter(ServletRequest request, ServletResponse
response,
        FilterChain next) throws IOException, ServletException {
        if (null == request.getCharacterEncoding()) {
            request.setCharacterEncoding(encoding);
        }
        response.setContentType("text/html; charset=UTF-8");
        response.setCharacterEncoding("UTF-8");
        next.doFilter(request, response);
    }
}


```

5.1.3 Validate kiểu dữ liệu, phạm vi, độ dài dữ liệu và định dạng dữ liệu, khuyến nghị nên thực hiện ép kiểu dữ liệu cho các tham số nhận được từ người dùng. Việc validate kiểu dữ liệu có thể được thực hiện như ví dụ sau:

```

public class InputAction extends ActionSupport {
    private String name;
    private int age;
    private Date dob;
    private String email;
    public String getName() {
        return name;
    }
    @RequiredStringValidator(message = "Name must be required",
shortCircuit = true, trim = true)
    @StringLengthFieldValidator(message = "Name length must be between 5
and 12", shortCircuit = true, trim = true, minLength = "5", maxLength =
"12")
    @RegexFieldValidator(message = "Name contain only letter, number,
_", type = ValidatorType.FIELD, regexExpression = "([a-zA-Z0-9_]*)"
    public void setName(String name) {
        this.name = name;
    }
}

```


| | | | | |
|--|--|----------|-------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 17/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

```

    }
    public int getAge() {
        return age;
    }
    @IntRangeFieldValidator(message = "Age must from 0 to 60",
shortCircuit = true, min = "0", max = "60")
    public void setAge(int age) {
        this.age = age;
    }
    public Date getDob() {
        return dob;
    }
    @DateRangeFieldValidator(message = "Date must in range 01/01/1955 to
31/12/2015", dateFormat = "dd/MM/yyyy", shortCircuit = true, min =
"01/01/1955", max = "31/12/2015")
    public void setDob(Date dob) {
        this.dob = dob;
    }
}

```

6 Kiểm soát dữ liệu đầu ra

6.1.1 Phải chỉ rõ character encoding cho dữ liệu đầu ra (thực hiện tương tự mục 5.1.2)

6.1.2 Response body phải được encode theo ngữ cảnh sử dụng. Một số trường hợp phổ biến:

- Đầu ra là html thì thực hiện encode html – mặc định JSF đã thực hiện encode tại các output sử dụng tag h:outputText (không được phép thiết lập escape="false" tại các tag h:outputText)

```

<h:outputText value="#{user.name}" />
<h:outputText value="#{user.name}" escape="true" />
<h:inputText value="#{user.name}" />

```

- Đầu ra là URL thì thực hiện encode URL (sử dụng hàm urlencode)

```

<h:outputLink value="#{bean.url}">Click
    <f:param name="param1" value="#{bean.urlParam1}" />
    <f:param name="param2" value="#{bean.urlParam2}" />
</h:outputLink>


```

- Đầu ra là json thì encode dữ liệu trả về dạng object/text plain

```

// Thay vì khởi tạo đối tượng kiểu ArrayList
List<String> lists = new ArrayList<String>(); //UNSAFE
// Khởi tạo đối tượng kiểu HashMap như sau
Map<String, String> maps = new HashMap<String, String>(); //SAFE

```

| | | | | |
|--|--|----------|--------------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 18/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

- 6.1.3 Response header: lọc bỏ các kí tự xuống dòng (\n, \r) trong dữ liệu do người dùng truyền lên.

```
header = header.replaceAll("\\p{Cntrl}", "");
```

- 6.1.4 Cookie trả về có đủ các thuộc tính sau: Domain, path, httponly, expire, secure. Tránh lưu trữ các dữ liệu quan trọng (username/password) trên cookie. Trong trường hợp cần lưu trữ thì phải thực hiện mã hóa bằng các thuật toán mã hóa mạnh như AES-CBC-256, khóa mã hóa phải được lưu giữ ở trên server.

```
Cookie cookie = new Cookie("cookieOnAdminPage", cookieValue);
/* 1. Giới hạn thuộc tính của cookie */


// ví dụ cookie cookieOnAdminPage để nhớ đăng nhập trên trang /admin/
cookie.setHttpOnly(true);
cookie.setMaxAge(7*24*60*60);
cookie.setPath("/admin/");
cookie.setComment("Cookie nho dang nhap nguoi dung, co gia tri tren link
/admin trong thoi gian 7 ngay");
//cookie.setSecure(true); // uncomment nếu sử dụng https
cookie.setValue(cookieValue);
response.addCookie(cookie);

/* Mã hóa cookie */
// Thực hiện tương tự mục 4. Tương tác với backend => Mã hóa các dữ liệu
nhập cảm trước khi lưu trữ
```

- 6.1.5 Không sử dụng dữ liệu từ phía người dùng gửi lên để tạo URL chuyển hướng, nếu bắt buộc phải chuyển hướng thì cần có cơ chế tạo whitelist danh sách các domain/ip được phép.

```
private static final String[] allowedURL = new
String[]{"voffice.viettel.vn", "thongtinnhansu.viettel.vn",
"viettelfamily.vn"};

public static boolean isAllowedURL(String url) {
    return Arrays.asList(allowedURL).contains(url);
}
```

| | | | | |
|--|--|----------|--------------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 19/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

7 Kiểm soát ngoại lệ và ghi log ứng dụng

7.1.1 Xử lý các ngoại lệ bằng try-catch và trả về các thông báo lỗi chung đã tùy chỉnh, thông báo lỗi trả về không được chứa các thông tin nhạy cảm của người dùng, hệ thống,... Cấu hình trong file web.xml của ứng dụng như sau:

```
<web-app>
  <display-name>Archetype Created Web Application</display-name>
  <error-page>
    <exception-type>java.lang.Exception</exception-type>
    <location>/WEB-INF/jsp/error.jsp</location>
  </error-page>
</web-app>
```

7.1.2 Các thông tin lỗi, ngoại lệ này phải được log lại để phục vụ bảo trì, xác định nguyên nhân lỗi ứng dụng.


7.1.3 File log phải được đặt tại thư mục an toàn ngoài thư mục web.

7.1.4 Không log lại các dữ liệu nhạy cảm (thông tin người dùng, session id, thông tin hệ thống).

7.1.5 Sử dụng log4j để ghi log. Cấu hình như sau:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">

  <appender name="ConsoleAppender"
    class="org.apache.log4j.ConsoleAppender">
    <layout class="org.apache.log4j.PatternLayout">
      <param name="ConversionPattern" value=
        "Time: %d{ISO8601} %-5p %n
        Location: %l%n
        Message: %m %n %n"/>
    </layout>
  </appender>
  <appender name="wflog"
    class="org.apache.log4j.DailyRollingFileAppender">
    <param name="file" value="${catalina.base}/logs/wflog.log" />
    <param name="append" value="true" />
    <param name="datePattern" value="'.'yyyy-MM-dd-HH" />
    <layout class="org.apache.log4j.PatternLayout">
      <param name="ConversionPattern"
        value="%d{HH:mm:ss.SSS} %t %C{1} %-5p %n%l%n%m%n%n"
      />
    </layout>
  </appender>
<!-- Ví dụ cấu hình logger cho package com.viettel -->
```

| | | | | |
|--|--|----------|-------------|-------------------------------|
|  Hãy nói theo cách của bạn | TỔNG CÔNG TY MẠNG LƯỚI VIETTEL | | | Mã hiệu: GL.00.CNTT.14 |
| | HƯỚNG DẪN LẬP TRÌNH AN TOÀN TRONG PHÁT TRIỂN ỨNG DỤNG WEB SỬ DỤNG FRAMEWORK JSF | | | Ngày có hiệu lực: 01/11/2015 |
| | | | | Ngày hết hiệu lực: 30/11/2016 |
| | | | | Lần ban hành: 01 |
| | | | | Trang: 20/20 |
| | ĐV áp dụng | Toàn TCT | ĐV kiểm tra | Phòng Công nghệ Thông tin |
| | Phạm vi áp dụng | Toàn TCT | | |

```

<logger name="com.viettel">
  <level value="error"/>
  <appender-ref ref="ConsoleAppender"/>
  <appender-ref ref="wflog"/>
</logger>

<root>
  <level value="error" />
  <appender-ref ref="ConsoleAppender"/>
  <appender-ref ref="wflog"/>
</root>
</log4j:configuration>

```

8 Sử dụng Framework, thư viện, các plugin (third-party components)

- 8.1.1 Loại bỏ các thành phần dư thừa, thư viện không cần thiết. Đối với các plugin từ bên thứ 3, cần cấu hình đảm bảo việc truy cập các plugin này phải tuân theo các logic nghiệp vụ của ứng dụng chính.
- 8.1.2 Sử dụng phiên bản mới nhất của framework tại thời điểm phát triển ứng dụng.
- 8.1.3 Thường xuyên cập nhật các bản vá lỗi cho framework.
- 8.1.4 Tắt chế độ development của framework khi triển khai ứng dụng thực tế. Loại bỏ tag **debug** có trong các **face** của **JSF**.

9 Xử lý nghiệp vụ (business logic)

- 9.1.1 Lập trình viên phải nắm rõ được toàn bộ luồng nghiệp vụ của chức năng do mình xây dựng. Quản trị dự án phải nắm được toàn bộ luồng nghiệp vụ của ứng dụng và phải xây dựng các testcase về ATTT – các test case này bao gồm các trường hợp không mong muốn của ứng dụng (chuyển tiền/cộng điểm âm, đọc/chỉnh sửa dữ liệu không thuộc quyền sở hữu của mình...
- 9.1.2 Các chức năng quan trọng (ví dụ chuyển khoản ngân hàng) phải có tính năng đồng bộ (lock flag), hoặc các hình thức tương đương để tránh lỗi về mất đồng bộ (race condition).