

**TÀI LIỆU HƯỚNG DẪN SỬ DỤNG**

**Mã hiệu dự án:**

**Mã hiệu tài liệu: HDSD**

**Mục lục**

[I. Giới thiệu tổng quan 4](#_Toc459795078)

[1. Mô hình hoạt động 4](#_Toc459795079)

[2. Mô hình kiến trúc code 5](#_Toc459795080)

[3. Sơ đồ Lib, các phụ thuộc Project và project 6](#_Toc459795081)

[II. Môi trường bắt buộc để lập trình trên FW mới 7](#_Toc459795082)

[III. Hướng dẫn Gọi webservice 11](#_Toc459795092)

[1. Tạo project để build file jar choc các webservice đã viết 11](#_Toc459795093)

[2. Add lib vừa build vào project erp-web 11](#_Toc459795094)

[3. Khai báo thông tin service 11](#_Toc459795095)

[4. Gọi webservice trong lớp controller bất kì 12](#_Toc459795096)

[IV. Các bước lập trình chức năng 14](#_Toc459795104)

[1. Sử dụng ToolGenCode 14](#_Toc459795105)

[1.1. Bật chương trình từ file jar 14](#_Toc459795106)

[1.2. Cấu hình DB. Sau đó nháy đúp vào hàng chứa dữ liệu DB đã được cấu hình 15](#_Toc459795107)

[1.3. Chọn đối tượng để gencode 16](#_Toc459795108)

[1.4. Nhấn nút Gencode và nhận Kết quả gencode 16](#_Toc459795109)

[1.5. Copy lần lượt các file trong từng thư mục vào các package tương ứng trong project 17](#_Toc459795110)

[1.6. Sau khi sử dụng tool, chúng ta có các webservice cơ bản lên một đối tượng trong CSDL 18](#_Toc459795111)

[1.7. Cấu hình để expose ra webservice từ file spring config 18](#_Toc459795112)

[2. Lập trình một webservice mới 19](#_Toc459795113)

[2.1. Code vào DAO 19](#_Toc459795114)

[2.2. Code business gọi dao: code một interface và một implement của interface này 19](#_Toc459795115)

[2.3. Code webservice gọi business: code một interface và một implement của interface này 20](#_Toc459795116)

[2.4. Cấu hình để expose ra webservice từ file spring config 20](#_Toc459795117)

[V. Hướng dẫn lập trình Restful service 21](#_Toc459795118)

[1. Khai báo webservice 21](#_Toc459795119)

[2. Khai báo cấu hình web service 21](#_Toc459795120)

[VI. Cấu hình Hibernate cho project service 22](#_Toc459795123)

[1. Mã hóa thông tin database bằng thư viên Jasypt 22](#_Toc459795124)

[2. Cấu hình mật khẩu mã hóa trong tomcat 23](#_Toc459795125)

[3. Thêm thông tin vào file cấu hình 23](#_Toc459795126)

Giới thiệu tổng quan

## Mô hình hoạt động

Erp-Service-Mobile

Erp-Service

Erp-ConsumerWS

Erp-Utils

## Sơ đồ Lib, các phụ thuộc Project và project

Erp-Service-Base

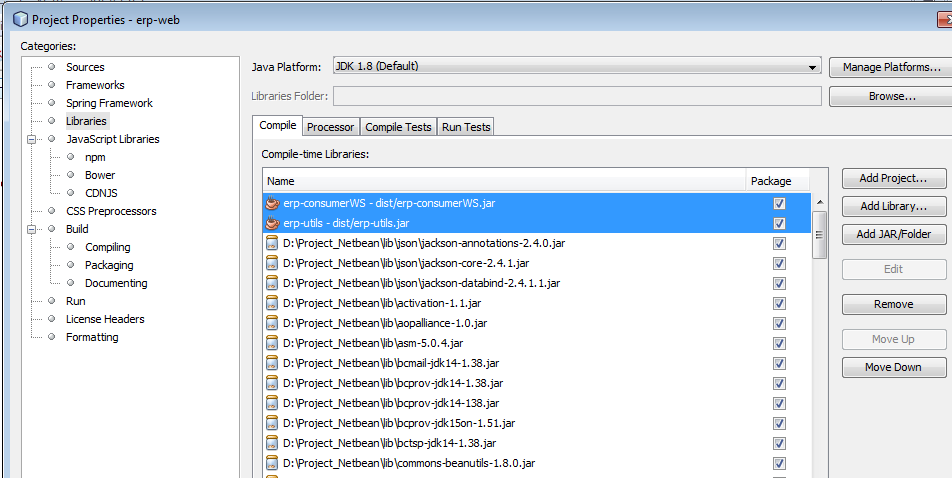
Erp-Libs

Mô tả: Chiều mũi tên là chỉ ra Project nguồn chứa project đích trong tham chiếu Lib.

Ví dụ dưới là Lib của erp-service: chứa erp-utils, erp-service-Base, Erp-Libs

Project erp-service-mobile :

* Project erp-service-mobile sẽ gọi các WS đã có trên Project erp-service
* Các nghiệp vụ mới sẽ Code trên Project erp-service-mobile theo hướng dẫn tạo WS restful bên dưới.



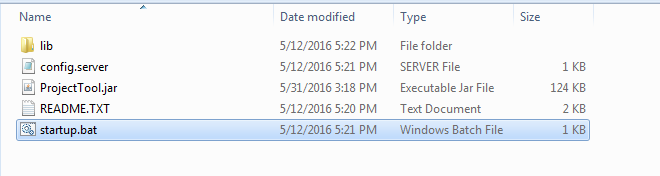
# Môi trường bắt buộc để lập trình trên FW mới

* 1. OS là window hoặc Linux
  2. Cài Eclipse
  3. Lấy bản Tomcat\_chuan trên thư mục source đã có các cấu hình
     1. Bin\setenv.bat có nội dung: set JAVA\_OPTS=-Dfile.encoding=UTF-8 -Xms128m -Xmx512m -XX:PermSize=64m -XX:MaxPermSize=256m -Dorg.apache.cxf.stax.allowInsecureParser=1
     2. Conf/ catalina.properties có thêm dòng này cuối: Các folder bên trong đặt theo nghiệp vụ .

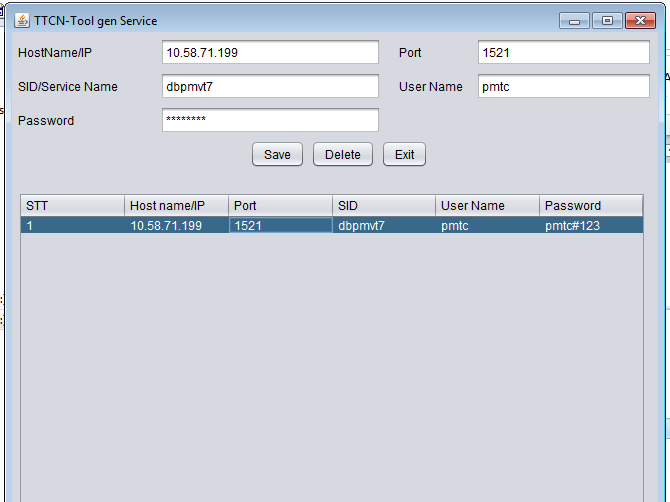
# Các bước lập trình chức năng

## Sử dụng ToolGenCode

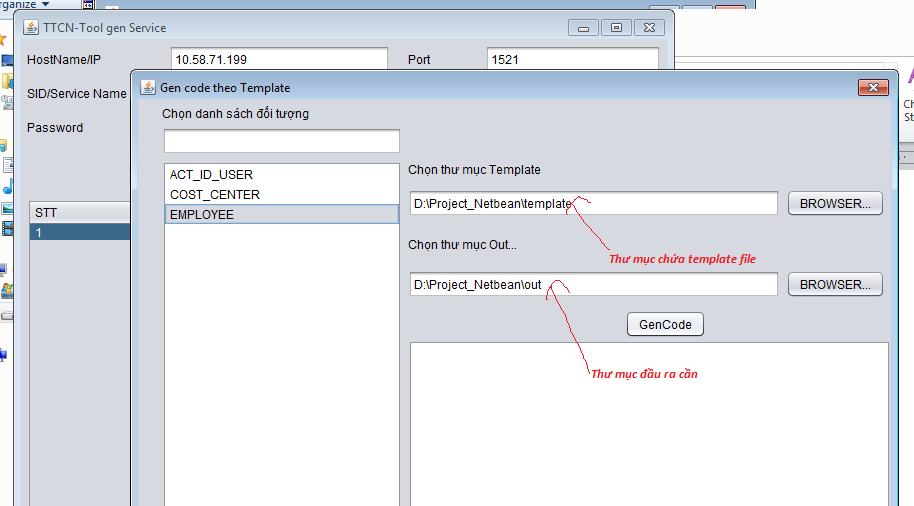
### Bật chương trình từ file jar



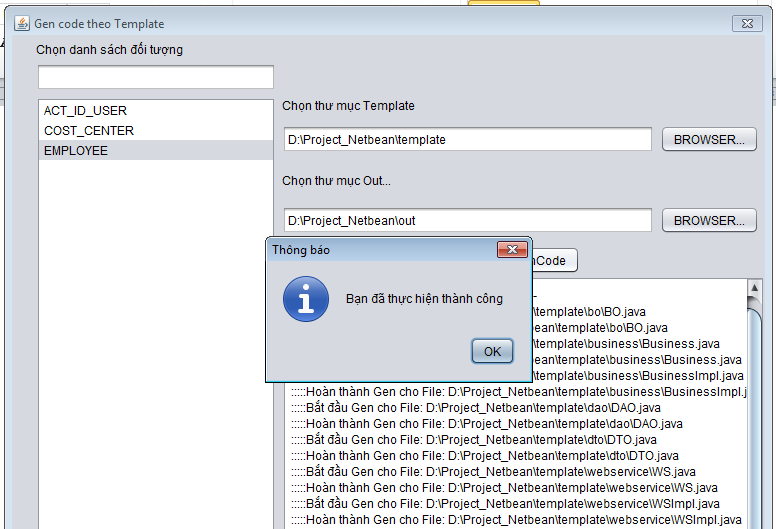
### Cấu hình DB. Sau đó nháy đúp vào hàng chứa dữ liệu DB đã được cấu hình

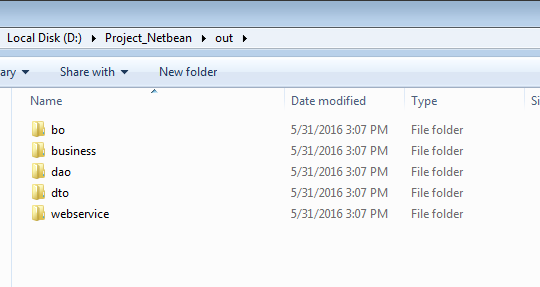


### Chọn đối tượng để gencode

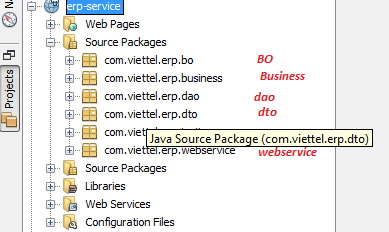


### Nhấn nút Gencode và nhận Kết quả gencode

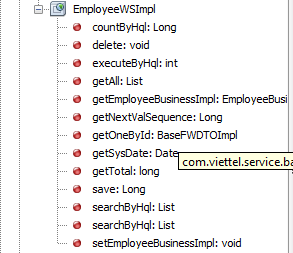




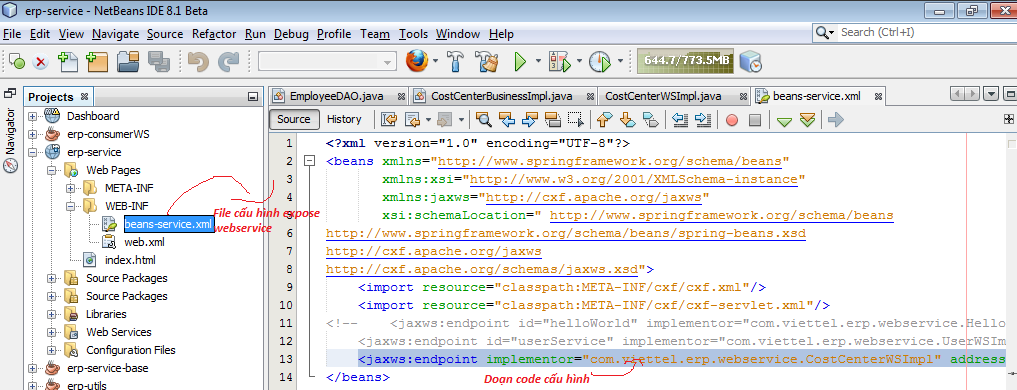
### Copy lần lượt các file trong từng thư mục vào các package tương ứng trong project



### Sau khi sử dụng tool, chúng ta có các webservice cơ bản lên một đối tượng trong CSDL



### Cấu hình để expose ra webservice từ file spring config



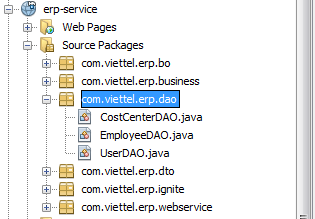
<jaxws:endpoint implementor="com.viettel.erp.webservice.CostCenterWSImpl" address="/costCenterService"/>

com.viettel.erp.webservice.CostCenterWSImpl: package trỏ đến webservice implement

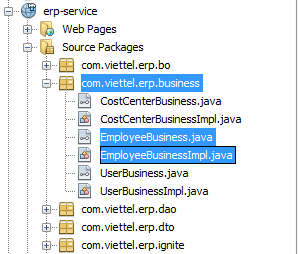
costCenterService: địa chỉ webservice để expose

## Lập trình một webservice mới

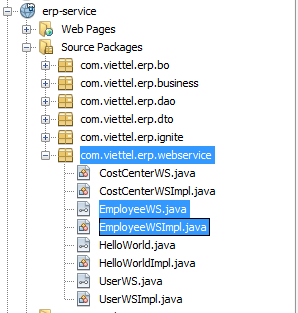
### Code vào DAO



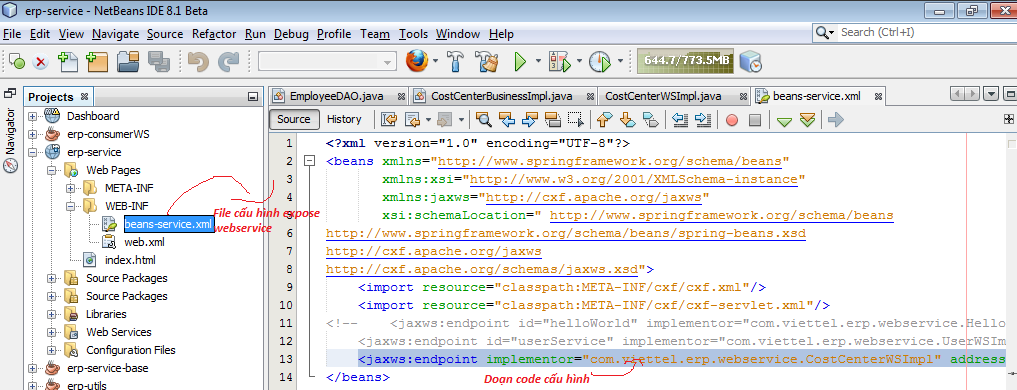
### Code business gọi dao: code một interface và một implement của interface này



### Code webservice gọi business: code một interface và một implement của interface này



### Cấu hình để expose ra webservice từ file spring config



<jaxws:endpoint implementor="com.viettel.erp.webservice.CostCenterWSImpl" address="/costCenterService"/>

com.viettel.erp.webservice.CostCenterWSImpl: package trỏ đến webservice implement

costCenterService: địa chỉ webservice để expose

# Hướng dẫn lập trình Restful service

## Khai báo webservice

* Restful webservice sẽ được code trong package **com.viettel.erp.rest** và **các sub\_package** tùy theo nghiệp vụ
* 1 web service gồm 2 file code
  + **<bussinesss>RestService**: vd : UserRestService.java , là interface của service. Tại đây sẽ khai báo các đầu hàm service

VD

|  |
| --- |
| @GET  @Path("/user/{id}")  @Consumes({MediaType.APPLICATION\_JSON, MediaType.APPLICATION\_XML})  @Produces({MediaType.APPLICATION\_JSON, MediaType.APPLICATION\_XML})  public Response getUserById(@PathParam("id") Long id); |

Trong đó

* + - **@GET**: là khai báo loại http request, tùy theo nghiệp vụ là GET,POST,PUT, DELETE
    - **@Path("/user")**: là đường dẫn của resource , đường dẫn này map với request của người dung. Tại VD: đang đặt đường dẫn là user. Đường dẫn request gọi tới hàm này như sau: <http://localhost:8084/erp-service/userServiceRest/user>.
    - **@Consumes({MediaType.APPLICATION\_JSON, MediaType.APPLICATION\_XML})**: khai báo định dạng của request . Tại ví dụ này, định dạng request cho phép cả json và xml
    - **@ Produces ({MediaType.APPLICATION\_JSON, MediaType.APPLICATION\_XML})**: khai báo định dạng của request . Tại ví dụ này, định dạng request cho phép cả json và xml: tương tự như Consumes, khai báo này định dạng response trả vể cho người dung gồm cả 2 định dạng json và xml
    - **@PathParam("id")**: khai báo cho phép map giá trị id trong request với giá trị đầu vào của hàm xử lý.
  + **<bussinesss>RestServiceImpl**: vd : UserRestServiceImpl.java , là implement của service tạo ở trên: tại đây sẽ lập trình xử lý nghiệp vụ của các hàm xử lý nghiệp vụ, Lập trình như hướng dẫn ở mục III Các bước lập trình chức năng

## Khai báo cấu hình web service

* Khai báo cấu hình vào file **WEB-INF/ beans-service.xml**
* Khai báo service như sau:

|  |
| --- |
| <jaxrs:server id="userServiceRest" address="/userServiceRest">  <jaxrs:providers>  <ref bean="jsonProvider" />  </jaxrs:providers>  <jaxrs:serviceBeans>  <bean id="userServiceClass" class="com.viettel.erp.rest.UserRestServiceImpl"/>  </jaxrs:serviceBeans>  </jaxrs:server> |

* Trong đó:
  + **id="userServiceRest":** là id đặt cho service
  + **address="/userServiceRest":** là địa chỉ của service. VD: Đường dẫn service khi gọi như sau: http://localhost:8084/erp-service/userServiceRest/user/1
  + **<bean id="userServiceClass" class="com.viettel.erp.rest.UserRestServiceImpl"/>**: là cấu hình bean trỏ đến file implement service ở bước.

## Cấu hình tomcat triển khai service

* Windows
  + Thêm dòng sau vào file <tomcat>/bin/setenv.bat

|  |
| --- |
| set JAVA\_OPTS= -Dorg.apache.cxf.stax.allowInsecureParser=1 |

* Linux
  + Thêm dòng sau vào file <tomcat>/bin/catalina.sh

|  |
| --- |
| JAVA\_OPTS="-Dorg.apache.cxf.stax.allowInsecureParser=1" |

# Cấu hình Hibernate cho project service

## Mã hóa thông tin database bằng thư viên Jasypt

Ví dụ mã hóa tại file com.viettel.service.base.utils.Decrypt, trong project erp-service-base

|  |
| --- |
| BasicTextEncryptor textEncryptor = new BasicTextEncryptor();  textEncryptor.setPassword("123456a@");  String plainText1 = textEncryptor.encrypt("jdbc:oracle:thin:@//10.58.71.199:1521/dbpmvt7");  String plainText2 = textEncryptor.encrypt("pmtc");  String plainText3 = textEncryptor.encrypt("pmtc#123"); |

* Trong đó textEncryptor.setPassword("123456a@") : thiết lập mật khẩu mã hóa lấy từ biến môi trường bên dưới

## Cấu hình mật khẩu mã hóa trong tomcat

* Cấu hình mậ khẩu trong file <tomcat>/conf/ catalina.properties
* Lưu ý: mật khẩu trùng với mật khẩu dung để mã hóa ở bước 1

|  |
| --- |
| PASSWORD\_ENV\_VARIABLE=123456a@ |

## Thêm thông tin vào file cấu hình

* Thêm thông tin mã hóa ở bước 1 vào file cấu hình: datasources.properties

|  |
| --- |
| db\_url = ENC(Paj5baOWcvmGe1oDaWHXjU2ByirJQVBQ2X0VSFP1JZbw/lUqyTzBpKHF9einSZJtCwYwOnCW/CY=)  db\_username = ENC(Dqs6CtyZe6dF74GQUJnqSw==)  db\_password = ENC(cYryY8VuyIkroyjyFJer86j/jFA+DgFR) |

* Cấu hình thông tin DB trong file hibernate-config.xml
* Cấu hình datasource trỏ đến DB vừa cấu hình

|  |
| --- |
| <bean id="dataSource"  class="com.mchange.v2.c3p0.ComboPooledDataSource" destroy-method="close">  <property name="driverClass" value="oracle.jdbc.OracleDriver"/>    <property name="jdbcUrl" value="${db\_url}"/>  <property name="user" value="${db\_username}"/>  <property name="password" value="${db\_password}"/>  <property name="acquireIncrement" value="2"/>  <property name="minPoolSize" value="10"/>  <property name="maxPoolSize" value="50"/>  <property name="maxIdleTime" value="600"/>  <property name="maxStatementsPerConnection" value="3000"/>  </bean> |

* File trong project mẫu đã được cấu hình thông tin này, nếu muốn thay đổi thông tin DB, chỉ cần cập nhật thông tin DB trong file **datasources.properties**

tomcatPassportTest\_8606/webapps/passportv3/WEB-INF/classes

