UstaEmpresarial

Equipo JBLESS



ESPECIFICACIÓN FUNCIONAL

**NOMINA EJB**

Versión 1.0

Mayo 06 de 2019

|  |  |  |  |
| --- | --- | --- | --- |
| **REGISTRO DE CAMBIOS AL DOCUMENTO** | | | |
|  | | | |
| **Versión** | **Fecha** | **Descripción del cambio** | **Autor** |
| 1.0 | 29/04/2019 | Creación de fachada | Equipo Jbless |
| 1.0 | 29/04/2019 | Creación del bean | Christian Montes |
| 1.0 | 29/04/2019 | Creación del | Oscar Gomez |
| 1.0 | 30/04/2019 | Mapeo de entidades | Mario Murcia  Angie Suárez |
| 1.0 | 06/05/2019 | Creación del documento | Oscar Gomez |
| 1.0 | 06/05/2019 | Edición de documento | Equipo Jbless |

**TABLA DE CONTENIDO**

INTRODUCCIÓN

1. Descripción del servicio Nómina
2. Descripción del Bean Nómina
3. Descripción de la fachada nómina
4. Tabla Nómina
5. Tabla Cargo
6. Tabla Concepto
7. Tabla Periodo
8. Tabla Hoja\_de\_Vida
9. Tabla Dependencia
10. Tabla Contrato
11. Persistencia
12. Servidor de aplicaciones
13. Módulo EJB

**INTRODUCCIÓN**

A continuación usted encontrará la documentación del módulo ejb del proyecto general *usta\_empresarial* con la cual se describe su funcionalidad en el proyecto. A su vez se explica cada una de las tablas del *schema nómina* de la base de datos *usta\_empresarial.*

Las tablas a definir son: *dependencia, cargo, hoja\_vida, contrato, periodo,concepto, nómina*.

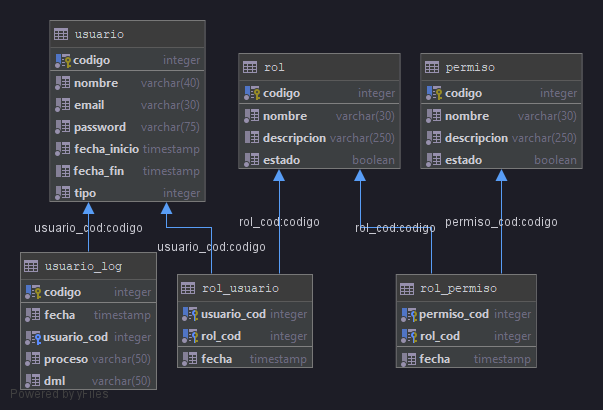
1. **Descripción del servicio nómina**

El servicio de seguridad está destinado a controlar el acceso al sistema mediante la autenticación de usuarios y el registro constante de las actividades que estos realizan en el sistema como la fechas de ingreso y qué acciones llevaron a cabo en este. Cada usuario tendrá un rol y el rol determinados permisos de acceso a funcionalidades.

1. **Descripción de la base de datos seguridad**

La base de datos seguridad consta de 5 tablas: usuario, rol, permiso, usuario\_log, rol\_usuario y rol\_permiso que ayudarán a controlar el acceso a las funcionalidades del sistema.

La siguiente imagen es el Modelo entidad-relacion de dicha base de datos.



*MER de la base de datos seguridad*

1. **Tabla Usuario**

En esta tabla se guardarán los registros de cada persona asociada a la empresa.

* 1. **Atributos**

**codigo:** Valor numérico entero(integer) auto\_increment que identifica a un usuario. Este campo no debe ser nulo.

**nombre:** Valor alfabético de 40 caracteres en el que especifica el nombre del usuario. Este campo no debe ser nulo.

**email:** Valor alfanumérico de 30 caracteres que corresponde a el correo con el que se registro el usuario. Este campo no debe ser nulo.

**password:** Valor alfanumérico de 75 caracteres que corresponde a la clave de inicio de sesión de cada email. Este campo no debe ser nulo.

**fecha\_inicio:** Valor tipo ‘timestamp’ que corresponde a la fecha y hora en la que se registró por primera vez el usuario. Su formato es “AAAA-MM-DD hh:mm:ss”. Este campo no debe ser nulo. Este valor está definido por defecto como la fecha en la que se realice el registro.

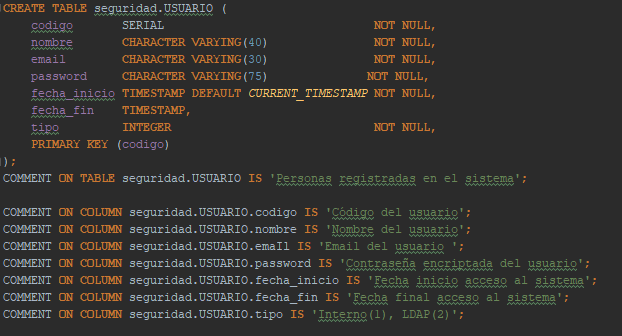
**fecha\_fin:** Valor tipo ‘timestamp’ que corresponde a la fecha y hora de el último acceso. Este campo no debe ser nulo. Su formato debe ser “AAAA-MM-DD hh:mm:ss”.

**tipo:** Valor numérico de un carácter que corresponde ala asignación de los usuarios internos(número 1) y los usuarios LDAP(número 2). Este campo no debe ser nulo.

* 1. **Índices**
     1. **Llave primaria**

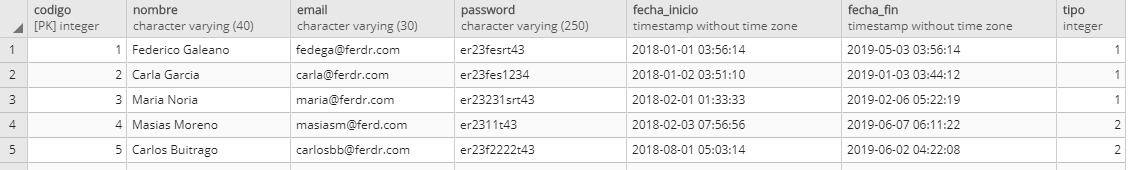
En esta tabla, la llave primaria es el código del usuario, ya que este es único e irrepetible debido a su carácter auto incremental.

* 1. **Sql**

****

**3.4 Ejemplos de registros**

A continuación, se muestran algunos registros de la tabla usuario

****

1. **Tabla rol**

En esta tabla se guardan los diferentes roles que existen en la empresa.

* 1. **Atributos**

**codigo:** Valor numérico entero mayor a cero auto\_increment e identificador único del rol. No puede ser nulo.

**nombre:** Valor alfabético (no se permiten números) de 30 caracteres como máximo. No puede ser nulo ni se acepta caracteres especiales ($, &, \*, ~, ° ). Hace referencia a un rol (administrador, secretaria(o), cliente, invitado) dentro de la empresa. Este campo no puede ser nulo.

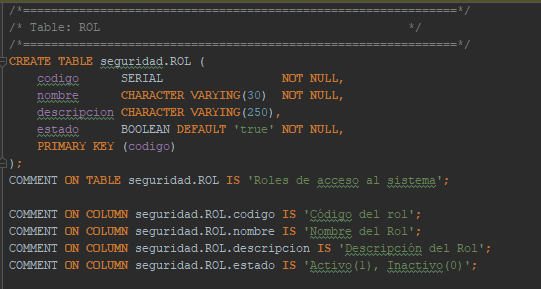
**descripción:** Valor alfanumérico máximo de 250 caracteres. descripción del rol, este campo puede ser nulo.

**estado:** Valor tipo ‘Boolean’ que retorna el estado true(1) o false(2) que hacen referencia a si el rol está activo o inactivo respectivamente. Este campo no debe ser nulo.

* 1. **Índices**
     1. **Llave primaria**

En esta tabla, la llave primaria es el código del rol, ya que este es único e irrepetible debido a su carácter auto incremental.

* 1. **Sql**

****

* 1. **Ejemplo de registro**

1. **Tabla permiso**

Tabla en la cual van todos los campos de permisos de acceso a módulos

* 1. **Atributos**

**codigo:**Valor numérico único auto-incrementable con el código del permiso.

**nombre:**Valor carácter con un tamaño máximo de 30 dígitos que indica el nombre del permiso (READ, DELETE,INSERT, DELETE) en su correspondiente módulo. Este campo no puede ser nulo.

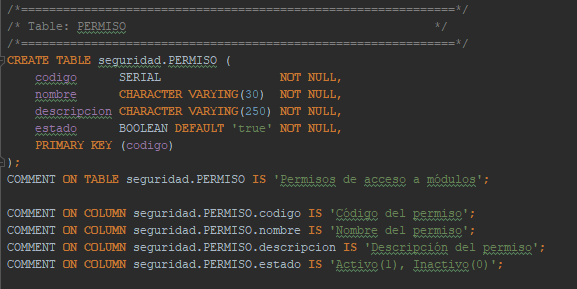
**descripción:** Valor carácter con un tamaño máximo de 250 dígitos que indica la descripción del permiso en el módulo especificado. Este campo no puede ser nulo.

**estado:**Valor tipo ‘boolean’ con el estado true(1) o false(0) nombrando si el permiso está activo(1) o inactivo(0). Este campo no puede ser nulo.

* 1. **Índices**
     1. **Llave primaria**

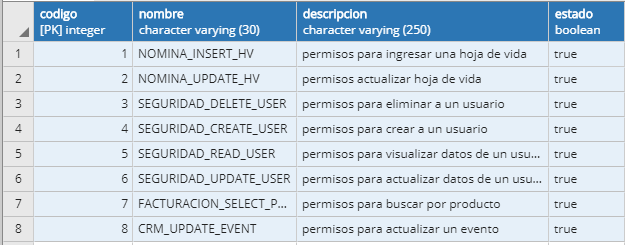
En esta tabla, la llave primaria, es el código del permiso, ya que este es único e irrepetible debido a su carácter auto incremental.

* 1. **Sql**

****

* 1. **Ejemplo de registro**

A continuación, se muestran algunos registros de la tabla permiso



1. **Tabla usuario\_log**

Tabla donde se almacena el historial de acciones de un usuario en el sistema.

* 1. **Atributos**

**codigo:** Valor numérico entero auto\_increment irrepetible que identifica cada registro. Este campo no debe ser nulo.

**fecha:** Valor tipo “timestam” el cual representa una fecha completa con formato “AAAA-MM-DD hh:mm:ss” que corresponde a la fecha en la que se realizó algún cambio. Este campo está definido por defecto por la fecha del sistema. No debe ser nulo.

**usuario\_cod:** Valor numérico entero que corresponde al código del usuario que realiza el cambio, este campo no debe ser nulo.

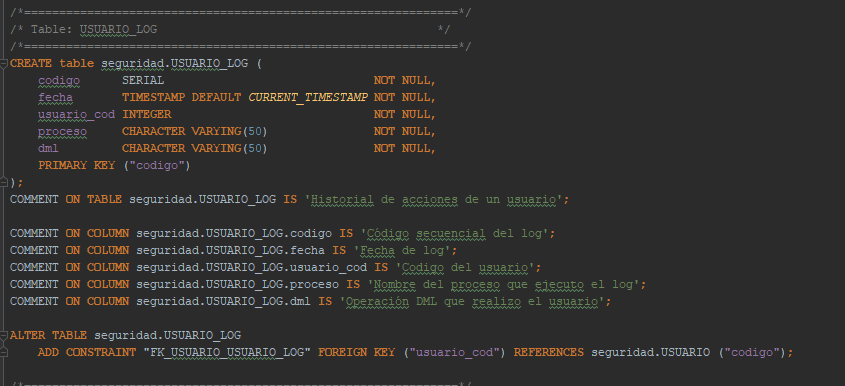
**proceso:** Valor alfanumérico con el cual se especifica que proceso ejecutó el log, este campo no debe ser nulo.

**dml:** Valor alfanumérico que especifica qué método se realizó “delete,update,insert”, este campo no debe ser nulo.

* 1. **Índices**
     1. **Llave primaria**

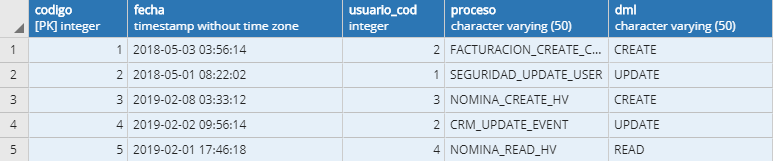
En esta tabla, la llave primari, es el código del usuario\_log, ya que este es único e irrepetible debido a su carácter auto incremental.

* 1. **Sql**

****

* 1. **Ejemplo de registros**

A continuación, se muestran algunos registros de la tabla usuario\_log

****

1. **Tabla rol\_usuario**

Tabla en la que se le asigna un rol a un usuario.

* 1. **Atributos**

**usuario\_cod:** Valor numérico entero mayor a cero heredado de la tabla *usuario* el cual representa a un usuario existente en el sistema.Este campo no debe ser nulo.

**rol\_cod:** Valor numérico entero mayor a cero heredado de la tabla *rol* el cual representa un rol existente en el sistema*.* Este campo no debe ser nulo.

**fecha:** Fecha tipo ‘timestamp’ en la que se realizó la asociación del rol con el usuario. La fecha tiene el formato “AAAA-MM-DD hh:mm:ss”. Este campo no debe ser nulo.

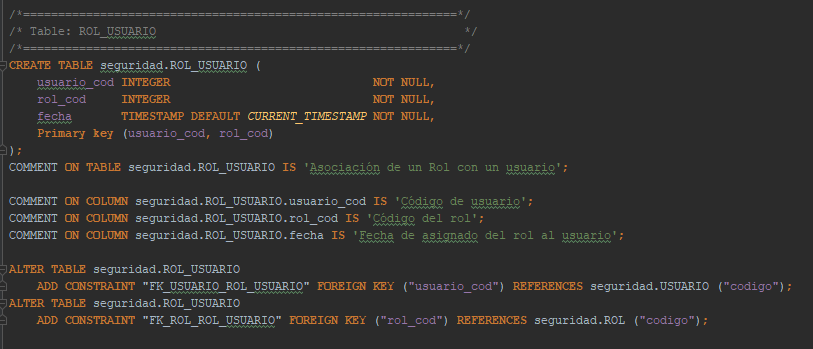
* 1. **Índices**
     1. **Llave primaria**

Llave compuesta por dos llaves foráneas: **“usuario\_cod”** y **“rol\_cod”**. Es compuesta debido a que un usuario solo puede tener un rol.

* + 1. **Llaves foráneas**

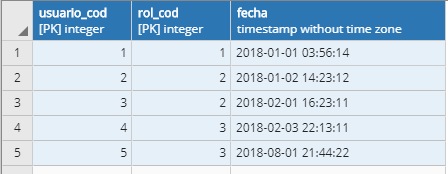
Esta tabla cuenta con dos llaves foráneas-

* **FK\_USUARIO\_ROL\_USUARIO** que asocia el campo *codigo (tabla usuario)* con *usuario\_cod(tabla rol\_usuario).*
* **FK\_ROL\_ROL\_USUARIO** que asocia el campo *codigo (tabla rol)* con *rol\_cod (tabla rol\_usuario)*.
  1. **Sql**

****

**7.3 Ejemplo de registros**

A continuación, se muestran algunos registros de la tabla rol\_usuario.



1. **Tabla rol\_permiso**

Tabla en la que se le asigna a un rol permisos de acceso a determinadas funcionalidades.

* 1. **Atributos**

**permiso\_cod:** Valor numérico entero mayor a cero, heredado de la tabla *permiso* el cual representa un permiso ya definido en el sistema*.* Este campo no debe ser nulo.

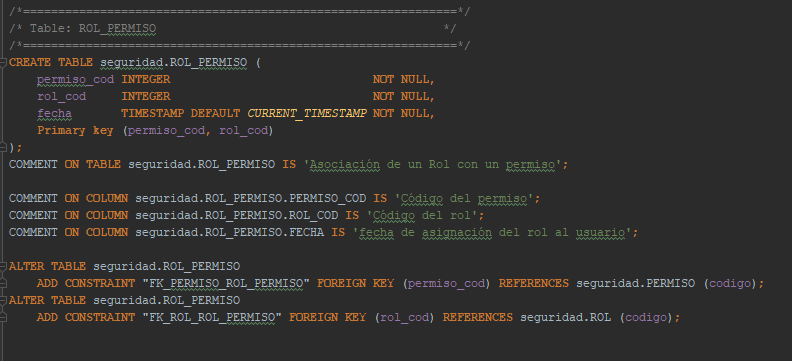
**rol\_cod:** Valor numérico entero mayor a cero, heredado de la tabla *rol* el cual representa un rol ya definido en el sistema*.* Este campo no debe ser nulo.

**fecha:** Fecha tipo ‘timestamp’ en la que se realizó la asociación del rol con el permiso. La fecha tiene el formato “AAAA-MM-DD hh:mm:ss” y está definida por defecto con la hora del sistema.

* 1. **Índices**
     1. **Llave primaria**

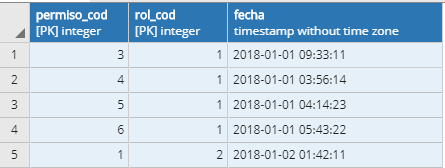
Llave compuesta por dos llaves foráneas: **“permiso\_cod”** y **“rol\_cod”**. Es compuesta debido a ...

* + 1. **Llaves foráneas**
  1. **Sql**

****

* 1. **Ejemplo de registros**

A continuación, se muestran algunos registros de la tabla rol\_permiso



1. **Persistencia**

El *persistence.xml* definido en el proyecto maven -> *SeguridadModelo* en el que se usa Hibernate.

En este archivo *persistence.xml* se define el *persistence-unit* como *SeguridadModelo*.

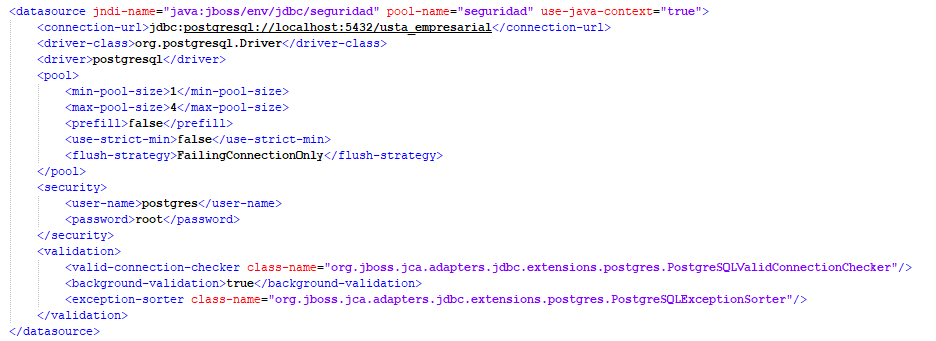
Se define el *jta-data-source como java:jboss/env/jdbc/seguridad* .

Se definen el mapeo de todas la entidades definidas anteriormente.

****

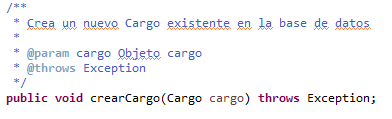
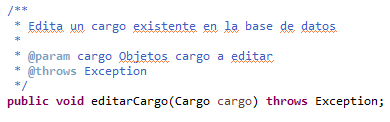
1. **Servidor de aplicación**
   1. **DataSource - Archivo Standalone.xml**

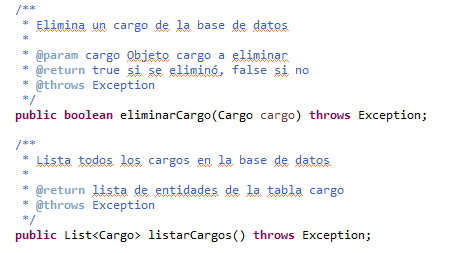
En el archivo *standalone.xml* del servidor *wilfly*, en el elemento *datasourse* se define el *jndi-name* como  *java:jboss/env/jdbc/seguridad* que se conecta al motor de Postgres y a la base de datos usta\_empresarial por medio del usuario *postgres* cuya clave es root. Además, este cuenta con un Pool de 1 a 4 conexiones.



1. **Proyecto EJB**
   1. **Hoja de Vida**
   2. **Cargo**
      1. **Cargo Fachada**

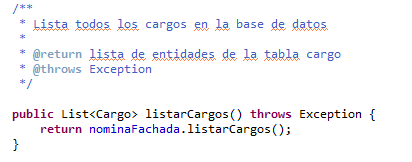
En este proyecto se llaman todos los métodos que se realizaron en el proyecto beam de la tabla Cargo

****

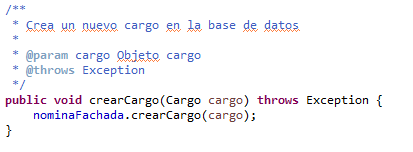
****

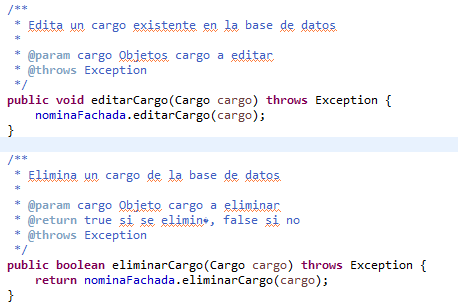
**11.1.2 Cargo servicio**

Este Servicio se encarga de conectar los métodos que se encuentran en el proyecto Bean al proyecto Web

****

Retorna la lista de los cargos que se encuentran guardados en la base de datos



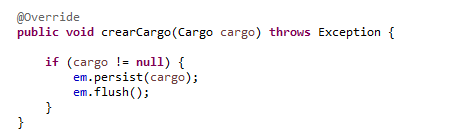


Retorna la eliminación de un cargo existente de la base de datos

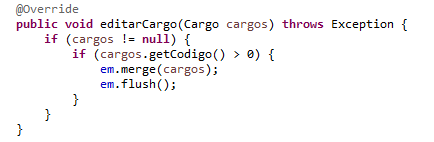
**11.1.3 Bean Cargo**

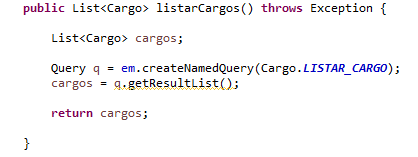
En este proyecto se implementa el crud de la tabla Cargo es decir:

* Crear Cargo
* Editar Cargo
* Listar Cargo
* Borrar Cargo

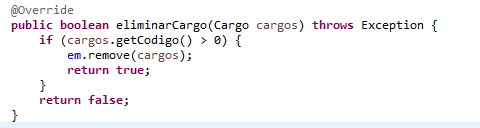


En este método se crea un nuevo cargo que debe ser diferente a null



****

En este método lista todos los cargos de la base de datos por medio de la consulta LISTAR\_CARGOS

****

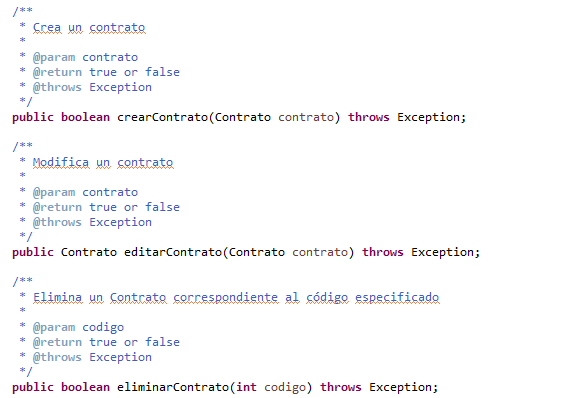
En este método borra el cargo por medio del código el cual debe ser mayor a 0

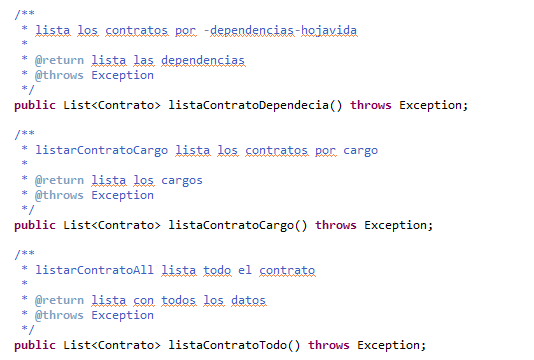
* 1. **Concepto**
  2. **Periodo**

**12.CONTRATO**

**12.1 fachada**

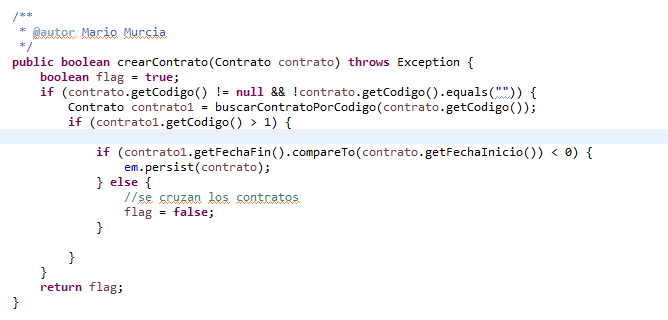
**se especifican los métodos con su documentación**



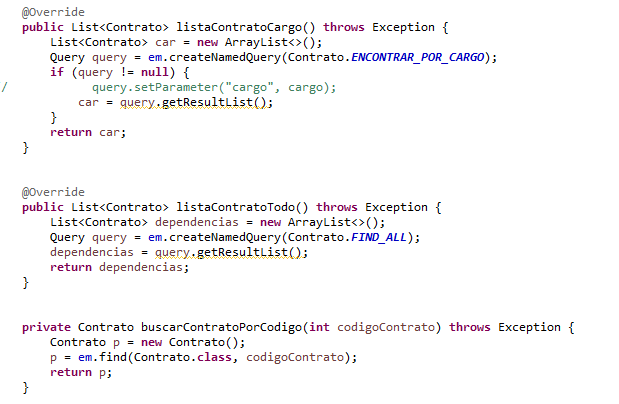


**12.2 beam**

1. se muestra lo que hace cada método
   1. crear contrato:crea un contrato con algunas especificaciones que son las reglas de negocio
   2. editar contrato:edita parámetros de los contratos
   3. eliminar contrato:deshabilita un contrato
   4. listar por dependencias: lista los contratos con sus dependencias
   5. listar contratos cargo: lista los contrató por sus cargos
   6. listar contrato all : lista todos los parámetros del contrato

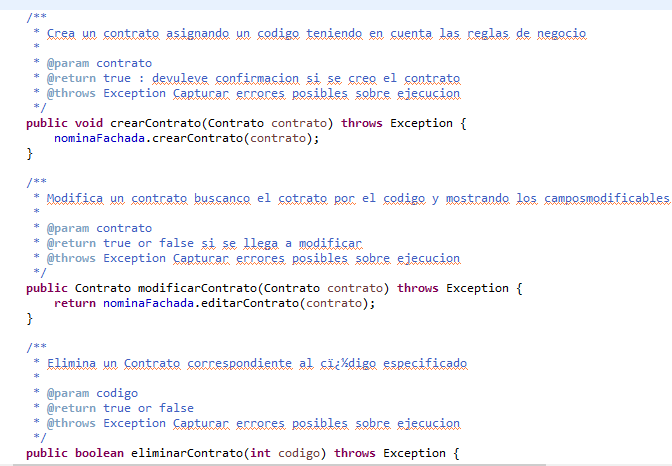
****

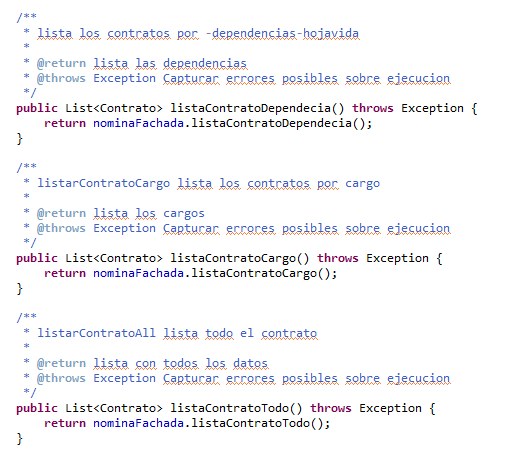
****

****

**12.3 servicio**

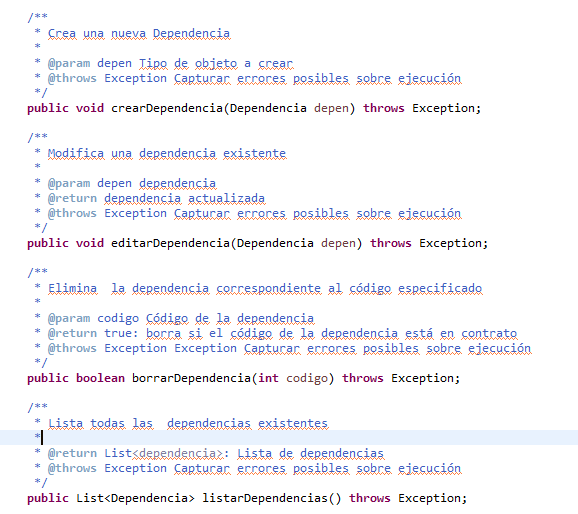
Este Servicio se encarga de conectar los métodos que se encuentran en el proyecto Bean al proyecto Web

****

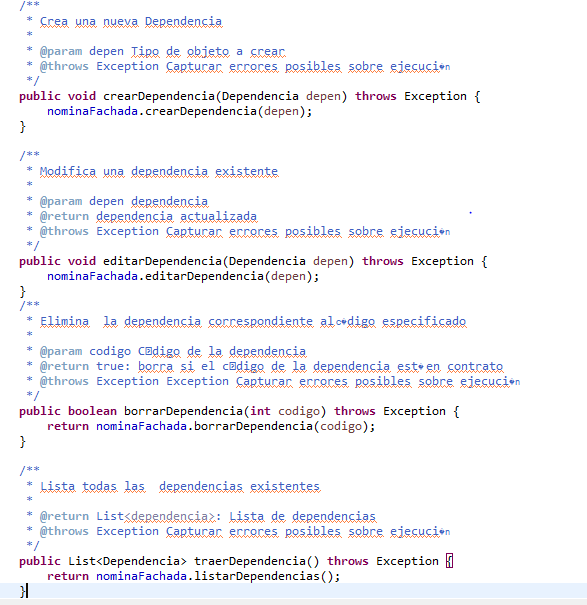
****

**12.4 Dependencia**

**12.4.1 Dependencia Fachada**

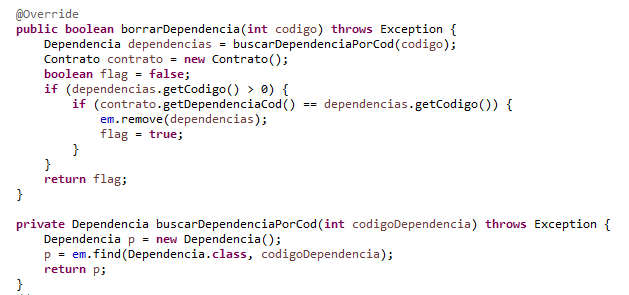
****

**12.4.2 Dependencia Servicio**

****

**12.4.3 Dependencia Bean**

****

****