

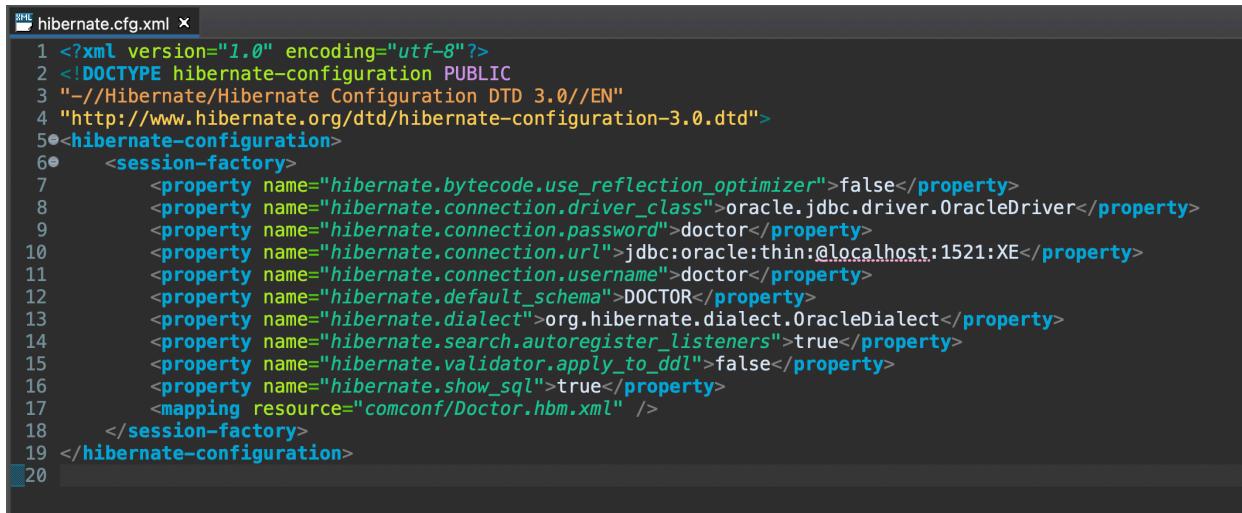
Alumno: Alejandro Herrera Mendoza

Descripción: Sistema web para registrar la información de un doctor con el fin de acceder con sus credenciales y mostrar su información registrada.

Tecnologías: JSF, Hibernate, Oracle 11g, primefaces.

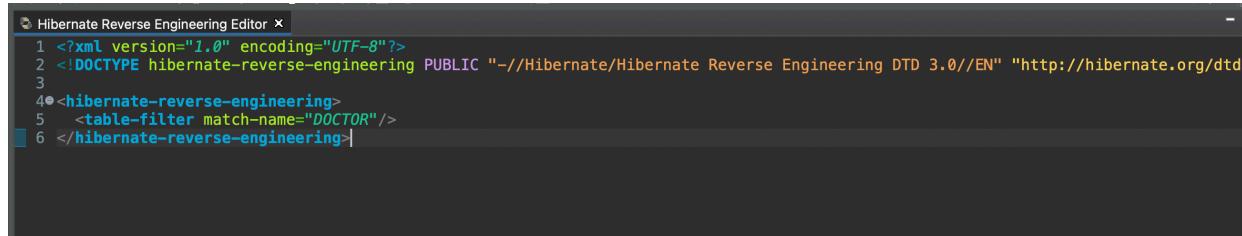
Código

Archivo hibernate.cfg.xml para tener la configuración necesaria para realizar la conexión al gestor de base de datos Oracle Database 11g.



```
hibernate.cfg.xml
1 <?xml version="1.0" encoding="utf-8"?>
2 <!DOCTYPE hibernate-configuration PUBLIC
3 "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4 "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5<.hibernate-configuration>
6  <session-factory>
7    <property name="hibernate.bytecode.use_reflection_optimizer">false</property>
8    <property name="hibernate.connection.driver_class">oracle.jdbc.driver.OracleDriver</property>
9    <property name="hibernate.connection.password">doctor</property>
10   <property name="hibernate.connection.url">jdbc:oracle:thin:@localhost:1521:XE</property>
11   <property name="hibernate.connection.username">doctor</property>
12   <property name="hibernate.default_schema">DOCTOR</property>
13   <property name="hibernate.dialect">org.hibernate.dialect.OracleDialect</property>
14   <property name="hibernate.search.autoregister_listeners">true</property>
15   <property name="hibernate.validator.apply_to_ddl">false</property>
16   <property name="hibernate.show_sql">true</property>
17   <mapping resource="comconf/Doctor.hbm.xml" />
18 </session-factory>
19 </hibernate-configuration>
20
```

Archivo hibernate.reveng.xml generado de forma automática después de haber mapeado la base de datos.



```
Hibernate Reverse Engineering Editor
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE hibernate-reverse-engineering PUBLIC "-//Hibernate/Hibernate Reverse Engineering DTD 3.0//EN" "http://hibernate.org/dtd/
3
4<hibernate-reverse-engineering>
5  <table-filter match-name="DOCTOR"/>
6 </hibernate-reverse-engineering|
```

Archivo Doctor.hbm.xml generado después de marcar la tabla con hibernate con los atributos de la misma.

La propiedad generator class="increment" en el campo id de la tabla se agregó para que hibernate reconociera que id se generará implícitamente.

```
Doctor.hbm.xml x
1 <?xml version="1.0"?>
2 <!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
3 "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
4 <!-- Generated Dec 29, 2020, 6:24:39 PM by Hibernate Tools 3.5.0.Final -->
5<hibernate-mapping>
6  <class name="comconf.Doctor" table="DOCTOR">
7    <id name="idDoctor" type="long">
8      <column name="ID_DOCTOR" precision="10" scale="0" />
9      <generator class="increment" />
10   </id>
11   <property name="nombre" type="string">
12     <column name="NOMBRE" length="100" />
13   </property>
14   <property name="apellidos" type="string">
15     <column name="APELLODOS" length="250" />
16   </property>
17   <property name="direccion" type="string">
18     <column name="DIRECCION" length="250" />
19   </property>
20   <property name="telefono" type="string">
21     <column name="TELEFONO" length="250" />
22   </property>
23   <property name="rfc" type="string">
24     <column name="RFC" length="50" />
25   </property>
26   <property name="especialidad" type="string">
27     <column name="ESPECIALIDAD" length="100" />
28   </property>
29   <property name="sexo" type="string">
30     <column name="SEXO" length="50" />
31   </property>
32   <property name="correo" type="string">
33     <column name="CORREO" length="100" />
34   </property>
35   <property name="contra" type="string">
36     <column name="CONTRA" length="50" />
37   </property>
38 </class>
39 </hibernate-mapping>
40
```

RegistroMB.java en donde se obtienen y envían los valores al servicio de registro.

```
RegistroMB.java x
77     this.sexo = sexo;
78 }
79
80● public String getCorreo() {
81     return correo;
82 }
83
84● public void setCorreo(String correo) {
85     this.correo = correo;
86 }
87
88● public String getContra() {
89     return contra;
90 }
91
92● public void setContra(String contra) {
93     this.contra = contra;
94 }
95
96● public String registrarDoctor() {
97
98     RegistroService service = new RegistroDao();
99     if (service.createdoctor(
100         new Doctor(nombre, apellidos, direccion, telefono, rfc, especialidad, sexo, correo, contra))) {
101         return "registrado";
102     } else {
103         return "registro";
104     }
105 }
106 }
107
108 }
109
```

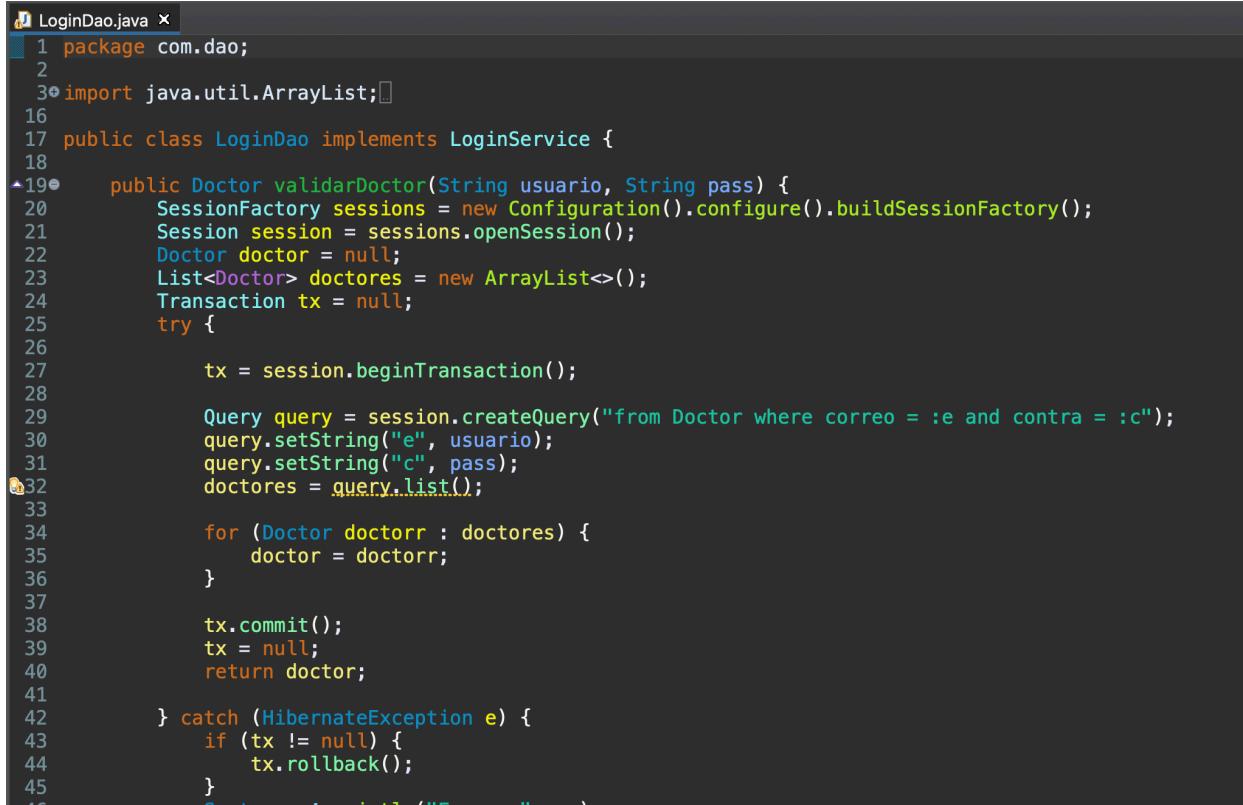
Archivo RegistroDao.java para registrar la información del doctor en el método createdoctor.

```
RegistroDao.java x
46     }
47
48● @Override
49 public boolean createdoctor(Doctor doctor) {
50
51     SessionFactory sessions = new Configuration().configure().buildSessionFactory();
52     Session session = sessions.openSession();
53
54     Transaction tx = null;
55     try {
56
57         tx = session.beginTransaction();
58
59         session.save(doctor);
60
61         tx.commit();
62         tx = null;
63         return true;
64     } catch (HibernateException e) {
65         if (tx != null) {
66             tx.rollback();
67         }
68
69         System.out.println("Error: " + e);
70         return false;
71     } finally {
72         session.close();
73     }
74
75 }
76
77 }
```

LoginMB.java para obtener las credenciales del doctor y mandar al servicio para ser validado.

```
J LoginMB.java x
41         return doctor;
42     }
43
44●    public void setDoctor(Doctor doctor) {
45        this.doctor = doctor;
46    }
47
48●    public String getMensaje() {
49        return mensaje;
50    }
51
52●    public void setMensaje(String mensaje) {
53        this.mensaje = mensaje;
54    }
55
56●    public String getFecha() {
57        Date fecha = new Date();
58        return fecha.toString();
59    }
60
61●    public String validarUsuario() {
62        LoginService service = new LoginDao();
63        doctor = service.validarDoctor(usuario, contra);
64        if (doctor != null) {
65            return "inicioMenu";
66        } else {
67            mensaje = "Credenciales incorrectas";
68            return "login";
69        }
70    }
71
72 }
73 }
```

LoginDao.java en dónde se realiza una consulta para saber si existe un usuario con esas credenciales.



The screenshot shows a code editor window with a dark theme. The title bar says "LoginDao.java x". The code is written in Java and implements a LoginService interface. It uses Hibernate to query a Doctor object from a database based on email and password. If found, it returns the doctor; otherwise, it returns null. Transaction management is handled using beginTransaction(), commit(), and rollback() methods.

```
1 package com.dao;
2
3 import java.util.ArrayList;
4
5 public class LoginDao implements LoginService {
6
7     public Doctor validarDoctor(String usuario, String pass) {
8         SessionFactory sessions = new Configuration().configure().buildSessionFactory();
9         Session session = sessions.openSession();
10        Doctor doctor = null;
11        List<Doctor> doctores = new ArrayList<>();
12        Transaction tx = null;
13        try {
14
15            tx = session.beginTransaction();
16
17            Query query = session.createQuery("from Doctor where correo = :e and contra = :c");
18            query.setString("e", usuario);
19            query.setString("c", pass);
20            doctores = query.list();
21
22            for (Doctor doctorr : doctores) {
23                doctor = doctorr;
24            }
25
26            tx.commit();
27            tx = null;
28            return doctor;
29        } catch (HibernateException e) {
30            if (tx != null) {
31                tx.rollback();
32            }
33        }
34    }
35}
```

Resultado

```
CREATE TABLE doctor (
    id_doctor      NUMBER(10),
    nombre         VARCHAR(100),
    apellidos     VARCHAR(250),
    direccion      VARCHAR(250),
    telefono       VARCHAR(250),
    rfc            VARCHAR(50),
    especialidad   VARCHAR(100),
    sexo           VARCHAR(50),
    correo          VARCHAR(100),
    contra          VARCHAR(50),
    CONSTRAINT doctor_pk PRIMARY KEY ( id_doctor )
);
```

The screenshot shows a web application interface for registering a new doctor. The URL in the address bar is `localhost:8080/HibernateOracleDoctor/faces/registro.xhtml?productId=`. The page title is "Registro". The form fields are as follows:

Field	Value
Nombre	Sandra
Apellidos	Herrera Mendoza
Direccion	privada 16 de septiemb
Telefono	24954775
Especialidad	Maestra
Rfc	SDANDH655
Sexo:	<input type="radio"/> Hombre <input checked="" type="radio"/> Mujer
Correo	sandy@gmail.com
Contraseña	sandy

At the bottom right of the form is a blue "Registrar" button.

SQL | All Rows Fetched: 13 in 0.005 seconds

ID_DOCTOR	NOMBRE	APELLIDOS	DIRECCION	TELEFONO	RFC	ESPECIALIDAD	SEXO	CORREO	CONTRA
1	1 Alejandro	Herrera Mendoza	Privada 16 de septiembre	24937474	HEMAHDHF76	Java	Hombre	alex@gmail.com	pass
2	2 Maria	Rosas	8 norte	249736474	RJFJ76	(null)	(null)	maria@gmail.com	password
3	3 Marco	Mendoza	8 sur este	24916745474	MARDGF765	(null)	Hombre	marco@gmail.com	marco
4	4 Martha	Sanchez	av 5 sur	249234657	MDRF655	(null)	Mujer	martha@gmail.com	martha
5	5 hkg	hgjg	hgj	hgjg	Hgj	(null)	Hombre	ghk@gmail.com	Hgjkh
6	6 ghjg	hgjg	hgj	hgk	Hgk	(null)	Mujer	ghk@gmail.com	Hgjk
7	7 Pedro	Rosas	av 5 sur	2494757	HDGG66	java	Hombre	pedro@gmail.com	pedro
8	21 Alejandro	Herrera Mendoza	privada 16 de septiembre	24915637	HEMAHG76	java	Hombre	alex@gmail.com	password
9	22 Alejandro	Herrera Mendoza	privada 16 de septiembre	24915637	HEMAHG76	java	Hombre	alex@gmail.com	password
10	41 Manuel	Mendoza Pérez	av 6 sur	249264747	HDMDG755	Veterinario	Hombre	manuel@gmail.com	manuel
11	42 Alejandro	rohum Hermen	3 sur	3242345	HDGFH6	Java	Hombre	a@gmail.com	pass
12	43 Raul	Mendez	Privada 16 norte	2484757	Doh	Doctor	Hombre	raul@gmail.com	raul
13	44 Sandra	Herrera Mendoza	privada 16 de septiembre	24954775	SDANDH655	Maestra	Mujer	sandy@gmail.com	sandy

localhost:8080/HibernateOracleDoctor/faces/login.xhtml

Login

Username	<input type="text" value="sandy@gmail.com"/>
Password	<input type="password"/>
Credenciales incorrectas	
<input type="button" value="Login"/>	
Registrar	

← → C ⌂ localhost:8080/HibernateOracleDoctor/faces/login.xhtml

Bienvenido Doctor(a):



Sandra Herrera Mendoza

Información

Dirección: privada 16 de septiembre

Especialidad: Maestra

Rfc: SDANDH655

Sexo: Mujer

Fecha: Wed Dec 30 13:50:15 CST 2020
