UNIVERSIDAD AUTONOMA DE NUEVO LEÓN

FACULTAD DE CIENCIAS FISICO MATEMATICAS

Lic. En Seguridad en Tecnologías en Información

Laboratorio 7

Bases de datos

Nombre del alumno: Guillermo Alejandro Camacho Martínez

Matricula: 1749496

Semestre: Segundo

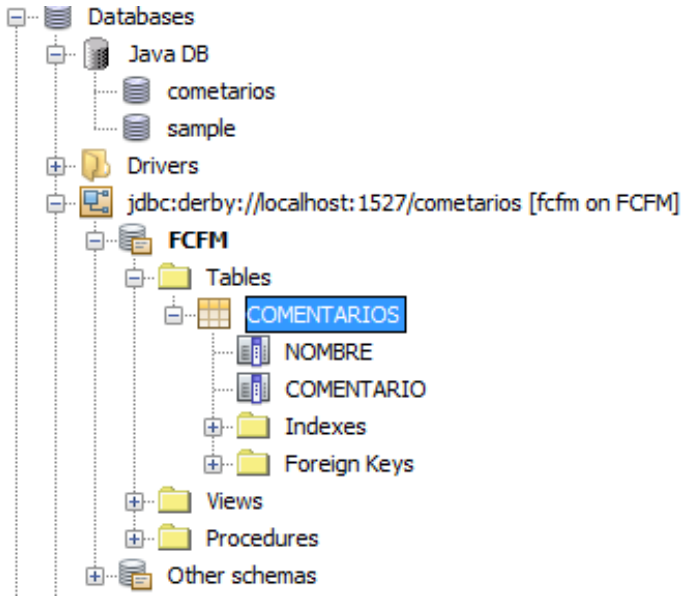
Grupo: 007

Materia: LDOO

San Nicolás, N.L, México a lunes 25 de Marzo del 2017

Actividad 1.- Crear una base de datos en javaDB.

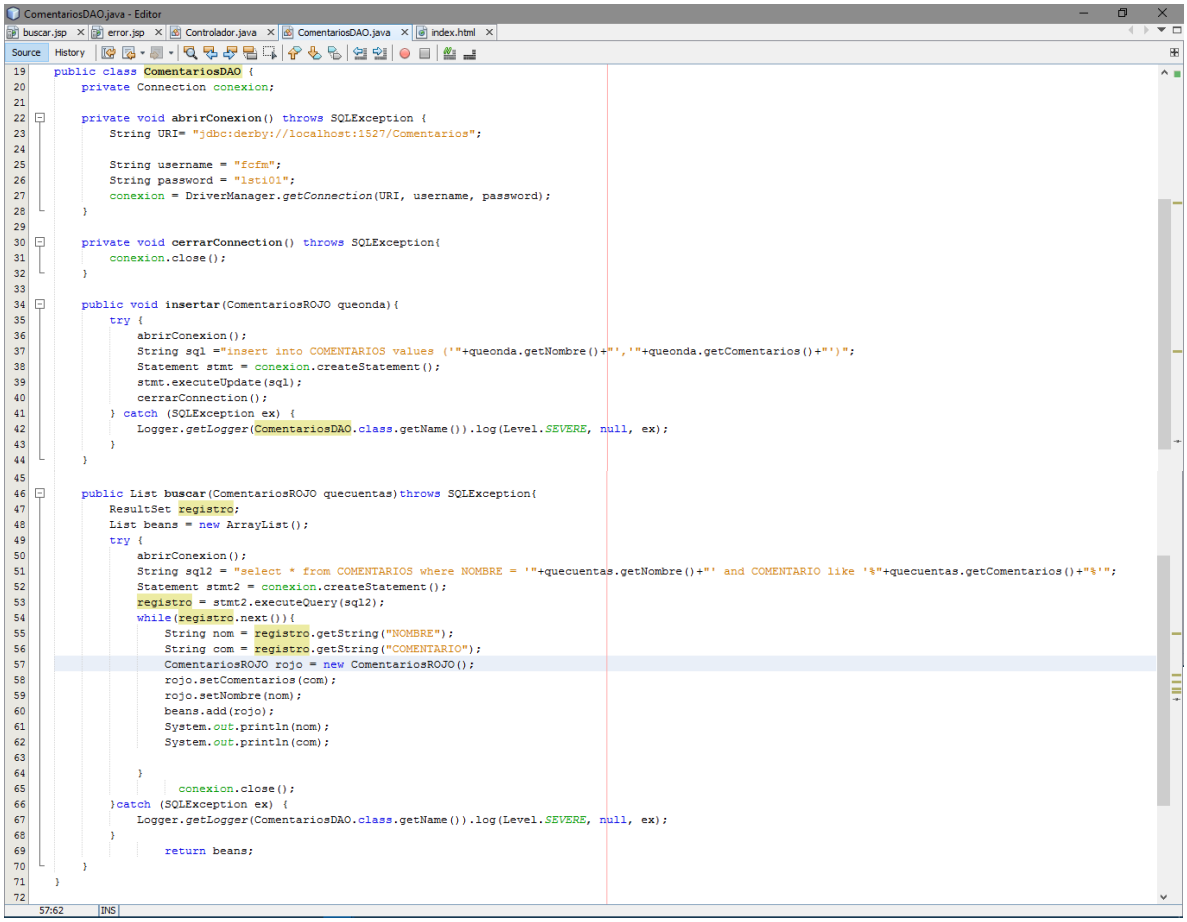
En la primera actividad, solo creamos una base de datos en el motor que tiene NetBeans, en la le asignaremos como nombre “Comentarios”, usuario “fcfm” y contraseña “lsti01”, después debemos de crear dos tablas llamada NOMBRE, de tipo VARCHAR, tamaño 50, no permite Null y otra llamada COMENTARIO, de tipo VARCHAR, tamaño 200, no permite Null.



Activiad 2.- Crear el modelo y una clase para la transerencia de datos.

Primero debemos de crear un nuevo paquete llamado modelo, en el cual se va a crear una nueva Java clase llamada ComentariosPOJO, en ella agregaremos 2 valiables de tipo String, nombre y comentarios y agregaremos los metodos set y get de ambos.

Ahora en este mismo paquete creamos una clase llamada ComentariosDAO, en la que tendemos 4 metodos, el priemro es un metodo privado para abrir la conexon, el segundo es para cerrar la conecxion, un tercer metodo que sea puglico llamado insertar que reciba como obejto de tipo comentarioPOJO y el ultimo para crear un objeto de tipo List, para tomar los datos en un formulario.




```java
public class ComentariosDAO {
    private Connection conexion;

    private void abrirConexion() throws SQLException {
        String URI= "jdbc:derby://localhost:1527/Comentarios";

        String username = "fcfm";
        String password = "lsti01";
        conexion = DriverManager.getConnection(URI, username, password);
    }

    private void cerrarConnection() throws SQLException{
        conexion.close();
    }

    public void insertar(ComentariosROJO queonda){
        try {
            abrirConexion();
            String sql ="insert into COMENTARIOS values ('"+queonda.getNombre()+"','"+queonda.getComentarios()+"')";
            Statement stmt = conexion.createStatement();
            stmt.executeUpdate(sql);
            cerrarConnection();
        } catch (SQLException ex) {
            Logger.getLogger(ComentariosDAO.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

    public List buscar(ComentariosROJO quecuentas)throws SQLException{
        ResultSet registro;
        List beans = new ArrayList();
        try {
            abrirConexion();
            String sql2 = "select * from COMENTARIOS where NOMBRE = '"+quecuentas.getNombre()+"' and COMENTARIO like '%"+quecuentas.getComentarios()+"%'";
            Statement stmt2 = conexion.createStatement();
            registro = stmt2.executeQuery(sql2);
            while(registro.next()){
                String nom = registro.getString("NOMBRE");
                String com = registro.getString("COMENTARIO");
                ComentariosROJO rojo = new ComentariosROJO();
                rojo.setComentarios(com);
                rojo.setNombre(nom);
                beans.add(rojo);
                System.out.println(nom);
                System.out.println(com);

            }
                conexion.close();
        }catch (SQLException ex) {
            Logger.getLogger(ComentariosDAO.class.getName()).log(Level.SEVERE, null, ex);
        }
            return beans;
    }
}
```

Actividad 3.- Crear las paginas HTML/JSP para la vista.

En la pagina html, que viene por defecto debemos de crear un formulario que envie datos por post y que conecte a nuestro controlador, lo que se debe de incluir en el formulario es lo siguiente:

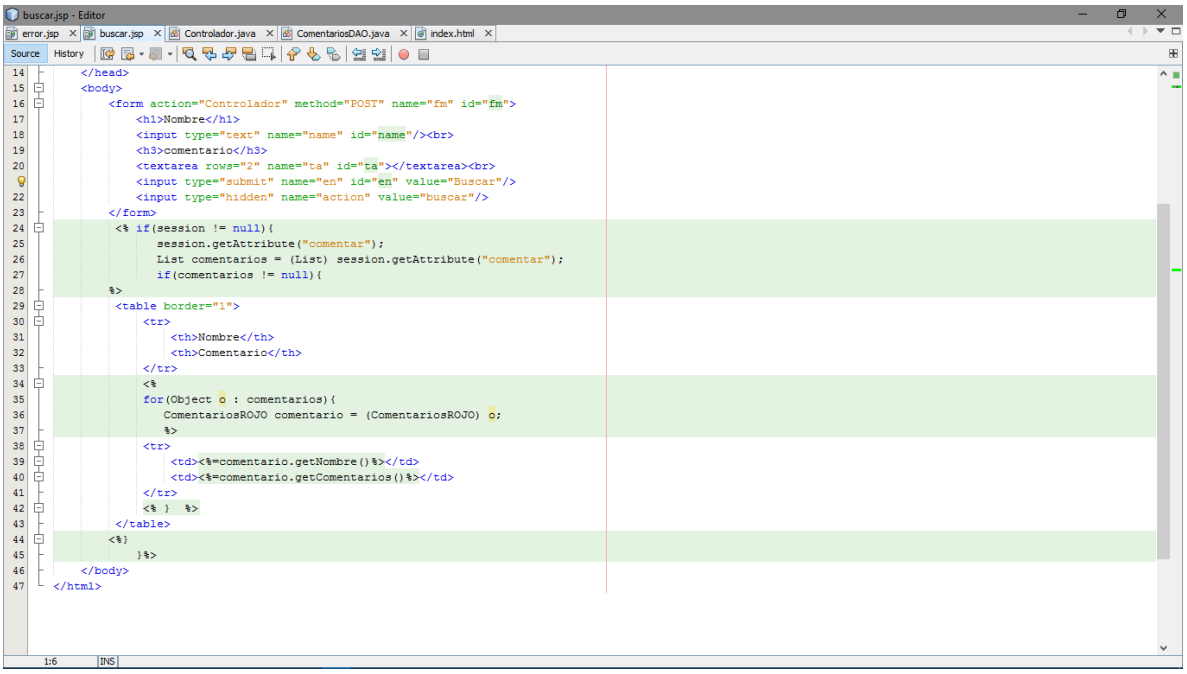
Un campo de texto para enviar el nombre.

ii. Un área de texto para enviar los comentarios.

iii. Un botón de submit.

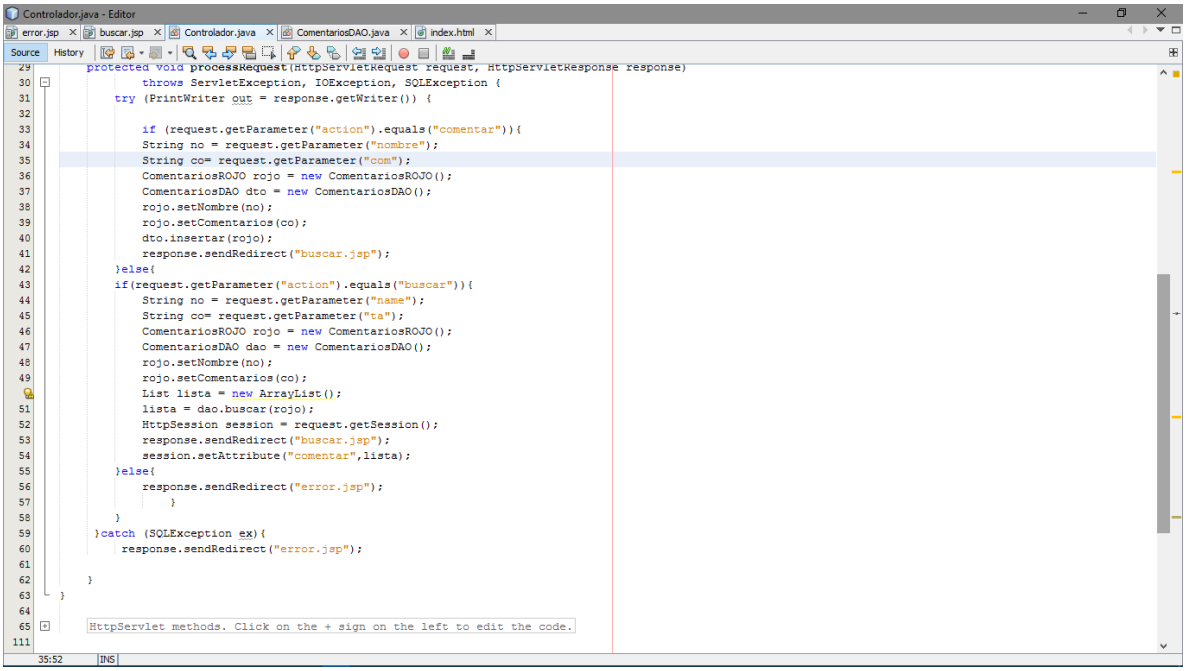
iv. Un campo oculto (**hidden**) de nombre “**accion**” con el valor “**comentar**”. Este campo oculto lo usará el controlador para determinar desde qué página lo están invocando.

Después, para consultar los comentarios, debemos de crear una página llamada buscar.jsp, en la cual tiene que incluir lo mismo que la anterior pero ahora agregaremos una tabla que mostrara los resultados en orden de esta manera:



Actividad 4.-Cear el controlador

La funcion del controlador es determinar la accion que se ente invocando en la pagina, por ejemplo si estas leyendo “action” que viene un caompo oculto en las paginas web. Si la accion viene de comentar se insertara en la base de datos y si no viene de ahí, y viene de buscar de invoca al mdelo para que busque los registros.



Preguntas de Reflexión

1. ¿Cuál piensas que es el propósito de haber hecho una clase DAO en el modelo en lugar de acceder a la base de datos directamente desde el controlador?

Pienso que eso la hace más reutilizable

2. ¿Para qué sirve un objeto POJO o JavaBean?

Para el encapsulamiento de los datos, ya que se hace uso del objeto en varios lugares

3. En caso de que los comentarios fueran muchos (digamos, cientos o miles) sería impráctico mostrarlos todos en una misma página. Generalmente los sitios de búsqueda (como Google) usan una técnica llamada “paginación”, para ir mostrando solo cierta cantidad de registros cada vez. Describe cómo harías esa paginación en esta aplicación (cuál es la lógica que seguirías en el programa).

Pues lo que teníamos que hacer es mostrar una cantidad que queramos, me imagino algo como un for en donde pongamos hasta que cantidad queramos que se vea y si sobrepasa esa cantidad puedas agregar algo así como una flecha para ver la otra página con los restantes

4. Cuando se muestra la tabla con los resultados de la búsqueda, desaparecen los valores de los campos de búsqueda. ¿Qué harías para que se sigan mostrando?

Hacer que te los mande

5. Haz una búsqueda, pero ahora, en lugar de escribir un nombre, escribe lo siguiente en el campo de búsqueda de nombre (la comilla inicial es importante, y también los dos guiones al final):

```
' or 1=1 --
```

¿Cuál fue el resultado de la búsqueda?

Pues, no sé si hice algo mal, lo copie al cual y no me aparece nada

6. A lo que hiciste en la pregunta anterior se le conoce como *SQL Injection* (**SQLi**), y es una de las vulnerabilidades más explotadas en las aplicaciones Web. De acuerdo a la cadena de búsqueda y a los resultados obtenidos, explica qué fue lo que ocurrió.

Si no mal recuerdo, siempre daba respuestas correctas

7. ¿Cómo piensas que puede evitarse un SQL injection como el de la pregunta 4? (A estas alturas del curso no se vale responder "*no sé*" a una pregunta así).

Esto se puede evitar validando campos, lo que se tiene que validar es que no haya campos con algún tipo de símbolo, por ejemplo, en nombre sabemos que ningún nombre lleva "<" o "-"

8. Elabora un diagrama donde muestres todos los elementos que construiste en esta práctica y cómo están relacionados entre ellos.