2024

GEOPAXIL

PROYECTO HIBERNATE

MIGUEL RICARDO SILVESTRE BERNARDINO Y DAVID PAJARES PARDAL

Contenido

1. Introducción:

Este proyecto consiste en una aplicación de gestión de información relacionada con comunidades autónomas, pueblos y ciudades en Java. El propósito principal es proporcionar una interfaz gráfica intuitiva para que los usuarios puedan ver, agregar, eliminar y modificar datos en una base de datos.

La tecnología utilizada incluye Java para el desarrollo de la aplicación, Hibernate para el mapeo objeto-relacional y la gestión de la base de datos, y MySQL como sistema de gestión de base de datos.

2. Objetivos del Proyecto:

a) Proporcionar una interfaz gráfica intuitiva: Se busca crear una interfaz de usuario clara que permita a los usuarios navegar fácilmente por la aplicación y realizar acciones como agregar, eliminar y modificar registros.
b) Integración de Hibernate y MySQL: Se pretende utilizar Hibernate para el mapeo objeto-relacional y la gestión de la base de datos MySQL. Esto implica establecer una conexión adecuada con la base de datos, definir las entidades de Java correspondientes y asegurar que las operaciones CRUD (crear, leer, actualizar, eliminar) se realicen correctamente.
c) Implementar funcionalidades de agregar, eliminar y modificar: Se busca proporcionar botones y acciones específicas en la interfaz de usuario que permitan a los usuarios realizar estas operaciones de manera intuitiva.
d) Mostrar datos en JList y text boxes: Se planea mostrar la información de la base de datos en componentes gráficos como JList y text boxes, lo que facilita la visualización y modificación de los registros.

Comenzamos estableciendo la estructura de la base de datos y definiendo las entidades de Hibernate. Luego, se desarrolló la interfaz de usuario utilizando WindowsBuilder, a la vez que se iba haciendo el código y por último juntamos el código con la interfaz.

3. Roles del equipo:

a) **Diseñador gráfico de la interfaz de usuario (David):** Se encargó de la creación y diseño de la interfaz de usuario de la aplicación. Se encargó de de la interfaz gráfica, el diseño de la disposición de los elementos visuales, la selección de colores, fuentes y estilos, y la creación de prototipos visuales para visualizar la estructura y funcionalidades de la aplicación. Utilizó WindowsBuilder para la creación de la interfaz gráfica. Ayudó también en la implementación de código en la interfaz.
b) **Desarrollador de código (Miguel):** Estuvo a cargo de la implementación de la lógica de la aplicación y la integración con la base de datos utilizando Hibernate y MySQL. Se encargó de la definición de las clases de entidad de Hibernate, la

escritura del código y la implementación de las operaciones CRUD (crear, leer, actualizar, eliminar).

4. Planificación de las tareas:

a. **Definición de requisitos y diseño de la base de datos:** Inicialmente, se definió la estructura de la base de datos y los requisitos del proyecto. Esto incluyó la identificación de las entidades y relaciones necesarias para almacenar la información sobre comunidades autónomas, pueblos y ciudades.

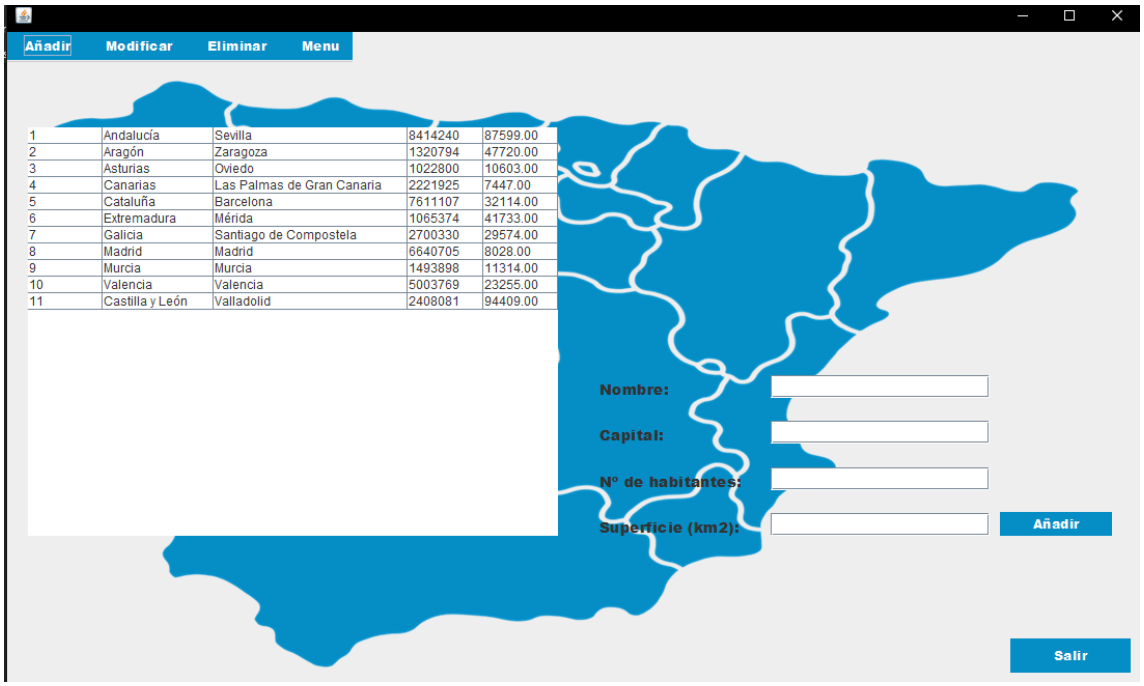
b. **Creación de la base de datos:** Una vez definida la estructura de la base de datos, se procedió a crearla utilizando MySQL. Esto implicó la creación de tablas, la definición de relaciones y la inserción de datos.

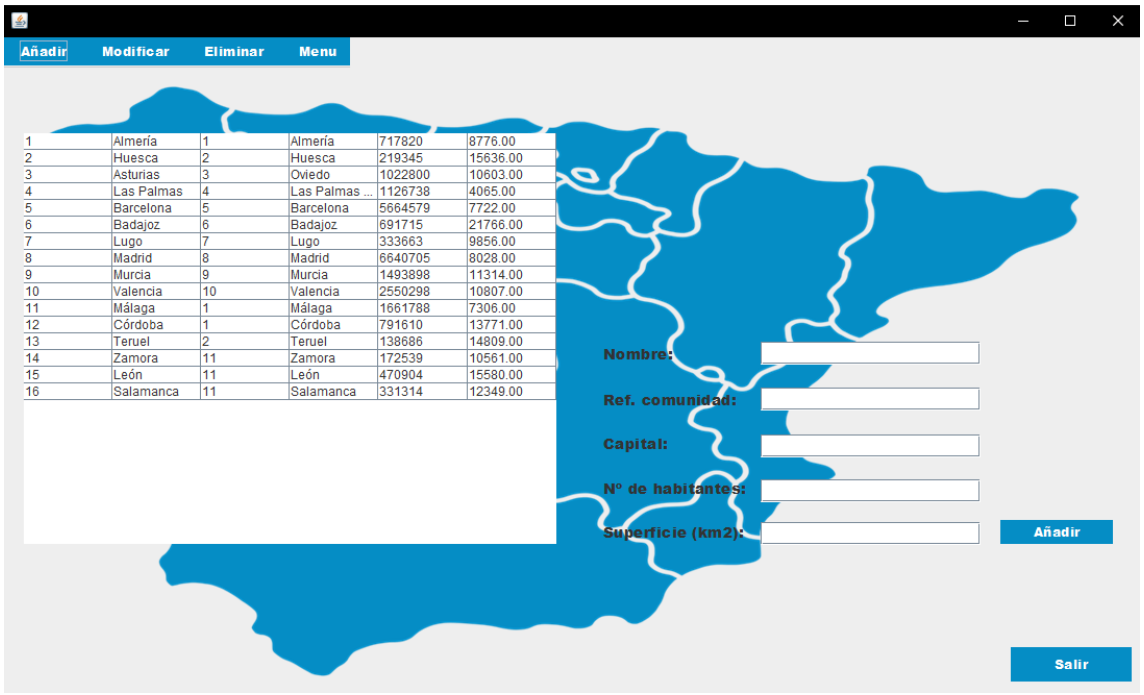
c. **Implementación de Hibernate:** Posteriormente, se realizó la implementación de Hibernate. Esto incluyó la configuración de Hibernate y la definición de las clases de entidad.

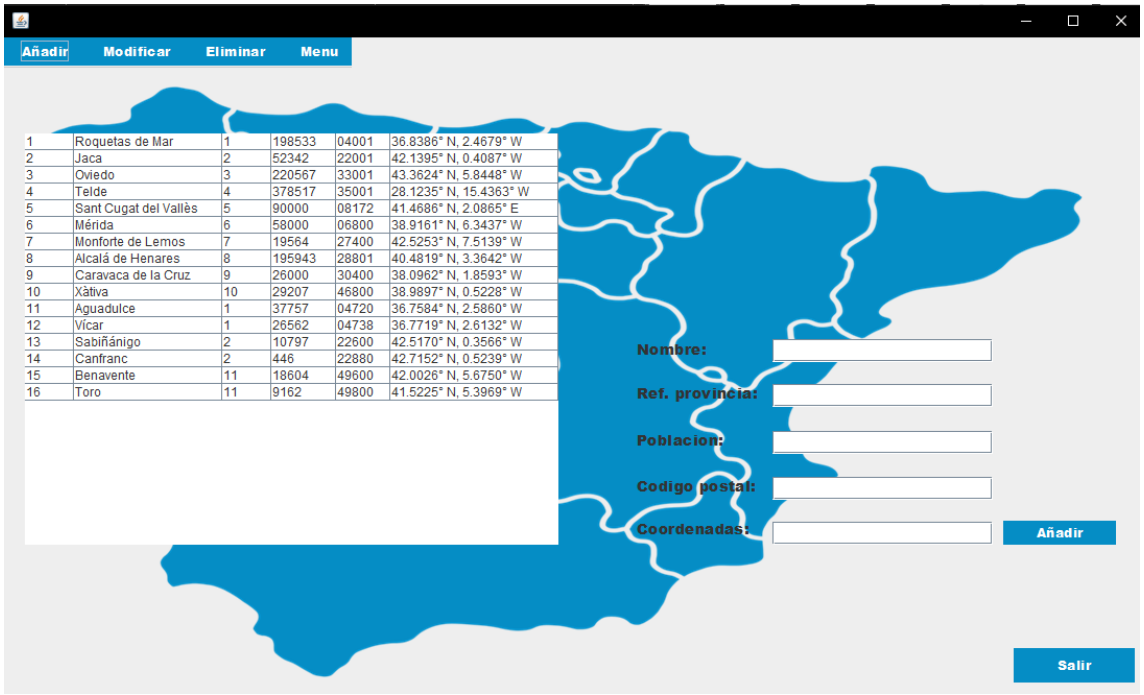
d. **Reparto de tareas y desarrollo paralelo:** Una vez completada la implementación de Hibernate, se llevó a cabo el reparto de tareas entre los dos miembros del equipo. Uno se centró en el diseño y desarrollo de la interfaz gráfica, mientras que el otro se ocupó del desarrollo del código y la lógica de la aplicación.

e. **Integración de la interfaz gráfica y el código:** Una vez finalizados ambos aspectos del proyecto, se procedió a integrar la interfaz gráfica diseñada con el código desarrollado. Esto implicó la conexión de los componentes visuales de la interfaz con la lógica de la aplicación, asegurando que la aplicación funcionara como se esperaba y que los datos se mostraran y actualizaran correctamente en la interfaz gráfica.

5. Exposición del proyecto:


Añadir
Modificar
Eliminar
Menu
1 Andalucía Sevilla 8414240 87599.00
2 Aragón Zaragoza 1320794 47720.00
3 Asturias Oviedo 1022800 10603.00
4 Canarias Las Palmas de Gran Canaria 2221925 7447.00
5 Cataluña Barcelona 7611107 32114.00
6 Extremadura Mérida 1065374 41733.00
7 Galicia Santiago de Compostela 2700330 29574.00
8 Madrid Madrid 6640705 8028.00
9 Murcia Murcia 1493898 11314.00
10 Valencia Valencia 5003769 23255.00
11 Castilla y León Valladolid 2408081 94409.00
Nombre:
Capital:
Nº de habitantes:
Superficie (km2):
Añadir
Salir


Añadir
Modificar
Eliminar
Menu
1 Almería 1 Almería 717820 8776.00
2 Huesca 2 Huesca 219345 15636.00
3 Asturias 3 Oviedo 1022800 10603.00
4 Las Palmas 4 Las Palmas ... 1126738 4065.00
5 Barcelona 5 Barcelona 5664579 7722.00
6 Badajoz 6 Badajoz 691715 21766.00
7 Lugo 7 Lugo 333663 9856.00
8 Madrid 8 Madrid 6640705 8028.00
9 Murcia 9 Murcia 1493898 11314.00
10 Valencia 10 Valencia 2550298 10807.00
11 Málaga 1 Málaga 1661788 7306.00
12 Córdoba 1 Córdoba 791610 13771.00
13 Teruel 2 Teruel 138686 14809.00
14 Zamora 11 Zamora 172539 10561.00
15 León 11 León 470904 15580.00
16 Salamanca 11 Salamanca 331314 12349.00
Nombre:
Ref. comunidad:
Capital:
Nº de habitantes:
Superficie (km2):
Añadir
Salir


Añadir
Modificar
Eliminar
Menu
1 Roquetas de Mar 1 198533 04001 36.8386° N, 2.4679° W
2 Jaca 2 52342 22001 42.1395° N, 0.4087° W
3 Oviedo 3 220567 33001 43.3624° N, 5.8448° W
4 Telde 4 378517 35001 28.1235° N, 15.4363° W
5 Sant Cugat del Vallès 5 90000 08172 41.4686° N, 2.0865° E
6 Mérida 6 58000 06800 38.9161° N, 6.3437° W
7 Monforte de Lemos 7 19564 27400 42.5253° N, 7.5139° W
8 Alcalá de Henares 8 195943 28801 40.4819° N, 3.3642° W
9 Caravaca de la Cruz 9 26000 30400 38.0962° N, 1.8593° W
10 Xàtiva 10 29207 46800 38.9897° N, 0.5228° W
11 Aguadulce 1 37757 04720 36.7584° N, 2.5860° W
12 Vícar 1 26562 04738 36.7719° N, 2.6132° W
13 Sabiñánigo 2 10797 22600 42.5170° N, 0.3566° W
14 Canfranc 2 446 22880 42.7152° N, 0.5239° W
15 Benavente 11 18604 49600 42.0026° N, 5.6750° W
16 Toro 11 9162 49800 41.5225° N, 5.3969° W
Nombre:
Ref. provincia:
Poblacion:
Codigo postal:
Coordenadas:
Añadir
Salir

6. Pruebas realizadas:

```java
package org.apache.maven.geopaxil;

import java.math.BigDecimal;

public class actualizar {
public static void main(String[] args) {
    int oldname = 0;
    String nombre = "";
    String capital = "";
    int poblacion = 0;;
    BigDecimal superficie = null;

    String nompro = "";
    String capitalpro = "";
    int comunidadpr = 0;
    int poblacionpro = 0;
    BigDecimal superficiepro = null;

    String nombrepu = "";
    Integer poblacionpu = 0;
    String codigoPostalpu = "";
    String coordenadasGeograficas= "";
    int numUpd;
    int numDel;
    try (Session s = HibernateUtil.getSessionFactory().openSession()) {
        s.getTransaction().begin();
        numUpd = s.createMutationQuery("UPDATE comunidadesautonomas SET nombre=:nombrec, capital=:capitalc, poblacion=:poblacionc, superficie_km2=:superficiec WHERE id=:oldName")
                .setParameter("oldName", oldname).setParameter("nombrec", nombre).setParameter("capitalc", capital).setParameter("poblacionc", poblacion).setParameter("superficiec", superficie)
                .executeUpdate();

        numUpd = s.createMutationQuery("UPDATE provincias SET nombre=:nombrepr, capital=:capitalpr, comunidad_id=:comunidadidpr, poblacion=:poblacionpr superficie_km2=:superficiepr WHERE id=:oldName")
                .setParameter("oldName", oldname).setParameter("nombrepr", nompro).setParameter("capitalpr", capital).setParameter("poblacionpr", poblacion).setParameter("comunidadidpr", comunidadpr).setParameter("su
                .executeUpdate();

        numUpd = s.createMutationQuery("UPDATE pueblos SET nombre=:nombrep, capital=:capitalc, poblacion=:poblacionc, superficie_km2=:superficiec WHERE id=:oldName")
                .setParameter("oldName", oldname).setParameter("nombrep", nombrepu)
                .executeUpdate();
        System.out.println("Se ha actualizadp: " + numUpd + " registros.");

    } catch (Exception e) {
        e.printStackTrace(System.err);
    }
}
}
```

```java
package org.apache.maven.geopaxil;

import java.math.BigDecimal;

public class crear {
    public static void main(String[] args) {
        String nombre = "aa";
        String capital = "asd";
        int poblacion = 98;;
        BigDecimal superficie = null;
//
        String nompro = "as";
        String capitalpro = "ass";
        int poblacionpro = 66;
        BigDecimal superficiepro = null;

//      String nombrepu = "";
//      Integer poblacionpu = 0;
//      String codigoPostalpu = "";
//      String coordenadasGeograficas= "";

        try(Session s = HibernateUtil.getSessionFactory().openSession()){
            s.getTransaction().begin();
            Comunidadesautonomas ca = new Comunidadesautonomas();
            ca.setNombre(nombre);
            ca.setCapital(capital);
            ca.setPoblacion(poblacion);
            ca.setSuperficieKm2(superficie);
            s.persist(ca);
            s.getTransaction().commit();

            s.getTransaction().begin();
            Provincias pro = new Provincias();
            pro.setNombre(nompro);
            pro.setCapital(capitalpro);
            pro.setComunidadesautonomas(ca);
            pro.setPoblacion(poblacionpro);
            pro.setSuperficieKm2(superficiepro);
            s.persist(pro);
            s.getTransaction().commit();

//          s.getTransaction().begin();
//          Pueblos pueblo = new Pueblos();
//          pueblo.setNombre(nombrepu);
```

```java
package org.apache.maven.geopaxil;

import org.hibernate.Session;

public class eliminar {
    public static void main(String[] args) {
        int eliminar = 0;
        int numDel;
        try (Session s = HibernateUtil.getSessionFactory().openSession()) {
            // Sentencia de Borrado
            numDel = s.createMutationQuery("DELETE comunidadesautonomas WHERE id=:nombre")
                    .setParameter("nombre", eliminar).executeUpdate();

            numDel = s.createMutationQuery("DELETE provincias WHERE id=:nombre").setParameter("nombre", eliminar)
                    .executeUpdate();

            numDel = s.createMutationQuery("DELETE pueblos WHERE id=:nombre").setParameter("nombre", eliminar)
                    .executeUpdate();
            System.out.println("Se han borrado: " + numDel + " registros.");

            // Necesario commit
            s.getTransaction().commit();
        } catch (Exception e) {
            e.printStackTrace(System.err);
        }
    }
}
```

7. Conclusiones:

a. **Aspectos destacados del proyecto:** El proyecto logró desarrollar con éxito una aplicación de gestión de información sobre comunidades autónomas, pueblos y ciudades utilizando Java, Hibernate y MySQL. Los puntos destacados incluyen la creación de una interfaz gráfica intuitiva y funcional, la integración efectiva de Hibernate para la gestión de la base de datos y la colaboración exitosa entre los dos miembros del equipo. La aplicación resultante permite a los usuarios realizar operaciones CRUD de manera eficiente.

b. **Lecciones aprendidas**:

- La planificación detallada y la definición clara de los requisitos.
- La colaboración entre diseñadores gráficos y desarrolladores de código requiere una comunicación efectiva y una comprensión mutua de los objetivos del proyecto.
- La división de tareas puede acelerar el proceso de desarrollo, pero es crucial coordinación para garantizar una buena integración al final del proyecto.
- La elección de herramientas y tecnologías adecuadas, como Hibernate para la persistencia de datos y WindowsBuilder para la interfaz gráfica, puede simplificar el desarrollo y mejorar la eficiencia.
- La realización de pruebas exhaustivas durante el desarrollo es esencial para identificar y solucionar problemas tempranamente.

En resumen, el proyecto destacó por su exitosa integración de Hibernate, una herramienta fundamental para la gestión eficiente de la base de datos.