Taller Agiles

Roles:

Scrum Máster: Nicolas Cadena.

Product Owner: Yasmani Tapia

Team Scrum: Harold Calderón y Juan Perdomo.

Productos:

Sprint 1:

Product Sprint 1 -> R1, R2.

R1: Permitir registrar Productos (nombre, cantidad, valor unitario) en dos diferentes motores de bases de datos. El motor de base de datos 1 debe almacenar los productos con valor unitario superior a 100.000 pesos en caso contrario se debe almacenar en el motor

2. Utilizar el patrón Sigleton y Factory Method.

R2: Permitir visualizar todos los productos.

Sprint 2:

Product Sprint 2 -> R3, R4.

R3: En el mismo programa se desea permitir crear productos utilizando registros existentes en el inventario (Utilizar patrón prototype).

R4: Se quiere implementar una funcionalidad que permita visualizar la cantidad total de productos y el valor total de inventario. Cada uno debe ser un elemento de interfaz diferente, la cantidad de productos mostrarlo en un una Etiqueta (JLabel) y el total de inventario en una caja de texto (JtextField). Los valores de estos productos se deben ver refrescados instantáneamente después de ingresar un nuevo producto. (Utilizar el patrón Observar).

Eventos:

Fecha inicio: 20/01/2024, 11:00 am.

Sprint 1:

Harold Calderón

¿Qué se ha hecho?

Las clases Main, cnxMySql y cnxPostgreSql para realizar la conexión a bases de datos PostgreSql y Mysql.

¿Qué dificultades hay?

La implementación de los patrones Singleton y Factory Method en la conexión.

¿Qué queda por hacer?

Insertar los datos en las bases de datos correspondientes según su valor unitario.

Juan Perdomo

¿Qué se ha hecho?

La instalación de dependencias necesarias y configuración de archivos del motor de bases de datos.

¿Qué dificultades hay?

La conexión de los datos realizado en cada patrón.

¿Qué queda por hacer?

Insertar los datos correspondientes según su valor en cada campo.

Yasmani Tapia

¿Qué se ha hecho?

Creación y diseño de la base de datos de PostgreSQL y Mysql.

¿Qué dificultades hay?

Problemas en el planteamiento de patrones de diseño.

¿Qué queda por hacer?

Insertar los datos correspondientes según su valor en cada campo.

Harold Calderón

¿Qué se ha hecho?

Los respectivos atributos y el método la clase CProducto para registrar los productos según su valor en las bases de datos.

¿Qué dificultades hay?

Implementación del patrón Factory Method

¿Qué queda por hacer?

El método para mostrar los productos en las bases de datos.

Juan Perdomo

¿Qué se ha hecho?

Se creó un formulario para el ingreso de datos.

¿Qué dificultades hay?

La implementación del patrón de diseño de Singleton.

¿Qué queda por hacer?

El método para mostrar los productos en las bases de datos.

Yasmani Tapia

¿Qué se ha hecho?

Se creo un método dentro la clase para visualizar los productos de la base de datos pertenecientes en la tabla

¿Qué dificultades hay?

Realizar una función que clasifique los datos por cada columna

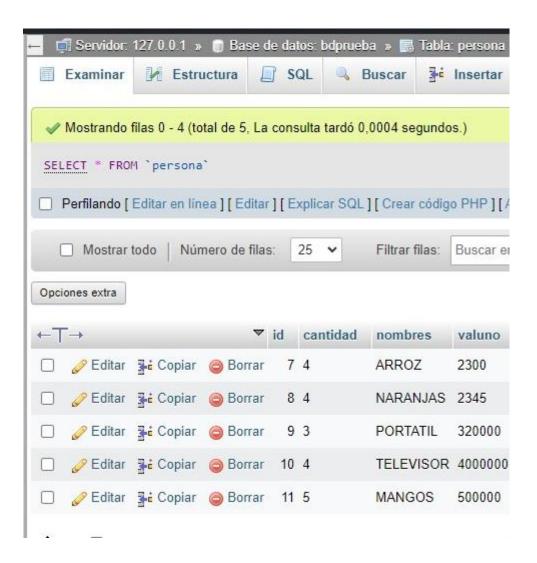
¿Qué queda por hacer?

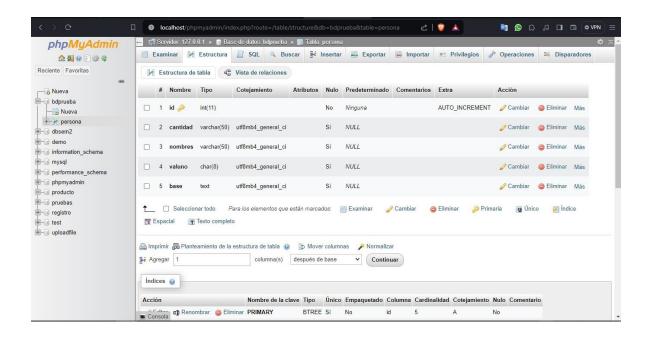
La visualización de cada producto que se ingresó en el formulario.

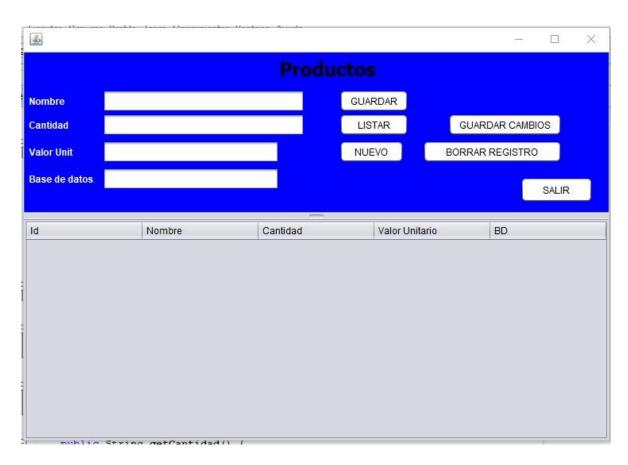
• **Review:** 23/01/2024, 7:00 pm a 08:00 pm.

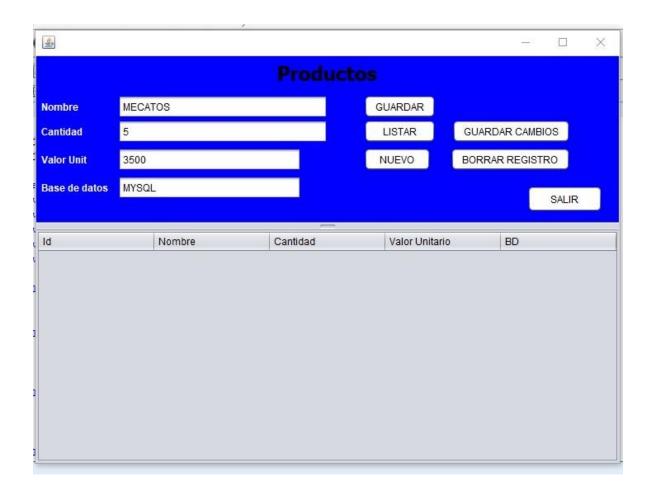
¿Qué porcentaje del producto sprint se terminó?

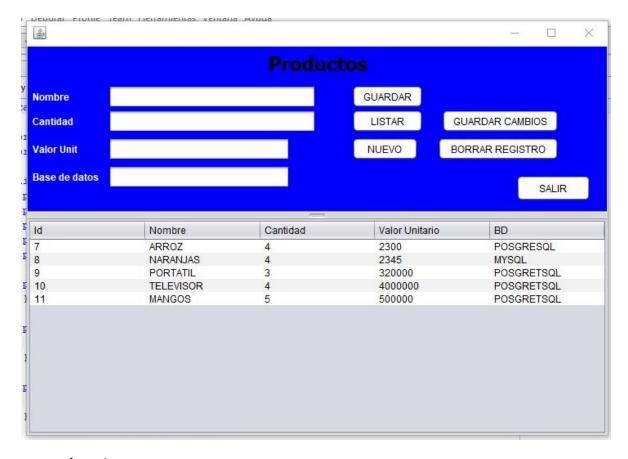
Se realizo un porcentaje de un 70%











Insertar

```
public class ProductoDAO extends Producto {
    @Override
    public void Insertar() throws SQLException {
        PreparedStatement pstmt=null;
        Connection cnx=null;
        try{
           String sql="INSERT INTO persona(cantidad, nombres, valuno, base)VALUES(?,?,?,?)";
           cnx=Conexion.Obtener();
           pstmt=cnx.prepareStatement(sql);
           pstmt.setString(1, this.getCantidad());
           pstmt.setString(2, this.getNombres());
           pstmt.setString(3, this.getValuno());
           pstmt.setString(4, this.getBase());
           pstmt.execute();
        }finally{
           if(pstmt!=null)
               pstmt.close();
```

• Listar

```
@Override
public List Listar() throws SQLException {
   PreparedStatement pstmt=null;
   ResultSet rs=null;
   Connection cnx=null;
   List<ProductoDAO> lista= new ArrayList<>();
   try{
        String sql="SELECT * FROM persona";
        cnx=Conexion.Obtener();
        pstmt=cnx.prepareStatement(sql);
        rs=pstmt.executeQuery();
        while(rs.next()){
            ProductoDAO oPrueba= new ProductoDAO();
           oPrueba.setId(rs.getInt(1));
           oPrueba.setNombres(rs.getString(2));
           oPrueba.setCantidad(rs.getString(3));
           oPrueba.setValuno(rs.getString(4));
           oPrueba.setBase(rs.getString(5));
           lista.add(oPrueba);
        return lista;
    }finally{
        if (pstmt!=null)
           pstmt.close();
        if(rs!=null)
          rs.close();
```

Interfaz

