Informe proyecto final Programación II

1. Introducción:

   La idea original nació de uno de mis hobbies que es la apicultura, me pareció una buena idea ya que encontré bastantes relaciones de composición y agregación, además de relaciones simples, herencias, etc.

   El programa en sí, pretende ser una herramienta para un apicultor, sus clientes y proveedores, facilitando las tareas de gestión, compra y venta de productos. Por cuestiones de tiempo, no fue posible implementar una interfaz gráfica pero es algo que estaba en los planes originales.

2. Estructura:

   El código está estructurado en tres grupos de clases.

   El primer grupo se encarga de derivar a los distintos tipos de usuarios, validarlos, verificar que las entradas sean correctas, que estén entre los parámetros deseados y maneja las excepciones relacionadas a un ingreso de un tipo de dato erroneo

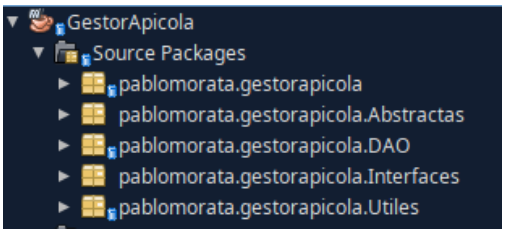
   El segundo grupo consta de las clases que los usuarios gestionan, siendo estas Producto, Consumible y Colmena, cada una de estas posee los métodos para gestionar sus atributos y comunicarselo a las clases DAO.
   Estas tres clases comparten una Interface común, que dicta los métodos: selectorOpciones(), registrar(), agregarAInventario(), eliminarDeInventario(), modificar(), eliminar() y mostrar().
   Tambien componen este grupo, los distintos tipos de usuarios, siendo estos Administrador, Cliente, Proveedor. En este sector se encuentra una herencia de 3 niveles:
   Persona → Usuarios → (Administrador, Cliente, Proveedor)

   Por último, las clases DAO se encargan exclusivamente del intercambio de información con la base de datos. Como mención, en este grupo se encuentran dos clases particulares, por un lado

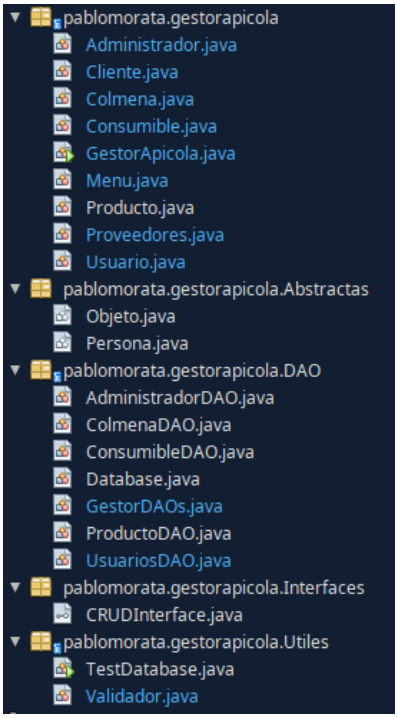
   “GestorDAOs” se encarga de tareas genericas relacionadas con las clases Producto, Consumible y Colmena. Por otro lado la clase UsuariosDAO que maneja situaciones comunes a Administrador, Cliente y Proveedor.

3. Dentro de Netbeans

Carpetas del proyecto:



Clases del proyecto:



Algunas Clases

Clase main → GestorApicola:

```java
package pablomorata.gestorapicola;

public class GestorApicola {

    public static void main(String[] args) {

        Menu menu = new Menu();

        menu.selectorOpciones();

    }
}
```

Clase → Menu (fragmento):

```java
public void selectorOpciones() {

    while (!salir) {

        System.out.println("""
                           Ingrese su usuario:
                           1: Administrador
                           2: Cliente
                           3: Proveedor
                           4: Crear nuevo usuario
                           0: Salir""");
        entrada = Validador.entreParametros(0, 4);


        ///validar entrada con excepciones

        switch (entrada) {
            case 0 ->
                salir = true;

            case 1 -> {

                if (validarUsuario("Administrador")) {
                    administrador.selectorOpciones(nombre);
                }
            }

            case 2 -> {

                if (validarUsuario("Cliente")) {
                    cliente.selectorOpciones(nombre);
                }
            }

            case 3 -> {

                if (validarUsuario("Proveedor")) {
                    proveedores.selectorOpciones(nombre);
                }
            }
            case 4 -> nuevoUsuario();
        }
    }
}
```

Interface → CRUDInterface:

```java
public interface CRUDInterface{

    public void selectorOpciones();

    public void registrar();

    public void agregarAInventario();

    public void eliminarDeInventario();

    public void modificar();

    public void eliminar();

    public void mostrar();

}
```

Abstracta → Persona:

```java
public abstract class Persona {

    String nombre;

    public Persona(){
    }


    public Persona(String nombre){

        this.nombre = nombre;

    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }
```

Clase → Database (fragmento):

```java
public class Database {

    private static final String URL = "jdbc:sqlite:/home/blitowsky/Documents/Facultad/Programación II/Proyecto Final/DBGestorApicola.db";
    private static Connection connection = null;

    public static Connection connect(){

        try{

            if(connection == null || connection.isClosed()){

                connection = DriverManager.getConnection(URL);
                System.out.println("Conexión exitosa.");

            }


        } catch(SQLException e){

            System.err.println("Conexión fallida: " + e.getMessage());

        }
        return connection;

    }
```

Clase → ColmenaDAO (fragmento):

```java
public class ColmenaDAO {

    public Colmena traerColmenas(int id) {

        String sql = "SELECT * FROM Colmena WHERE id = ?";

        try (Connection conn = Database.connect(); PreparedStatement pstmt = conn.prepareStatement(sql)) {

            pstmt.setInt(1, id); // Asignar el valor del parámetro

            try (ResultSet rs = pstmt.executeQuery()) {
                if (rs.next()) { // Mover el cursor a la primera fila
                    int colmenaId = rs.getInt("id");
                    boolean abejas = rs.getBoolean("abejas");
                    int miel = rs.getInt("miel");
                    int marcos = rs.getInt("marcos");
                    String estado = rs.getString("observaciones");
                    int peso = rs.getInt("peso");
                    String utilidad = rs.getString("utilidad");
                    int prioridad = rs.getInt("prioridad");
                    // Crear y devolver el objeto Colmena
                    return new Colmena(colmenaId, abejas, miel, marcos, estado, peso,prioridad ,utilidad);
                } else {
                    return null;
                }
            }

        } catch (SQLException e) {
            System.err.println("Error al retornar el objeto colmena colmenas: " + e.getMessage());
        }
        return null;

    }
```

4. Conclusiones:

   Realizar este proyecto fue todo un desafío. A lo largo de los últimos 2 meses tuve que aprender desde 0 y sin experiencia en el paradigma POO, un lenguaje tan amplio como Java, entender el funcionamiento de una base de datos, la sintaxis básica de SQLite y, no menos importante, como conectarlo con Java para poder lograr una aplicación funcional.

   El tiempo fue un gran limintante al momento de aplicar todas las ideas que tenía en mente, a pesar de eso, considero que fue suficiente para implementar las cosas principales y que cuente con las bases para construir las demas ideas pendientes.