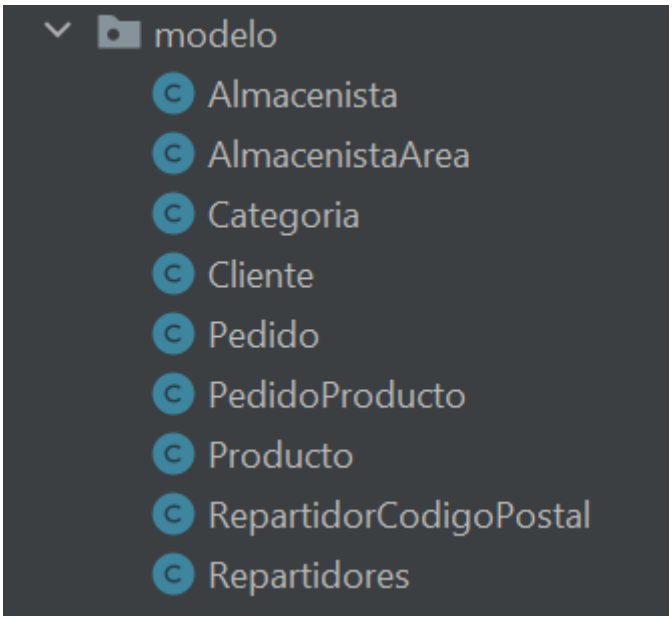
CONEXIÓN DE JAVA CON LA BASE DE DATOS SUPERMERCADO

CONEXIÓN A LA BASE DE DATOS: Se estableció una conexión a la base de datos utilizando la clase ConexionBD. Esta clase proporciona métodos estáticos para obtener y cerrar conexiones con la base de datos.

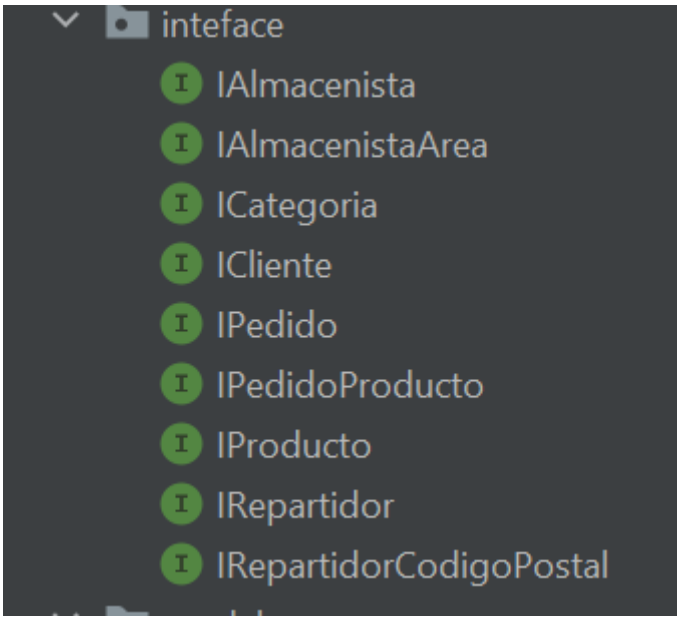
```java
public class ConexionBD {
    1 usage
    private static final String URL_BD = "jdbc:mysql://localhost:3306/supermercado";
    1 usage
    private static final String USER = "root";
    1 usage
    private static final String PASSWORD = "admin";

    16 usages  Devon Alvarez
    public static Connection getConnection() throws SQLException {
        return DriverManager.getConnection(URL_BD, USER, PASSWORD);
    }
}
```

DEFINICIÓN DE LAS CLASES DE MODELOS: Se crearon clases de modelos para representar las entidades del sistema, como Cliente, Categoria, Almacenista, Repartidor, RepartidorCodigoPostal, AlmacenistaArea, Producto, Pedido y PedidoProducto. Cada clase tiene sus atributos correspondientes y métodos para acceder y modificar estos atributos.



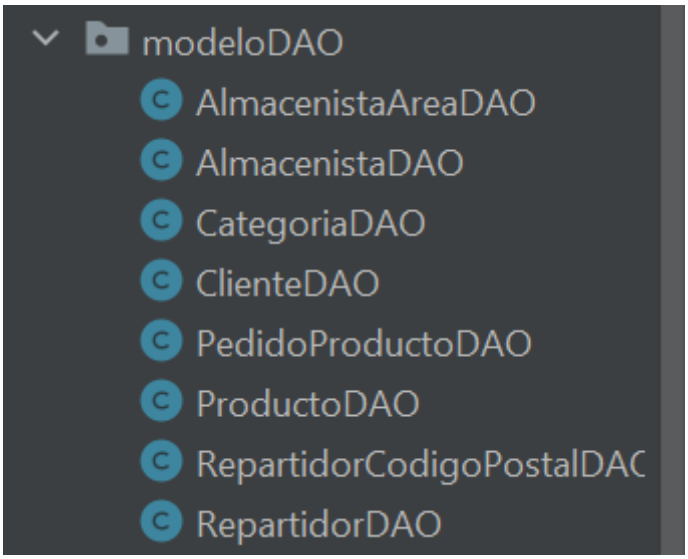
(Ver detalles de las clases en el proyecto)

CREACIÓN DE LAS INTERFACES DE ACCESO A DATOS (DAO): Se crearon interfaces para definir los métodos que serán implementados por las clases DAO. Estas interfaces incluyen métodos para insertar y mostrar datos relacionados con cada entidad del sistema.



(Ver detalles de las interfaces en el proyecto)

IMPLEMENTACIÓN DE LAS CLASES DAO: Se crearon clases DAO para implementar la lógica de acceso a datos definida en las interfaces. Estas clases incluyen métodos para insertar y mostrar datos relacionados con cada entidad del sistema, como **ClienteDAO**, **CategoriaDAO**, **AlmacenistaDAO**, **RepartidorDAO**, **RepartidorCodigoPostalDAO**, **AlmacenistaAreaDAO**, **ProductoDAO**, **PedidoDAO** y **PedidoProductoDAO**.



(Ver detalles de las interfaces en el proyecto)

UTILIZACIÓN EN LA CLASE MAIN: Se creó una clase Main para probar las funcionalidades de las clases DAO. En esta clase, se crearon objetos de las entidades del sistema, se utilizaron los métodos de las clases DAO para insertar estos objetos en la base de datos, y se mostraron los datos insertados utilizando los métodos de las clases DAO correspondientes.

```
Devon Alvarez
public static void main(String[] args) {


    //CLIENTE

    //Insertar y mostrar Cliente
    System.out.println("Mostrando cliente");
    ClienteDAO clienteDAO = new ClienteDAO();
    clienteDAO.mostrarCliente();


    System.out.println("Insertando clientes...");

    List<Cliente> clientes = new ArrayList<Cliente>();
    clientes.add(new Cliente( id: 18,  cedula: "1234567890",  nombres: "Juan",  apellidos: "Pérez",  direccion: "Calle Principal 1
    clientes.add(new Cliente( id: 19,  cedula: "0987654321",  nombres: "María",  apellidos: "López",  direccion: "Avenida Central

    clienteDAO.insertarClientes(clientes);
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

id	cedula	nombres	apellidos	direccion	telefono	codigo_postal	email	contrasenia
16	1003049780	Camilo	Sandobal	Avenida 25 #30-20	3164906738	660676	camila.sandobal@gmail.com	contraseña16
17	982376590	Diego	Ortiz	Calle 18 #8-40	3170478327	770787	diego.ortiz@gmail.com	contraseña17
18	1234567890	Juan	Pérez	Calle Principal 123	123456789	12345	juan@example.com	contraseña123
19	0987654321	María	López	Avenida Central 456	987654321	54321	maria@example.com	password456
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

CATEGORIA

```
//CATEGORIA

//Insertar y mostrar Categoria
List<Categoria> categorias = new ArrayList<>();
categorias.add(new Categoria( nombre: "Cereales",  condicionAlmacenaje: "Seco",  observaciones: "Categoria para cereales y granos"))
categorias.add(new Categoria( nombre: "Tuberculos",  condicionAlmacenaje: "Refrigerado",  observaciones: "Categoria para productos a

CategoriaDAO categoriaDAO = new CategoriaDAO();

categoriaDAO.insertarCategorias(categorias);
categoriaDAO.mostrarCategorias();
```

Cereales	Seco	Categoría para cereales y granos
Tuberculos	Refrigerado	Categoría para productos alimenticios refrigerados
NULL	NULL	NULL

ALMACENISTA

```
//ALMACENISTA
// Insertar y mostrar Almacenistas
 List<Almacenista> almacenistas = new ArrayList<>();
almacenistas.add(new Almacenista( id: 11,  nombre: "Petrona Rodriguez"));
almacenistas.add(new Almacenista( id: 22,  nombre: "Arleth Casola"));

AlmacenistaDAO almacenistaDAO = new AlmacenistaDAO();
  almacenistaDAO.insertarAlmacenistas(almacenistas);
almacenistaDAO.mostrarAlmacenistas();
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wra

id	nombre
9	Pablo Díaz
10	Elena Ruiz
11	Petrona Rodrig...
22	Arleth Casola
NULL	NULL

PRODUCTO

```
//PRODUCTO
 // Insertar y mostrar Productos
List<Producto> productos = new ArrayList<>();
productos.add(new Producto( id: 12,  nombre: "Mantequilla",  marca: "Rama",  origen: "Colombia",  volumen: "50 g",  peso: "50 g",
productos.add(new Producto( id: 13,  nombre: "Salsa de tomate",  marca: "Fruco",  origen: "Colombia",  volumen: "150 ml",  peso:

ProductoDAO productoDAO = new ProductoDAO();
productoDAO.insertarProductos(productos);
//| productoDAO.mostrarProductos();
```

id	nombre	marca	origen	volumen	peso	fotografia	precio	stock	nombre_categoria
10	Desodorante	Axe	Colombia	150 ml	150 ml	desodorante.jpg	6000	80	Higiene
11	cuchilla de hoja	Dorco	Colombia	2g	2g	Cuchilla.jpg	600	60	Higiene
12	Mantequilla	Rama	Colombia	50 g	50 g	Mantequilla.jpg	3500	50	Lacteos
13	Salsa de tomate	Fruco	Colombia	150 ml	200 g	salsa.jpg	7600	30	Alimentos
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

REPARTIDORES

```
//REPARTIDORES
// Insertar y mostrar Repartidores
List<Repartidores> repartidores = new ArrayList<>();
repartidores.add(new Repartidores( id: 13,  nombre: "Duvan Molina",  matriculaFurgoneta: "CJt568"));
repartidores.add(new Repartidores( id: 14,  nombre: "Juan Perez",  matriculaFurgoneta: "UQd894"));

RepartidorDAO repartidorDAO = new RepartidorDAO();
repartidorDAO.insertarRepartidores(repartidores);
// repartidorDAO.mostrarRepartidores();
```

Result Grid | Filter Rows: | Edit: | Export/Import:

id	nombre	matricula_furgoneta
11	Manuel Lozano	CJK345
12	Jairo Marulanda	UQC356
13	Duvan Molina	CJt568
14	Juan Perez	UQd894
NULL	NULL	NULL

PEDIDO_PRODUCTO

```java
//PEDIDO_PRODUCTO
// Insertar y mostrar pedido_producto
List<PedidoProducto> pedidoProductos = new ArrayList<>();
pedidoProductos.add(new PedidoProducto( idPedido: 6,  idProducto: 8,  cantidad: 3,  precioUnitario: 2600));
pedidoProductos.add(new PedidoProducto( idPedido: 5,  idProducto: 8,  cantidad: 3,  precioUnitario: 3400));
pedidoProductos.add(new PedidoProducto( idPedido: 3,  idProducto: 8,  cantidad: 3,  precioUnitario: 4700));

PedidoProductoDAO pedidoProductoDAO = new PedidoProductoDAO();
pedidoProductoDAO.insertarPedidoProductos(pedidoProductos);
pedidoProductoDAO.mostrarPedidoProductos();
```

id_pedido	id_producto	cantidad	precio_unitario
5	1	2	3500
5	9	1	3000
6	8	3	2600
5	8	3	3400
3	8	3	4700

REPARTIDOR_CODIGO_POSTAL

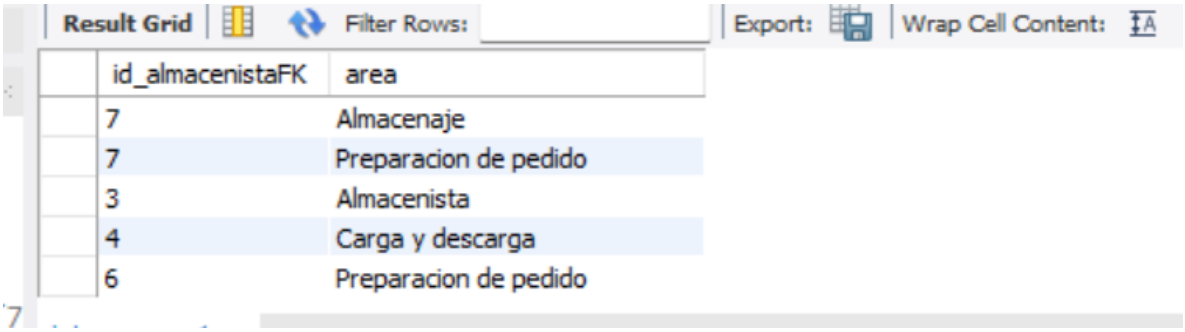
```java
//REPARTIDOR_CODIGO_POSTAL
// Insertar y mostrar repartido y codigo postal
List<RepartidorCodigoPostal> repartidoresCodigosPostales = new ArrayList<>();
repartidoresCodigosPostales.add(new RepartidorCodigoPostal( idRepartidor: 7,  codigoPostal: "11022"));
repartidoresCodigosPostales.add(new RepartidorCodigoPostal( idRepartidor: 6,  codigoPostal: "11023"));
repartidoresCodigosPostales.add(new RepartidorCodigoPostal( idRepartidor: 5,  codigoPostal: "11024"));

// Crear una instancia de RepartidorCodigoPostalDAO
RepartidorCodigoPostalDAO repartidorCodigoPostalDAO = new RepartidorCodigoPostalDAO();
repartidorCodigoPostalDAO.insertarRepartidoresCodigosPostales(repartidoresCodigosPostales);
repartidorCodigoPostalDAO.mostrarRepartidoresCodigosPostales();
```

Result Grid | Filter Rows: | Edit:

id_repartidor	codigo_postal
5	11024
5	220232
6	11023
7	11022
NULL	NULL

ALMACENISTA_AREAS

```java
//ALMACENISTA_AREA
// Insertar y mostrar almacenista_area
List<AlmacenistaArea> almacenistasAreas = new ArrayList<>();
almacenistasAreas.add(new AlmacenistaArea( idAlmacenista: 3,  area: "Almacenista")); // ID Almacenista, Área
almacenistasAreas.add(new AlmacenistaArea( idAlmacenista: 4,  area: "Carga y descarga"));
almacenistasAreas.add(new AlmacenistaArea( idAlmacenista: 6,  area: "Preparacion de pedido"));

AlmacenistaAreaDAO almacenistaAreaDAO = new AlmacenistaAreaDAO();
almacenistaAreaDAO.insertarAlmacenistasAreas(almacenistasAreas);
almacenistaAreaDAO.mostrarAlmacenistasAreas();
```



id_almacenistaFK	area
7	Almacenaje
7	Preparacion de pedido
3	Almacenista
4	Carga y descarga
6	Preparacion de pedido

Este resumen se proporciona una visión completa de los pasos necesarios para establecer una conexión a la base de datos supermercado y acceder a los datos desde Java.