UNIVERSIDAD AUTÓNOMA GABRIEL RENÉ MORENO

FACULTAD DE INGENIERÍA EN CIENCIAS DE LA COMPUTACIÓN Y TELECOMUNICACIONES

**Fecha de Presentación:** 27/9/2024

**Software de Nota de Venta para Emprendedores con Arquitectura MVC(MODELO – VISTA – CONTROLADOR)**

Primer Parcial

Estudiante: Edberto Ybanera Herrera - 219059829

**Materia:** Arquitectura de Software (INF-552)

**Docente:** Ing. Josue Obed Veizaga Gonzales

**Gestión:** 2-2024

Santa Cruz de la Sierra – Bolivia

Contenido

[INTRODUCCION 3](#_Toc178286857)

[1. REQUERIMIENTOS 3](#_Toc178286858)

[1.1. Requerimientos Funcionales 3](#_Toc178286859)

[1.1.1. Identificación de Actores 3](#_Toc178286860)

[1.1.2. Identificación de Cados de Uso 3](#_Toc178286861)

[1.1.3. Detalle de Casos de Uso 4](#_Toc178286862)

[1.2. Requerimientos No Funcionales 9](#_Toc178286863)

[1.2.1. Mantenibilidad 9](#_Toc178286864)

[1.2.2. Portabilidad 9](#_Toc178286865)

[2. ANALISIS 9](#_Toc178286866)

[2.1 Identificación de Paquetes 10](#_Toc178286867)

[2.1.1. Registro 10](#_Toc178286868)

[2.1.2. Ventas 10](#_Toc178286869)

[2.2. Especificación de Paquetes 10](#_Toc178286870)

[2.2.1. Registro 10](#_Toc178286871)

[2.2.2. Ventas 10](#_Toc178286872)

[3. DISEÑO 11](#_Toc178286873)

[3.1. Arquitectura Lógica 11](#_Toc178286874)

[3.1.1 Registro 11](#_Toc178286875)

[3.1.2 Ventas 11](#_Toc178286876)

[IDENTIFICACION DE LOS CASOS DE USO 11](#_Toc178286877)

[CASOS DE USO 12](#_Toc178286878)

[1. CU1 Gestionar Persona 12](#_Toc178286879)

# INTRODUCCION

El propósito de la aplicación para emprendedores es proporcionar una herramienta completa para gestionar productos, categorías, proveedores y ventas. Está diseñada para ayudar a los emprendedores a organizar y administrar eficientemente su negocio, desde la gestión de inventario hasta el seguimiento de ventas. Utilizando el patrón de diseño MVC y SQLite, la aplicación ofrece una estructura sólida y una base de datos integrada para almacenar y gestionar la información de manera eficaz. Además, la inclusión de elementos de diseño de interfaz de usuario y animaciones mejora la experiencia del usuario al interactuar con la aplicación.

# REQUERIMIENTOS

## Requerimientos Funcionales

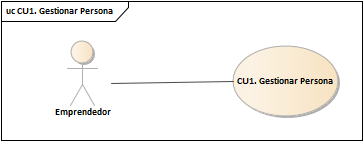
### Identificación de Actores

### Identificación de Cados de Uso

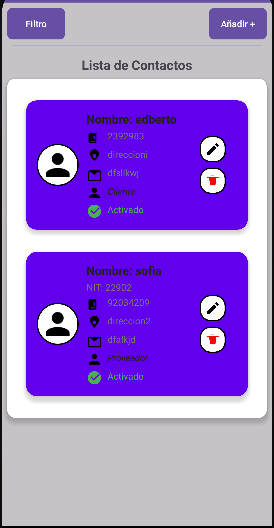
* CU1. Gestionar Persona
* CU2. Gestionar Proveedor
* CU3. Gestionar Categoría
* CU4. Gestionar Producto
* CU5. Gestionar Nota Venta

### Detalle de Casos de Uso

#### CU1. Gestionar Persona



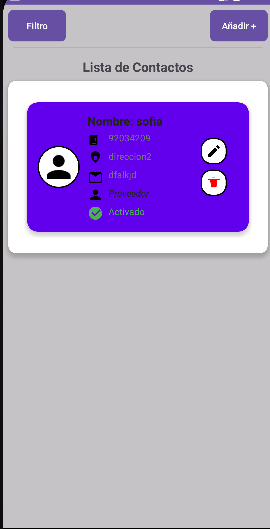
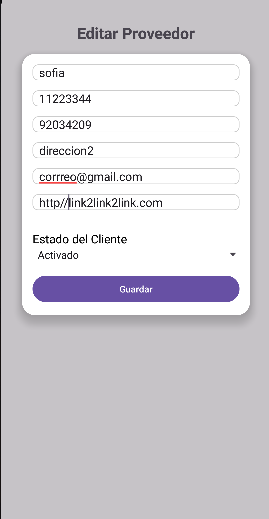
|  |  |
| --- | --- |
| **Caso de Uso** | **CU1. Gestionar Persona** |
| Propósito | Gestionar Persona |
| Descripción | Permite gestionar las personas registradas en el sistema |
| Actores | Emprendedor |
| Actor Iniciador | Emprendedor |
| Precondición | Ninguna |
| Proceso | 1. Agregar Persona    1. Obtener los datos de Persona    2. Guardar datos 2. Modificar Persona    1. Seleccionar Persona a modificar    2. Guardar Datos Modificados 3. Eliminar Persona    1. Selecciona Persona a eliminar    2. Confirmar eliminación    3. Eliminar Persona de la base de datos |
| Postcondición | Gestionar Proveedor, Gestionar Venta, Gestionar Proveedor |
| Excepciones | No modifica el tipo cliente |

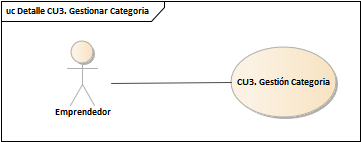
#### CU2. Gestionar Proveedor



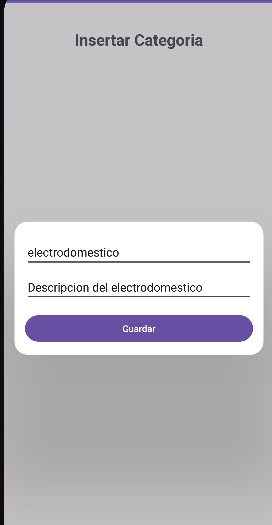
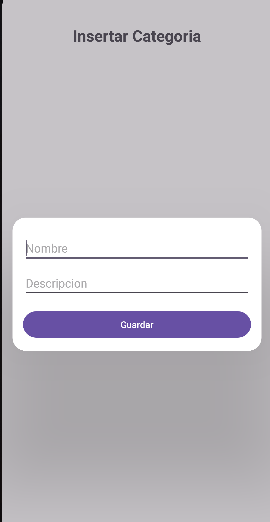
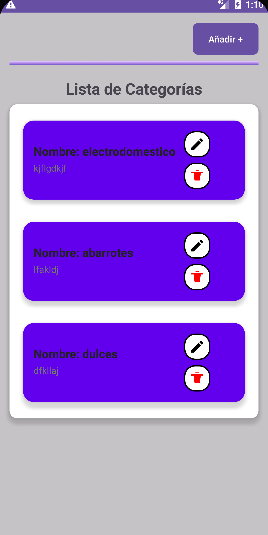
|  |  |
| --- | --- |
| **Caso de Uso** | **CU2. Gestionar Proveedor** |
| Propósito | Gestionar Proveedor |
| Descripción | Permite gestionar a los proveedores que están registradas en el sistema |
| Actores | Emprendedor |
| Actor Iniciador | Emprendedor |
| Precondición | Ninguno |
| Proceso | 1. Agregar Proveedor    1. Obtener los datos de Proveedor    2. Guardar datos 2. Modificar Proveedor    1. Seleccionar Proveedor a modificar    2. Confirmar Datos    3. Guardar Datos Modificados 3. Eliminar Proveedor    1. Selecciona Proveedor a eliminar    2. Confirmar eliminación    3. Eliminar Proveedor de la base de datos |
| Postcondición | Gestionar Producto |
| Excepciones | No modifica el tipo cliente |

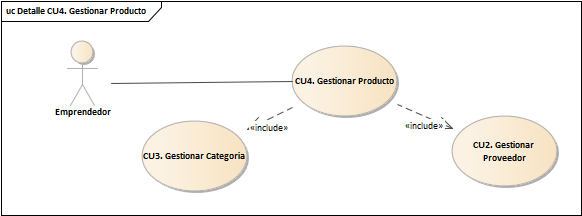
#### CU3. Gestionar Categoría



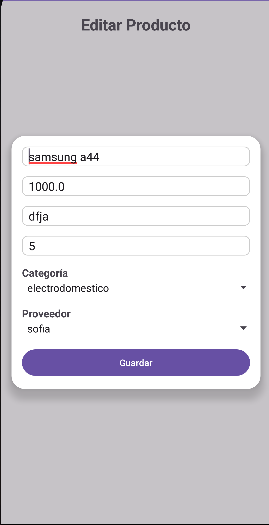
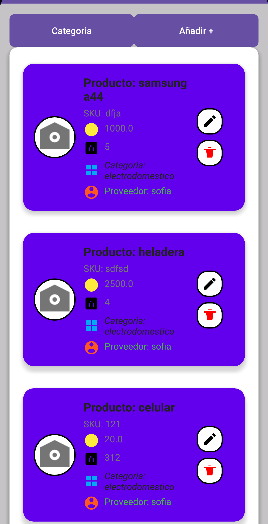
|  |  |
| --- | --- |
| **Caso de Uso** | **CU1. Gestionar Categoría** |
| Propósito | CU3. Gestionar Categoría |
| Descripción | Permite gestionar Categoría que están registradas en el sistema |
| Actores | Emprendedor |
| Actor Iniciador | Emprendedor |
| Precondición | Ninguna |
| Proceso | 1. Agregar Categoría    1. Obtener los datos de Categoría    2. Guardar datos 2. Modificar Categoría    1. Seleccionar Categoría a modificar    2. Modificar Datos    3. Guardar Datos Modificados 3. Eliminar Categoría    1. Selecciona Categoría a eliminar    2. Confirmar eliminación    3. Eliminar Categoría de la base de datos |
| Postcondición | Gestionar Producto |
| Excepciones | Ninguna |



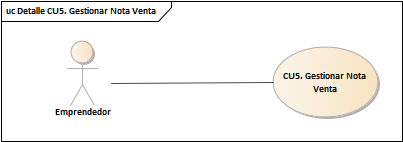
#### CU4. Gestionar Producto



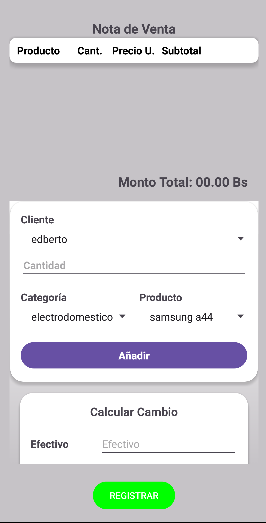
|  |  |
| --- | --- |
| **Caso de Uso** | **CU1. Gestionar Producto** |
| Propósito | Gestionar Producto |
| Descripción | Permite gestionar Producto que están registradas en el sistema |
| Actores | Emprendedor |
| Actor Iniciador | Emprendedor |
| Precondición | Gestionar Categoría |
| Proceso | 1. Agregar Producto    1. Obtener los datos de Producto    2. Guardar datos 2. Modificar Producto    1. Seleccionar Producto a modificar    2. Modificar Datos    3. Guardar Datos Modificados 3. Eliminar Producto    1. Selecciona Producto a eliminar    2. Confirmar eliminación    3. Eliminar Producto de la base de datos |
| Postcondición | Gestionar Nota de Venta |
| Excepciones | Ninguna |



#### CU5. Gestionar Nota Venta



|  |  |
| --- | --- |
| **Caso de Uso** | **CU1. Gestionar Nota de Venta** |
| Propósito | Gestionar Nota de Venta |
| Descripción | Permite gestionar Nota de Venta que están registradas en el sistema |
| Actores | Emprendedor |
| Actor Iniciador | Emprendedor |
| Precondición | Gestionar Producto, Gestionar Persona, Gestionar Categoría |
| Proceso | 1. Agregar Producto    1. Obtener los datos de Producto    2. Guardar datos 2. Modificar Producto    1. Seleccionar Producto a modificar    2. Modificar Datos    3. Guardar Datos Modificados 3. Eliminar Producto    1. Selecciona Producto a eliminar    2. Confirmar eliminación    3. Eliminar Producto de la base de datos |
| Postcondición | Ninguna |
| Excepciones | Aun no está habilitado para modificaciones |



## Requerimientos No Funcionales

### Mantenibilidad

Los casos de uso implementados en el software estarán aislados de tal manera que se puedan identificar.

### Portabilidad

La aplicación móvil es compatible con Android 5.1 en adelante.

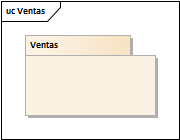
# ANALISIS

## 2.1 Identificación de Paquetes

### 2.1.1. Registro

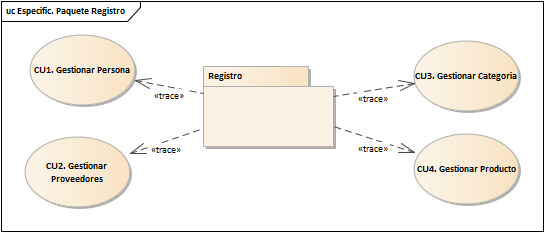


### 2.1.2. Ventas

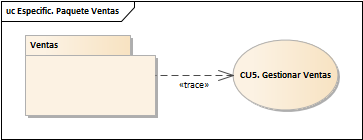


## 2.2. Especificación de Paquetes

### 2.2.1. Registro



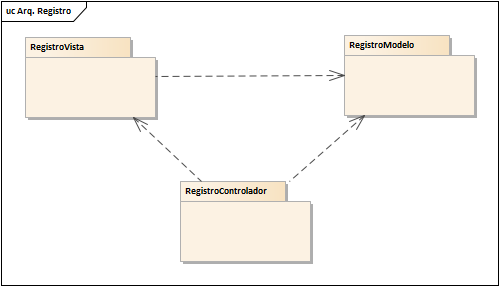
### 2.2.2. Ventas



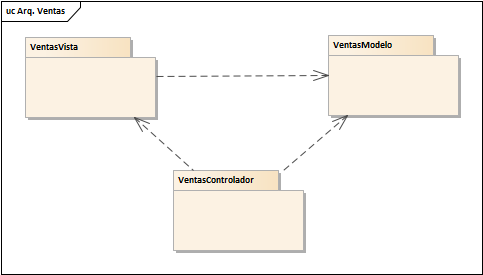
# DISEÑO

## Arquitectura Lógica

### 3.1.1 Registro

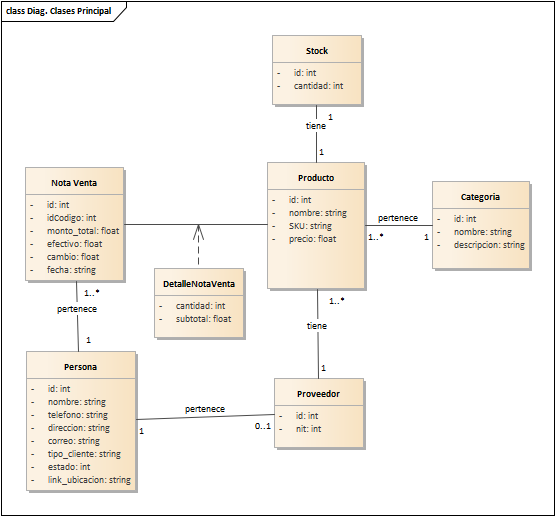


### 3.1.2 Ventas



## Base de Datos

### Diseño Conceptual



### Diseño Lógico



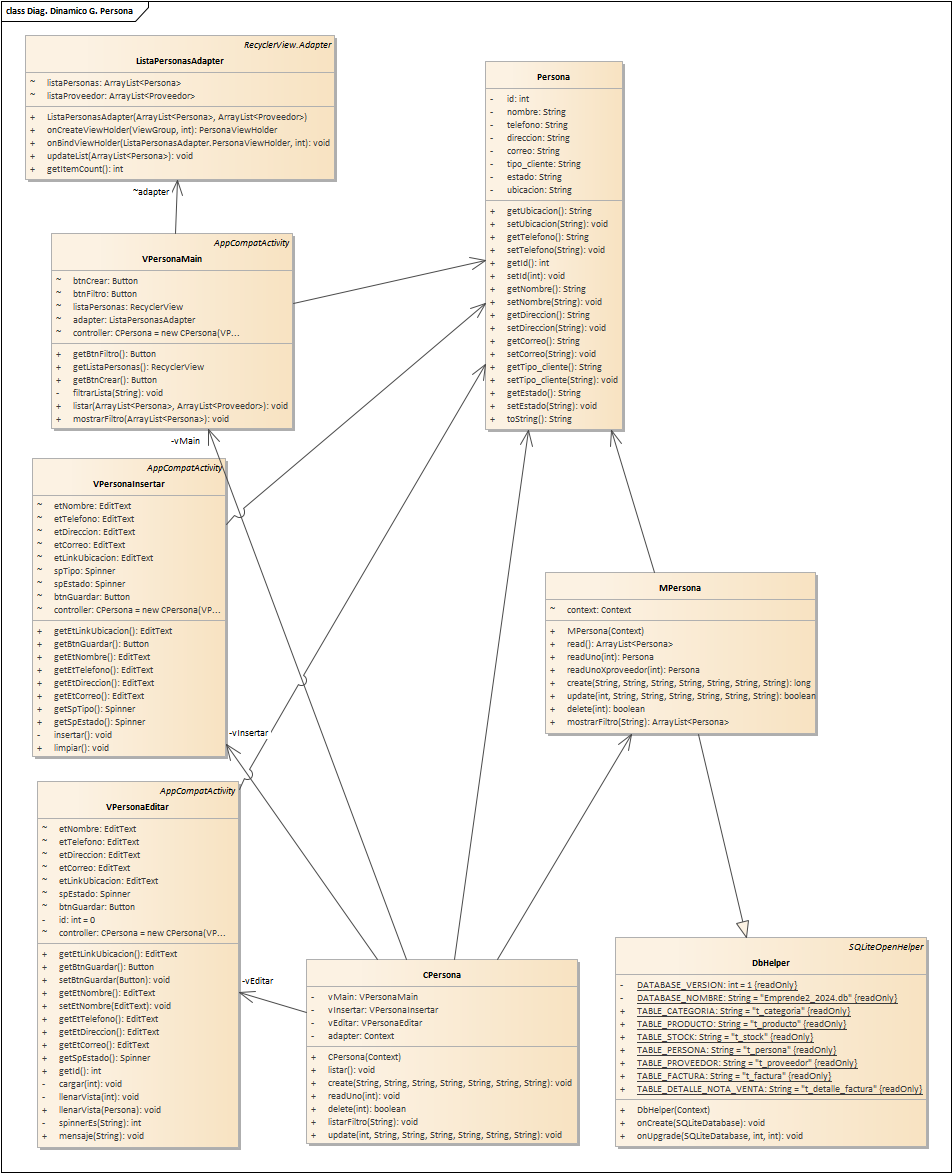
### Diseño Físico

1. -- Crear la tabla de Categoría  
   CREATE TABLE categoria (  
    id INTEGER PRIMARY KEY AUTOINCREMENT,  
    nombre TEXT,  
    descripcion TEXT  
   );  
     
   -- Crear la tabla de Persona  
   CREATE TABLE persona (  
    id INTEGER PRIMARY KEY AUTOINCREMENT,  
    nombre TEXT,  
    telefono TEXT,  
    direccion TEXT,  
    correo TEXT,  
    tipo\_cliente TEXT,  
    estado TEXT,  
    link\_ubicacion TEXT  
   );  
     
   -- Crear la tabla de Proveedor  
   CREATE TABLE proveedor (  
    id INTEGER PRIMARY KEY AUTOINCREMENT,  
    nit TEXT,  
    id\_persona INTEGER,  
    FOREIGN KEY(id\_persona) REFERENCES persona(id)  
   );  
     
   -- Crear la tabla de Producto  
   CREATE TABLE producto (  
    id INTEGER PRIMARY KEY AUTOINCREMENT,  
    nombre TEXT,  
    SKU TEXT,  
    precio REAL,  
    id\_categoria INTEGER,  
    id\_proveedor INTEGER,  
    FOREIGN KEY(id\_categoria) REFERENCES categoria(id),  
    FOREIGN KEY(id\_proveedor) REFERENCES proveedor(id)  
   );  
     
   -- Crear la tabla de Stock  
   CREATE TABLE stock (  
    id INTEGER PRIMARY KEY AUTOINCREMENT,  
    cantidad INTEGER,  
    id\_producto INTEGER,  
    FOREIGN KEY(id\_producto) REFERENCES producto(id)  
   );  
     
   -- Crear la tabla de Factura  
   CREATE TABLE factura (  
    id INTEGER PRIMARY KEY AUTOINCREMENT,  
    id\_codigo INTEGER,  
    id\_persona INTEGER,  
    monto\_total REAL,  
    efectivo REAL,  
    cambio REAL,  
    fecha DATE  
   );  
     
   -- Crear la tabla de Detalle de Nota de Venta  
   CREATE TABLE detalle\_nota\_venta (  
    id INTEGER PRIMARY KEY AUTOINCREMENT,  
    id\_factura INTEGER,  
    id\_producto INTEGER,  
    cantidad REAL,  
    subtotal REAL,  
    FOREIGN KEY(id\_factura) REFERENCES factura(id),  
    FOREIGN KEY(id\_producto) REFERENCES producto(id)  
   );

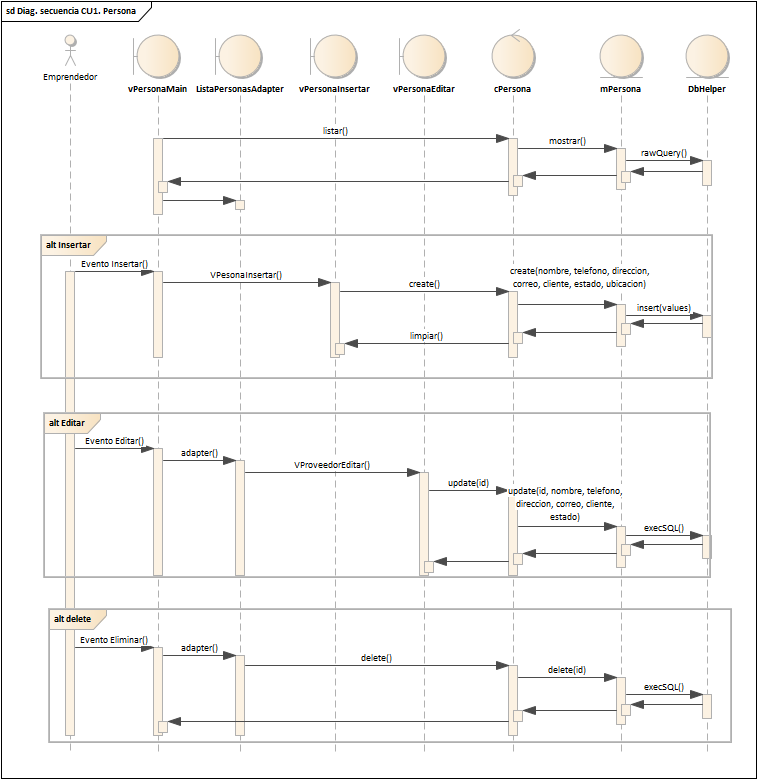
## Detalle Procedimental

### 3.4.1 CU1. Gestionar Persona

#### Diagrama de clases dinámico

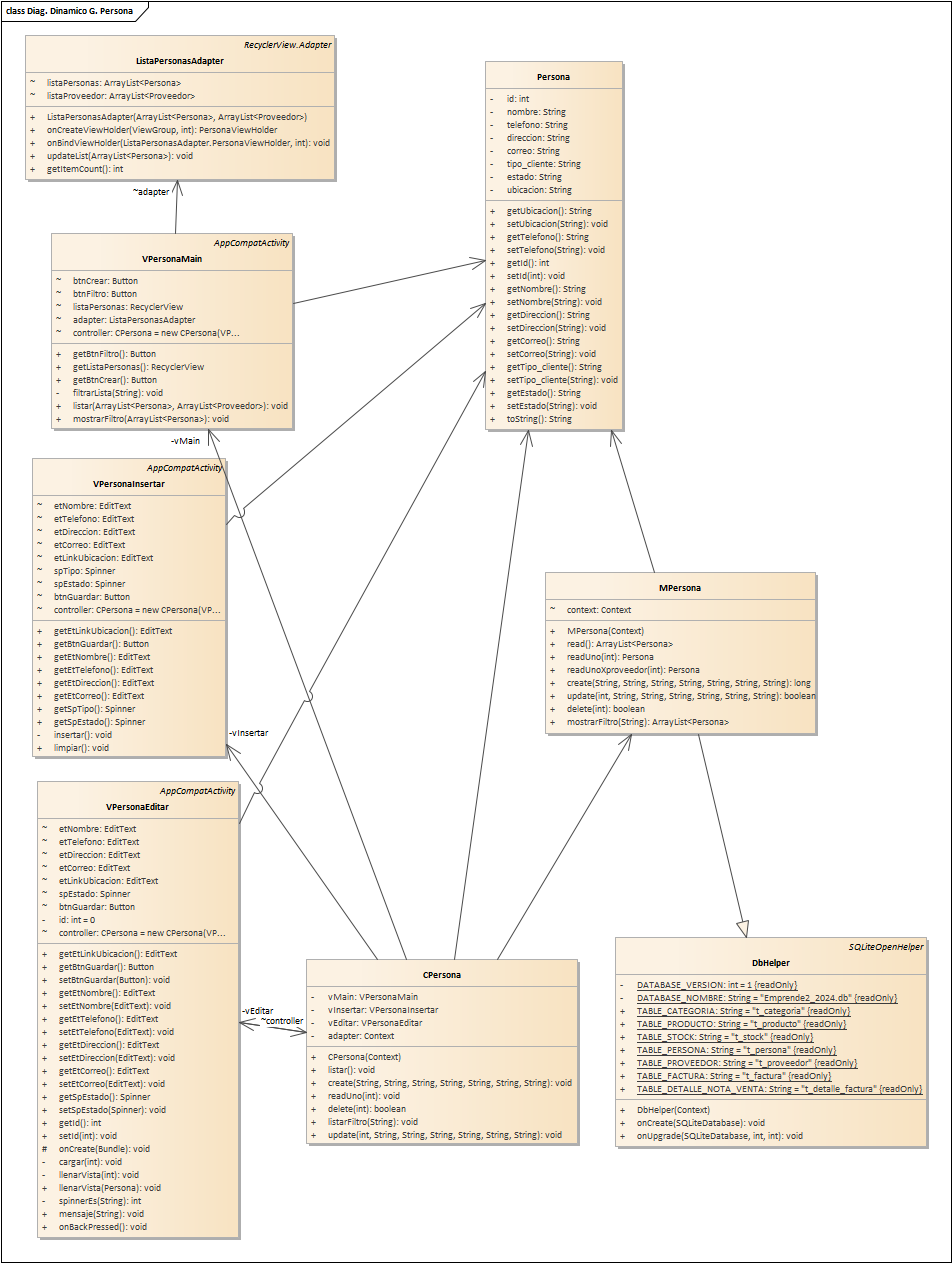


#### Diagrama de secuencia

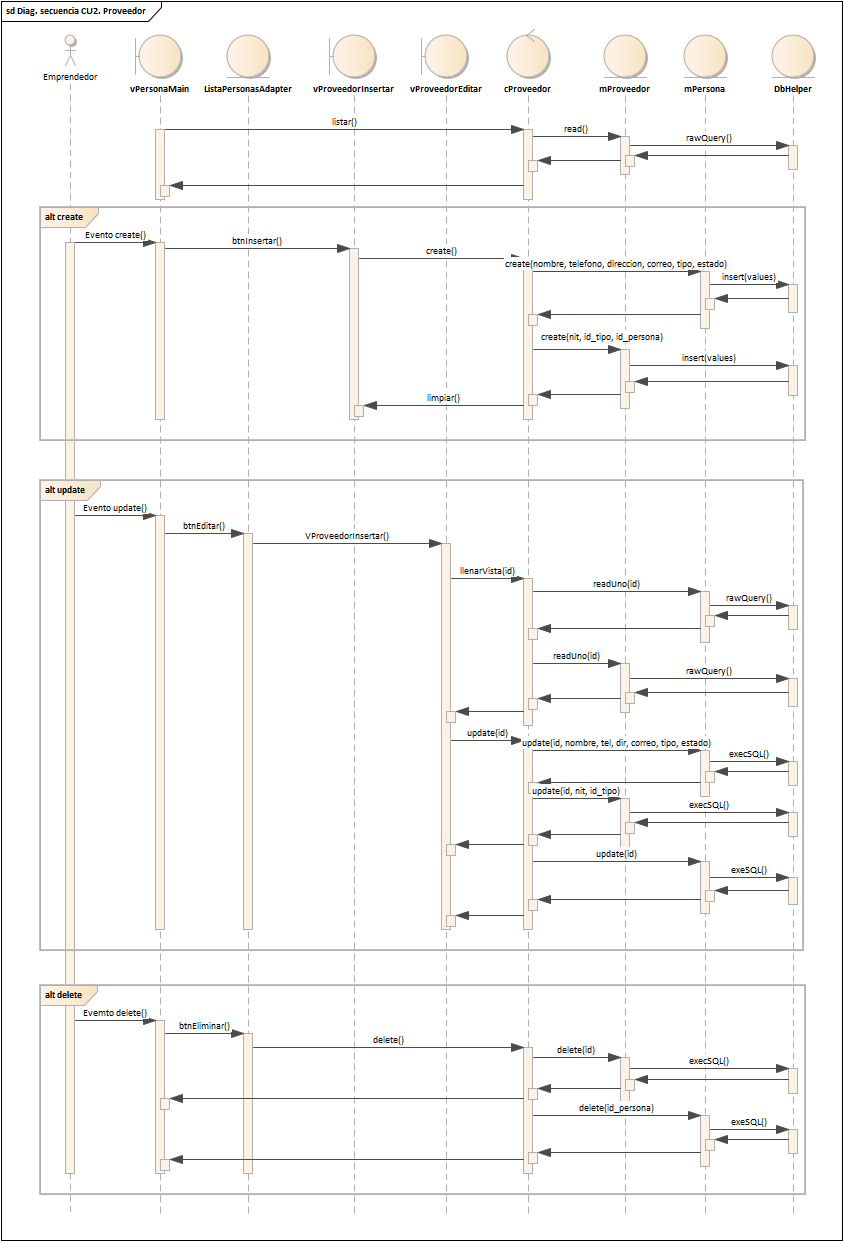


### CU2. Gestionar Proveedor

#### Diagrama de clases dinámico

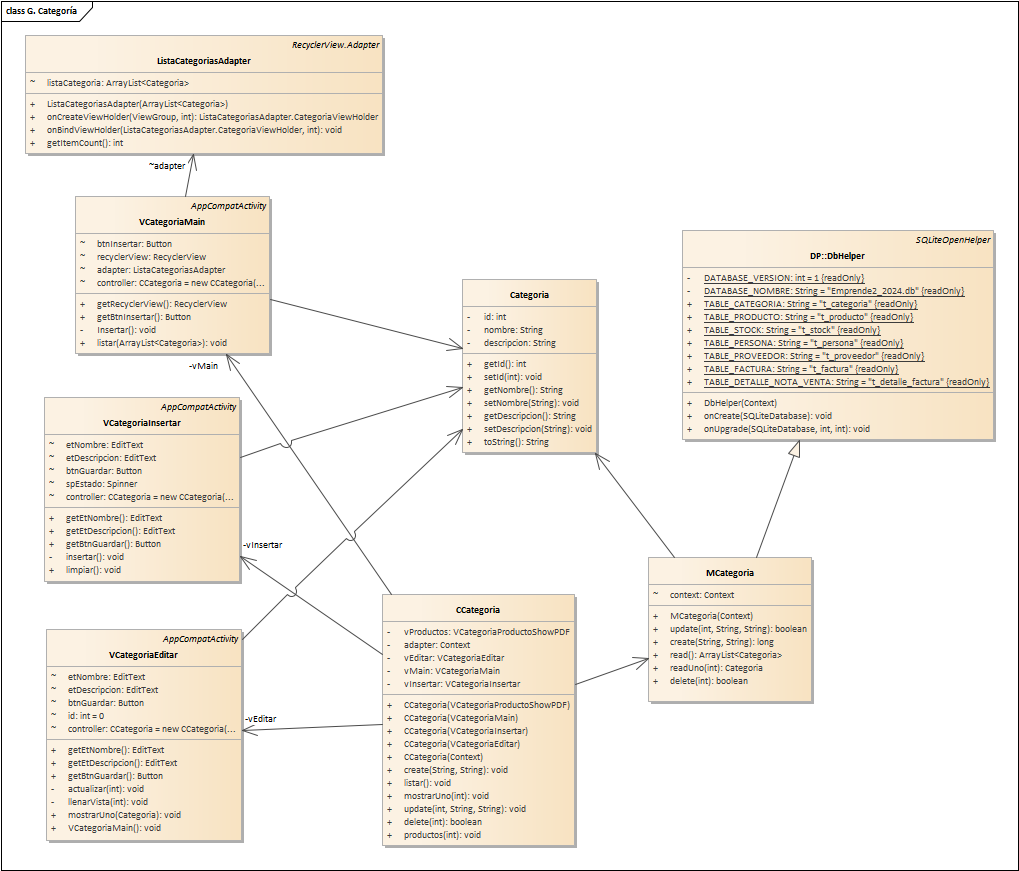


#### Diagrama de secuencia

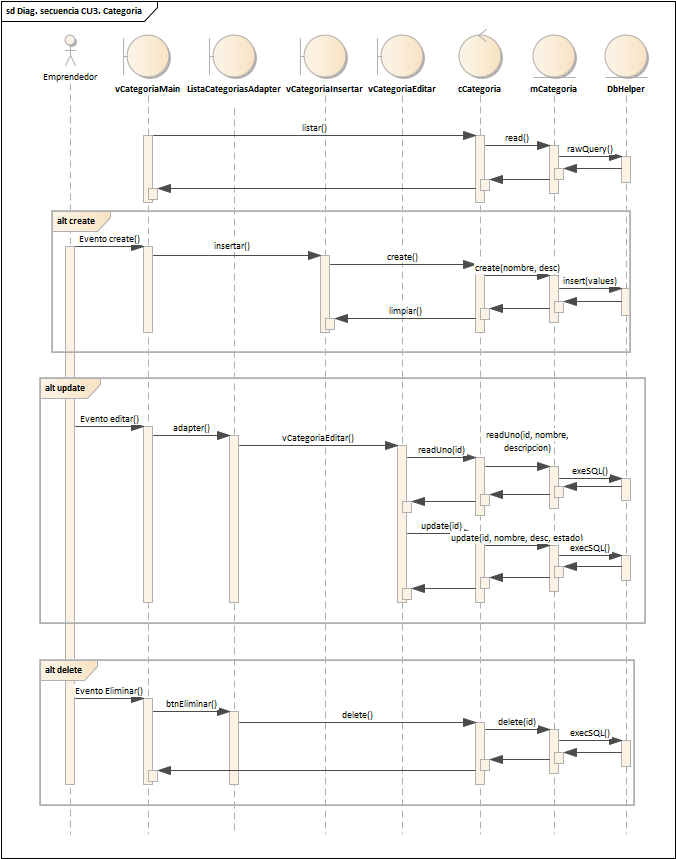


### CU3. Gestionar Categoría

#### Diagrama de clases dinámico

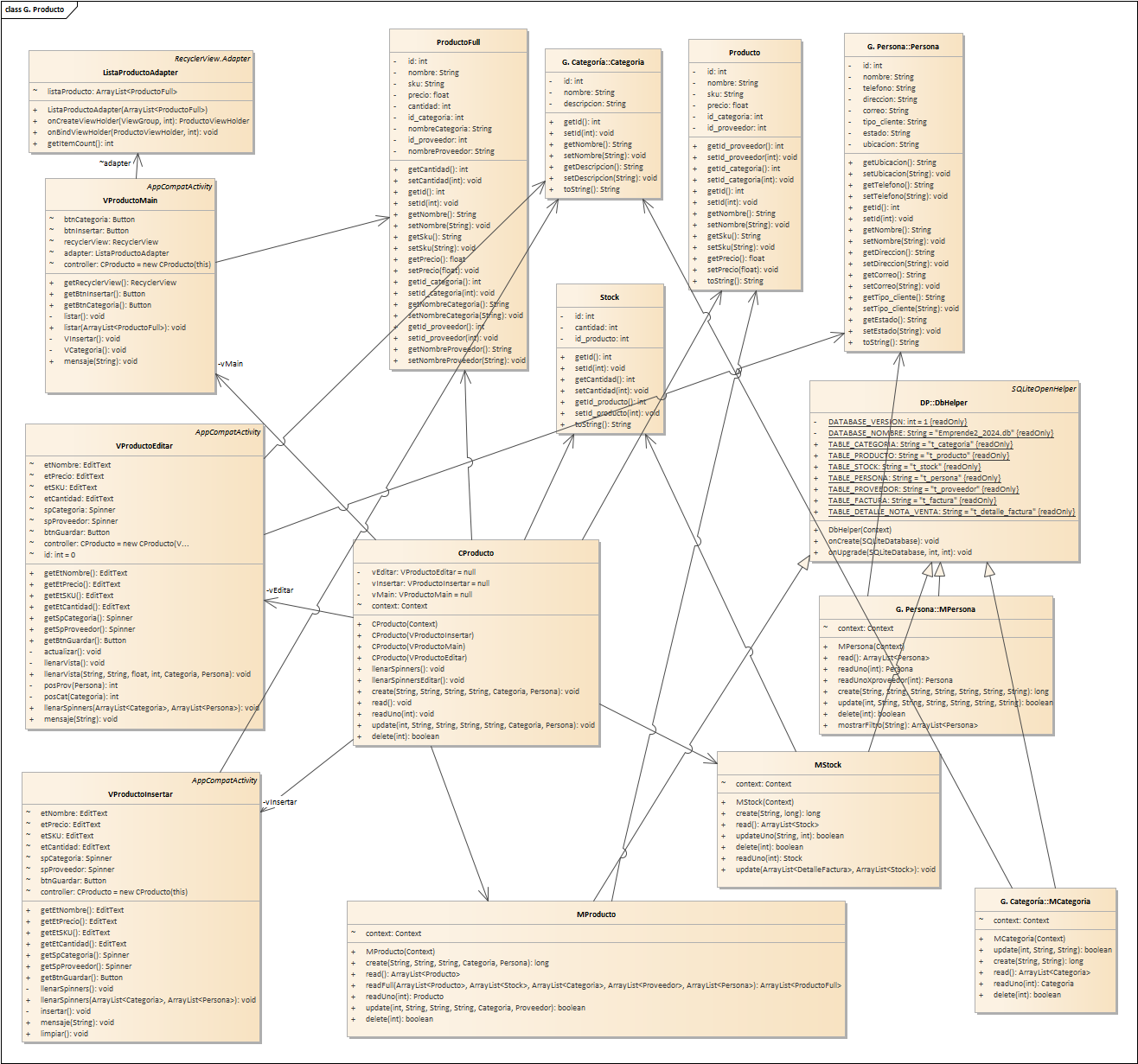


#### Diagrama de secuencia

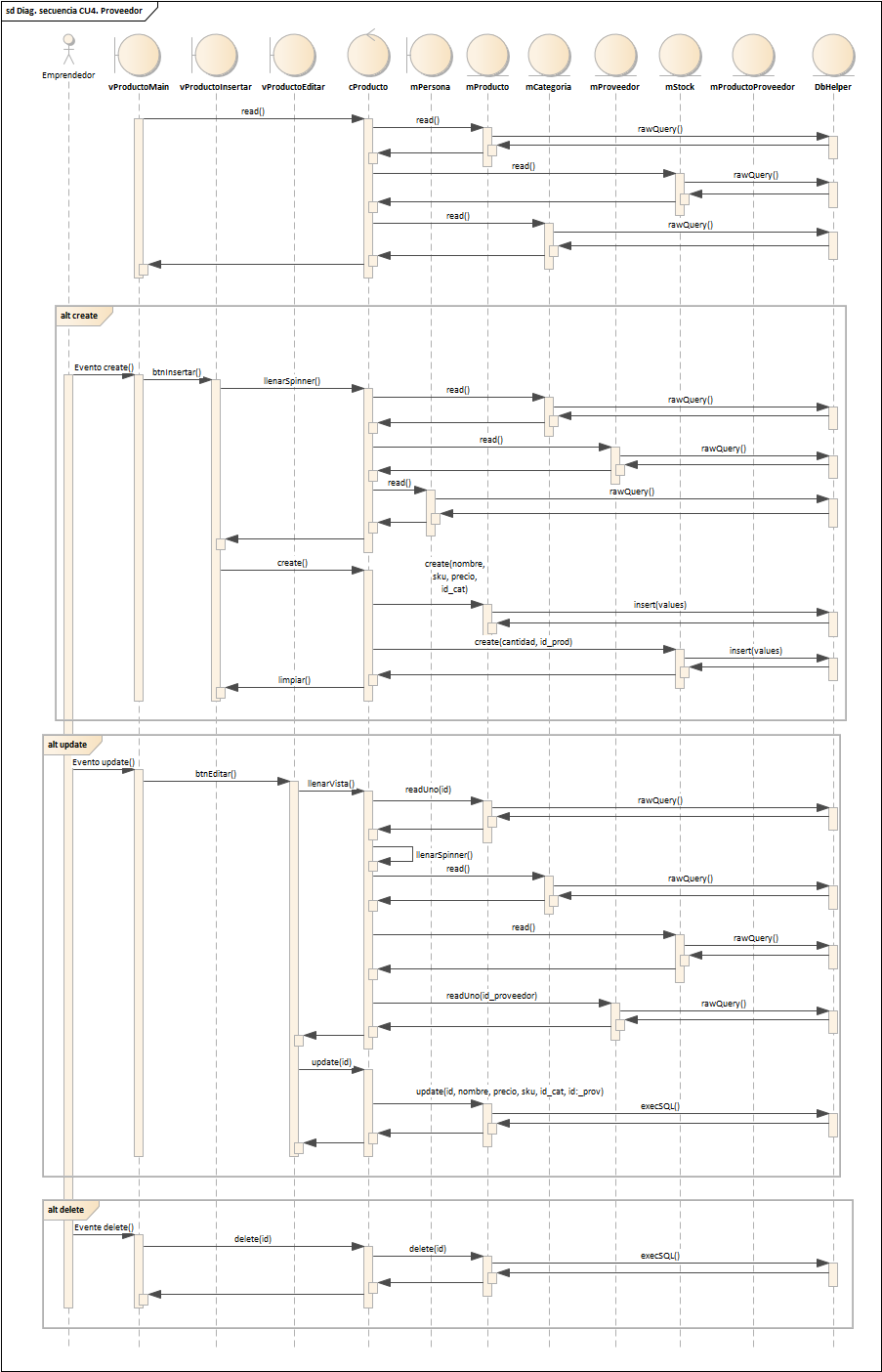


### CU4. Gestionar Producto

#### Diagrama de clases dinámica

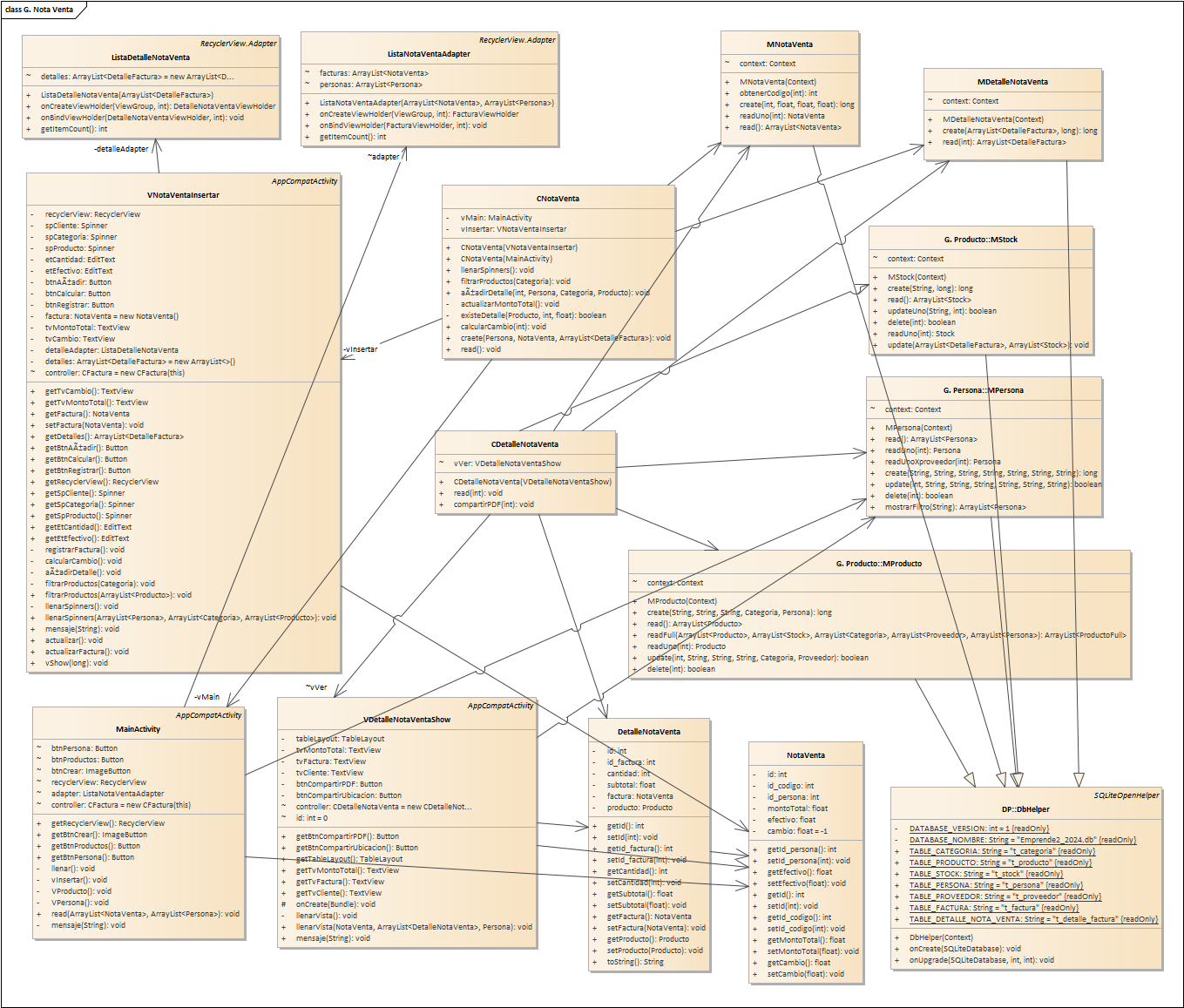


#### Diagrama de secuencia

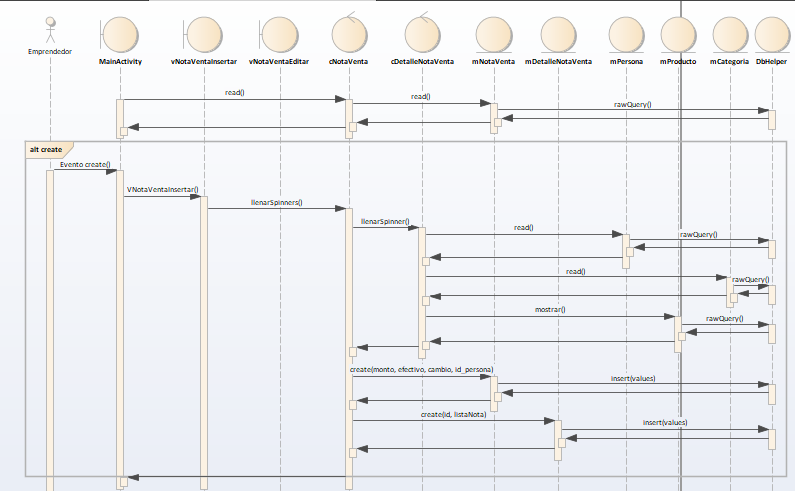


### CU5. Gestionar Nota Venta

#### Diagrama de clases dinámico



#### Diagrama de secuencia



# CODIGO

## VCategoria

### VCategoriaMain

package com.app.emprende2\_2024.view.VCategoria;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.widget.Button;  
import android.widget.Toast;  
  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.recyclerview.widget.LinearLayoutManager;  
import androidx.recyclerview.widget.RecyclerView;  
  
import com.app.emprende2\_2024.R;  
import com.app.emprende2\_2024.controller.CCategoria.CCategoria;  
import com.app.emprende2\_2024.model.MCategoria.Categoria;  
import com.app.emprende2\_2024.view.VProducto.VProductoMain;  
  
import java.util.ArrayList;  
  
  
public class VCategoriaMain extends AppCompatActivity {  
 Button btnInsertar;  
 RecyclerView recyclerView;  
 ListaCategoriasAdapter adapter;  
  
 public RecyclerView getRecyclerView() {  
 return findViewById(R.id.rvCategoria);  
 }  
  
  
  
 public Button getBtnInsertar() {  
 return findViewById(R.id.btnInsertarCategoria);  
 }  
  
 CCategoria controller = new CCategoria(VCategoriaMain.this);  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_vcategoria\_main);  
 getBtnInsertar().setOnClickListener(v -> Insertar());  
 controller.listar();  
  
 }  
 private void Insertar() {  
 Intent intent = new Intent(this, VCategoriaInsertar.class);  
 startActivity(intent);  
 }  
 public void onBackPressed() {  
 // Inicia la actividad que deseas cuando se presiona el botón de atrás  
 super.onBackPressed();  
 Intent intent = new Intent(this, VProductoMain.class);  
 startActivity(intent);  
 finish(); // Opcional, para finalizar la actividad actual si no deseas volver a ella al presionar atrás en la actividad destino  
 }  
  
 public void listar(ArrayList<Categoria> mostrar) {  
 adapter = new ListaCategoriasAdapter(mostrar);  
 try {  
 recyclerView = findViewById(R.id.rvCategoria);  
 recyclerView.setLayoutManager(new LinearLayoutManager(this));  
 recyclerView.setAdapter(adapter);  
  
 }catch (Exception e){  
 Toast.makeText(this, "ERROR ERROR ERROR", Toast.LENGTH\_SHORT).show();  
 e.printStackTrace();  
 }  
 }  
  
  
}

### VCategoriaInsertar

package com.app.emprende2\_2024.view.VCategoria;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.Spinner;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import com.app.emprende2\_2024.R;  
import com.app.emprende2\_2024.controller.CCategoria.CCategoria;  
  
public class VCategoriaInsertar extends AppCompatActivity {  
 EditText etNombre, etDescripcion;  
 Button btnGuardar;  
 Spinner spEstado;  
  
 public EditText getEtNombre() {  
 return findViewById(R.id.etNombreCategoriaInsertar);  
 }  
  
 public EditText getEtDescripcion() {  
 return findViewById(R.id.etDescripcionCategoriaInsertar);  
 }  
  
 public Button getBtnGuardar() {  
 return findViewById(R.id.btnGuardarCategoriaInsertar);  
 }  
  
  
 CCategoria controller = new CCategoria(VCategoriaInsertar.this);  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_vcategoria\_insertar);  
 getBtnGuardar().setOnClickListener(v -> insertar());  
 }  
  
 private void insertar() {  
 String nombre = getEtNombre().getText().toString();  
 String descripcion = getEtDescripcion().getText().toString();  
 controller.create(nombre,descripcion);  
 }  
  
 public void limpiar() {  
 getEtNombre().setText("");  
 getEtDescripcion().setText("");  
 }  
  
  
 public void onBackPressed() {  
 // Inicia la actividad que deseas cuando se presiona el botón de atrás  
 super.onBackPressed();  
 Intent intent = new Intent(this, VCategoriaMain.class);  
 startActivity(intent);  
 finish(); // Opcional, para finalizar la actividad actual si no deseas volver a ella al presionar atrás en la actividad destino  
 }  
}

### VCateogoriaEditar

package com.app.emprende2\_2024.view.VCategoria;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.widget.Button;  
import android.widget.EditText;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import com.app.emprende2\_2024.R;  
import com.app.emprende2\_2024.controller.CCategoria.CCategoria;  
import com.app.emprende2\_2024.model.MCategoria.Categoria;  
  
public class VCategoriaEditar extends AppCompatActivity {  
  
 EditText etNombre, etDescripcion;  
 Button btnGuardar;  
  
 public EditText getEtNombre() {  
 return findViewById(R.id.etNombreCategoriaEditar);  
 }  
  
 public EditText getEtDescripcion() {  
 return findViewById(R.id.etDescripcionCategoriaEditar);  
 }  
  
 public Button getBtnGuardar() {  
 return findViewById(R.id.btnGuardarCategoriaEditar);  
 }  
 int id = 0;  
 CCategoria controller = new CCategoria(VCategoriaEditar.this);  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_vcategoria\_editar);  
 if (savedInstanceState == null){  
 Bundle extras = getIntent().getExtras();  
 if (extras == null)  
 id = Integer.parseInt(null);  
 else  
 id = extras.getInt("ID");  
 }else{  
 id = (int) savedInstanceState.getSerializable("ID");  
 }  
 llenarVista(id);  
 getBtnGuardar().setOnClickListener(v -> update(id));  
 }  
  
 private void update(int id) {  
 String nombre = getEtNombre().getText().toString();  
 String descripcion = getEtDescripcion().getText().toString();  
 controller.update(id,nombre,descripcion);  
 }  
  
 private void llenarVista(int id) {  
 controller.readUno(id);  
 }  
  
 public void readUno(Categoria categoria) {  
 String nombre = categoria.getNombre();  
 String descripcion = categoria.getDescripcion();  
  
 getEtNombre().setText(nombre);  
 getEtDescripcion().setText(descripcion);  
 }  
  
 public void VCategoriaMain() {  
 Intent intent = new Intent(this, VCategoriaMain.class);  
 startActivity(intent);;  
 finish();  
 }  
  
 public void onBackPressed() {  
 // Inicia la actividad que deseas cuando se presiona el botón de atrás  
 super.onBackPressed();  
 Intent intent = new Intent(this, VCategoriaMain.class);  
 startActivity(intent);  
 finish(); // Opcional, para finalizar la actividad actual si no deseas volver a ella al presionar atrás en la actividad destino  
 }  
}

## VPersona

### VPersonaMain

### VPersonaEditar

### VPersonaInsertar

## VProducto

## VProveedor

## VNotaVenta