



# PROYECTO

DISEÑO DE SOFTWARE

UNIVERSIDAD AUTÓNOMA DE YUCATÁN  
FMAT – UMT

PROYECTO  
ABARROTES “EL FORÁNEO”  
TIPO PICK-UP

LICENCIATURA EN INGENIERÍA DE  
SOFTWARE

INTEGRANTES  
BALDERAS PECH ÁNGEL ALEJANDRO  
CANCHE TINTORE JORGE ARMANDO  
LAGUNEZ RODRÍGUEZ LUIS MANUEL  
TEJERO GAMBOA LETICIA DEL CARMEN

PROFR.  
DR. JOSÉ LUIS LÓPEZ MARTÍNEZ

FECHA DE ENTREGA  
23/05/2023

## ÍNDICE

Introducción .....	3
Objetivo .....	5
Vistas del proyecto .....	6
Ejemplo de funcionamiento .....	11
Enlace de muestra de funcionamiento .....	15
Enlace a Git-Hub del código .....	15
Requerimientos funcionales .....	15
Diagramas .....	19
Diagrama de casos de uso	
Log-in Usuario .....	19
Realizar la Compra .....	20
Registro de artículos .....	20
Registro de usuario .....	21
Ticket de compra .....	22
Diagrama de clases	
CLogin .....	23
CRArticulos .....	24
CRCliente .....	25
Objetos .....	26
Artículo DAO .....	27
Cliente Compra .....	28
Diagrama de secuencia .....	29
Diagrama de estados	
Producto .....	30
Orden de compra .....	31
Diagrama de colaboración .....	32
Diagrama de actividad .....	33
Conclusión .....	34

## INTRODUCCIÓN

El presente proyecto se enfoca en el diseño y desarrollo de un sistema de tienda de abarrotes tipo pick up, utilizando diversos patrones de diseño y principios SOLID para garantizar la calidad y mantenibilidad del software. El sistema está diseñado para brindar una experiencia eficiente y cómoda tanto para los usuarios como para los administradores, se ha implementado utilizando Java como lenguaje de programación, JavaFX para las interfaces UX/UI y SQLite como gestor de base de datos.

El sistema ofrece a los usuarios la posibilidad de registrarse o iniciar sesión para acceder a las funcionalidades principales. Una vez que han ingresado, se les presenta un menú desde el cual pueden cambiar su información personal o realizar compras. Al seleccionar la opción de compra, se despliega un menú que permite buscar productos por categoría y agregarlos a una lista de compras. Por otro lado, si el usuario desea actualizar su información personal, se le proporcionan diferentes menús para modificar datos básicos, de dirección y de ciudad, lo que le permite mantener su perfil actualizado en todo momento.

Además, el sistema cuenta con una funcionalidad específica para los administradores. Estos tienen la capacidad de registrar y editar los artículos de venta disponibles en la tienda. Para cada artículo, se pueden ingresar datos como el nombre del producto, el precio al público, el precio del proveedor, la cantidad total en stock, la categoría del producto y el código del mismo. Esta funcionalidad brinda a los administradores un control completo sobre los productos ofrecidos, permitiendo una gestión eficiente y precisa de los artículos disponibles.

Con esta combinación de patrones de diseño como el DAO, MVC y Singleton, se busca lograr un sistema estructurado, modular y fácilmente mantenible. La aplicación de los principios SOLID asegura un diseño orientado a objetos sólido y flexible, lo que facilita futuras mejoras y adaptaciones del sistema. Además, el uso de Java, *JavaFX* y SQLite proporciona una base tecnológica sólida y confiable para el desarrollo de la aplicación.

En resumen, este proyecto de diseño de software de un sistema de tienda de abarrotes tipo pick up ofrece una solución completa y eficiente para la gestión de una tienda de abarrotes. Con su enfoque en los patrones de diseño y principios SOLID, combinado con las tecnologías seleccionadas, se garantiza la calidad, la escalabilidad y la facilidad de mantenimiento del software. A través de interfaces intuitivas y funcionales, tanto los usuarios como los administradores podrán disfrutar de una experiencia fluida y satisfactoria en el manejo de la tienda de abarrotes.

### ***Glosario de palabras***

*SOLID: Es un acrónimo que representa cinco principios de diseño de software: Single Responsibility Principle (Principio de responsabilidad única), Open-Closed Principle (Principio de abierto/cerrado), Liskov Substitution Principle (Principio de sustitución de Liskov), Interface Segregation Principle (Principio de segregación de interfaces) y Dependency Inversion Principle (Principio de inversión de dependencias). Estos principios ayudan a crear código modular, flexible y mantenible.*

*Java: Es un lenguaje de programación de alto nivel y orientado a objetos. Fue desarrollado por Sun Microsystems (ahora propiedad de Oracle) y se utiliza ampliamente en el desarrollo de aplicaciones empresariales, aplicaciones de escritorio, aplicaciones móviles y desarrollo web.*

*JavaFX: Es un conjunto de bibliotecas y herramientas para la creación de interfaces de usuario (UI) ricas y interactivas en Java. Proporciona una forma de construir aplicaciones de escritorio y aplicaciones móviles con una interfaz gráfica de usuario moderna y atractiva.*

*SQLite: Es un motor de base de datos relacional de código abierto y ligero. A diferencia de otros sistemas de gestión de bases de datos, SQLite no funciona como un servidor independiente, sino que se integra directamente con la aplicación. Es ampliamente utilizado en aplicaciones móviles y otras aplicaciones de bajo a mediano tráfico.*

*Interfaces UX/UI: UX (User Experience) se refiere a la experiencia general de un usuario al interactuar con una aplicación, mientras que UI (User Interface) se refiere a la interfaz gráfica con la que el usuario interactúa directamente. Las interfaces UX/UI se centran en diseñar y crear experiencias de usuario intuitivas, atractivas y satisfactorias al interactuar con una aplicación o sitio web.*

*DAO: Es el acrónimo de "Data Access Object" (Objeto de Acceso a Datos). Es un patrón de diseño que se utiliza para separar la lógica de acceso a datos de la lógica de negocio en una aplicación. Proporciona una capa de abstracción entre la lógica de la aplicación y la fuente de datos, lo que facilita el intercambio de la fuente de datos sin afectar la lógica de la aplicación.*

*MVC: Es el acrónimo de "Model-View-Controller" (Modelo-Vista-Controlador). Es un patrón de diseño arquitectónico que separa los componentes de una aplicación en tres partes principales: el modelo (representación de los datos y la lógica de negocio), la vista (presentación visual de la información) y el controlador (gestión de la interacción y coordinación entre el modelo y la vista).*

*Singleton: Es un patrón de diseño creacional que se utiliza para restringir la creación de una clase a una única instancia y proporcionar un punto de acceso global a dicha instancia. El objetivo principal del patrón Singleton es asegurar que una clase solo tenga una instancia y proporcionar un medio para acceder a ella. Esto puede ser útil cuando se necesita tener un punto centralizado de control sobre una única instancia de una clase en toda la aplicación.*

## OBJETIVO

El objetivo de este proyecto es diseñar y desarrollar un sistema de tienda de abarrotes tipo pick up que brinde una experiencia eficiente y cómoda tanto para los usuarios como para los administradores. El sistema se basará en patrones de diseño como el DAO, MVC y Singleton, y aplicará los principios SOLID para garantizar la calidad, escalabilidad y mantenibilidad del software.

1. Proporcionar una plataforma intuitiva y amigable para que los usuarios puedan registrar sus cuentas, iniciar sesión y realizar compras de manera ágil y conveniente.
2. Ofrecer a los usuarios la posibilidad de buscar productos por categoría y agregarlos a una lista de compras, facilitando así la selección y compra de los productos deseados.
3. Permitir a los usuarios actualizar su información personal de manera sencilla, ya sea en cuanto a datos básicos, dirección o ciudad, para mantener sus perfiles actualizados y garantizar una comunicación efectiva.
4. Brindar a los administradores las herramientas necesarias para registrar y editar los artículos de venta, incluyendo información como nombre del producto, precios, cantidad en stock, categoría y código del producto.
5. Implementar patrones de diseño como DAO, MVC y Singleton para garantizar una arquitectura sólida y modular, que facilite la reutilización del código y promueva la mantenibilidad y escalabilidad del sistema a largo plazo.
6. Aplicar los principios SOLID para lograr un diseño orientado a objetos de alta calidad, con una alta cohesión y bajo acoplamiento, facilitando futuras modificaciones y mejoras en el sistema.
7. Utilizar Java como lenguaje de programación principal, aprovechando su robustez y amplia adopción en el desarrollo de software, junto con JavaFX para crear interfaces de usuario modernas y atractivas.
8. Emplear SQLite como gestor de base de datos para garantizar un almacenamiento eficiente y confiable de la información del sistema, asegurando la integridad y disponibilidad de los datos.


En resumen, el objetivo de este proyecto es diseñar y desarrollar un sistema de tienda de abarrotes tipo pick up que cumpla con los requisitos funcionales y no funcionales establecidos, proporcionando una experiencia satisfactoria tanto para los usuarios como para los administradores. Se busca aplicar los mejores estándares y prácticas de diseño de software, utilizando patrones de diseño, principios SOLID y tecnologías adecuadas para garantizar un sistema de calidad, escalable y fácilmente mantenible.

## VISTAS DEL PROYECTO

Vistas hechas en Figma: <https://www.figma.com/file/ArE6ltPrtSIMO6snGMand8/Untitled?type=design&node-id=0%3A1&t=tYKP8l1JsJe2F3nJ-1>


Vistas (Scene Builder):

Vista para ingresar como usuario o administrador.



A login form design with a light teal background. At the top center is a green circular profile icon. Below it are two tabs: 'Ingresa' (highlighted in teal) and 'Regístrate' (highlighted in yellow). The form contains two input fields: 'Correo' with the placeholder text 'example@domain' and 'Contraseña'. At the bottom are two dark blue buttons: 'Ingresar' and 'Salir'.

Vista para registrarse como usuario.



A registration form design with a yellow background. At the top center is a green circular profile icon. Below it are two tabs: 'Ingresa' (highlighted in teal) and 'Regístrate' (highlighted in yellow). The form contains three input fields: 'Correo' with the placeholder text 'example@domain', 'Contraseña', and 'Repetir Contraseña'. At the bottom are two dark blue buttons: 'Continuar' and 'Salir'.

Vista para registrar y editar artículos, solo para administrador.

# Registro y Edición de Artículos

Artículos [Volver](#)

Nombre

Precio al público

Precio del proveedor

Cant. total (existencia)

Categoría 

S/C

Código Artículo

Registrar

Buscar

Codigo	Nombre	Precio	Exist.
Tabla sin contenido			

Actualizar

Eliminar

Vista de menú para clientes, puede elegir entre iniciar una compra o actualizar su información personal, de contacto y dirección.


# nombre\_cliente

Menú [Volver](#)

Información

Comprar

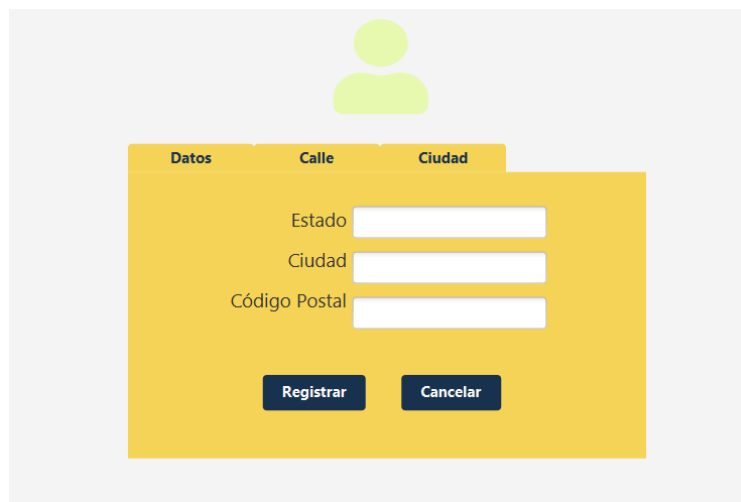
Vista para proporcionar la información personal, de contacto y dirección del usuario al momento del registro.



A user registration form with a light gray background. At the top center is a green silhouette of a person's head and shoulders. Below it is a yellow form with three tabs: 'Datos', 'Calle', and 'Ciudad'. The 'Datos' tab is selected. Inside the form, there are three white input fields labeled 'Nombre', 'Apellido Paterno', and 'Teléfono'. At the bottom of the form are two dark blue buttons: 'Continuar' and 'Cancelar'.



A user registration form with a light gray background. At the top center is a green silhouette of a person's head and shoulders. Below it is a yellow form with three tabs: 'Datos', 'Calle', and 'Ciudad'. The 'Calle' tab is selected. Inside the form, there are three white input fields labeled 'Calle', 'Número', and 'Colonia'. At the bottom of the form are two dark blue buttons: 'Continuar' and 'Cancelar'.



A user registration form with a light gray background. At the top center is a green silhouette of a person's head and shoulders. Below it is a yellow form with three tabs: 'Datos', 'Calle', and 'Ciudad'. The 'Ciudad' tab is selected. Inside the form, there are three white input fields labeled 'Estado', 'Ciudad', and 'Código Postal'. At the bottom of the form are two dark blue buttons: 'Registrar' and 'Cancelar'.



Vista para editar la información personal, de contacto y dirección del usuario.

example@domain

Cientes

Volver

DatosCalleCiudad



Correo

Contraseña

Repetir contraseña

Nombre

Apellido Paterno

Telefono


Actualizar

example@domain

Cientes

Volver

DatosCalleCiudad



Calle

Numero

Colonia


Actualizar

example@domain

Cientes

Volver

DatosCalleCiudad



Estado

Ciudad

Codigo Postal

Actualizar

Vista para usuario para realizar una compra.

# Comprar

Compras Volver



Artículos a comprar

Nombre	Precio	Cantidad	Total
Tabla sin contenido			

Eliminar Artículo

Total:0000.00

Categoria 

Categoria ▼

 Buscar

Nombre	Precio	Exist.
Tabla sin contenido		

Cantidad  Añadir

## EJEMPLO DE FUNCIONAMIENTO

Funcionamiento de administrador.



Interface for user login and registration. It features a green header with a user icon and two tabs: 'Ingresa' (selected) and 'Regístrate'. Below the tabs are input fields for 'Correo' (admin@abarotes.com) and 'Contraseña' (masked with dots). At the bottom are 'Ingresar' and 'Salir' buttons.



Interface for 'Registro y Edición de Artículos'. It includes a green header with 'Artículos' and a 'Volver' button. A search bar is at the top right. The main form has fields for 'Nombre', 'Precio al público', 'Precio del proveedor', 'Cant. total (existencia)', 'Código Artículo', and 'Categoría'. The 'Categoría' dropdown is open, showing options: '✓ S/C', 'LIMPIEZA', 'BOTANA', 'COMIDA', 'FRUTA/VERDURA', and 'BEBIDA'. A table on the right shows 'Tabla sin contenido'. At the bottom right are 'Actualizar' and 'Eliminar' buttons.



Interface for 'Registro y Edición de Artículos' with the form filled out. The 'Nombre' field contains 'Jugo narajna "Del Valle"', 'Precio al público' is 18, 'Precio del proveedor' is 14, 'Cant. total (existencia)' is 20, 'Categoría' is 'BEBIDA', and 'Código Artículo' is 1234. The 'Registrar' button is at the bottom left. The table on the right still shows 'Tabla sin contenido'. The 'Actualizar' and 'Eliminar' buttons are at the bottom right.

## Registro y Edición de Artículos

Artículos

[Volver](#)



Buscar

Nombre

Precio al público

Precio del proveedor

Cant. total (existencia)

Categoría

Código Artículo

[Registrar](#)

Codigo	Nombre	Precio	Exist.
140716	Jugo narajn...	18.0	20

[Actualizar](#)

[Eliminar](#)

## Registro y Edición de Artículos

Artículos

[Volver](#)



Buscar

Nombre

Precio al público

Precio del proveedor

Cant. total (existencia)

Categoría

Código Artículo

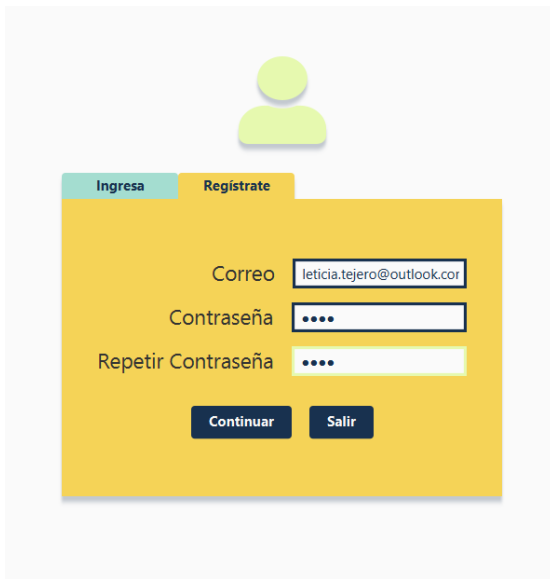
[Registrar](#)

Codigo	Nombre	Precio	Exist.
750107130...	Frijol Negro...	22.0	30
750105532...	Refresco Co...	19.5	30
807680019...	Espaguetis ...	28.5	30
1234567855	Frijor Sierra ...	23.0	18
140716	Jugo narajn...	18.0	20

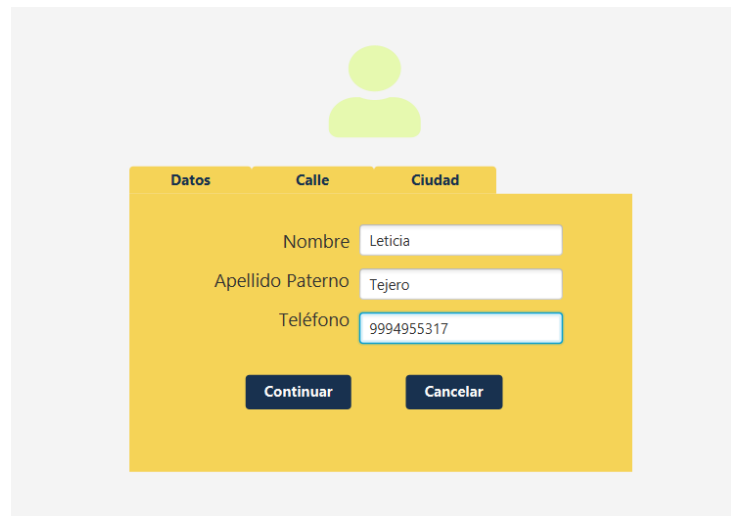
[Actualizar](#)

[Eliminar](#)

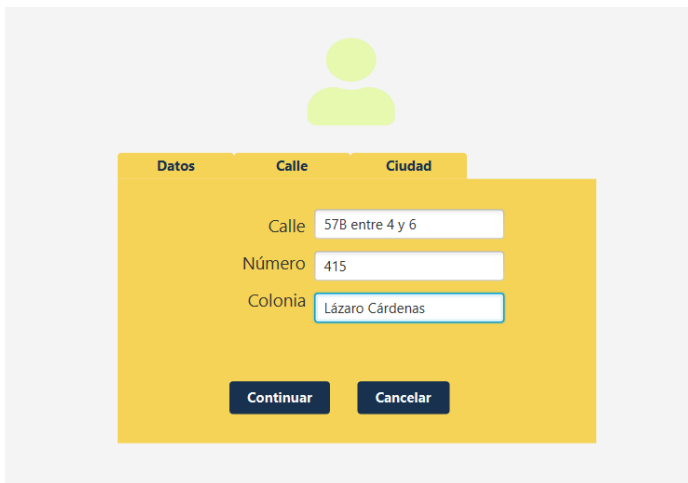
## Funcionamiento de usuario.



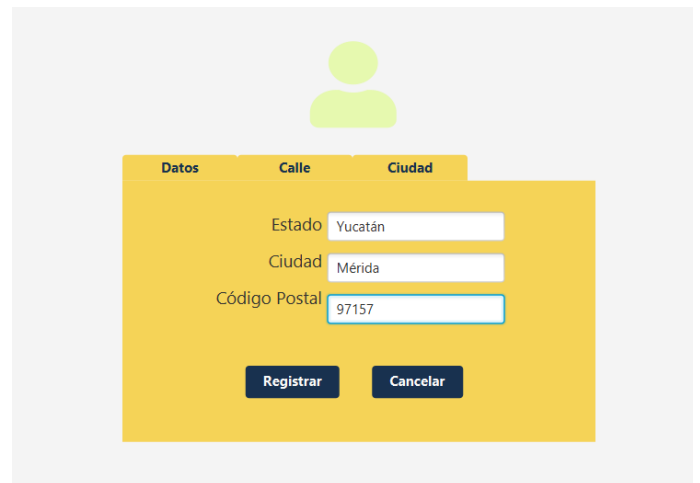
Registration form (Step 1) with tabs "Ingresa" and "Regístrate". Fields include "Correo" (leticia.tejero@outlook.cor), "Contraseña" (masked), and "Repetir Contraseña" (masked). Buttons: "Continuar", "Salir".



Registration form (Step 2) with tabs "Datos", "Calle", and "Ciudad". Fields include "Nombre" (Leticia), "Apellido Paterno" (Tejero), and "Teléfono" (9994955317). Buttons: "Continuar", "Cancelar".



Registration form (Step 3) with tabs "Datos", "Calle", and "Ciudad". Fields include "Calle" (57B entre 4 y 6), "Número" (415), and "Colonia" (Lázaro Cárdenas). Buttons: "Continuar", "Cancelar".



Registration form (Step 4) with tabs "Datos", "Calle", and "Ciudad". Fields include "Estado" (Yucatán), "Ciudad" (Mérida), and "Código Postal" (97157). Buttons: "Registrar", "Cancelar".



Login form with tabs "Ingresa" and "Regístrate". Fields include "Correo" (leticia.tejero@outlook.cor) and "Contraseña" (masked). Buttons: "Ingresar", "Salir".



User dashboard for "Leticia Tejero". Header: "Leticia Tejero", "Menú", "Volver". Main area: "Información" (person icon) and "Comprar" (shopping cart icon).

leticia.tejero@outlook.c...
Clientes
Volver

Datos
Calle
Ciudad



Correo
leticia.tejero.g@outlook.com

Contraseña

Repetir contraseña

Nombre
Leticia

Apellido Paterno
Tejero

Telefono
9994955317

Actualizar

Comprar
Compras
Volver



Artículos a comprar

Categoria
Categoria
Buscar
1234

Nombre	Precio	Cantidad	Total
Tabla sin contenido			

Nombre	Precio	Exist.
Tabla sin contenido		

Eliminar Artículo

Total:0000.00

Cantidad
Añadir

## ENLACE A VIDEO DEMOSTRACIÓN DE FUNCIONAMIENTO

<https://youtu.be/PeDxDimoRWA>

## ENLACE A GIT-HUB DEL CÓDIGO

<https://github.com/a-bals21/Abarrotes-PF>

## REQUERIMIENTOS FUNCIONALES

RF001	Registro de usuario.
<b>Descripción</b>	El sistema pedirá los datos correspondientes para registrar un nuevo usuario dentro del software y este se almacenará utilizando como persistencia de datos en el gestor SQLite donde se almacenará la información de los usuarios.
<b>Prioridad</b>	Alta
<b>Entradas</b>	<ul style="list-style-type: none"><li>• Nombre, Apellido Paterno, Correo Electrónico, Contraseña, Calle, Número, Colonia, Código Postal, Ciudad, Estado, Teléfono.</li></ul>
<b>Salidas</b>	<ul style="list-style-type: none"><li>• Mensaje de error en el caso de no haber llenado algún campo.</li><li>• Mensaje de error en caso de ingresar incorrectamente los datos, es decir que el formato de los datos sea incorrecto.</li><li>• Mensaje que despliegue una ventana de registro exitoso en caso de haber rellenado todos los campos de manera correcta.</li></ul>
<b>Información adicional</b>	<p>El sistema debe proporcionar un ID al usuario para su registro que será almacenado junto a la hoja de información del software.</p> <p>El usuario al registrarse será redirigido al log-in del sistema, es decir, a la interfaz de cliente.</p>

RF002	Log-in de usuario.
<b>Descripción</b>	El sistema pedirá los datos de entrada necesarios para realizar el inicio de sesión dentro del software, dicha información será validada desde el gestor de datos.
<b>Prioridad</b>	Alta
<b>Entradas</b>	<ul style="list-style-type: none"> <li>• Correo electrónico.</li> <li>• Contraseña.</li> </ul>
<b>Salidas</b>	<ul style="list-style-type: none"> <li>• Mensaje de error en el caso de no haber llenado algún campo.</li> <li>• Mensaje de error en caso de ingresar incorrectamente los datos, es decir que el formato de los datos sea incorrecto.</li> <li>• Mensaje que despliegue una ventana de registro exitoso en caso de haber rellenado todos los campos de manera correcta.</li> </ul>
<b>Información adicional</b>	<p>El sistema debe proporcionar un ID al usuario para su registro que será almacenado junto a la hoja de información del software.</p> <p>El usuario al registrarse será redirigido al log-in del sistema, es decir, a la interfaz de cliente.</p>

RF003	Registro de artículos.
<b>Descripción</b>	El sistema pedirá los datos correspondientes para registrar un nuevo artículo dentro del software y este se almacenará utilizando como persistencia de datos en el gestor SQLite donde se almacenará la información de los artículos.
<b>Prioridad</b>	Alta
<b>Entradas</b>	<ul style="list-style-type: none"> <li>• ID, Nombre artículo, precio al público, precio del proveedor, total en existencia.</li> </ul>
<b>Salidas</b>	<ul style="list-style-type: none"> <li>• Mensaje de error en el caso de no haber llenado algún campo.</li> <li>• Mensaje de error en caso de ingresar incorrectamente los datos, es decir que el formato de los datos sea incorrecto.</li> <li>• Mensaje que despliegue una ventana de registro exitoso en caso de haber rellenado todos los campos de manera correcta.</li> </ul>



<b>Información adicional</b>	
------------------------------	--

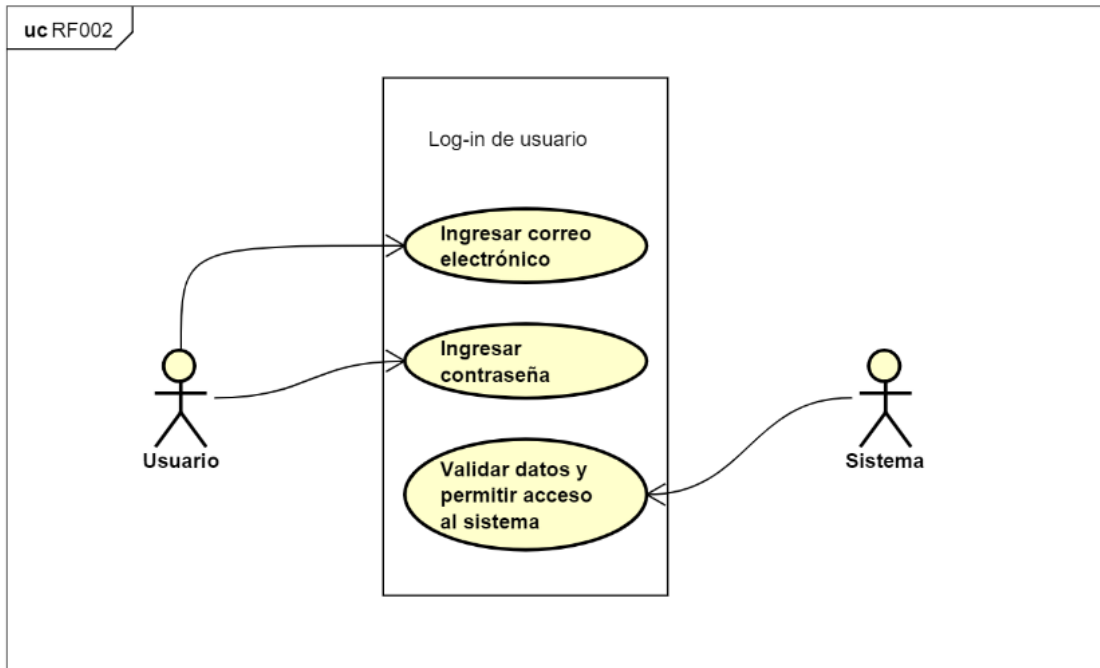
RF004	Realizar compra.
<b>Descripción</b>	El sistema permitirá que el usuario pueda seleccionar los artículos que decida comprar y almacenarlos en un carrito de compras siempre y cuando estos se encuentren con disponibilidad en stock.
<b>Prioridad</b>	Alta
<b>Entradas</b>	<ul style="list-style-type: none"> <li>• Artículos por seleccionar.</li> </ul>
<b>Salidas</b>	<ul style="list-style-type: none"> <li>• Precio unitario de los artículos.</li> <li>• Sumatoria del precio unitario de artículos.</li> <li>• Importe por pagar (incluyendo IVA).</li> </ul>
<b>Información adicional</b>	El número de artículos en stock debe ser visible al usuario, en caso de sobrepasar el límite se deberá desplegar un mensaje de advertencia.

RF005	Ticket de compra.
<b>Descripción</b>	El sistema permitirá desplegar al usuario un ticket donde se encuentre la información de compra que realizó.
<b>Prioridad</b>	Alta
<b>Entradas</b>	
<b>Salidas</b>	<ul style="list-style-type: none"> <li>• Nombre de la tienda.</li> <li>• Dirección de la tienda.</li> <li>• Información fiscal de la tienda.</li> <li>• Número de Ticket.</li> <li>• Fecha de compra.</li> <li>• Nombre del cliente.</li> <li>• Precio unitario de los artículos.</li> <li>• Precio total de artículos.</li> <li>• Importe por pagar (incluyendo IVA).</li> </ul>
<b>Información adicional</b>	De preferencia este archivo deberá guardarse con extensión .pdf.

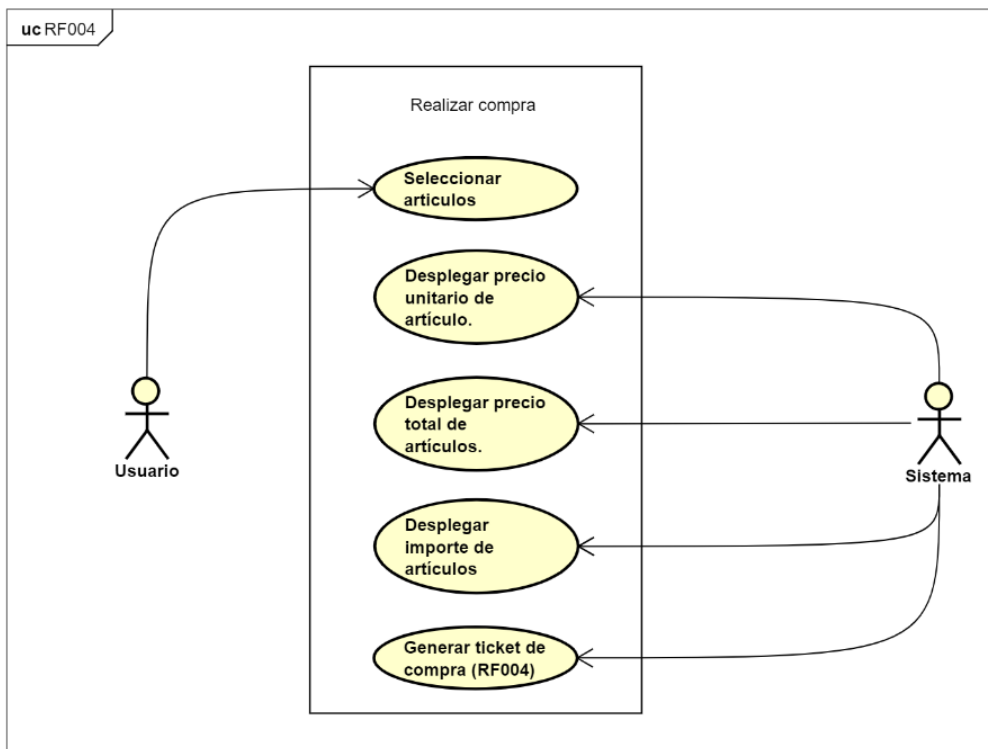
## DIAGRAMAS

### Diagramas de caso de usos

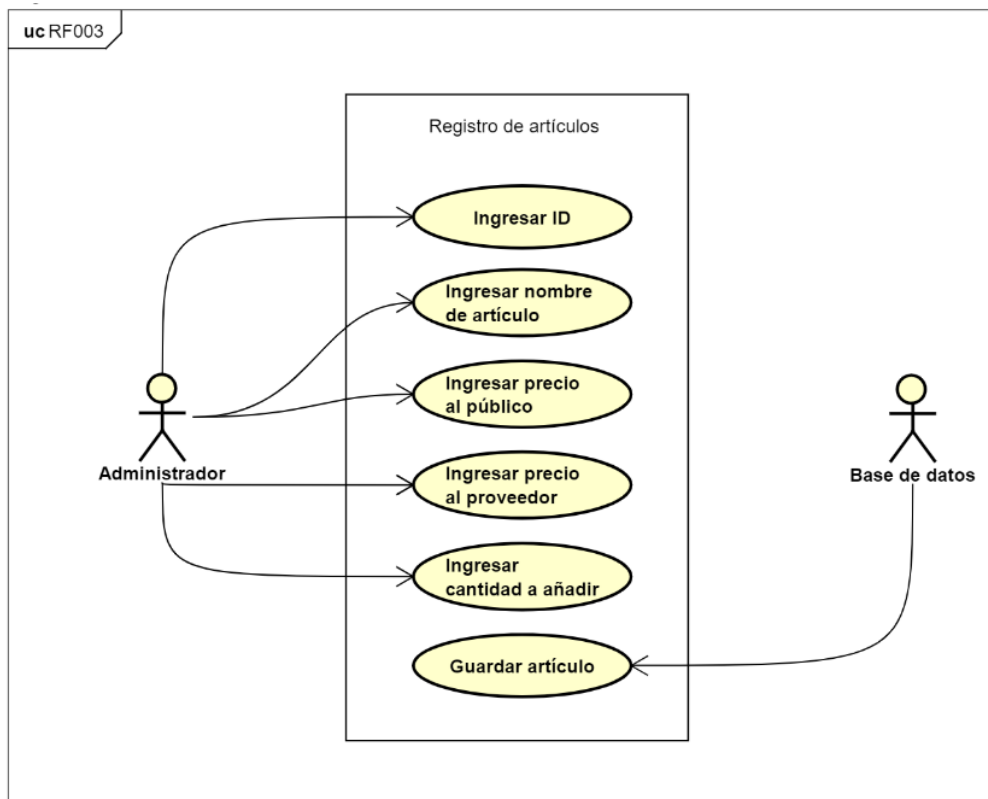
#### Log-in usuario



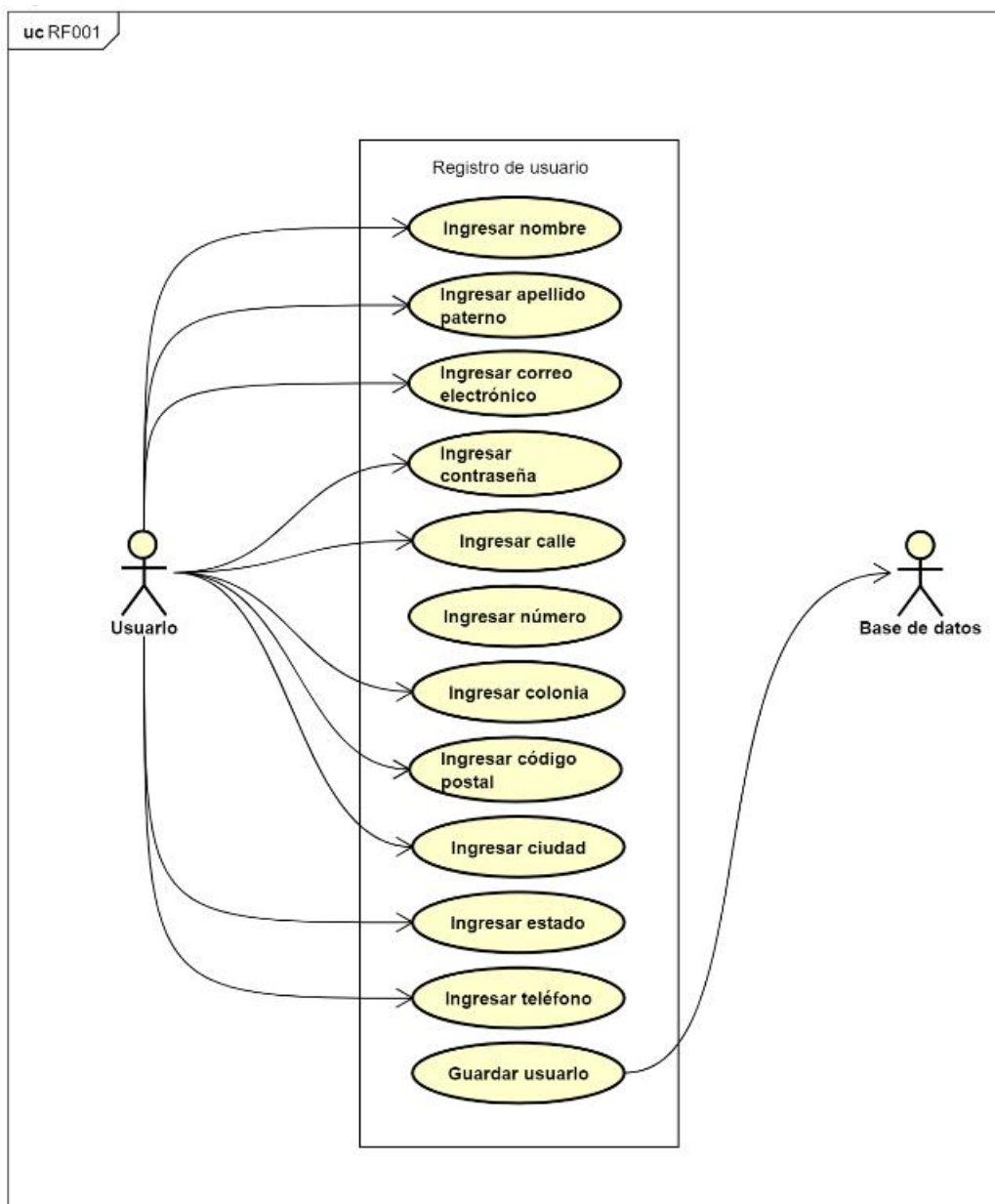
## Realizar la compra



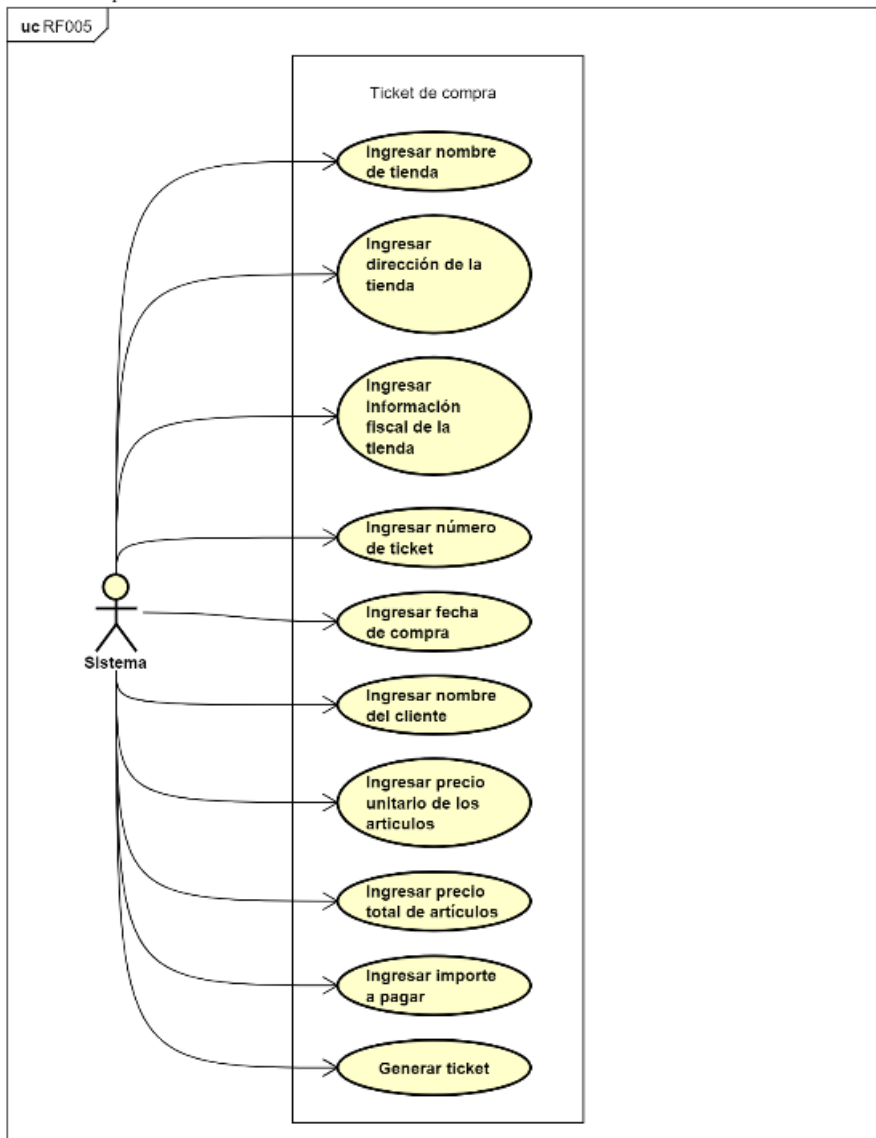
## Registro de artículos



## Registro de usuario

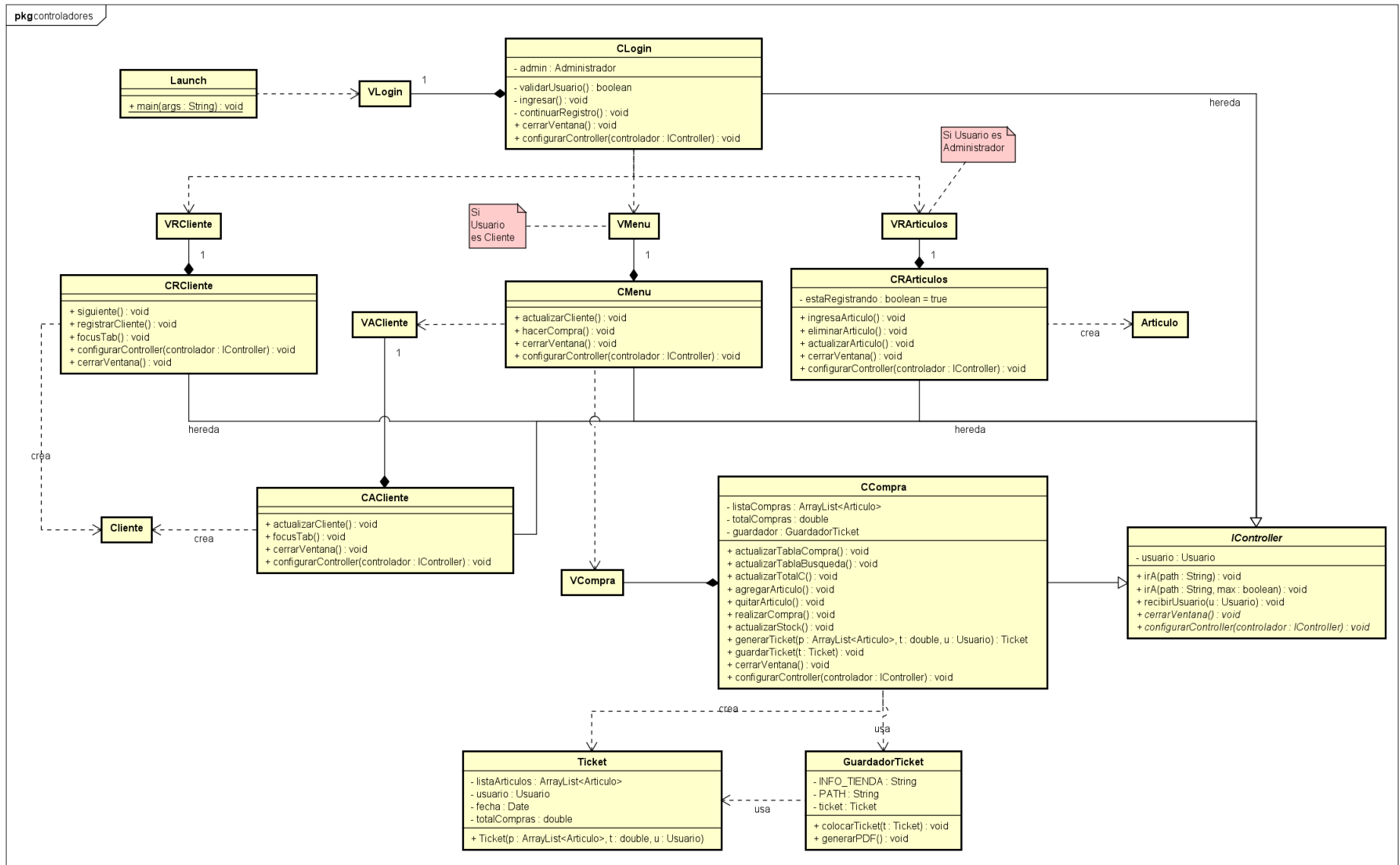


## Ticket de compra

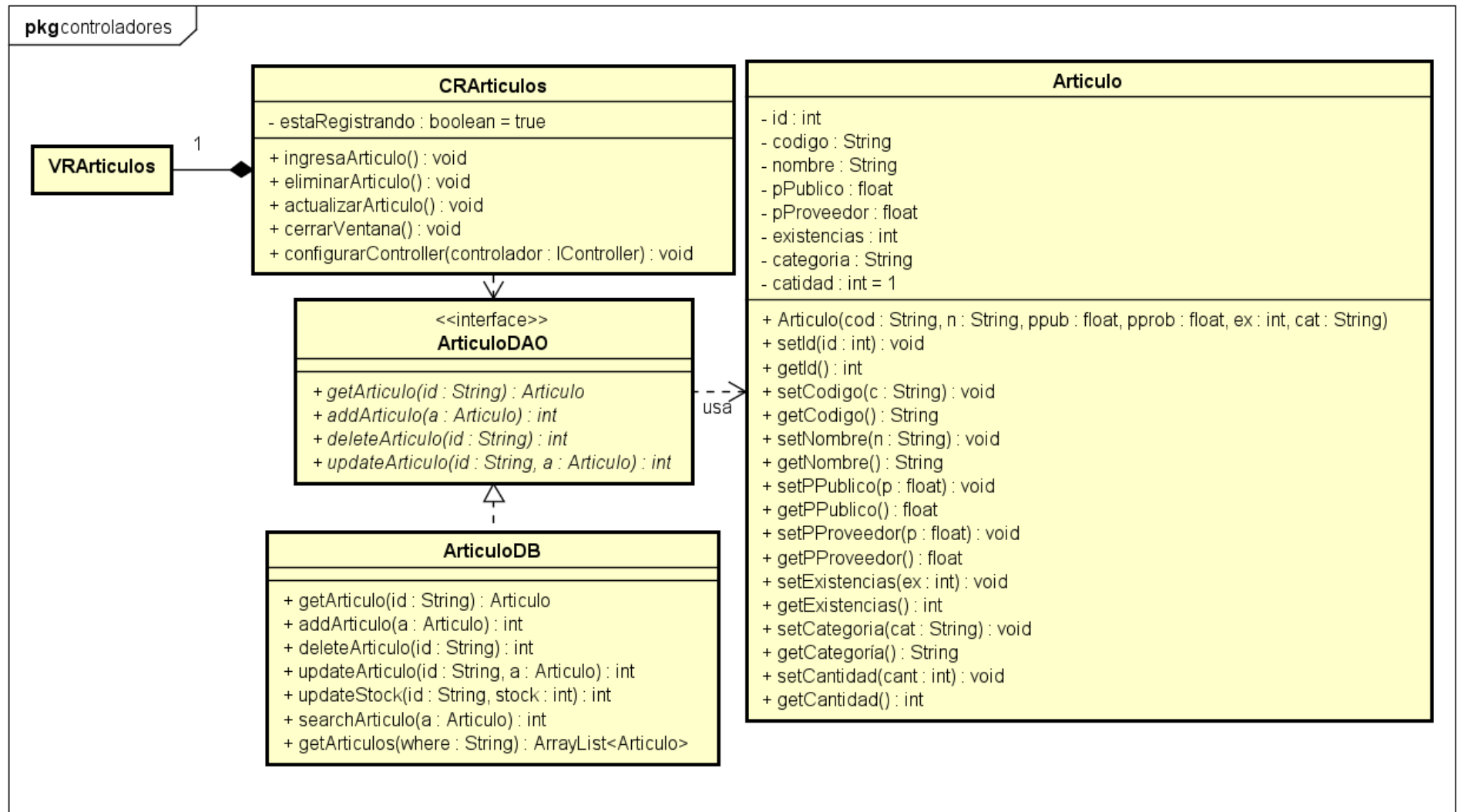


## Diagramas de clases

### CLogin

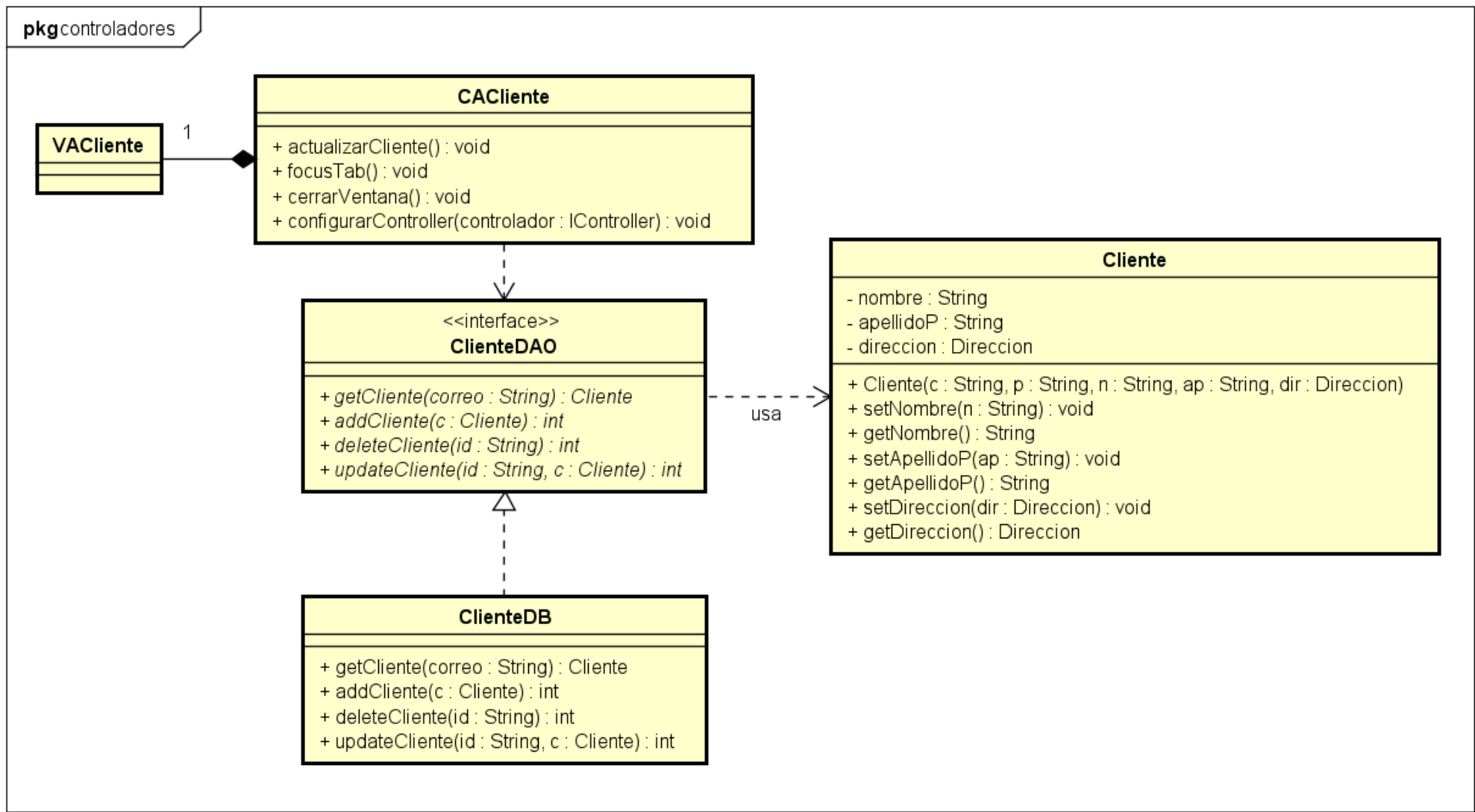


## CRArticulos

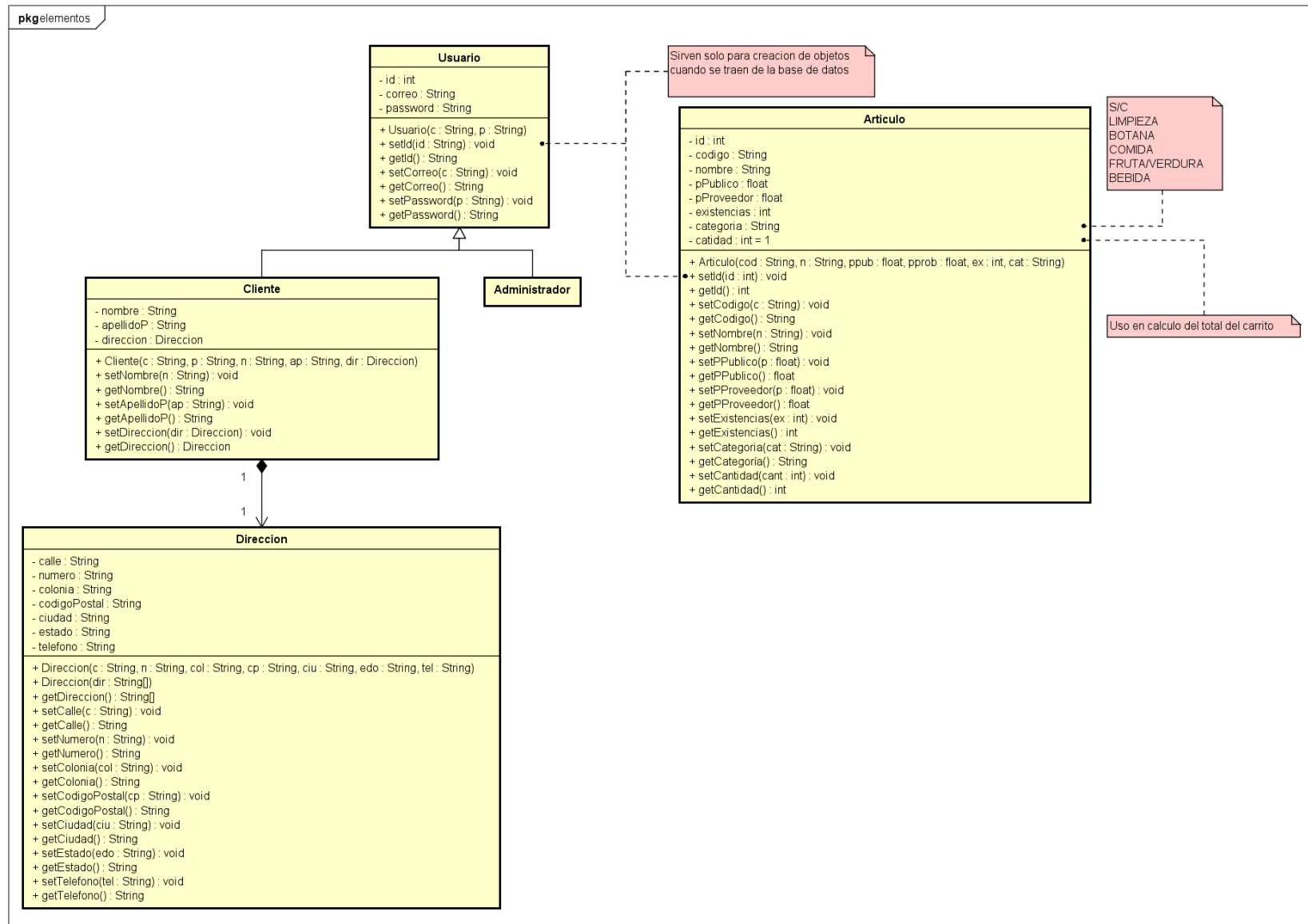




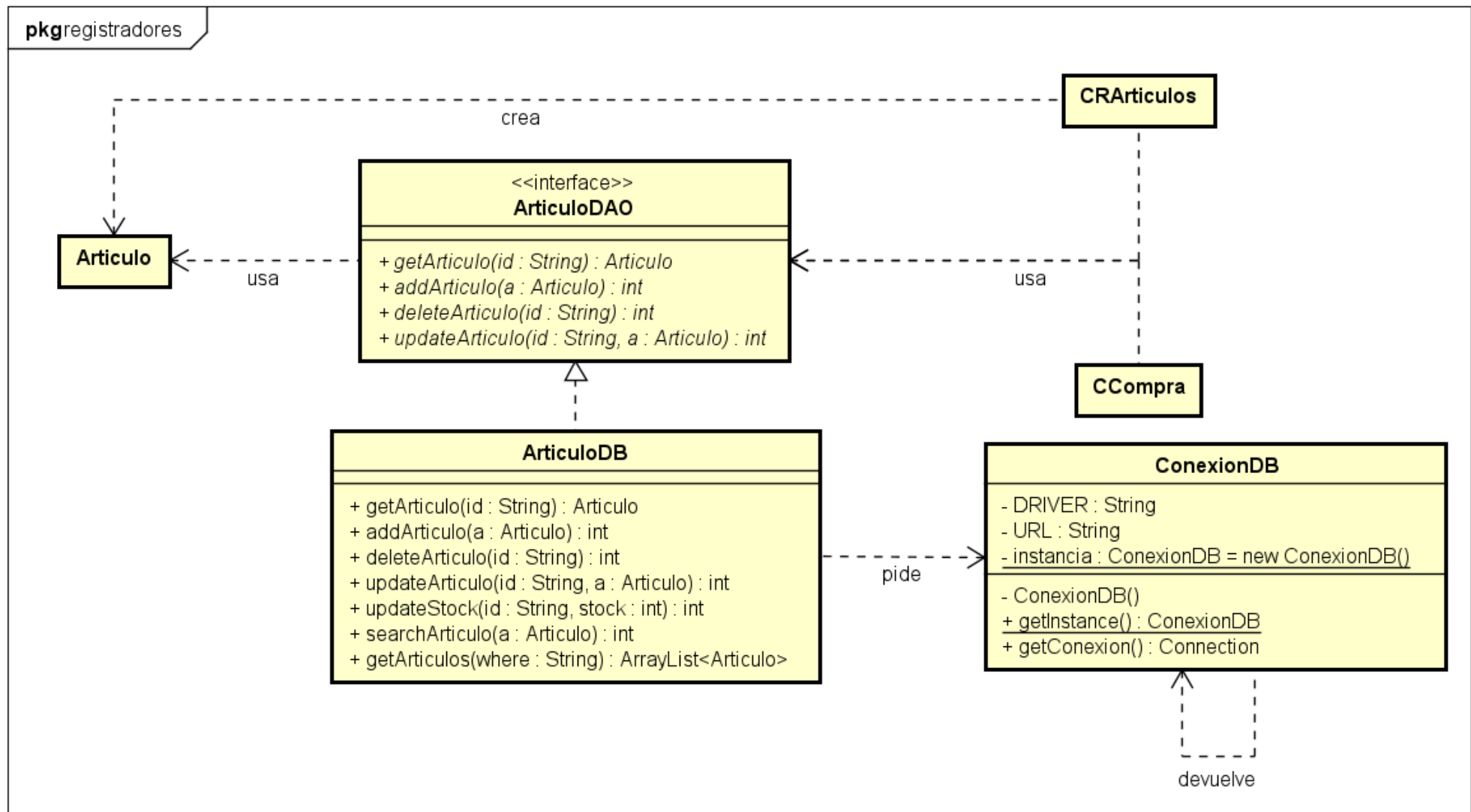
## CRCliente



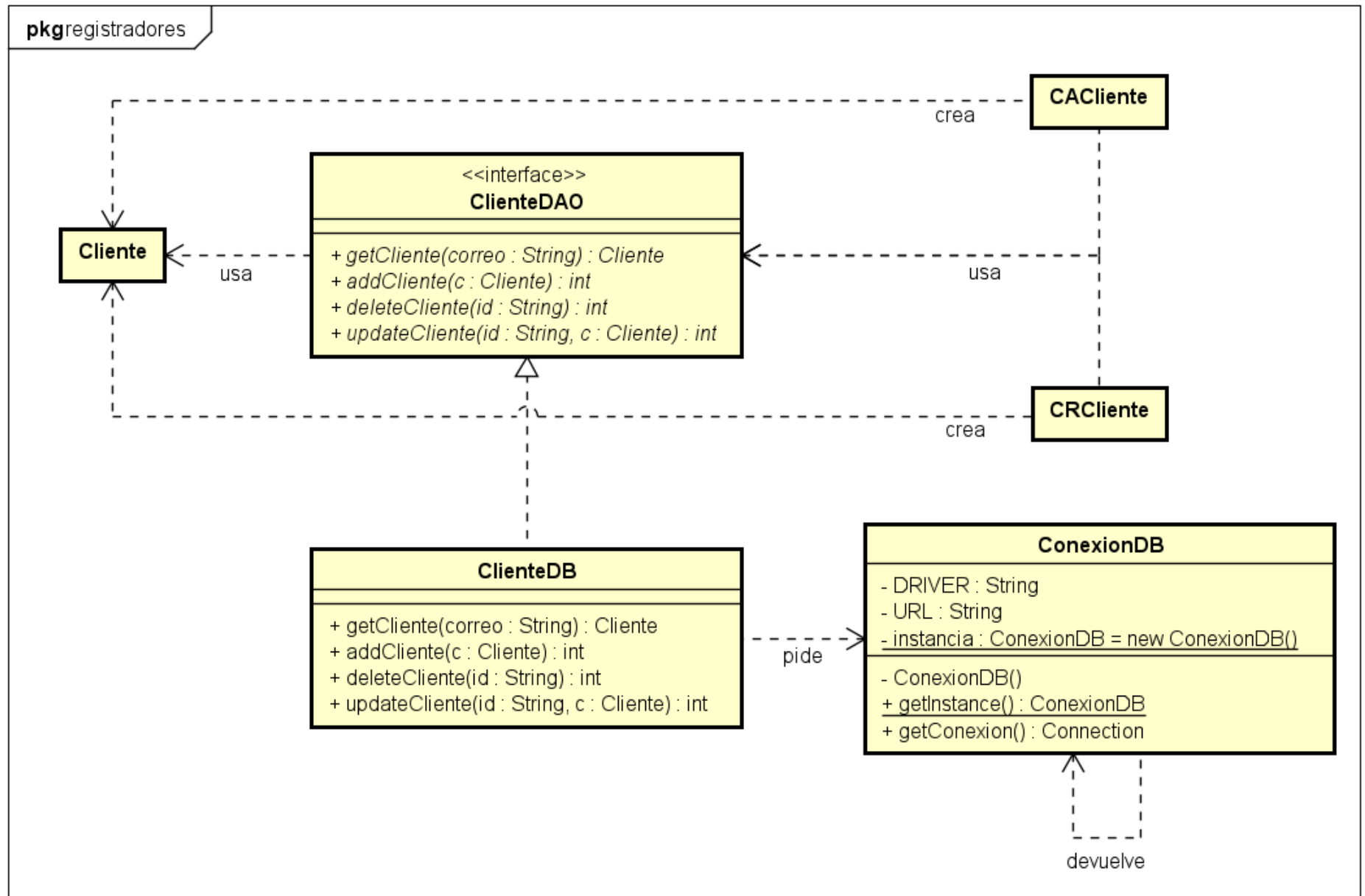
## Objetos



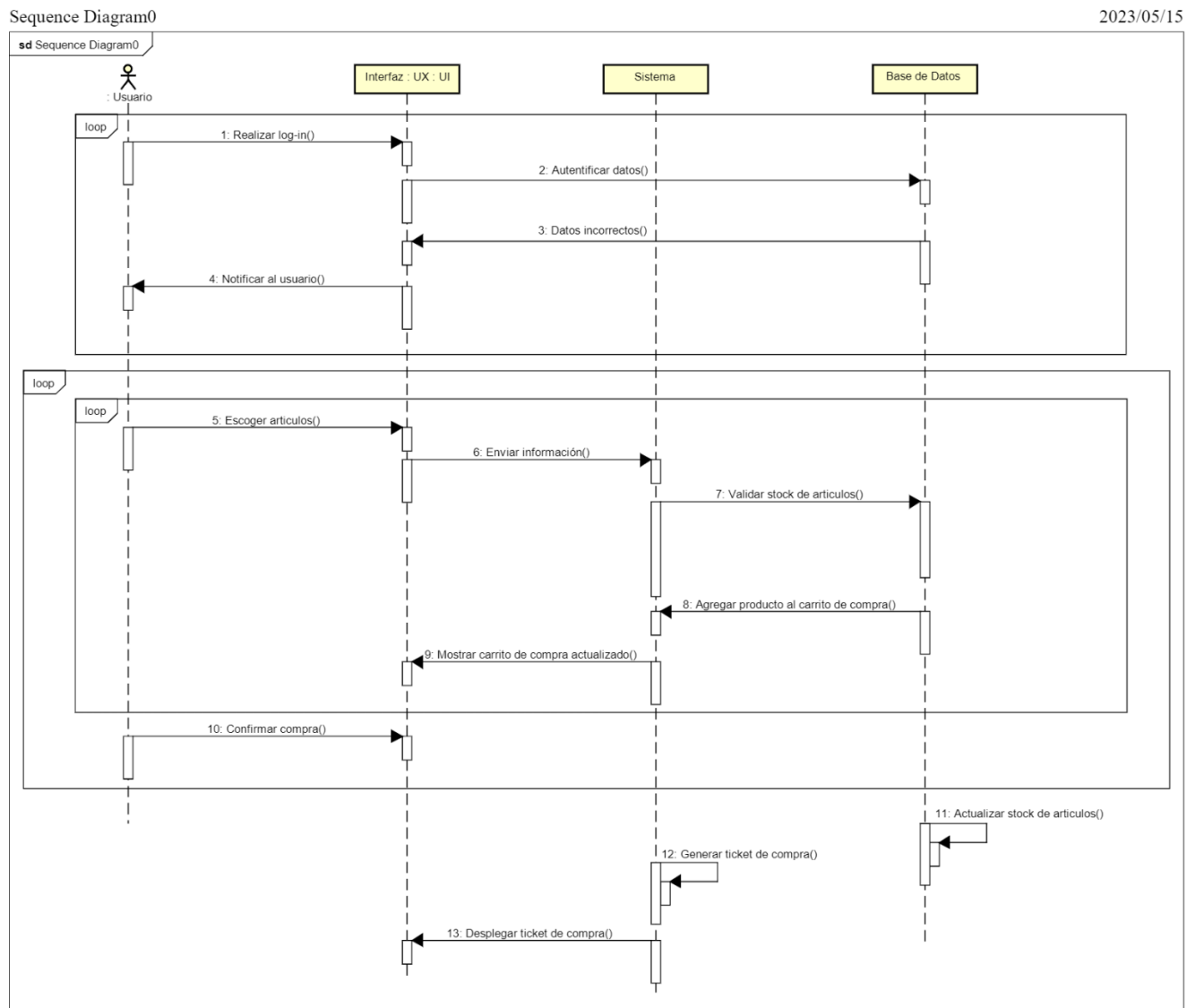
## Artículo DAO



## Cliente DAO

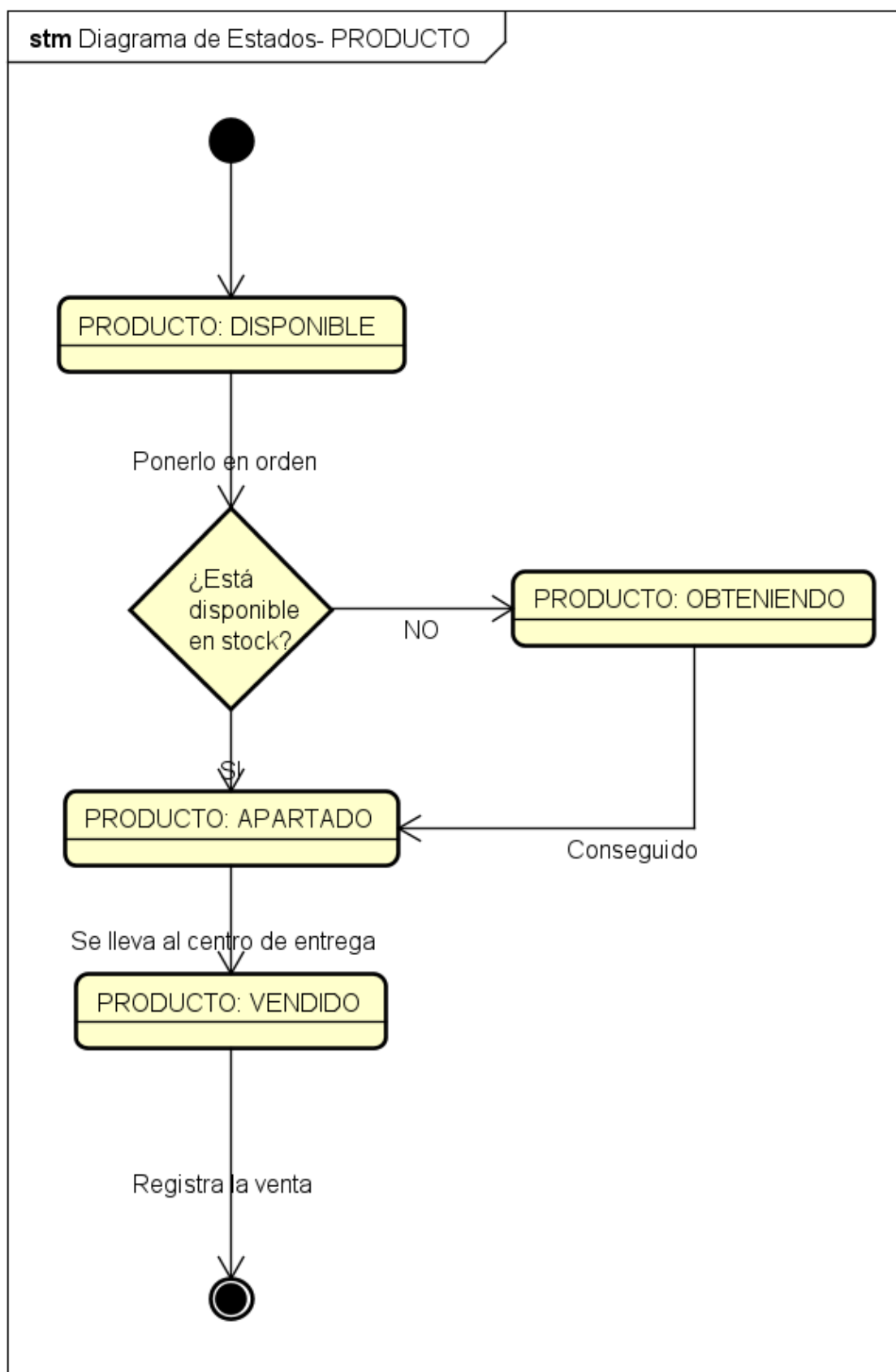


## Diagrama de secuencia

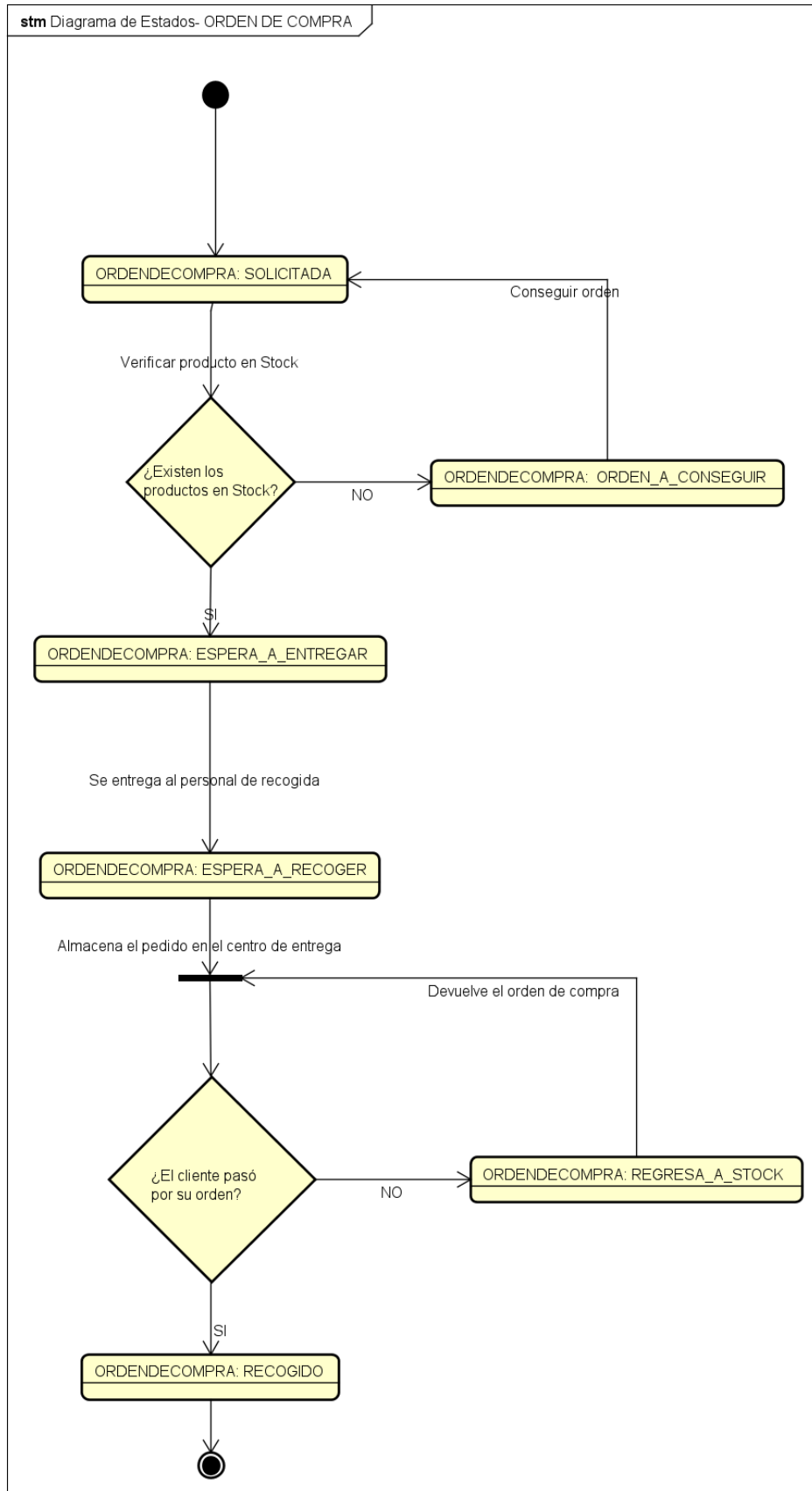


## Diagrama de estado

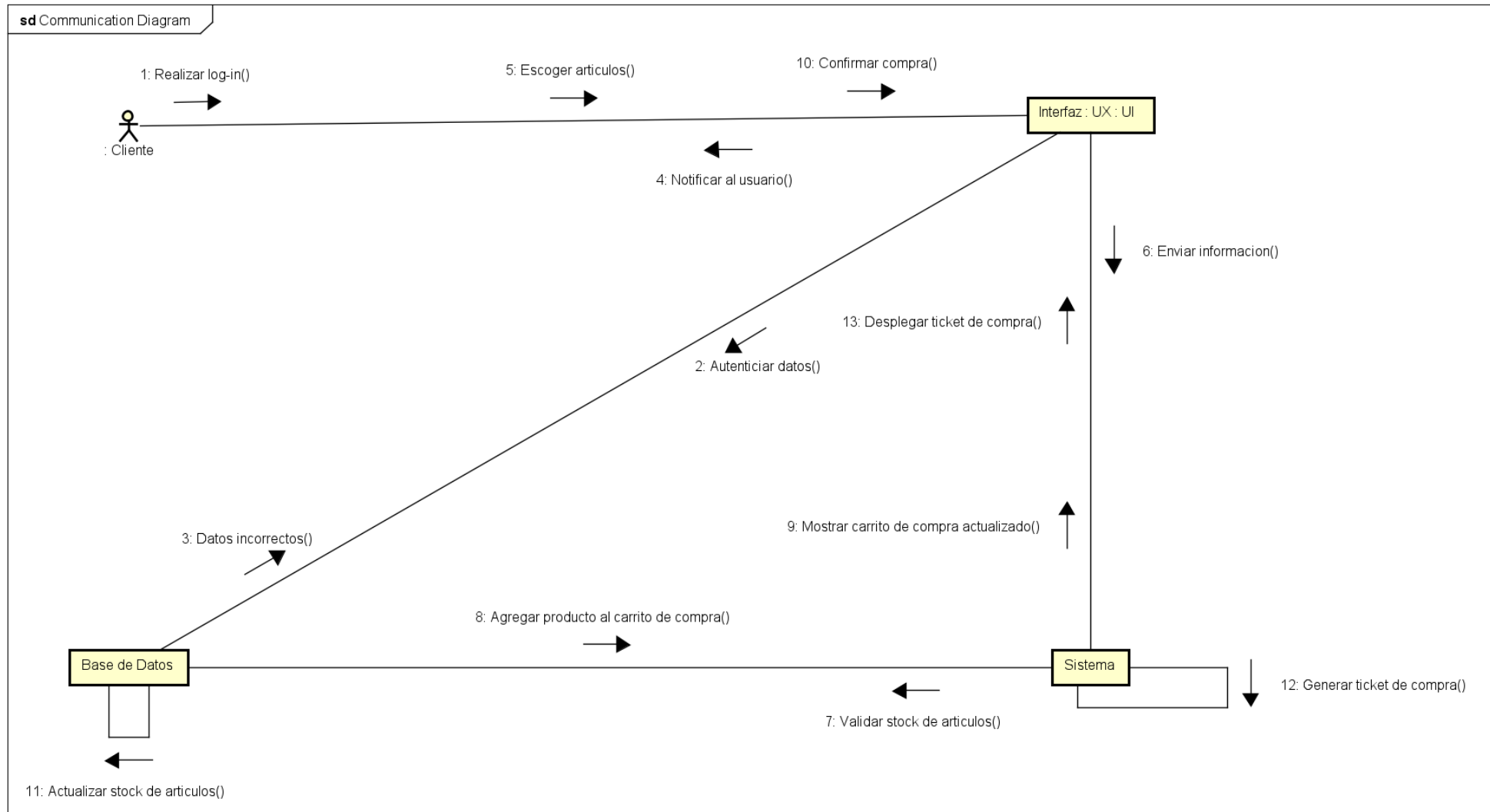
Producto



## Orden de compra

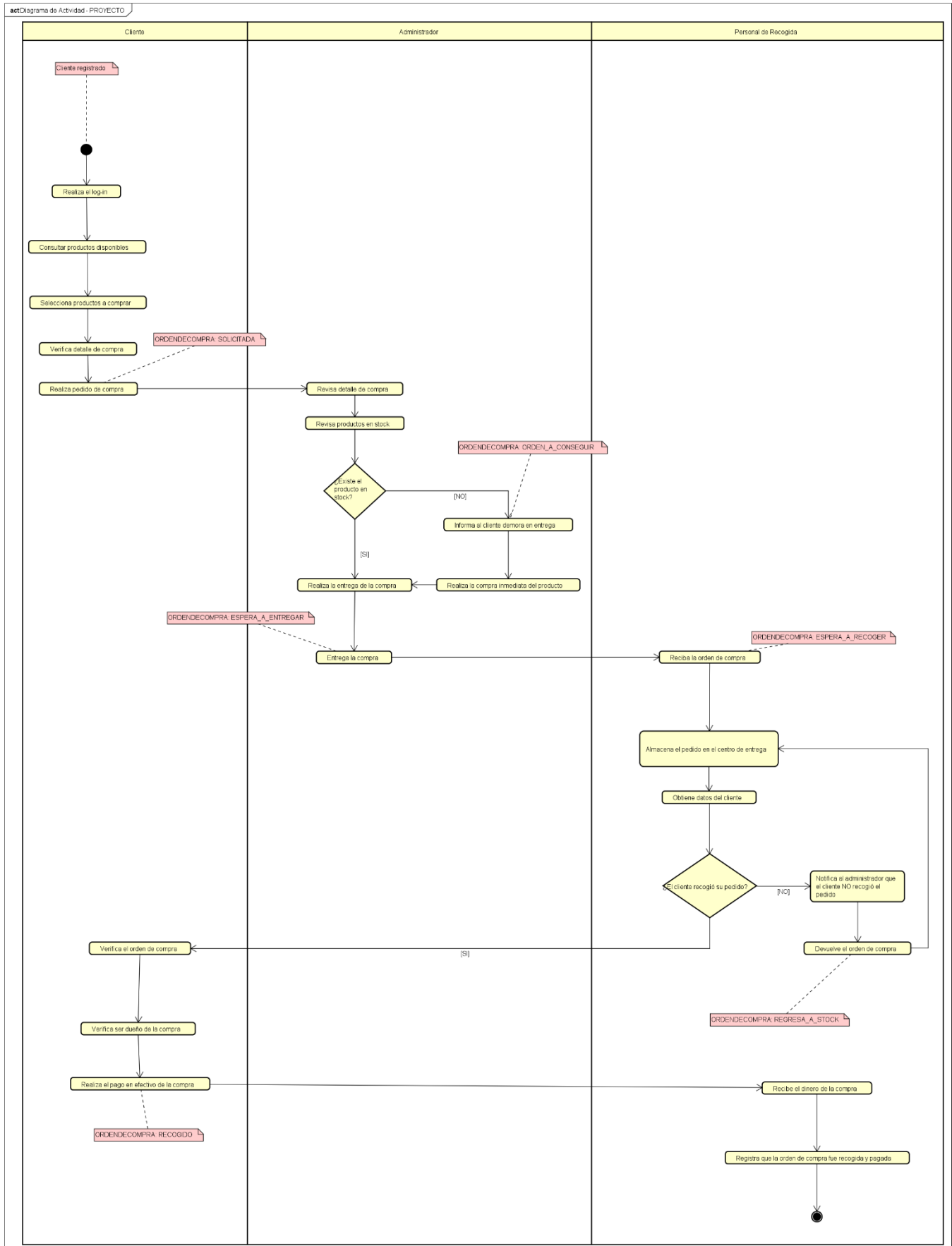


## Diagrama de colaboración





## Diagrama de actividad



## CONCLUSIONES

En conclusión, el desarrollo de este proyecto de diseño de software para un sistema de tienda de abarrotes tipo pick up ha sido exitoso y ha logrado cumplir con los objetivos planteados. El uso de patrones de diseño como el DAO, MVC y *Singleton* ha permitido una estructura modular y flexible, facilitando la reutilización de código y la mantenibilidad del sistema. Asimismo, la aplicación de los principios SOLID ha garantizado un diseño orientado a objetos sólido y de alta calidad.

El sistema ofrece a los usuarios la posibilidad de registrarse, iniciar sesión y realizar compras de manera fácil y cómoda. Los menús intuitivos y la posibilidad de modificar la información personal proporcionan una experiencia de usuario fluida y personalizada. Por otro lado, los administradores tienen el control completo sobre los productos de venta, lo que les permite gestionar de manera eficiente el inventario y asegurar una oferta actualizada y atractiva para los clientes.

El uso de Java como lenguaje de programación principal, junto con JavaFX para las interfaces UX/UI y SQLite como gestor de base de datos, ha demostrado ser una combinación sólida y confiable para el desarrollo de la aplicación. Estas tecnologías ofrecen una gran compatibilidad, rendimiento y escalabilidad, lo que garantiza el funcionamiento óptimo del sistema incluso en entornos de alto tráfico y demanda.

En resumen, este proyecto de diseño de software ha logrado crear un sistema de tienda de abarrotes tipo pick up eficiente y de calidad, que satisface las necesidades de los usuarios y administradores. El enfoque en los patrones de diseño, los principios SOLID y las tecnologías seleccionadas ha permitido obtener un sistema robusto, escalable y fácilmente mantenible. Este proyecto sienta las bases para futuras mejoras y ampliaciones, brindando a la tienda de abarrotes una herramienta poderosa para su gestión y éxito continuo.