UNIVERSIDAD AUTÓNOMA GABRIEL RENÉ MORENO

FACULTAD DE INGENIERÍA EN CIENCIAS DE LA COMPUTACIÓN Y TELECOMUNICACIONES

Sistema de Punto de Venta (POS) con Inteligencia Artificial para el Reconocimiento de Voz y Recomendaciones de productos a comprar

Docente: Ing. Garzón Cuellar Angélica

Materia: Sistemas de Información II - SA

Integrantes: GRUPO# 37

- Moreno Montenegro Paul 220001960
- Jesus Anthony Bruno Stroebel 220000581

SANTA CRUZ – BOLIVIA

ÍNDICE

Contenido

1. Perfil

1.1 Introducción

En la actualidad, las soluciones tecnológicas están revolucionando la forma en que las personas realizan sus compras, optimizando tiempo, presupuesto y experiencia de usuario. En este contexto, nace Smart Cart, una plataforma diseñada para convertirse en el asistente de compras inteligente de los consumidores.

Smart Cart surge como respuesta a la necesidad de muchos usuarios de contar con una herramienta que les permita planificar, organizar y ejecutar sus compras de manera eficiente. A diferencia de las tiendas en línea tradicionales, esta aplicación no se limita únicamente a mostrar productos, sino que ofrece una experiencia personalizada, orientada al ahorro, la organización y la conveniencia.

El objetivo principal de Smart Cart es ayudar a los usuarios a gestionar su lista de compras, acceder a productos de diferentes tiendas, recibir recomendaciones inteligentes y comparar precios entre supermercados o tiendas asociadas. Todo esto mediante una plataforma intuitiva, disponible tanto en su versión web como en su aplicación móvil.

Con una interfaz amigable y funcionalidades prácticas, como control de gastos, historial de compras, recomendaciones basadas en hábitos del usuario y visualización en tiempo real de precios y disponibilidad, Smart Cart promete transformar la experiencia de compra cotidiana, haciéndola más rápida, ordenada y económica.

1.2 Objetivo General

Desarrollar una plataforma inteligente, accesible vía web y aplicación móvil, que permita a los usuarios organizar, optimizar y ejecutar sus compras de manera eficiente mediante funcionalidades como listas dinámicas, comparación de precios, historial de productos y recomendaciones personalizadas.

1.3 Objetivos Específicos

- Gestión de listas de compras: Permitir que los usuarios creen, editen y administren listas de productos, incluyendo cantidades, categorías y prioridades.

- Catálogo digital de productos: Integrar catálogos actualizados de productos provenientes de distintas tiendas o supermercados asociados, permitiendo búsquedas, filtros por precio, marca o categoría.

- Recomendaciones inteligentes: Implementar un sistema de recomendaciones basadas en el historial de compras y preferencias del usuario, usando lógica predictiva o IA.

- Sincronización multiplataforma: Garantizar que las listas y preferencias del usuario estén disponibles y sincronizadas en tiempo real tanto en la versión web como móvil.

- Notificaciones y alertas: Integrar un sistema de alertas sobre productos en descuento, reposiciones o recordatorios de compra programada.

- Gestión de usuarios y roles: Desarrollar una interfaz de acceso con distintos roles (administrador y cliente), permitiendo funcionalidades exclusivas según el tipo de usuario.

- Visualización de historial de compras: Permitir a los usuarios revisar sus compras pasadas, con detalles como fechas, productos adquiridos y montos totales.

1.4 Descripción del Problema

En la actualidad, muchas personas enfrentan dificultades al momento de realizar sus compras en supermercados o tiendas, ya sea por falta de tiempo, desorganización o por olvidar productos importantes. A esto se suma la necesidad de comparar precios, disponibilidad y características de los productos entre distintos establecimientos, lo cual puede ser tedioso y poco eficiente.

Smart Cart surge como una solución innovadora a estos problemas, ofreciendo una plataforma que permita a los usuarios organizar sus listas de compras de manera eficiente, acceder a catálogos digitales actualizados, recibir recomendaciones personalizadas y gestionar su experiencia de compra desde una sola aplicación, tanto en versión web como móvil.

Además, se busca implementar un sistema que identifique el tipo de usuario (cliente o administrador), permitiendo distintas funcionalidades según el rol. Esto mejorará la experiencia general, optimizando el tiempo y facilitando decisiones informadas durante las compras.

1.5 Alcance

Módulo 1: Gestión de Usuarios

- **Inicio de Sesión y Cierre de Sesión**:
  - Permitir a las clientas, administradores y empleados iniciar y cerrar sesión de manera segura, tanto en la plataforma web como en la aplicación móvil.
- **Gestión de Usuarios**:
  - Administrar cuentas para los diferentes tipos de usuarios (clientas, administradores, empleados), permitiendo la creación de perfiles y la asignación de roles específicos.
- **Roles y Niveles de Acceso**:
  - Establecer distintos niveles de acceso a la plataforma según el rol del usuario, garantizando la seguridad y privacidad de los datos.

Módulo 2: Gestión de Productos

- **Registro de Productos**:
  - Administrar el inventario de la tienda registrando productos con detalles como tallas, colores, descripciones, fotos y precios.
- **Categorías y Subcategorías**:
  - Organizar los productos por categorías y subcategorías para facilitar la navegación y búsqueda en el catálogo.
- **Control de Stock**:
  - Mantener un sistema de inventario actualizado que refleje la disponibilidad de productos en tiempo real.

Módulo 3: Catálogo de Productos

- **Catálogo Digital**:
  - Publicar un catálogo de productos completo y accesible desde la web y la app móvil, con filtros avanzados para facilitar la búsqueda de productos.
- **Detalles del Producto**:
  - Proporcionar información detallada de los productos, incluyendo fotos, características y disponibilidad en sucursales.

Módulo 4: Carrito de Compras y Pedidos

- **Carrito de Compras**:
  - Permitir a las clientas agregar productos al carrito, revisarlos y modificar su selección antes de proceder al pago.
- **Gestión de Pedidos**:
  - Administrar el proceso de pedidos, mostrando información actualizada sobre el estado de los mismos, desde la confirmación hasta la entrega o recogida en tienda.

Módulo 5: Pagos, Entregas y Reportes

- **Métodos de Pago**:
  - Integrar diversas formas de pago, tales como tarjetas de crédito/débito, pagos mediante QR, transferencias bancarias y pago contra entrega.
- **Pasarelas de Pago**:
  - Implementar pasarelas de pago nacionales como Libélula e internacionales como Stripe para garantizar transacciones seguras.
- **Opciones de Entrega y Recogida**:
  - Ofrecer a las clientas la opción de entrega a domicilio, con cálculo automático de costos según distancia y características del pedido, o la posibilidad de reservar y recoger los productos en una sucursal.
- **Generación de Reportes de Venta**:
  - Crear informes detallados sobre ventas, permitiendo a los administradores analizar el rendimiento del e-commerce, con datos como productos más vendidos, zonas de mayor demanda y canal de venta.

Módulo 7*:* Compras mediante Voz y Recomendaciones con IA

- **Reconocimiento de voz:**
  Permitir a los usuarios agregar productos al carrito o realizar operaciones como calcular el total, utilizando comandos de voz (por ejemplo: *"Agregar un mouse inalámbrico"* o *"Mostrar total"*).
- **Procesamiento de comandos:**
  Convertir la voz en texto y ejecutar acciones específicas en la app. Ofrecer confirmaciones visuales o verbales de las acciones.
- **Recomendaciones inteligentes en tiempo real:**
  Mientras se realiza la compra, mostrar sugerencias de productos relacionados utilizando algoritmos como Apriori o reglas de asociación.
- **Interacción accesible:**
  Mejorar la experiencia de compra para personas con discapacidad o usuarios que deseen una experiencia más rápida, especialmente en tiendas físicas concurridas.

Parte I – Fundamentación Teórica

a) E-commerce

El comercio electrónico, o e-commerce, es la compra y venta de bienes y servicios a través de medios electrónicos, especialmente Internet. Este modelo de negocio permite a las empresas ampliar su alcance, operar 24/7, reducir costos y ofrecer a los consumidores una experiencia de compra más cómoda y personalizada.

a. Como usuario

i. Amazon

Amazon es una de las plataformas de e-commerce más grandes del mundo. Ofrece una experiencia de compra altamente optimizada: búsqueda avanzada, recomendaciones personalizadas, opiniones de usuarios, pagos seguros, y entregas rápidas con opciones como Amazon Prime. Su ecosistema incluye múltiples categorías de productos y permite a terceros vender dentro de su Marketplace.

ii. Alibaba

Alibaba opera principalmente como un Marketplace B2B (Business to Business). Los usuarios pueden contactar proveedores mayoristas, solicitar cotizaciones, negociar precios y realizar compras a gran escala. Su modelo favorece la conexión entre fabricantes y empresas distribuidoras o minoristas.

iii. Shopify

Shopify permite a cualquier persona crear su propia tienda en línea sin necesidad de conocimientos avanzados de programación. Como usuario, la experiencia es muy fluida: múltiples formas de pago, catálogos visuales y adaptabilidad a dispositivos móviles. Cada tienda tiene su propio diseño, pero todas comparten la estabilidad y rapidez de la plataforma.

b. Como desarrollador

i. Magento

Magento es una plataforma de e-commerce robusta y flexible desarrollada en PHP. Es ideal para negocios medianos o grandes que requieren personalización avanzada. Ofrece gestión de productos, usuarios, pasarelas de pago, cupones, reportes, y es altamente escalable.

ii. PrestaShop

PrestaShop es una solución gratuita y de código abierto que facilita la creación de tiendas en línea. Ofrece una interfaz amigable para gestionar productos, pedidos, clientes y marketing. También cuenta con módulos para integrar pasarelas de pago, logística y temas visuales.

iii. WooCommerce

WooCommerce es un plugin para WordPress que transforma un sitio en una tienda en línea completamente funcional. Es ideal para pequeños negocios y para quienes ya manejan WordPress. Ofrece muchas extensiones para pagos, envíos, gestión de inventario y más.

b) Pasarelas de Pago

Las pasarelas de pago son servicios que permiten procesar pagos electrónicos de forma segura entre el comprador y el vendedor. Actúan como intermediarios entre la tienda en línea, el cliente y las entidades bancarias o emisoras de tarjetas.

i. Formas comunes de pago online

Tarjetas de crédito y débito: Permiten al usuario pagar ingresando los datos de su tarjeta. Las pasarelas validan la información con el banco emisor y autorizan la transacción.

Pagos mediante QR: El usuario escanea un código QR generado por la tienda, lo que redirige a un enlace de pago desde una app bancaria o billetera virtual.

Transferencias bancarias: Se realiza el pago directamente desde una cuenta bancaria a otra, a veces integradas por medio de enlaces o plugins dentro de la tienda.

Pago contra entrega: Aunque no es online, puede registrarse como opción en el sistema y confirmar el pago en físico al recibir el producto.

ii. Pasarela de Pago LIBÉLULA

Libélula es una pasarela de pagos desarrollada en Bolivia que permite recibir pagos en línea con tarjetas de débito, crédito y otros medios locales. Se integra con tiendas online por medio de su API, permitiendo registrar pagos y verificar transacciones. Su ventaja principal es la adecuación al sistema bancario y regulaciones locales.

iii. Pasarela de Pago STRIPE

Stripe es una pasarela de pagos internacional ampliamente utilizada por su facilidad de integración, documentación completa y soporte para pagos en múltiples monedas. Permite aceptar pagos con tarjetas, Apple Pay, Google Pay, y transferencias bancarias. También ofrece servicios avanzados como facturación recurrente, prevención de fraude y reportes financieros.

c) Deliverys

Los servicios de delivery son fundamentales para el e-commerce, ya que se encargan de la distribución física de los productos hasta el domicilio del cliente.

i. Funcionamiento

Aplicaciones como Pedidos Ya, Glovo o Uber Eats permiten a los usuarios realizar pedidos desde su celular y recibir productos en cuestión de minutos u horas. Los repartidores reciben una notificación, recogen el producto y lo entregan al destino indicado. El sistema permite seguimiento en tiempo real, evaluación del servicio y notificaciones automáticas.

ii. Cálculo de pagos de entrega

El precio de una entrega se calcula generalmente en función de:

Distancia entre el punto de partida (restaurante o tienda) y el destino del cliente.

Peso y volumen del producto, que afectan el tipo de vehículo requerido.

Tiempo estimado de entrega y demanda actual (tarifas dinámicas).

Frecuencia de uso (algunos servicios ofrecen descuentos o tarifas planas a clientes frecuentes o suscritos).

Zonas de cobertura y restricciones logísticas.

d) Proceso Unificado De Desarrollo De Software (PUDS)

Para el Sistema de Punto de Venta (POS) con Inteligencia Artificial y Reconocimiento de Voz, el PUDS proporciona un marco estructurado que permite transformar requerimientos complejos del comercio electrónico y físico en una solución funcional, eficiente y moderna. Este enfoque es ideal para desarrollar funcionalidades innovadoras como recomendaciones inteligentes, ventas por voz y gestión omnicanal.

Etapas del PUDS

- **Fase de Inicio**

Se define la visión del proyecto enfocándose en las necesidades del entorno de tiendas electrónicas modernas: agilidad en la atención al cliente, recomendaciones automatizadas, y reconocimiento de voz. Se identifican actores clave (vendedores, clientes, responsables de almacén) y se establece la propuesta de valor basada en accesibilidad, velocidad y automatización.

- **Fase de Elaboración**

Se realiza un análisis detallado de los casos de uso críticos, como: gestión de productos, recomendaciones inteligentes en tiempo real, procesamiento de ventas mediante comandos de voz, y generación automática de reportes. Se diseña una arquitectura escalable y segura que permite integrar módulos de Machine Learning, servicios de voz y múltiples dispositivos cliente.

- **Fase de Construcción**

Durante esta fase se desarrolla iterativamente el sistema completo, incluyendo el frontend (web y móvil), backend con APIs RESTful y módulos de inteligencia artificial. Se construyen componentes como: el motor de recomendaciones, reconocimiento de voz con procesamiento de lenguaje natural (PLN), integración con pasarelas de pago y módulos de gestión de inventario.

- **Fase de Transición**

El sistema es validado con pruebas funcionales, de carga y de experiencia de usuario. Se capacita al personal de tienda y se recopila retroalimentación para ajustes finales. También se configura la generación automática de reportes y notificaciones para garantizar una operación fluida desde el primer día.

e) Lenguaje de Modelado Unificado (UML):

UML es esencial para modelar visualmente tanto la estructura como el comportamiento del sistema POS, facilitando la comunicación entre analistas, diseñadores, desarrolladores y otros stakeholders. Permite entender cómo interactúan los componentes del sistema con los usuarios, cómo se comportan los procesos automáticos, y cómo se organizan los datos.

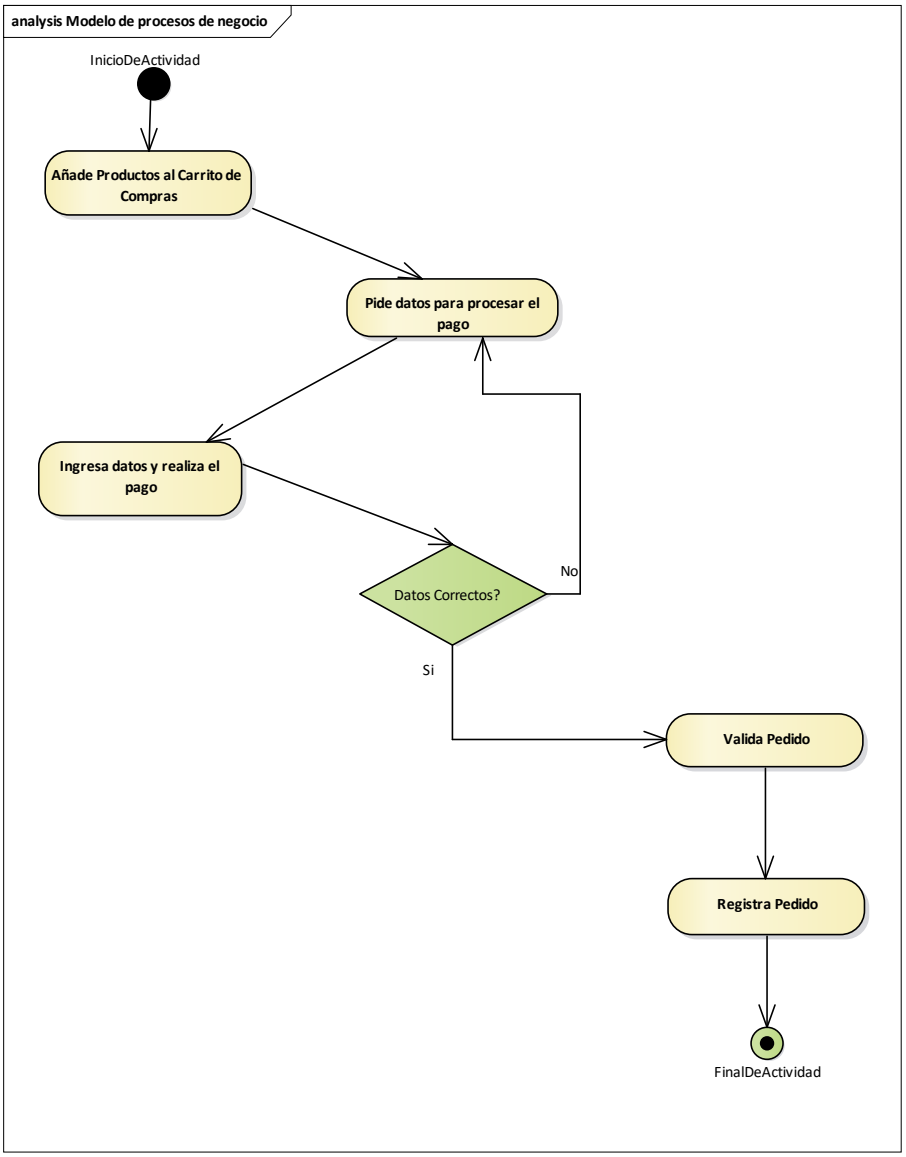
Aplicación de UML

- **Modelo Funcional**
  Se emplean **diagramas de casos de uso** para representar interacciones como: agregar productos por voz, sugerencias automáticas de compra, realizar un pago, o generar un reporte de ventas. Estos diagramas permiten identificar claramente las funciones esperadas por cada actor.
- **Modelo de Objetos**
  Los **diagramas de clases** describen la estructura de datos y relaciones entre entidades como: productos, clientes, transacciones, historial de compras, usuarios del sistema y recomendaciones. Este modelo es clave para diseñar la base de datos y la lógica del backend.
- **Modelo Dinámico**
  Mediante **diagramas de secuencia y estados**, se modela el comportamiento de procesos complejos como:
  - el flujo de reconocimiento de voz hasta la acción ejecutada (por ejemplo, “agregar producto”),
  - el cambio de estado del carrito durante una venta,
  - el proceso de recomendación en tiempo real según compras anteriores.

Parte II – Proceso de desarrollo

2.1. Flujo de Trabajo Captura de Requisitos

Modelo de Negocios (Diagrama de Actividades)



2.1.1. Identificar actores

- Usuario
- Administrador
- Cliente

2.1.2. identificar Casos de Uso

- CU1: Inicio de Sesión
- CU2: Cerrar Sesión
- CU3: Gestionar Usuario
- CU4: Gestionar Cliente
- CU5: Gestionar Roles
- CU6: Gestionar Productos

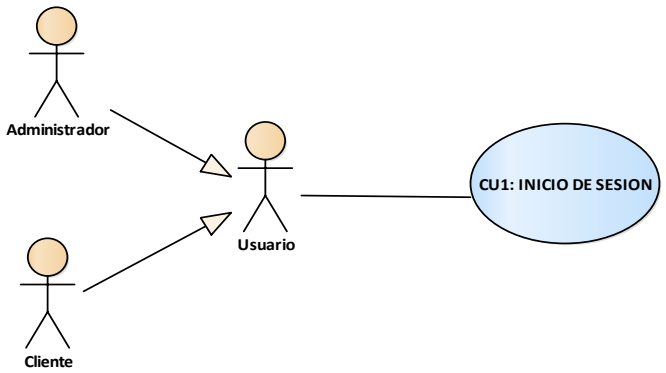
- CU7: Gestionar Carrito de Compras
- CU8: Gestionar Catalogo
- CU9: Procesar Método de Pago
- CU10: Generar Reportes de Ventas
- CU11: Consultar Historial de Ventas
- CU12: Mostrar Recomendaciones de Productos
- CU13: Agregar Producto al Carrito por Voz

2.1.2. Priorizar Casos de Uso

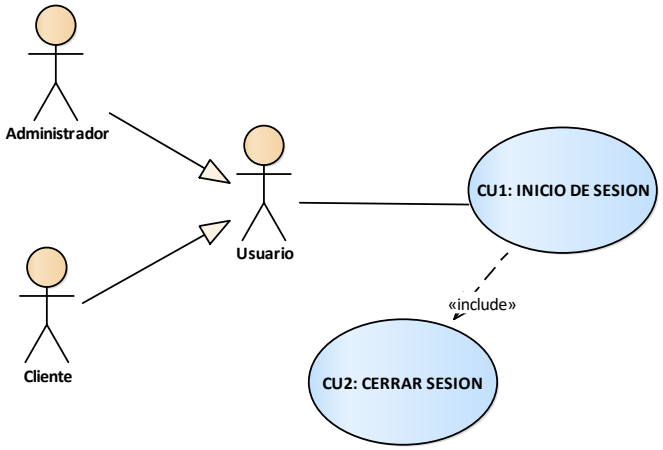
ID	Caso de Uso	Prioridad	Actores
CU1	Inicio de Sesión	Alta	Usuario, Administrador, Cliente
CU2	Cerrar Sesión	Alta	Usuario, Administrador, Cliente
CU3	Gestionar Usuario	Alta	Administrador
CU4	Gestionar Cliente	Media	Administrador
CU5	Gestionar Roles	Media	Administrador
CU6	Gestionar Productos	Alta	Administrador
CU7	Gestionar Carrito de Compras	Alta	Usuario, Cliente
CU8	Gestionar Catálogo	Media	Administrador, Usuario
CU9	Procesar Método de Pago	Alta	Cliente, Usuario
CU10	Generar Reportes de Ventas	Media	Administrador
CU11	Consultar Historial de Ventas	Media	Administrador, Usuario
CU12	Mostrar Recomendaciones de Productos	Alta	Sistema (IA), Cliente
CU13	Agregar Producto al Carrito por Voz	Alta	Usuario, Cliente, Sistema de Voz

2.1.3. Diagrama Casos de Uso

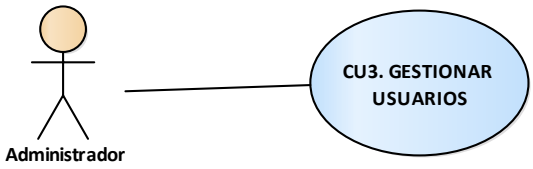
- CU1: Inicio de Sesión



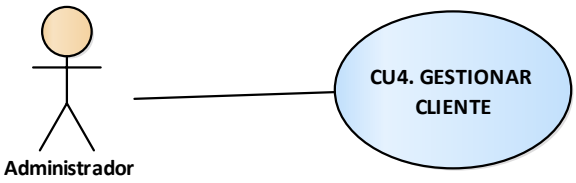
- CU2: Cerrar Sesión



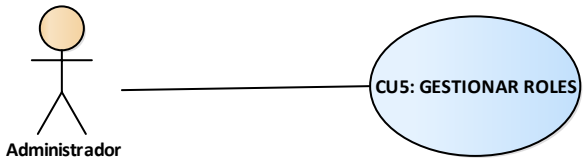
- CU3: Gestionar Usuario



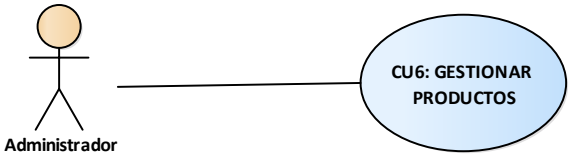
- CU4: Gestionar Cliente



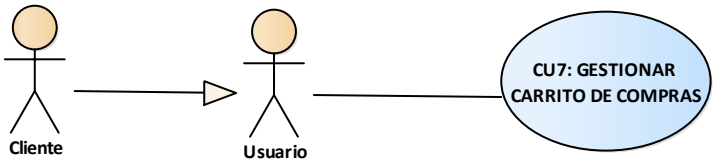
- CU5: Gestionar Roles



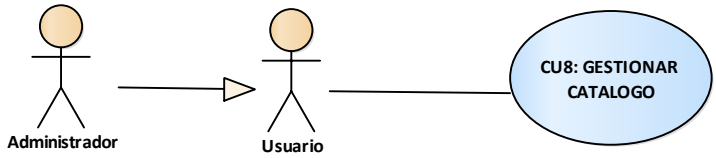
- CU6: Gestionar Productos



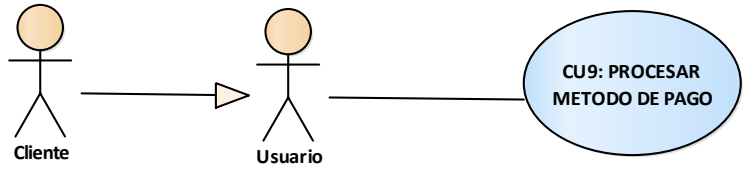
- CU7: Gestionar Carrito de Compras



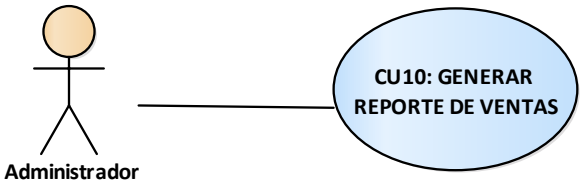
- CU8: Gestionar Catalogo



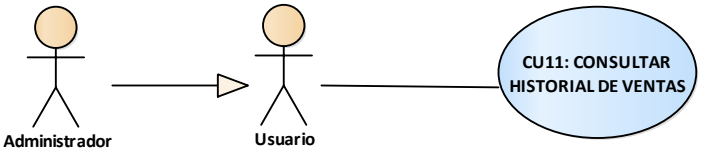
- CU9: Procesar Método de Pago



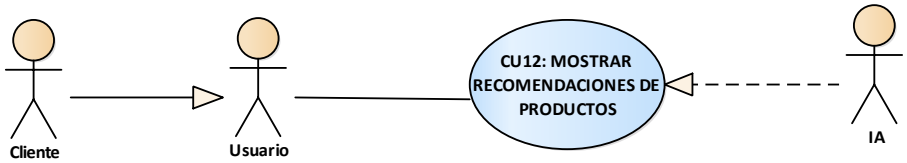
- CU10: Generar Reportes de Ventas



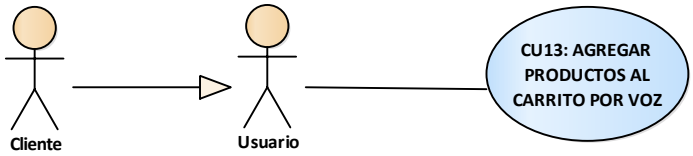
- CU11: Consultar Historial de Ventas



- CU12: Mostrar Recomendaciones de Productos



- CU13: Agregar Producto al Carrito por Voz



2.1.4. Detallar Casos de Uso

CU1: Inicio de Sesión	Descripción
Propósito/Objetivo	Validar las credenciales del usuario e iniciar una sesión segura.
Resumen	El usuario ingresa credenciales, se validan y se accede al sistema.
Actores	Usuario, Administrador, Cliente
Actor Iniciador	Usuario
Flujo de Suceso	1. Acceder a login 2. Ingresar datos 3. Validar 4. Acceso
Precondición	Usuario registrado
Postcondición	Sesión iniciada
Excepción	Credenciales incorrectas, usuario inactivo

CU2: Cerrar Sesión	Descripción
Propósito/Objetivo	Finalizar la sesión de forma segura.
Resumen	El usuario selecciona cerrar sesión y es redirigido al login.
Actores	Usuario, Administrador, Cliente
Actor Iniciador	Usuario
Flujo de Suceso	1. Seleccionar cerrar sesión 2. Invalidar sesión 3. Redirección
Precondición	Sesión activa
Postcondición	Sesión finalizada
Excepción	Error de red o sesión ya cerrada

CU3: Gestionar Usuario	Descripción
Propósito/Objetivo	Administrar los usuarios del sistema.
Resumen	Crear, modificar, eliminar usuarios.
Actores	Administrador
Actor Iniciador	Administrador
Flujo de Suceso	1. Acceder a usuarios 2. CRUD 3. Guardar
Precondición	Permisos de admin
Postcondición	Cambios reflejados en base de datos
Excepción	Datos inválidos o duplicados

CU4: Gestionar Cliente	Descripción
Propósito/Objetivo	Administrar los clientes del sistema.
Resumen	Crear, modificar y eliminar los clientes.
Actores	Administrador
Actor Iniciador	Administrador
Flujo de Suceso	1. Acceder a clientes. 2. CRUD 3. Guardar
Precondición	Permisos de admin

Postcondición	Cambios reflejados en base de datos
Excepción	Datos inválidos o duplicados

CU5: Gestionar Roles	**Descripción**
Propósito/Objetivo	Administrar los roles de los usuarios.
Resumen	Crear, modificar y eliminar roles para asignar permisos a los usuarios.
Actores	Administrador
Actor Iniciador	Administrador
Flujo de Suceso	1. Acceder a roles. 2. CRUD de roles. 3. Guardar cambios
Precondición	Permisos de admin
Postcondición	Cambios reflejados en base de datos
Excepción	Rol con nombre duplicado o permisos no válidos.

CU6: Gestionar Productos	**Descripción**
Propósito/Objetivo	Administrar los usuarios del sistema.
Resumen	Crear, modificar y eliminar productos en el inventario.
Actores	Administrador
Actor Iniciador	Administrador
Flujo de Suceso	1. Acceder a productos 2. CRUD de productos 3. Guardar cambios
Precondición	Permisos de admin
Postcondición	Cambios reflejados en base de datos
Excepción	Producto con nombre o código duplicado.

CU7: Gestionar Carrito de Compras	**Descripción**
Propósito/Objetivo	Administrar los productos en el carrito de compras del cliente.
Resumen	Agregar, eliminar y modificar la cantidad de productos en el carrito
Actores	Cliente
Actor Iniciador	Cliente
Flujo de Suceso	1. Acceder al carrito de compras. 2. Agregar, eliminar o modificar cantidad de productos. 3. Guardar cambios.
Precondición	El cliente debe estar autenticado.
Postcondición	Los productos del carrito se actualizan correctamente.
Excepción	Producto sin stock disponible.

CU8: Gestionar Catalogo	Descripción
Propósito/Objetivo	Organizar y gestionar el catálogo de productos para los clientes.
Resumen	El administrador puede agregar productos al catálogo y asignarles categorías.
Actores	Administrador
Actor Iniciador	Administrador
Flujo de Suceso	1. Acceder al catálogo de productos. 2. Agregar o modificar productos en el catálogo 3. Guardar cambios.
Precondición	Permisos de admin
Postcondición	El catálogo se actualiza con los nuevos productos o cambios.
Excepción	Producto con nombre o código duplicado

CU9: Procesar Método de Pago	Descripción
Propósito/Objetivo	Procesar el pago de una compra realizada por el cliente
Resumen	Permite al cliente seleccionar el método de pago y completar la compra.
Actores	Cliente
Actor Iniciador	Cliente
Flujo de Suceso	1. Acceder al método de pago. 2. Seleccionar método de pago (tarjeta, PayPal, etc.) 3. Confirmar el pago
Precondición	El cliente debe tener productos en su carrito
Postcondición	El pago se procesa correctamente y la compra es completada
Excepción	Pago no autorizado o error en el sistema de pago

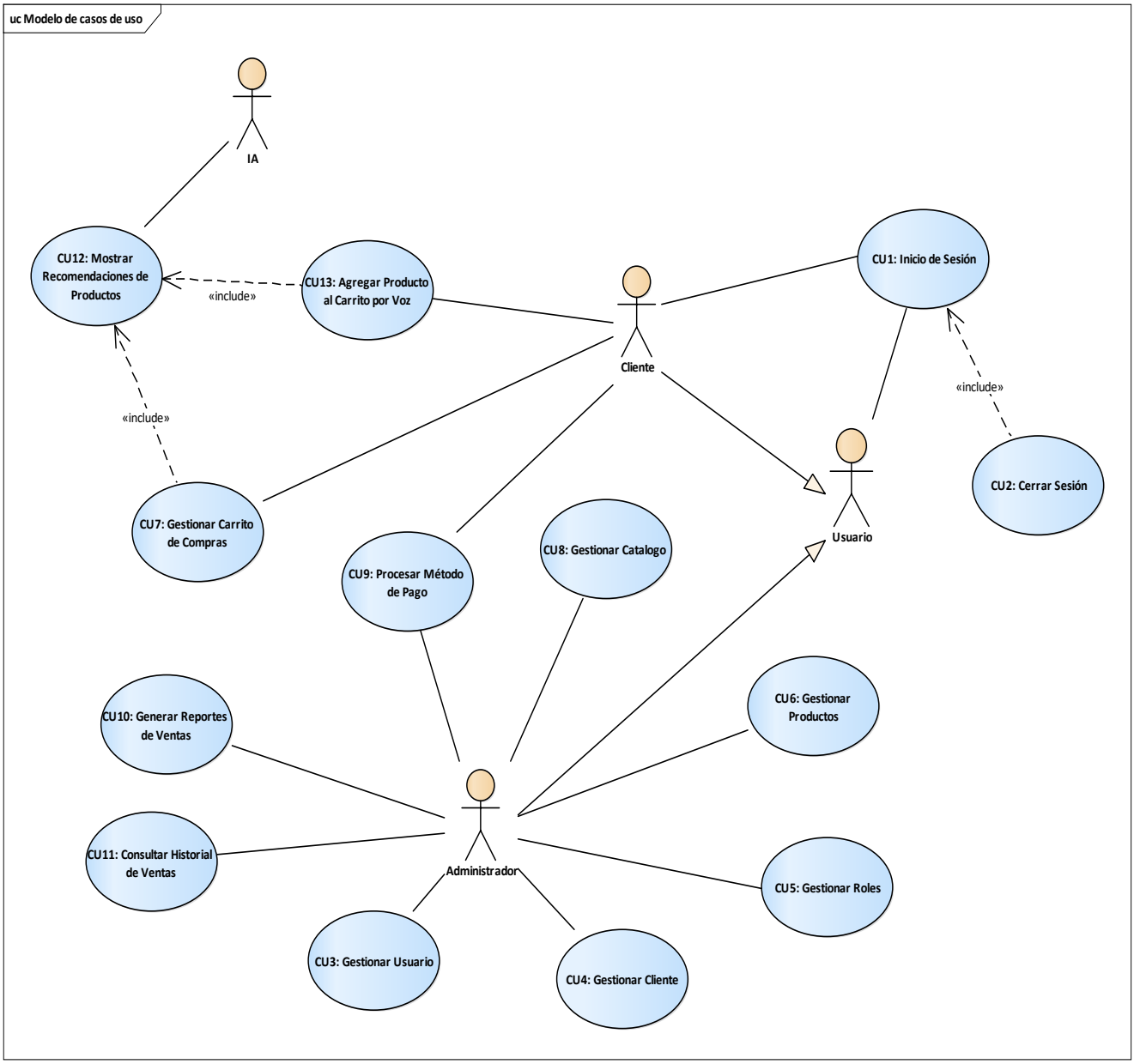
CU10: Generar Reportes de Ventas	Descripción
Propósito/Objetivo	Generar reportes sobre las ventas realizadas.
Resumen	Permite al administrador generar informes sobre las ventas y estadísticas de productos.
Actores	Administrador
Actor Iniciador	Administrador
Flujo de Suceso	1. Acceder a la sección de reportes. 2. Seleccionar los filtros para el reporte (fechas, productos, ventas). 3. Generar el reporte.
Precondición	Permisos de admin
Postcondición	El reporte se genera y muestra en el sistema.
Excepción	No hay datos disponibles para el reporte solicitado

CU11: Consultar Historial de Ventas	**Descripción**
Propósito/Objetivo	Consultar el historial de ventas realizadas en la tienda.
Resumen	Permite al administrador ver las ventas previas realizadas en la tienda.
Actores	Administrador
Actor Iniciador	Administrador
Flujo de Suceso	1. Acceder al historial de ventas. 2. Ver detalles de ventas previas
Precondición	Permisos de admin
Postcondición	El historial de ventas se muestra correctamente
Excepción	No hay ventas registradas en el período consultado

CU12: Mostrar Recomendaciones de Productos	**Descripción**
Propósito/Objetivo	Sugerir productos relevantes al cliente según sus preferencias o historial de compras
Resumen	El sistema ofrece recomendaciones personalizadas al cliente basado en su comportamiento.
Actores	Cliente
Actor Iniciador	Cliente
Flujo de Suceso	1. El cliente visualiza productos sugeridos 2. El cliente interactúa con las recomendaciones
Precondición	El cliente debe haber interactuado previamente con productos.
Postcondición	El cliente ve productos recomendados que le pueden interesar
Excepción	No hay productos recomendados por falta de datos

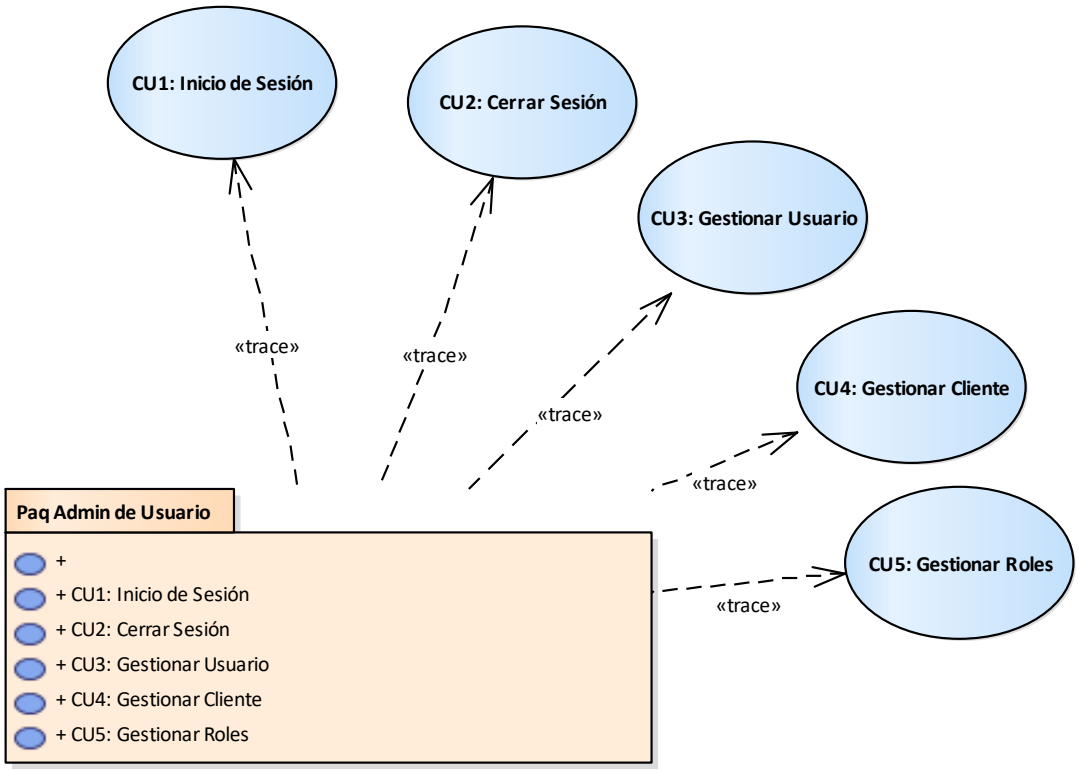
CU13: Agregar Producto al Carrito por Voz	**Descripción**
Propósito/Objetivo	Permitir al cliente agregar productos al carrito mediante comandos de voz.
Resumen	El cliente puede añadir productos al carrito utilizando su voz.
Actores	Cliente
Actor Iniciador	Cliente
Flujo de Suceso	1. El cliente emite un comando de voz para agregar un producto 2. El sistema interpreta la voz y agrega el producto al carrito
Precondición	El cliente debe estar autenticado.
Postcondición	El producto se agrega correctamente al carrito.
Excepción	El sistema no entiende el comando o el producto no está disponible

2.1.5. Estructura del modelo de Caso de Uso

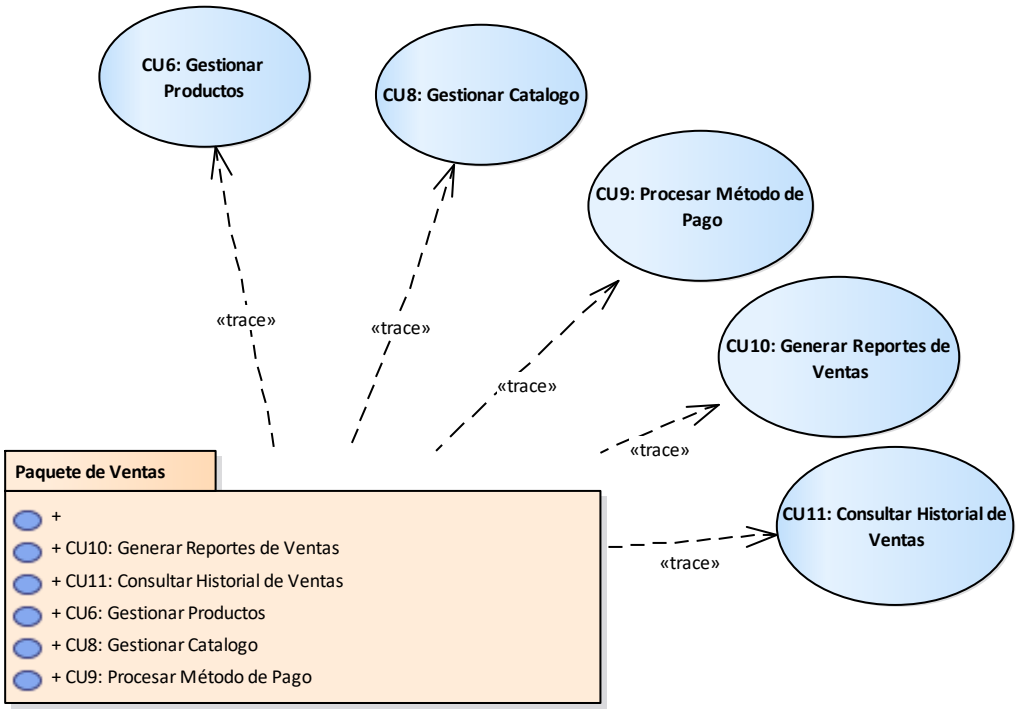


2.2. Flujo de Trabajo Análisis

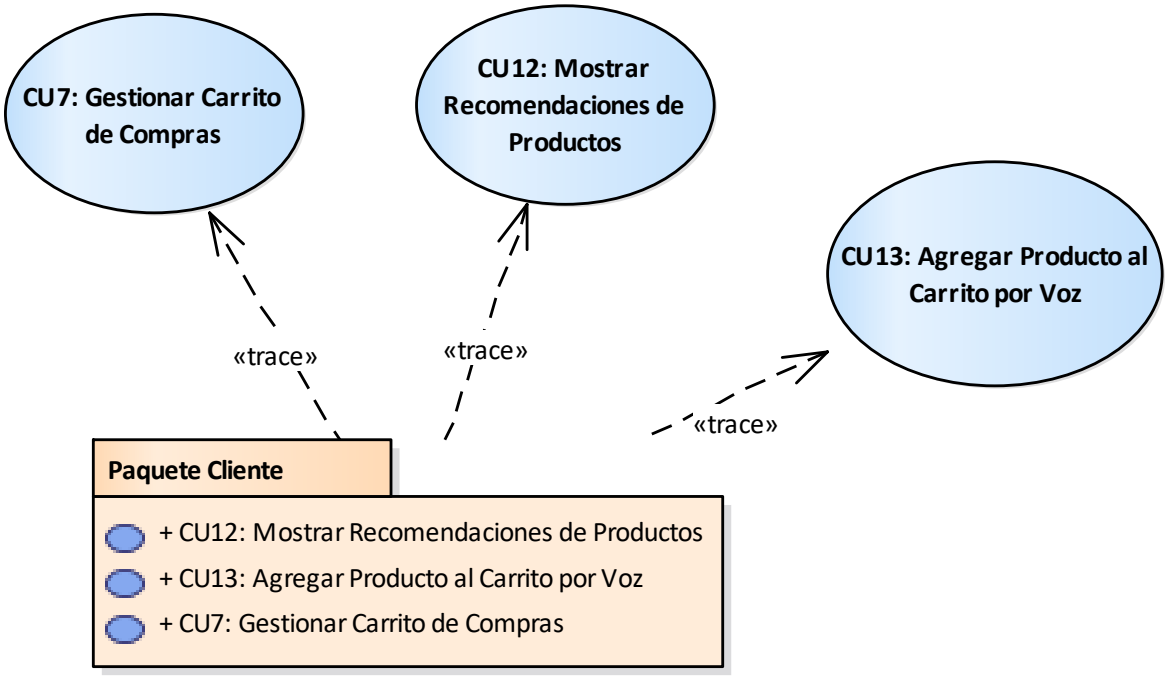
2.2.1. Análisis de Arquitectura



(from Modelo de análisis)



(from Modelo de análisis)

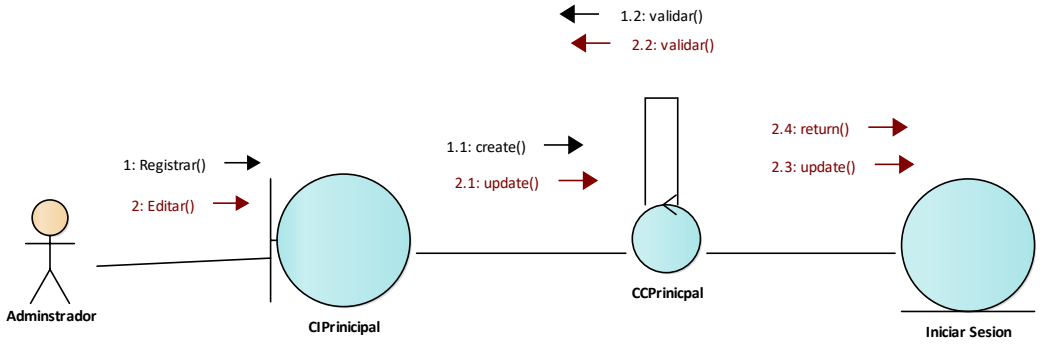


(from Modelo de análisis)

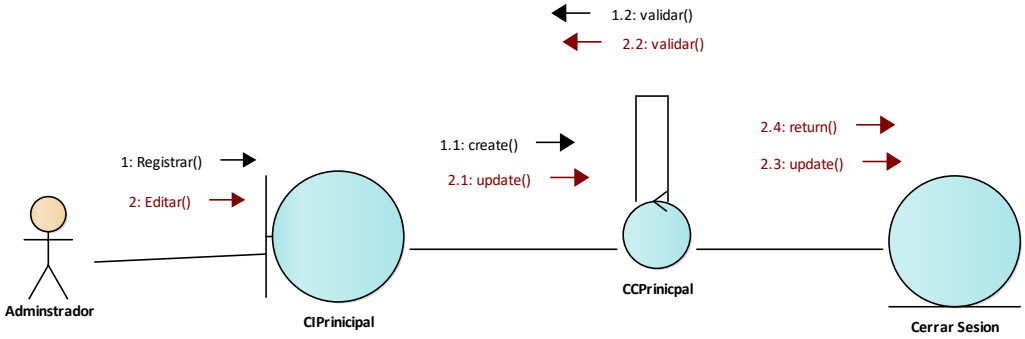
2.2.2. Análisis de Caso de Uso

Diagrama de Comunicaciones

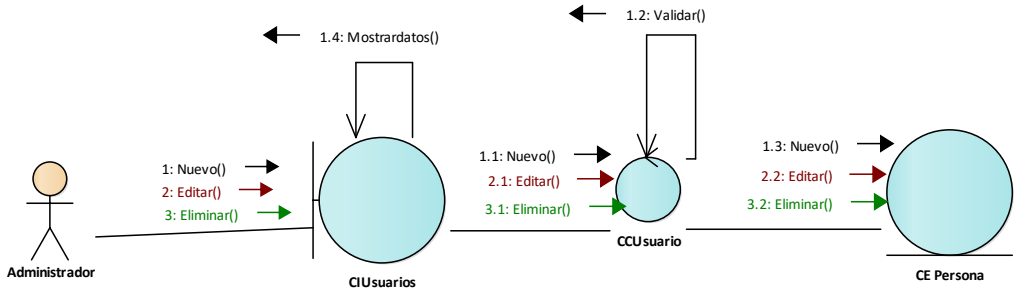
- CU1: Inicio de Sesión



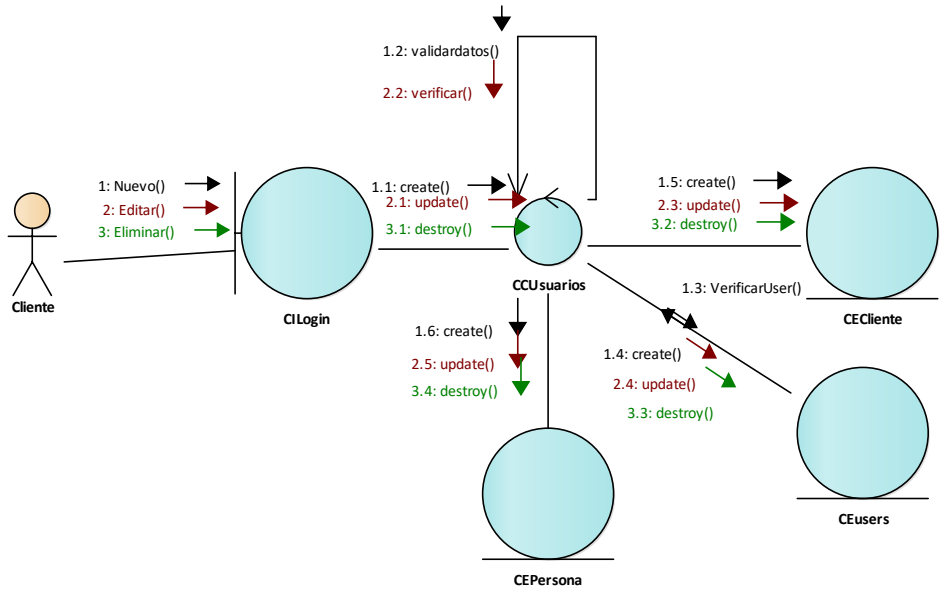
- CU2: Cerrar Sesión



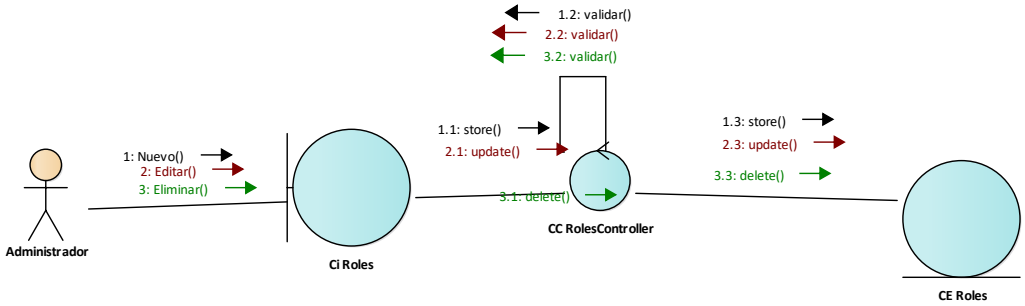
- CU3: Gestionar Usuario



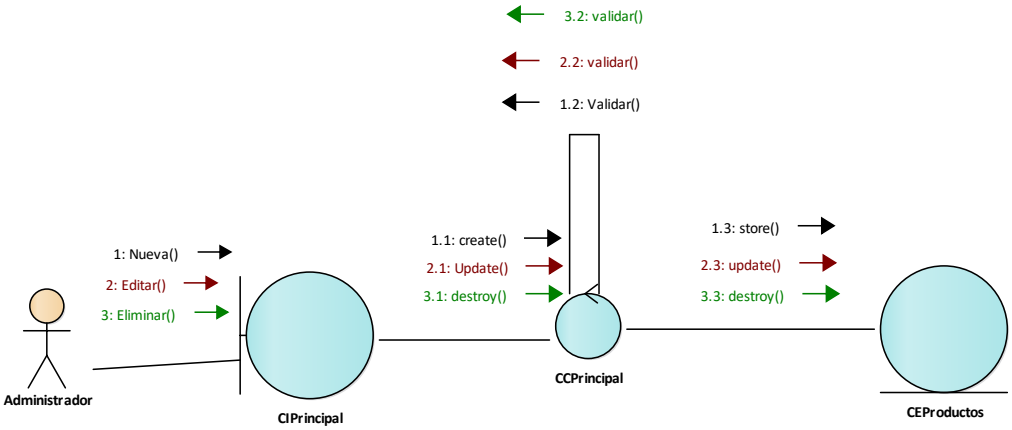
- CU4: Gestionar Cliente



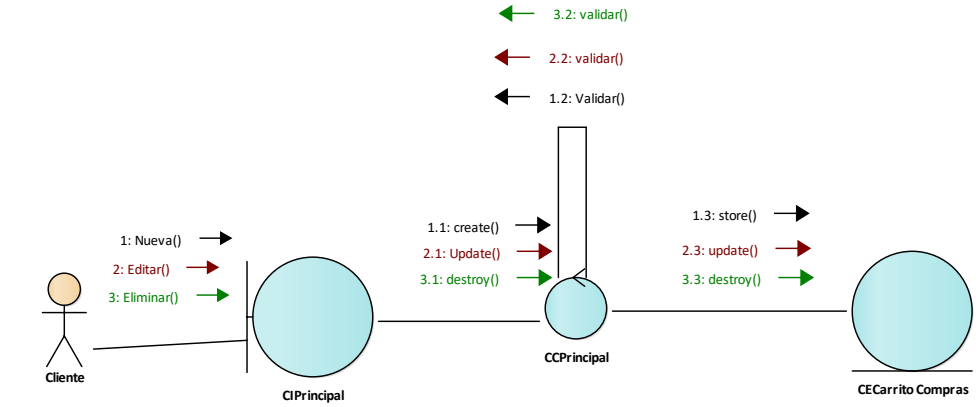
- CU5: Gestionar Roles



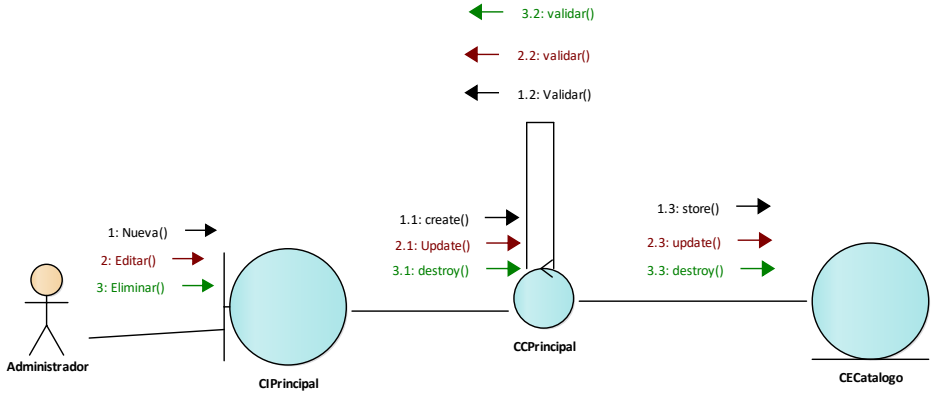
- CU6: Gestionar Productos



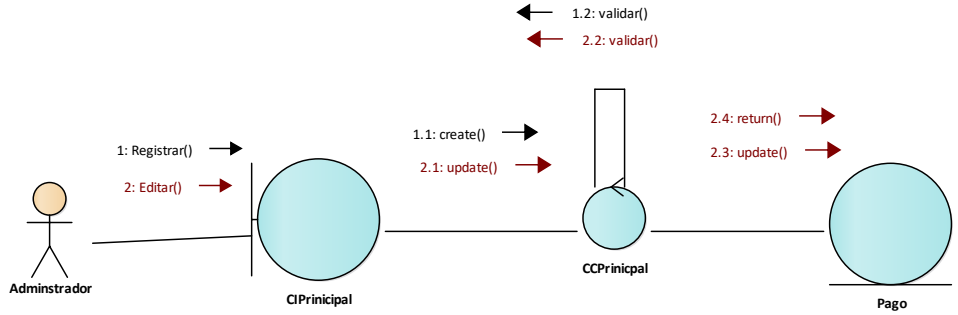
- CU7: Gestionar Carrito de Compras



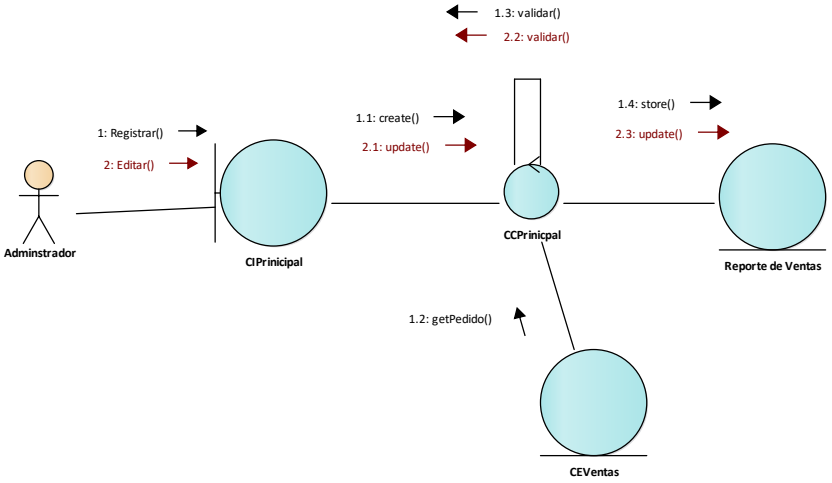
- CU8: Gestionar Catalogo



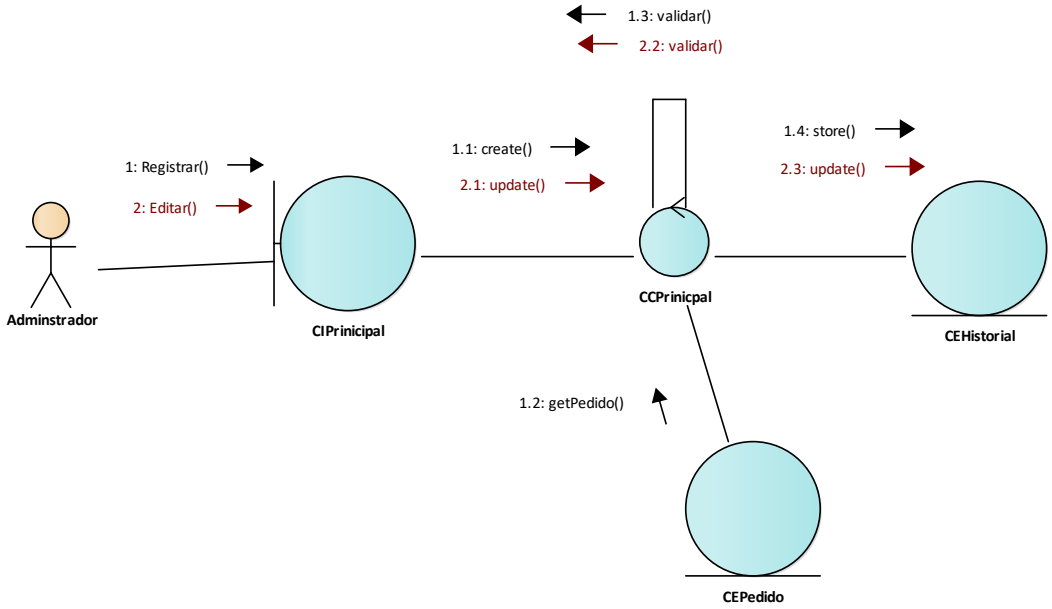
- CU9: Procesar Método de Pago



- CU10: Generar Reportes de Ventas



- CU11: Consultar Historial de Ventas

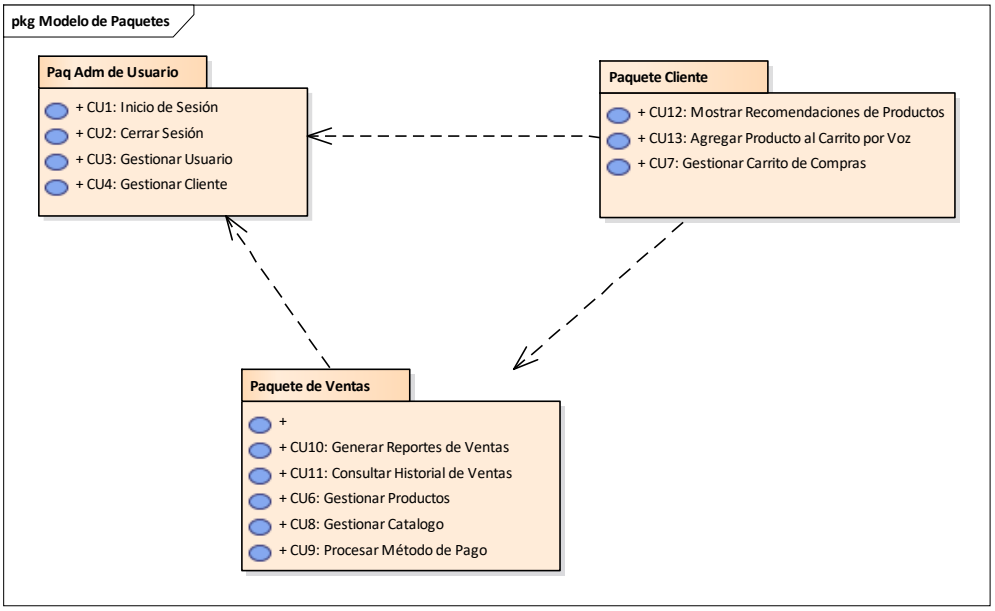


2.2.3. Análisis de Clase

- CU1: Inicio de Sesión

(usuario con clases)

2.2.4. Análisis de Paquete



2.3. Flujo de Trabajo Diseño

2.3.1. Diseño Arquitectura

Diagrama de Despliegue

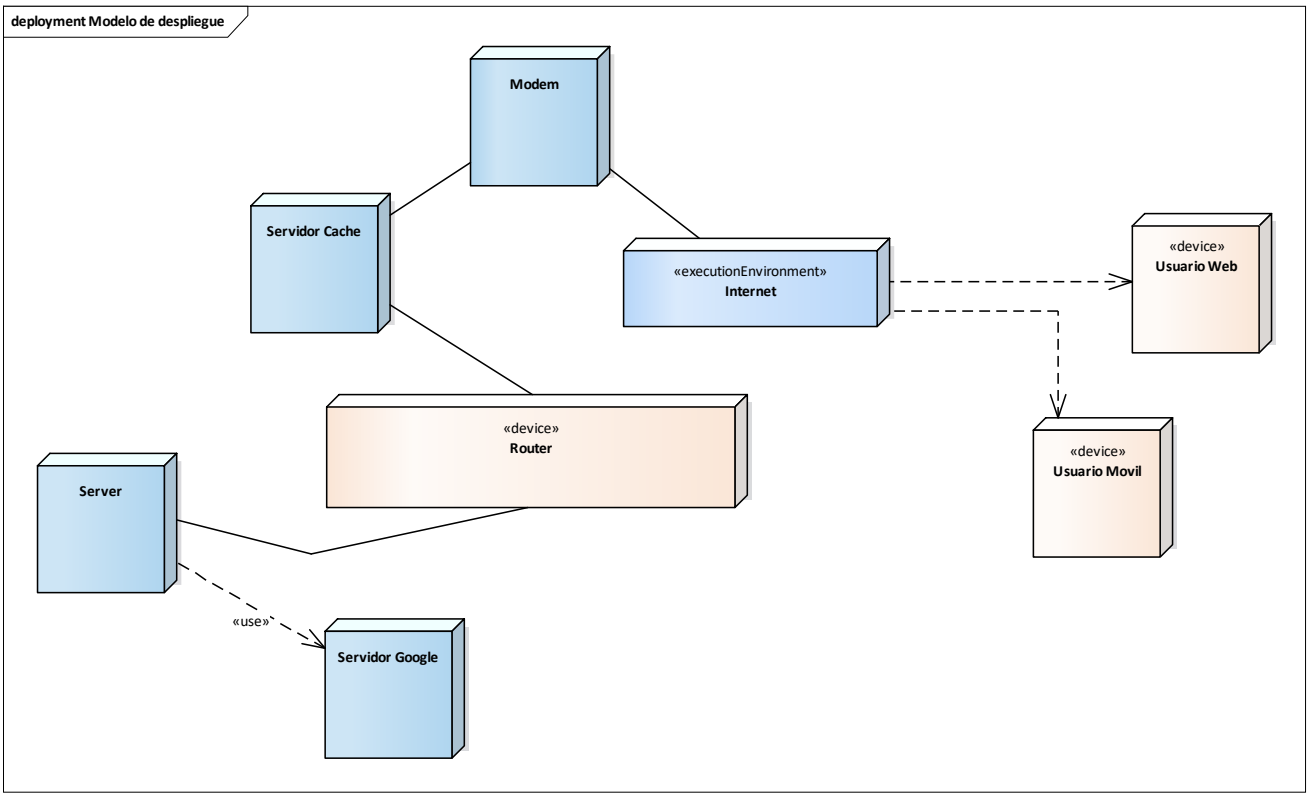
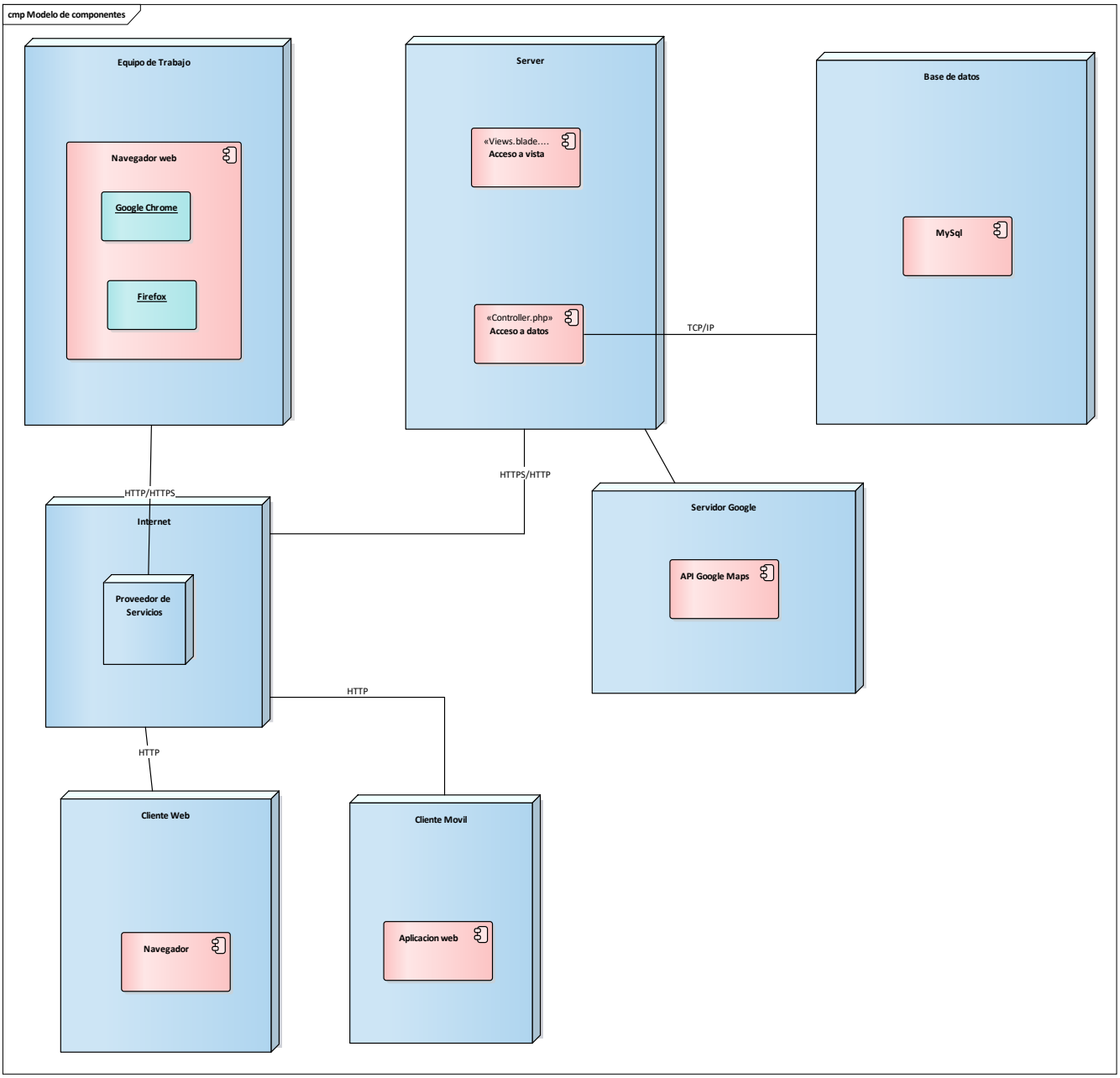


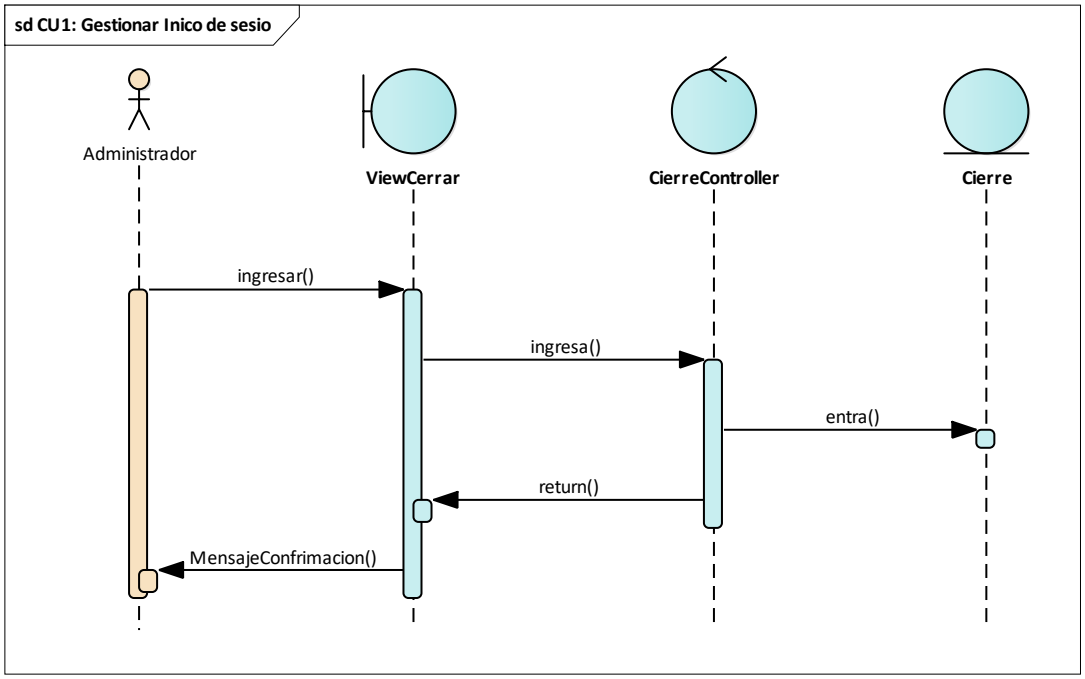
Diagrama organizado por capas

Diagrama de Componentes

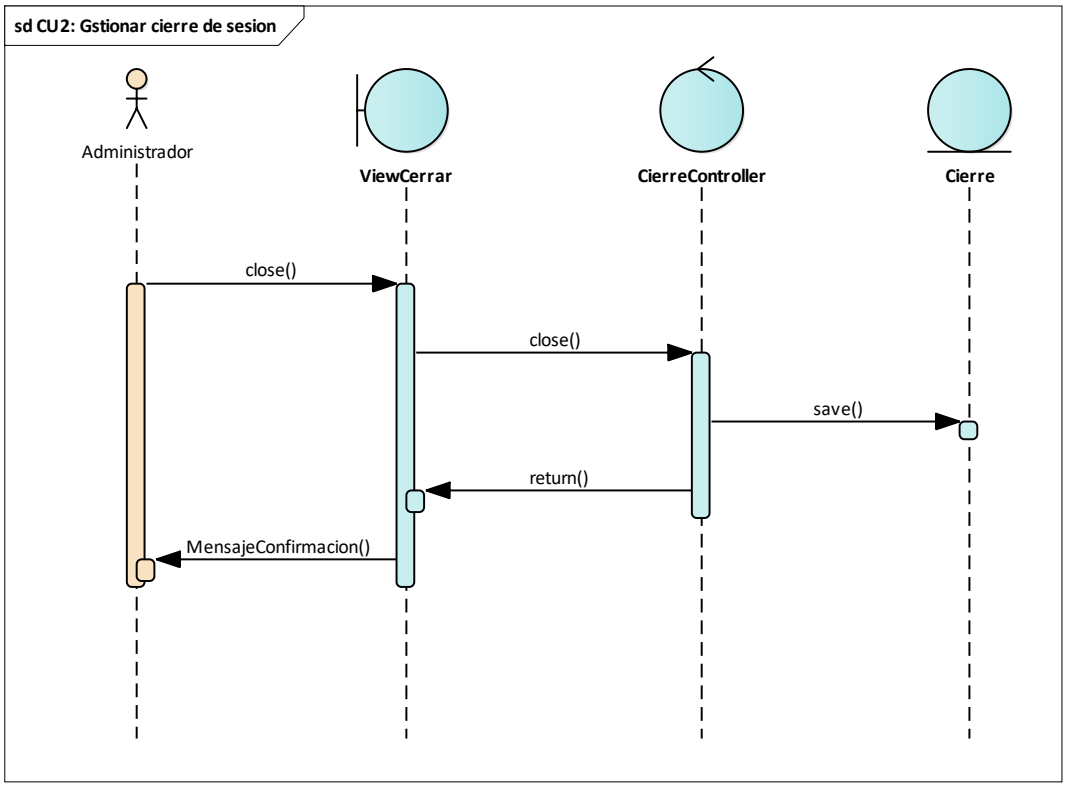


2.3.2. Diseño Casos de Uso

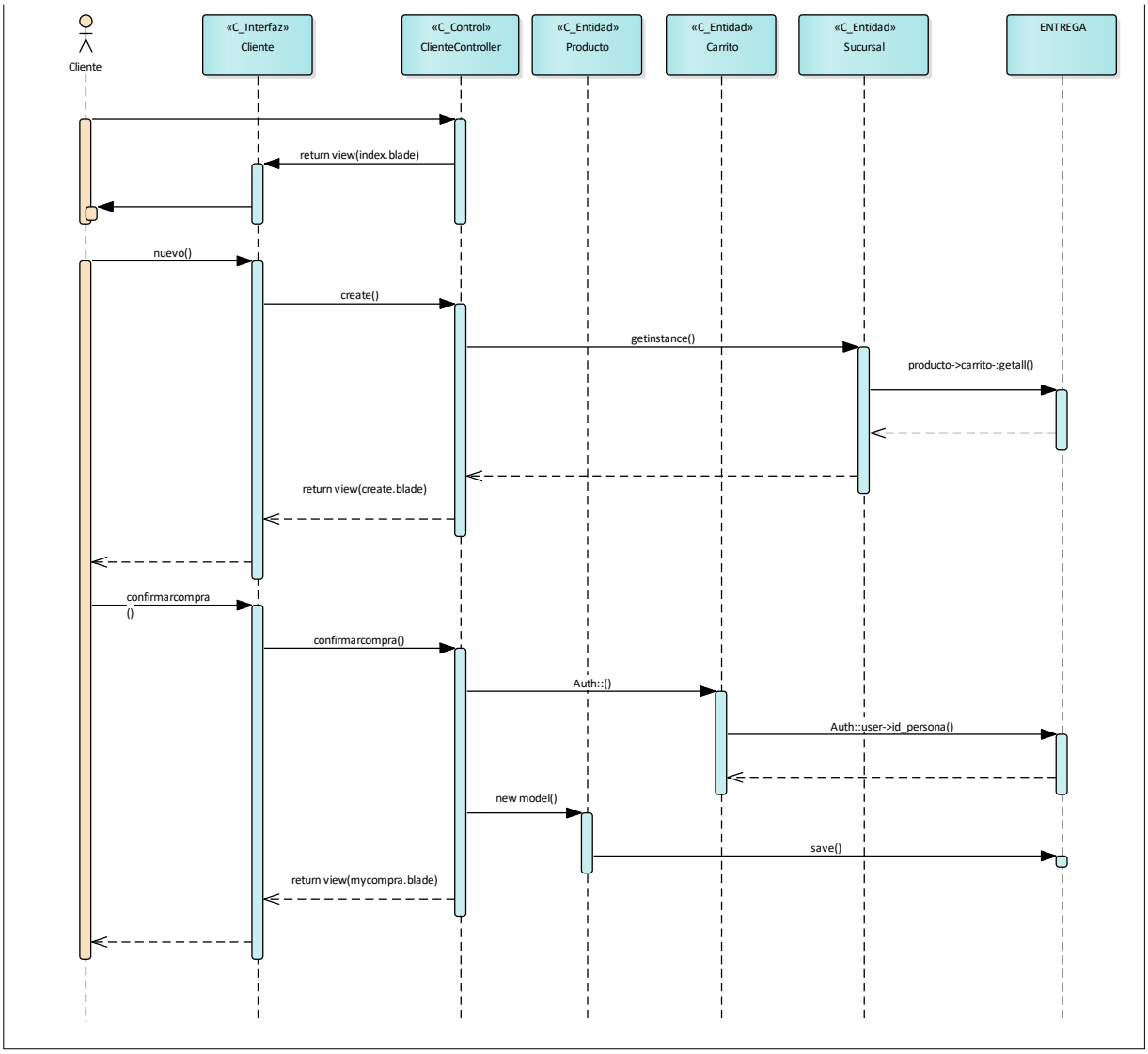
- CU1: Inicio de Sesión



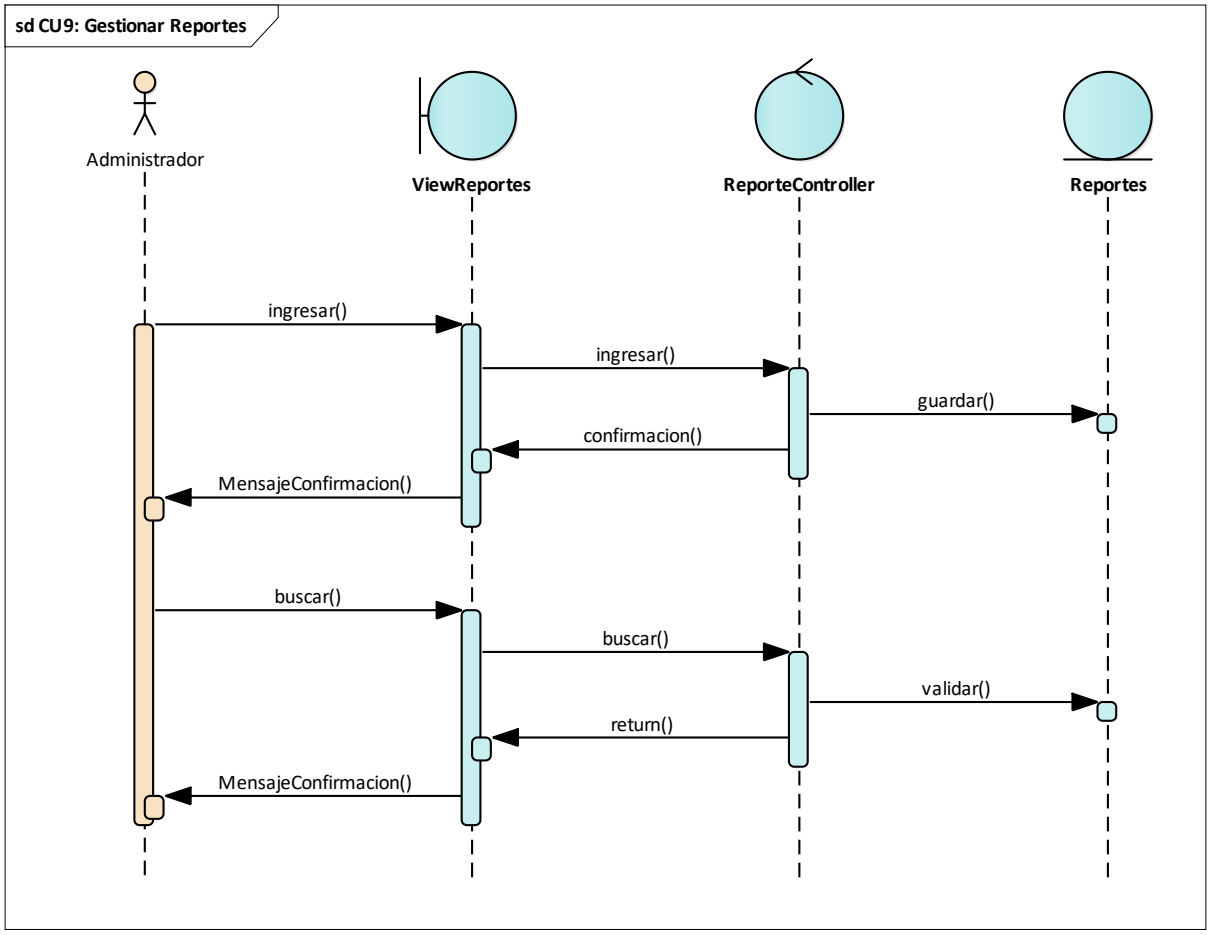
- CU2: Cerrar Sesión



- CU7: Gestionar Carrito de Compras

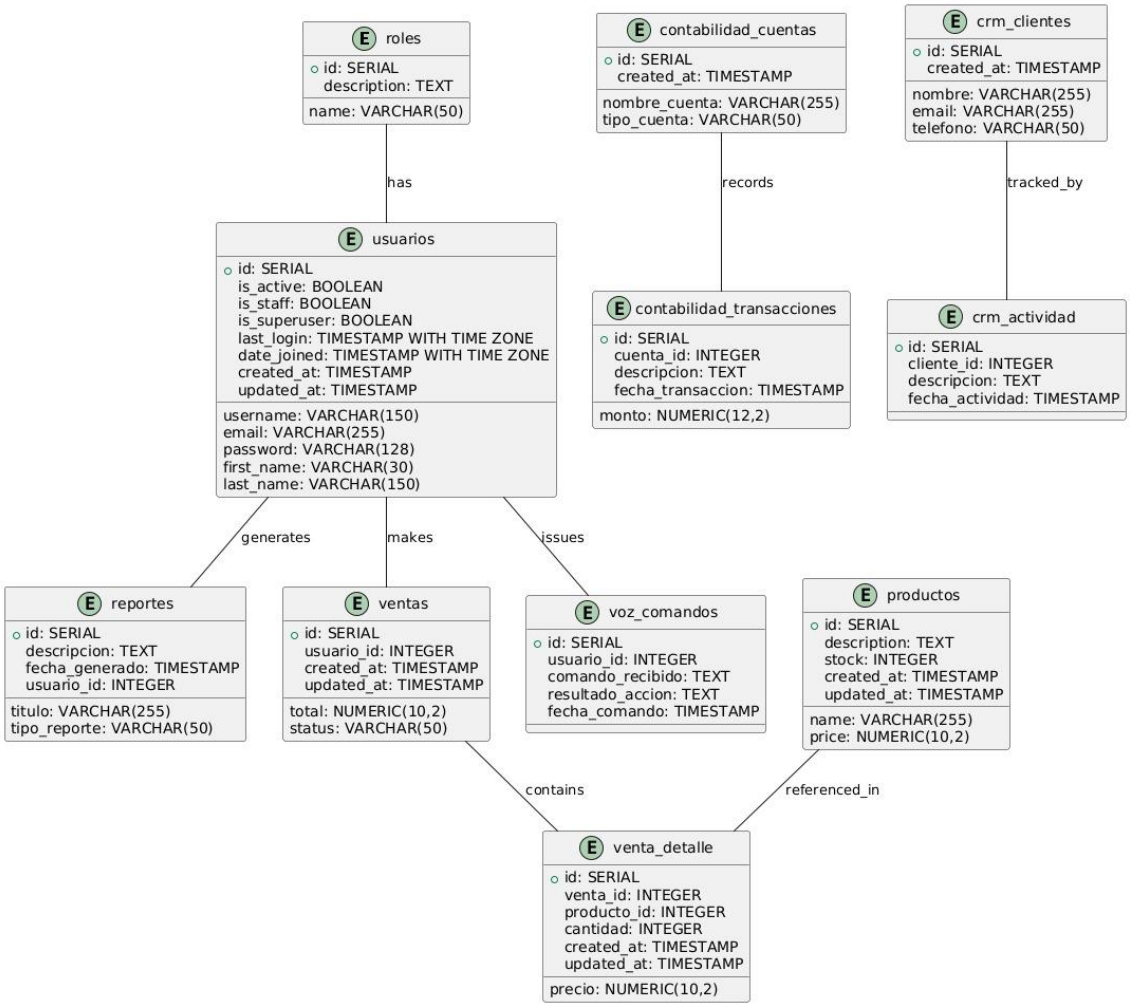


- CU9: Gestionar Reportes



2.3.3. Diseño de Datos

Diseño Conceptual



Mapeo

tabla: usuario

id	username	email	password	first_name	last_name	is_active	is_staf	is_superuser	last_login	date_joined	create_at	update_id	role_id

tabla: roles

id	name	description

tabla: productos

id	name	description	price	stock	create_at	updated_at

tabla: contabilidad_cuentas

id	cuenta_id	tipo_cuenta	create_at

tabla: contabilidad_transacciones

id	cuenta_id	monto	descripcion	fecha_transaccion

tabla: crm_clientes

id	nombre	email	telefono	created_at

tabla: crm_actividad

id	cliente_id	descripcion	fecha_actividad

tabla: reportes

id	titulo	descripcion	tipo_reporte	fecha_generado	usuario_id

tabla: ventas

id	usuario_id	total	status	created_at	updated_at

tabla: venta_detalle

id	venta_id	producto_id	cantidad	precio	created_at	updated_at

tabla: voz_comandos

id	usuario_id	comando_recibido	resultado_accion	fecha_comando

Script

```
--   CREATE DATABASE ventas

-- ============================================================
-- 1) Tabla roles
--    Maneja los diferentes perfiles (admin, cliente, empleado, etc.)
-- ============================================================
CREATE TABLE roles (
    id SERIAL PRIMARY KEY,
    name VARCHAR(50) NOT NULL UNIQUE,
    description TEXT
);

-- Inserción en la tabla roles
INSERT INTO roles (name, description)
VALUES
  ('cliente', 'Cliente del sistema'),
  ('empleado', 'Empleado del sistema'),
  ('admin', 'Administrador del sistema');

-- ============================================================
-- 2) Tabla usuarios
--    Almacena datos de usuarios y referencia a su rol
-- ============================================================
CREATE TABLE usuarios (
    id SERIAL PRIMARY KEY,
    username VARCHAR(150) NOT NULL UNIQUE,
    email VARCHAR(255) NOT NULL UNIQUE,
    password VARCHAR(128) NOT NULL,
    first_name VARCHAR(30) NOT NULL,
    last_name VARCHAR(150) NOT NULL,
    is_active BOOLEAN NOT NULL DEFAULT TRUE,
    is_staff BOOLEAN NOT NULL DEFAULT FALSE,
    is_superuser BOOLEAN NOT NULL DEFAULT FALSE,
    last_login TIMESTAMP WITH TIME ZONE NULL,
    date_joined TIMESTAMP WITH TIME ZONE NOT NULL DEFAULT
CURRENT_TIMESTAMP,
    role_id INTEGER REFERENCES roles(id),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Inserción en la tabla usuarios (usuario ejemplo de cada rol)
INSERT INTO usuarios (username, email, password, first_name, last_name, role_id)
VALUES
  ('juan_cliente', 'juan_cliente@example.com', 'password', 'Juan', 'Pérez', (SELECT id
FROM roles WHERE name='cliente')),
```

```
  ('maria_empleado', 'maria_empleado@example.com', 'password', 'María', 'López',
(SELECT id FROM roles WHERE name='empleado')),
  ('admin', 'admin@example.com', 'password', 'Admin', 'User', (SELECT id FROM roles
WHERE name='admin'));

-- ============================================================
-- 3) Tabla productos
--    Catálogo básico de productos (para ventas, inventario, etc.)
-- ============================================================
CREATE TABLE productos (
    id SERIAL PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    description TEXT,
    price NUMERIC(10,2) NOT NULL,
    stock INTEGER NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Inserción en la tabla productos
INSERT INTO productos (name, description, price, stock) VALUES
  ('Laptop', 'Laptop de gama media', 800.00, 10),
  ('Smartphone', 'Smartphone de última generación', 600.00, 15),
  ('Tablet', 'Tablet de 10 pulgadas', 300.00, 20);

-- ============================================================
-- 4) CONTABILIDAD
--    Ejemplo simplificado de manejo de cuentas y transacciones
-- ============================================================

-- 4a) Tabla contabilidad_cuentas
CREATE TABLE contabilidad_cuentas (
    id SERIAL PRIMARY KEY,
    nombre_cuenta VARCHAR(255) NOT NULL,
    tipo_cuenta VARCHAR(50) NOT NULL,          -- Ej: 'Activo', 'Pasivo', 'Patrimonio',
etc.
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Inserción en contabilidad_cuentas
INSERT INTO contabilidad_cuentas (nombre_cuenta, tipo_cuenta)
VALUES
  ('Caja', 'Activo'),
  ('Ventas', 'Ingreso'),
  ('Gastos Generales', 'Gasto');

-- 4b) Tabla contabilidad_transacciones
CREATE TABLE contabilidad_transacciones (
```

```
    id SERIAL PRIMARY KEY,
    cuenta_id INTEGER NOT NULL REFERENCES contabilidad_cuentas(id),
    monto NUMERIC(12,2) NOT NULL,
    descripcion TEXT,
    fecha_transaccion TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Inserción en contabilidad_transacciones
INSERT INTO contabilidad_transacciones (cuenta_id, monto, descripcion)
VALUES (
  (SELECT id FROM contabilidad_cuentas WHERE nombre_cuenta='Caja'),
  1200.00,
  'Ingreso inicial de caja'
);

-- ============================================================
-- 5) CRM
--    Ejemplo básico de clientes y posibles actividades de seguimiento
-- ============================================================

-- 5a) Tabla crm_clientes
CREATE TABLE crm_clientes (
    id SERIAL PRIMARY KEY,
    nombre VARCHAR(255) NOT NULL,
    email VARCHAR(255),
    telefono VARCHAR(50),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Inserción en crm_clientes
INSERT INTO crm_clientes (nombre, email, telefono)
VALUES
  ('Juan Pérez', 'juan.perez@example.com', '555-1234');

-- 5b) Tabla crm_actividad
CREATE TABLE crm_actividad (
    id SERIAL PRIMARY KEY,
    cliente_id INTEGER REFERENCES crm_clientes(id),
    descripcion TEXT,
    fecha_actividad TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Inserción en crm_actividad
INSERT INTO crm_actividad (cliente_id, descripcion)
VALUES
  (1, 'Llamada de seguimiento para venta de laptop');

-- ============================================================
```

```
-- 6) REPORTES
--    Almacén simple de reportes generados o programados
-- ============================================================
CREATE TABLE reportes (
    id SERIAL PRIMARY KEY,
    titulo VARCHAR(255) NOT NULL,
    descripcion TEXT,
    tipo_reporte VARCHAR(50),      -- Podrías tener 'PDF', 'Excel', etc.
    fecha_generado TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    usuario_id INTEGER REFERENCES usuarios(id)
);

-- Inserción en reportes
INSERT INTO reportes (titulo, descripcion, tipo_reporte, usuario_id)
VALUES (
  'Reporte de Ventas Mensual',
  'Informe de ventas del mes actual',
  'PDF',
  (SELECT id FROM usuarios WHERE username='admin')
);

-- ============================================================
-- 7) VENTAS
--    Estructura para manejo de ventas/órdenes y su detalle
-- ============================================================

-- 7a) Tabla ventas (orden principal)
CREATE TABLE ventas (
    id SERIAL PRIMARY KEY,
    usuario_id INTEGER REFERENCES usuarios(id),
    total NUMERIC(10,2) NOT NULL,
    status VARCHAR(50) NOT NULL DEFAULT 'pendiente',
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Inserción en ventas
INSERT INTO ventas (usuario_id, total, status)
VALUES (
  (SELECT id FROM usuarios WHERE username='admin'),
  800.00,
  'pagada'
);

-- 7b) Tabla venta_detalle (items de cada venta)
CREATE TABLE venta_detalle (
    id SERIAL PRIMARY KEY,
    venta_id INTEGER REFERENCES ventas(id),
```

```
    producto_id INTEGER REFERENCES productos(id),
    cantidad INTEGER NOT NULL,
    precio NUMERIC(10,2) NOT NULL,       -- precio unitario en el momento de la venta
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Inserción en venta_detalle
INSERT INTO venta_detalle (venta_id, producto_id, cantidad, precio)
VALUES (
  (SELECT id FROM ventas WHERE usuario_id=(SELECT id FROM usuarios WHERE
username='admin') ORDER BY id DESC LIMIT 1),
  (SELECT id FROM productos WHERE name='Laptop'),
  1,
  800.00
);

-- ============================================================
-- 8) VOZ
--    Ejemplo minimalista para registro de comandos de voz
-- ============================================================
CREATE TABLE voz_comandos (
    id SERIAL PRIMARY KEY,
    usuario_id INTEGER REFERENCES usuarios(id),
    comando_recibido TEXT NOT NULL,
    resultado_accion TEXT,          -- Se puede guardar un log de lo que ejecutó
    fecha_comando TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Inserción en voz_comandos
INSERT INTO voz_comandos (usuario_id, comando_recibido, resultado_accion)
VALUES (
  (SELECT id FROM usuarios WHERE username='admin'),
  'Agregar Laptop al carrito',
  'Laptop agregada correctamente'
);
```

2.4. Flujo de Trabajo Implementación

6.1. Elección de plataforma de desarrollo del software

Proceso de desarrollo

Para el desarrollo del presente proyecto se utilizó un proceso basado en PUDS (Proceso Unificado de Desarrollo de Software), empleando modelos basados en UML 2.5+, los cuales permitieron definir con claridad los requerimientos, actores, casos de uso, y modelos de comportamiento del sistema. La documentación y diseño se realizó mediante herramientas CASE (Computer-Aided Software Engineering), como Enterprise Architect o Modelio, facilitando así la creación de diagramas estructurados (casos de uso, clases, secuencia, etc.) y asegurando un desarrollo robusto y bien estructurado desde las etapas iniciales hasta la implementación.

Backend

- **Lenguaje:** Python
  Se eligió Python como lenguaje principal del backend debido a su sintaxis clara, soporte activo, y amplia comunidad. Es ideal para proyectos que integran tanto lógica de negocio como capacidades de análisis inteligente, como machine learning.
- **Framework:** Flask o Django
  Para la creación de la API RESTful se considera el uso de **Flask** o **Django**, dos frameworks altamente robustos y populares en el desarrollo backend. Flask ofrece mayor flexibilidad y simplicidad para microservicios, mientras que Django proporciona una estructura más completa para aplicaciones de mayor escala.
- **API RESTful:**
  La comunicación entre el backend y los frontends (web y móvil) se implementa mediante una arquitectura **RESTful**, permitiendo una integración modular, segura y escalable.
- **Machine Learning:**
  Se planea incorporar reglas de asociación mediante librerías como **Scikit-learn** o **Mlxtend** para ofrecer recomendaciones o automatizaciones inteligentes dentro del sistema. Estas bibliotecas permiten generar modelos predictivos y analíticos con facilidad e integrarlos directamente con el backend.
- **Reconocimiento de Voz:**
  Para comandos de voz y accesibilidad, se contempla el uso de **Google Cloud Speech-to-Text** o **Vosk**. Estas herramientas permiten transcribir voz a texto en tiempo real, facilitando la interacción natural del usuario con el sistema.

Frontend Web

- **Framework:** React o Angular
  Para la interfaz web del sistema se utiliza **React** o **Angular**, tecnologías modernas basadas en componentes reutilizables y orientadas a aplicaciones SPA (Single Page Application), lo que permite una experiencia de usuario fluida, responsiva y altamente interactiva.

Frontend Móvil

- **Framework:** Flutter
  Para la implementación de la versión móvil del sistema se utiliza **Flutter**, un framework desarrollado por Google que permite la creación de aplicaciones nativas multiplataforma (iOS y Android) desde una sola base de código, utilizando el lenguaje Dart. Flutter permite construir interfaces modernas, rápidas y con excelente rendimiento.

Base de Datos

- **Motor:** PostgreSQL
  Se eligió **PostgreSQL** por su potencia, confiabilidad, y soporte para características avanzadas como integridad referencial, funciones almacenadas, transacciones ACID, y extensibilidad. Es ideal para manejar grandes volúmenes de datos y operaciones complejas.

Herramientas Adicionales

- **Control de versiones:** Git / GitHub
  Para el control de versiones y colaboración en el equipo de desarrollo se utiliza **Git** junto con **GitHub**, permitiendo una gestión eficiente del código fuente, manejo de ramas y control de versiones seguro.
- **Servicios en la nube:** AWS, Google Cloud o Azure
  El sistema será desplegado en la nube, utilizando plataformas como **Amazon Web Services (AWS)**, **Google Cloud Platform (GCP)** o **Microsoft Azure**. Estas plataformas ofrecen servicios de alta disponibilidad, balanceo de carga, escalabilidad y seguridad. Dependiendo del entorno disponible, el backend puede estar alojado en un contenedor o instancia virtual (por ejemplo, EC2 en AWS), mientras que el frontend puede estar desplegado en servicios como Firebase Hosting o Cloud Run.
- **Architect Enterprise:** Es una herramienta comprensible de diseño y análisis UML, cubriendo el desarrollo de software desde el paso de los requerimientos a través de las etapas del análisis, modelos de diseño, pruebas y mantenimiento. Permitiendo la creación de distintos diagramas, paquetes y clases para un mejor entendimiento del sistema y una mejor abstracción del mismo.

2.5. Flujo de Trabajo Pruebas