INSTITUTO TECNOLÓGICO NACIONAL DE MÉXICO

CAMPUS CULIACÁN

M.C. MARTHA ESTELA VALENZUELA TIRADO
DR. CLEMENTE GARCIA GERARDO
DOCENTES

PROYECTO INTEGRADOR DE INGENIERIA DE SOFTWARE
INGENIERIA WEB
MATERIAS

13:00-14:00 HORAS
HORARIO DE CLASE

AGOSTO-DICIEMBRE
SEMESTRE

BERRELLEZA BELTRÁN ABRAHAM

GUERRERO ROMO FRANCISCO JAVIER

HERRERA ROMERO JORGE ARATH

URIBE LÓPEZ MARCO ARTURO

ELABORADO POR

CULIACÁN, SINALOA, MÉXICO, A MES DE DICIEMBRE DEL AÑO 2021

Índice

Descripción del problema

Suponga que Ud. como Ingeniero en Sistemas Computacionales, ha sido contactado por la empresa agrícola “Palmera S.A. de C.V.”, para solicitarle sus servicios profesionales referente al desarrollo de aplicaciones software para la web y app para dispositivos móviles.

La empresa tiene como giro principal la producción de dátiles (orgánicos y no orgánicos) de diferentes variedades. Actualmente el control del proceso de producción, cosecha, envasado y venta de dátiles se realiza de forma manual (los empleados administrativos y del campo con que cuenta la empresa), debido a que la empresa no cuenta con ningún sistema OLTP que les facilite el trabajo que se realiza diariamente. De ahí la necesidad de desarrollar la aplicación web y la app para tener una presencia en todo el país y facilite a los empleados realizar las labores del día sin necesidad de esperar ordenes de sus jefes. Los clientes podrán realizar los pedidos del producto sin la intervención de los empleados.

La agrícola hoy en día tiene destinados N predios (lotes, parcelas, área, terrenos) donde tienen plantadas una serie de palmeras para la producción de los dátiles.

La agrícola tiene varios especialistas en palmeras que realizan las siguientes actividades:

- Registrar predios
- Determinar qué predio son adecuados para producción orgánica del dátil. Esto significa que el 100% de las palmeras producen dátil orgánico.
- Determinar qué palmera pueden producir dátil orgánico en un predio no adecuado para producción orgánico.
- Definir las actividades que se tiene que realizar de manera particular a cada una de las palmeras que producen dátil orgánico en un predio no orgánico (diferentes tareas a cada palmera).
- Definir las actividades que se tiene que realizar a las palmeras que producen dátil orgánico en un predio orgánico (las mismas tareas para todas las palmeras).
- Definir las actividades que se tiene que realizar a las palmeras que están en lotes que no son para producción orgánica.

Ejemplo de actividades:

- Eliminar hierba
- Podar palmera
- Preparar tasa para riego
- Poner veneno
- Realizar riego
- Poner fertilizante
- Aclarar racimo (para hacer que el dátil sea de más calidad).
- Poner red
- Recolectar (esta actividad se puede realizar más de una ocasión)
- Quitar red

La recolección en palmeras que produce dátiles orgánicos tiene un control estricto en el cumplimiento de las fechas de todas las actividades que se la hayan asignado (tolerancia, máximo dos actividades puede retrasarse un día), de no cumplirlas la producción será considerada no orgánica.

Algunas políticas de la empresa:

Los predios son identificados con el prefijo “P” concatenado con número entero de 3 dígitos (ejemplo P001, P010).

Las palmeras son identificadas con el prefijo “PAL” y número consecutivo de 3 dígitos(ejemplo PAL001,PAL009).

La venta del producto se realiza con cargo a tarjeta de crédito.

La cosecha se almacena en contenedores de 1,000 kilogramos.

Se puede vender a partir de un kg.

El método usado para el control del producto en almacén es el PEPS.

La compañía “Palmera S.A. de C.V.” no tiene presencia Web, quiere lanzar un sitio web (www.palmeradatil.com) y una aplicación móvil (consultar la existencia de productos), donde los diferentes usuarios puedan gestionar todos los procesos relacionados con la actividad del dátil.

Tecnología a utilizar:

- Metodología de desarrollo: PU (proceso unificado), SCRUM O XP.
- CASE: Enterprise architect
- Notación de modelado: UML y UWE.
- Marco de desarrollo: YII, LARAVEL (MVC -modelo vista controlador-)
- Paradigma de desarrollo: Orientado a objetos
- Android estudio.
- Sistema Gestor de Base de Datos (OO, Relacional).
- Servicio web para el predio.
- Debe desarrollar un servicio web que realice la autorización oficial de los predios para la producción de dátil orgánico.
- La aplicación debe sugerir el precio de venta del producto.

Fase de inicio

Visión y análisis del negocio

Historial de versiones

Versión	Fecha	Descripción	Autor
Borrador de inicio	17/Sep/2021	Primer borrador: Para refinarse durante la elaboración.	Equipo 5

Introducción

La empresa “Palmera S.A. de C.V.” requiere de un sistema OLTP con aplicación web y móvil, en el que los empleados puedan llevar a cabo las actividades que se llevaban previamente a mano como: administrar los predios orgánicos y no orgánicos, gestionar los cuidados que necesitan las palmeras individualmente y los predios.

Enunciado del problema

Llevar el control de los procesos que realiza la empresa de forma manual es ineficiente y ralentiza las actividades de esta. el hecho que el control de información importante, como las actividades que se les van a realizar a las palmeras y que actividades ya fueron hechas, se registre con medios analógicos y se transmita de manera oral, hace más lenta la comunicación y permite más errores. Un sistema OLTP facilitaría el acceso a la información que los empleados necesitan para trabajar. También cada vez que un cliente quiera realizar un pedido, este necesita que un empleado lo atienda, limitando la accesibilidad y alcance de los servicios. En conclusión, los agricultores necesitan saber rápidamente que actividades van a hacer, los especialistas necesitan almacenar la información sobre los predios y palmeras de manera fiable y los clientes necesitan poder realizar pedidos en cualquier momento y desde cualquier lugar.

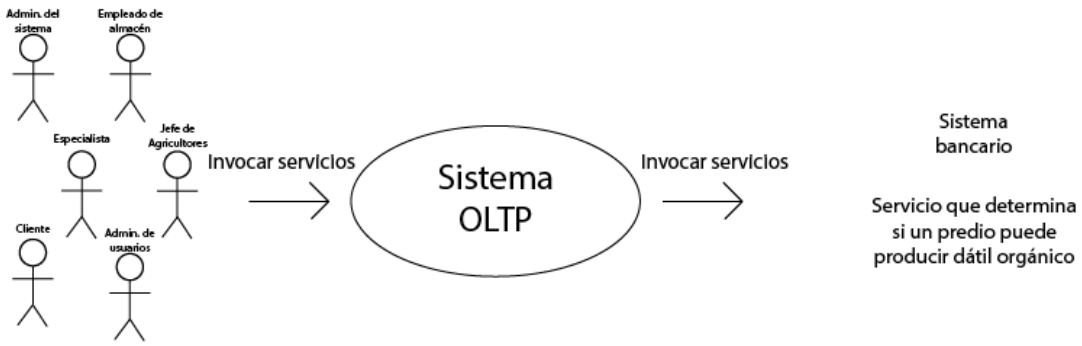
Objetivos de alto nivel y problemas claves del personal involucrado

Objetivo de alto nivel	Prioridad	Problema e inquietudes	Soluciones actuales
Obtener las actividades que se le van a realizar a las palmeras y reportar su finalización.	Alta	La asignación de actividades y la finalización de estas, es información que debe ser comunicada de la manera más rápida y fiable	Los trabajadores se reportan con un superior cada vez que finalizan sus actividades para que se les asignen nuevas
Llegar a la mayor cantidad de clientes posibles	Alta	Atender los pedidos de los clientes	Los empleados atienden a los clientes de forma presencial
Manejar la producción de dátiles orgánicos y no orgánicos	Alta	Saber que palmeras y que lotes producen dátiles orgánicos y no orgánicos	Se lleva un registro de forma analógica
Manejar las entradas y salidas de producto a contenedores	Alta	El registro de las entradas y salidas de producto tiene que ser fiable	Se lleva un registro de forma analógica

Objetivos a nivel usuario

- **Especialista**: Registrar predios, determinar qué predio son adecuados para producción orgánica, determinar qué palmera pueden producir dátil orgánico, definir las actividades que se tiene que realizar de manera particular a cada una de las palmeras que producen dátil orgánico en un predio no orgánico, a cada predio orgánico y a todos los predios no orgánicos.
- **Jefe de agricultores**: Asignar a los agricultores las actividades agendadas por el especialista.
- **Cliente**: Realizar pedidos.
- **Administrador de almacenes:** Realizar el alta baja de contenedores del almacén.
- **Empleado de almacén:** Realizar las entradas y salidas del producto en el almacén.
- **Administrador del sistema:** Realizar el alta baja de usuarios.

Perspectiva del producto



Resumen de beneficios

Característica soportada	Beneficio del personal involucrado
Control de procesos de producción, cosecha, envasado y venta	Aumento de la velocidad de las actividades y reducción del esfuerzo requerido
Pedidos en línea	Ya no se necesitarán empleados para atender todos los pedidos de los clientes
Manejo de la producción de dátiles orgánicos y no orgánicos	Se reduce la probabilidad de errores

Resumen de las características del sistema

- Registrar predios.
- Administración de predios orgánicos y no orgánicos.
- Administración de palmeras orgánicas y no orgánicas.
- Administración de actividades.
- Realización de pedidos.
- Gestión de contenedores en el almacén.
- Gestión de usuarios.

Especificación Complementaria

Historial de revisiones

Versión	Fecha	Descripción	Autor
Borrador de inicio	17-09-2021	Primer borrador: Para refinarse durante la elaboración.	Equipo 5

Introducción

Este documento es el repositorio de todos los requisitos del Proyecto “Palmera S.A. de C.V.” que no se capturan en los casos de uso.

Funcionamiento

Registro y gestión de errores:
Se rechazarán los datos capturados erróneamente y se le notificará al usuario.

Seguridad:

- El acceso al sistema requerirá autenticación de los usuarios.
- La información de los clientes estará protegida de agentes externos.

Facilidad de uso

- La navegación por las diversas interfaces del sistema será sencilla.
- Las interfaces de registro serán minimalistas.
- Comunicar errores con claridad.
- Prevenir errores.

Fiabilidad

Capacidad de recuperación:
Si se produce algún fallo al usar un servicio externo, intentar tratar el error con una solución local para la continuidad del proceso que se esté llevando a cabo.

Rendimiento

- El servicio de registro de los predios deberá completarse en un minuto o menos el 90% de las veces.
- El cliente debe de poder hacer un pedido a la empresa en un minuto o menos, en el 90% de las veces.
- El servicio que determina si el predio es apto para producción orgánica deberá completarse en un minuto o menos el 90% de las veces.
- El servicio que determina si una palmera es orgánica o no deberá completarse en un minuto o menos el 90% de las veces.
- El servicio de asignación de actividades deberá completarse en un minuto o menos el 90% de las veces.
- El servicio de registrar una salida o entrada del almacén deberá completarse en un minuto o menos el 90% de las veces.
- El servicio de registrar o dar de baja un usuario deberá completarse en un minuto o menos el 90% de las veces.

Interfaces

Interfaces y hardware destacable:

- Computadores.
- Dispositivos móviles.
- Servidor web.
- Servidor de base de datos.

Interfaces software:

- Sistema bancario.
- Servicio que determina si un predio es orgánico o no.

Reglas del dominio (negocio)

ID	Regla	Grado de variación	Fuente
REGLA1	Se requiere identificar un predio con el prefijo “P” concatenado con un número entero de 3 dígitos.	Sin grado de variación.	Políticas de la empresa.
REGLA2	Se requiere identificar una palmera con el prefijo “PAL” y número consecutivo entero de 3 dígitos.	Sin grado de variación.	Políticas de la empresa.
REGLA3	La venta del producto se realizará con cargo a tarjeta de crédito.	Sin grado de variación.	Políticas de la empresa.
REGLA4	La venta será a partir de un kg.	Sin grado de variación.	Políticas de la empresa.
REGLA5	La cosecha será almacenada en contenedores de 1,000 kg.	Sin grado de variación.	Políticas de la empresa.
REGLA6	El método PEPS será utilizado para el control del producto en almacén.	Sin grado de variación.	Políticas de la empresa.
REGLA7	Las actividades tienen como tolerancia máxima 1 día de retraso,	Sin grado de variación.	Organismo externo de clasificación de dátiles.

	de no cumplirse, la cosecha será considerada no orgánica.		

Modelo de casos de uso

Lista Actor-Semántica

Actores	Descripción
Especialista	Empleado que se encarga de la evaluación de las palmeras y los predios, así como de la asignación de cuidados de estos
Jefe de agricultores	Empleado a cargo de los agricultores, encargado de asignarles las actividades agendadas por el especialista
Cliente	Persona que desea comprar un producto de la empresa
Administrador de almacenes	Empleado encargado de realizar la alta y baja de los contenedores del almacén
Empleado de almacén	Empleado encargado de llenar y vaciar los almacenes
Administrador del sistema	Empleado encargado de registrar la alta y baja en consulta de usuarios y sus permisos en el sistema

Lista Actor-Objetivo

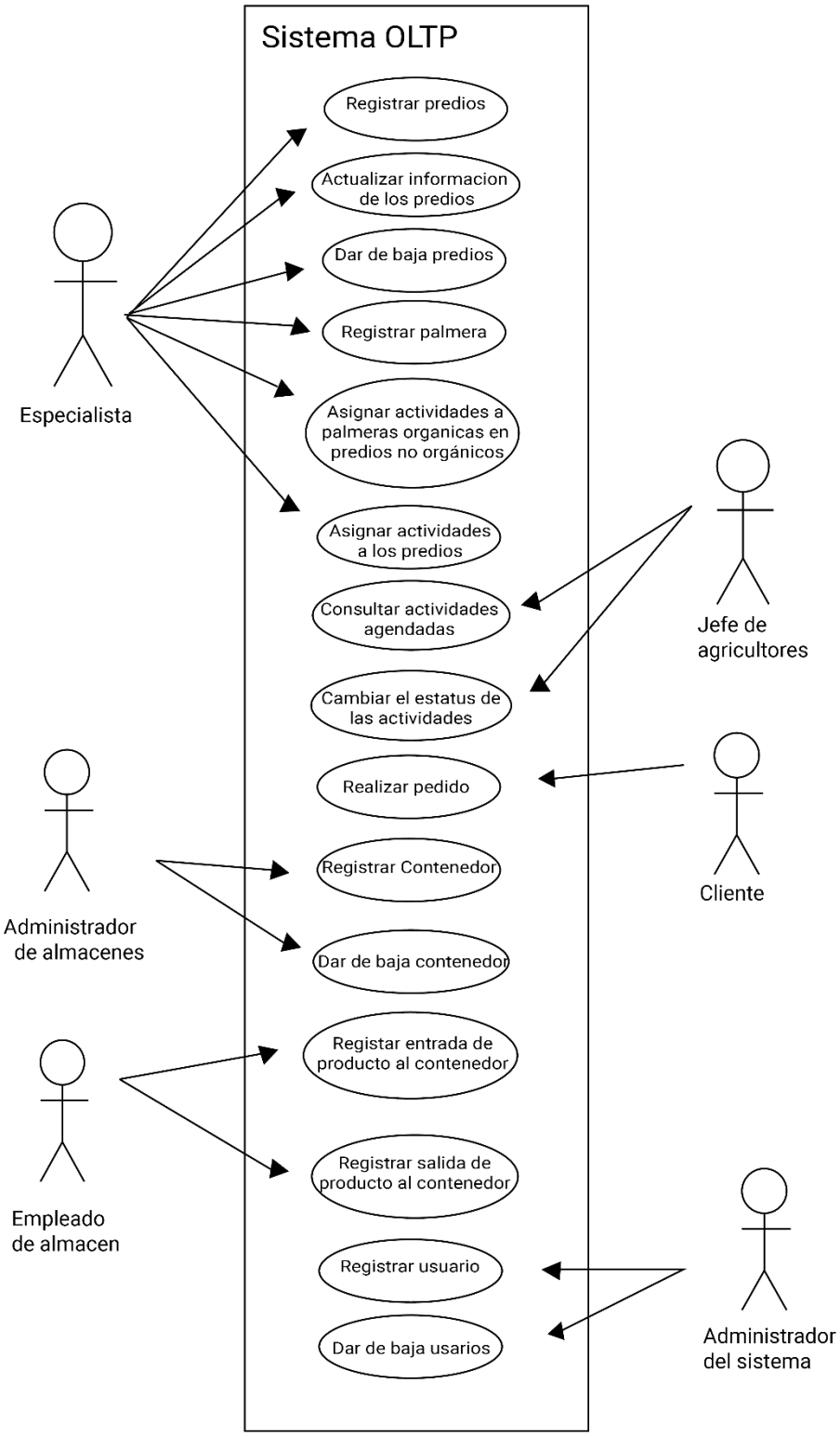
Actores	Objetivo
Especialista	● Registrar predios ● Registrar palmeras ● Evaluar predios ● Evaluar palmeras ● Asignar actividades a predios ● Asignar actividades a palmeras
Jefe de agricultores	● Consultar actividades agendadas ● Cambiar el estatus de las actividades
Cliente	● Realizar pedido
Administrador de almacenes	● Registrar contenedor ● Dar de baja contenedores
Empleado de almacén	● Registrar entrada de producto al contenedor ● Registrar salida de producto al contenedor
Administrador del sistema	● Registrar usuario

	• Dar de baja usuarios

Lista de casos de uso

- CU01 - Registrar predios.
- CU02 - Actualizar información de predios.
- CU03 - Dar de baja predios.
- CU04 - Registrar palmeras.
- CU05 - Asignar actividades a predios.
- CU06 - Asignar actividades a palmeras orgánicas en predios no orgánicos.
- CU07 - Consultar actividades agendadas.
- CU08 - Cambiar el estatus de las actividades.
- CU09 - Realizar pedido.
- CU10 - Registrar contenedor.
- CU11 - Dar de baja contenedores.
- CU12 - Registrar entrada de producto al contenedor.
- CU13 - Registrar salida de producto al contenedor.
- CU14 - Registrar usuario.
- CU15 - Dar de baja usuarios.

Diagrama de casos de uso



Descripción breve de casos de uso

- Registrar predios
    1. El especialista captura los datos (como la localización, su tamaño, humedad del suelo, temperatura del suelo, etc.) del predio.
    2. El sistema da de alta el predio.
- Actualizar información de predios
    1. El sistema muestra los predios y su información.
    2. El especialista selecciona el predio cuya información va a actualizar.
    3. El especialista captura la información actualizada.
    4. El sistema actualiza la información del predio.
- Dar de baja predios
    1. El sistema muestra los predios y su información.
    2. El especialista selecciona el predio que dará de baja.
    3. El sistema da de baja el predio.
- Registrar palmeras
    1. El especialista captura los datos (variedad de dátil, predio en el que se encuentra, altura, ancho, horas de luz recibidas al año, color de las hojas, etc.) de la palmera.
    2. El sistema invoca el servicio externo de valoración que indica si una palmera es apta para producción orgánica, muestra el resultado y da de alta la palmera.
- Asignar actividades a predios
    1. El especialista selecciona el predio a los que les asignará actividades.
    2. El especialista selecciona las actividades que necesita el predio, cada cuanto tiempo y por cuanto tiempo serán realizadas.
    3. El sistema agenda las actividades a todas las palmeras en ese predio.
- Asignar actividades a palmeras orgánicas en predios no orgánicos.
    1. El especialista selecciona la palmera a la que les asignará actividades.
    2. El especialista selecciona las actividades que necesita la palmera, cada cuanto tiempo y por cuanto tiempo serán realizadas.
    3. El sistema agenda las actividades a la palmera.
- Consultar actividades agendadas
    1. El sistema le muestra al usuario las actividades que deben realizarse ese día y las actividades atrasadas.
    2. El jefe de agricultores asigna las actividades a los agricultores.
    3. El sistema marca las actividades como “En proceso”.
    4. El agricultor comienza a realizar sus actividades.
- Cambiar el estatus de las actividades
    1. El agricultor termina su actividad(es) y notifica a su jefe.
    2. El jefe de agricultores selecciona el predio o la palmera al que pertenece la actividad.
    3. El sistema muestra las actividades asignadas a ese predio o palmera.
    4. El jefe de agricultores selecciona la actividad(es) que ya se realizaron.
    5. El sistema marca la actividad(es) como realizada(s).
- Realizar pedido

1. El sistema muestra las variedades de dátil disponibles.
2. El cliente selecciona una o más variedades y la cantidad que quiere de cada una.
3. El sistema valida de nuevo la disponibilidad de los productos.
4. El sistema se comunica con el sistema bancario para validar el pago.
5. El sistema registra la compra y agenda la entrega.

- Registrar contenedor
    1. El administrador de almacenes captura los datos (como capacidad y localización) del nuevo contenedor.
    2. El sistema da de alta el contenedor.
- Dar de baja contenedores
    1. El administrador de almacenes selecciona el contenedor a dar de baja.
    2. El administrador de almacenes captura el motivo de la baja.
    3. El sistema da de baja el contenedor.
- Registrar entrada de producto al contenedor
    1. El empleado de almacén captura la cantidad de producto y el contenedor al que ingresó.
    2. El sistema registra la entrada de producto.
- Registrar salida de producto al contenedor
    1. El sistema muestra la variedad y cantidad de dátil de cada pedido que se encuentre pendiente por embolsar.
    2. El trabajador del almacén saca el producto y lo lleva a embolsar.
    3. El trabajador del almacén marca el pedido como listo para enviar.
    4. El sistema registra la salida de producto.
- Registrar usuario
    1. El administrador del sistema captura los datos (como nombre, apellidos, correo, puesto, etc.) personales del empleado.
    2. El sistema da de alta al usuario.
- Dar de baja usuarios
    1. El administrador del sistema busca al empleado por nombre y lo selecciona.
    2. El sistema muestra los datos del empleado.
    3. El administrador del sistema captura el motivo de la baja de ese usuario.
    4. El sistema da de baja al usuario.

Caso de uso CU01: Registrar predios

Actor Principal:

Especialista

Personal involucrado e intereses:

Especialista: Quiere la correcta captura de los datos de los predios y las palmeras.

Jefe de agricultores: Quiere saber en qué predios están las palmeras.

Empresa: Quiere que sus empleados dispongan de la información necesaria para la correcta realización de sus actividades en todo momento.

Precondiciones:

El especialista se identifica y autentica.

Garantías de éxito (postcondiciones):

- Se registra el predio.
- Se registra si el predio es apto para producción orgánica o no.

Escenario principal de éxito (o Flujo Básico):

1. El especialista inicia el registro del predio.
2. El especialista captura la humedad del suelo, el tamaño, localidad en la que se encuentra, temperatura promedio, y el pH del suelo.
3. El especialista captura el tipo del predio.
4. El sistema registra el predio.
5. El especialista captura las coordenadas de los vértices del predio.
6. El sistema guarda las coordenadas de los vértices del predio.

Extensiones (o Flujos Alternativos):

a. En cualquier momento el sistema falla:

1. El especialista reinicia el sistema, inicia sesión y vuelve a empezar el registro.

3. El especialista captura que el predio es orgánico:

1. El sistema envía los datos del predio al servicio de valoración.

1a. El servicio de valoración no responde

1. El especialista pospone el registro

2. El sistema muestra el resultado de la valoración.

2a. El servicio determina que el predio es orgánico.

1. Se retoma el flujo desde el paso 4 del flujo básico.

2b. El servicio determina que el predio no es orgánico.

1a. El especialista está de acuerdo con el resultado y retoma desde el paso 4 del flujo básico.

1b. El especialista no está de acuerdo con el resultado y cancela el registro del predio.

4a.El especialista capturó datos inválidos:

1. El sistema alerta al especialista que los datos son inválidos.
2. El sistema retoma desde el paso 2 del flujo básico.

4c.El especialista capturó menos de 3 coordenadas de vértices del predio:

1. El sistema alerta al especialista que el predio debe tener cuando menos 3 coordenadas.
2. El sistema retoma desde el paso 5 del flujo básico.

Requisitos especiales:

- El servicio de registro de los predios deberá completarse en un minuto o menos el 90% de las veces.
- El servicio que determina si el predio es apto para producción orgánica deberá completarse en un minuto o menos el 90% de las veces.

Frecuencia:

Cada que se compra un terreno o hay un nuevo predio.

Temas abiertos:

- ¿Los predios se evalúan más de una vez?
- ¿La empresa tiene intenciones de añadir la evaluación de los predios a su sistema, sin utilizar un servicio externo?

Caso de uso CU09: Realizar pedido

Actor Principal:

Cliente

Personal involucrado e intereses:

Cliente: Quiere realizar su compra con satisfacción.

Empleado de almacén: Quiere conocer cuánto y que producto debe sacar del almacén para envasar.

Empresa: Quiere que su producto esté disponible para la compra, llevar el proceso de venta con satisfacción y de manera confiable para sus clientes, que su inventario se actualice automáticamente.

Precondiciones:

El cliente se identifica y autentica.

Garantías de éxito (postcondiciones):

- Se registra la venta.
- Se envía la información de la venta al sistema de contabilidad.
- Se actualiza el inventario.

Escenario principal de éxito (o Flujo Básico):

1. El cliente inicia una compra.
2. El sistema muestra las variedades de dátiles que vende la empresa.
3. El cliente selecciona la variedad de dátil que quiere comprar y la cantidad.
4. El sistema añade al carrito la compra.
5. El sistema muestra los detalles (precio del producto, cantidad del producto, precio total a pagar) del pedido.
6. El cliente proporciona los datos de su tarjeta de crédito y dirección, y confirma la compra.
7. El sistema envía los datos del cliente al sistema bancario para validar el pago.
8. El sistema registra la venta.

Extensiones (o Flujos Alternativos):

1. El cliente ya contaba con productos en su carrito de compras:

    1. El sistema recupera el carrito de compras.
    2. El sistema continua el flujo básico.

1-4a. El sistema falla:

    1. El cliente sale de la aplicación, vuelve a iniciar sesión y vuelve a empezar la compra.

1-5b. El cliente quiere cancelar su compra:

    1. El cliente cancela la compra.
    2. El sistema regresa al paso 1 del flujo básico.

3a. La variedad de dátil no se encuentra disponible:

    1. El sistema muestra los detalles del producto y la leyenda “no disponible”
    2. El sistema regresa al paso 2 del flujo básico.

3b. El cliente no se ha autenticado:

    1. El sistema le pide al cliente que inicie sesión
    2. El sistema regresa al paso 1 del flujo básico.

4-7a. El sistema falla:

    1. El cliente sale de la aplicación y vuelve a iniciar sesión.
    2. El sistema recupera el carrito de compras.
    3. El sistema regresa al paso 2 del flujo básico.

4-7b. El cliente quiere cancelar su compra:

    4. El sistema deshace el carrito de compras.
    5. El sistema regresa al paso 1 del flujo básico.

5a. La cantidad deseada es mayor a la disponible:

    1. El sistema alerta al cliente que no hay la cantidad deseada y muestra la cantidad máxima que puede pedir.
    2. El sistema regresa al paso 3 del flujo básico.

7a. El sistema bancario falla:

    1. El sistema alerta al cliente que ocurrió un fallo con la compra.
    2. El sistema regresa al paso 6 del flujo básico.

8b. Los datos de la tarjeta de crédito son erróneos:

1. El sistema alerta al cliente que los datos de la tarjeta de crédito son erróneos y que ingrese una nueva tarjeta de crédito valida.
2. El sistema regresa al paso 6 del flujo básico.

Requisitos especiales:

- El cliente debe de poder hacer un pedido a la empresa en un minuto o menos, en el 90% de las veces.

Frecuencia:

Puede realizarse muchas veces a lo largo de todo el día.

Temas abiertos:

- ¿Se pretenden implementar otros métodos de pago a futuro?
- ¿El sistema maneja devoluciones?

Caso de uso CU05: Asignar actividades a predios

Actor Principal:

Especialista

Personal involucrado e intereses:

Especialista: Quiere que se realicen las actividades necesarias a las palmeras.

Jefe de agricultores: Quiere saber que actividades le tienen que hacer las palmeras.

Empresa: Quiere maximizar su producción con el correcto cuidado a las palmeras y que sus empleados dispongan de la información necesaria para la correcta realización de sus actividades en todo momento.

Precondiciones:

El especialista se identifica y autentica.

Garantías de éxito (postcondiciones):

- Se agendan las actividades para su realización.
- Los jefes de agricultores podrán consultar las actividades agendadas.

Escenario principal de éxito (o Flujo Básico):

1. El especialista inicia la asignación de actividades.
2. El especialista captura la clave del predio cuyas palmeras se les asignarán las actividades.
3. El sistema muestra las actividades que ya tienen agendadas las palmeras de ese predio.
4. El especialista captura las actividades que se le realizarán al predio, cuantas veces a la semana, así como las fechas de inicio y fin del periodo en el que se realizarán las actividades.
5. El sistema registra las asignaciones de las actividades a todas las palmeras del predio seleccionado, exceptuando las palmeras orgánicas en predios no orgánicos.

Extensiones (o Flujos Alternativos):

a. En cualquier momento el sistema falla:

1. El especialista reinicia el sistema, inicia sesión y vuelve a empezar la asignación de actividades.

2a. La clave del predio ingresado no existe:

1. El sistema alerta al especialista que la clave no corresponde a ningún predio existente.
2. El sistema regresa al paso 1 del flujo básico.

Requisitos especiales:

- El servicio de asignación de actividades deberá poder completarse en un minuto o menos el 90% de las veces.

Frecuencia:

En cualquier momento de todo el año.

Temas abiertos:

- ¿Hay actividades que se realicen al suelo de todo el predio?

Caso de uso CU06: Asignar actividades a palmeras orgánicas en predios no orgánicos.

Actor Principal:

Especialista

Personal involucrado e intereses:

Especialista: Quiere que se realicen las actividades necesarias a las palmeras.

Jefe de agricultores: Quiere saber que actividades le tienen que hacer las palmeras.

Empresa: Quiere maximizar su producción con el correcto cuidado a las palmeras y que sus empleados dispongan de la información necesaria para la correcta realización de sus actividades en todo momento.

Precondiciones:

El especialista se identifica y autentica.

Garantías de éxito (postcondiciones):

- Se agendan las actividades para su realización.
- Los jefes de agricultores podrán consultar las actividades agendadas.

Escenario principal de éxito (o Flujo Básico):

1. El especialista inicia la asignación de actividades.
2. El especialista captura la clave de la palmera a la que le asignará las actividades.
3. El sistema muestra las actividades que tiene asignadas la palmera.
4. El especialista captura las actividades que se le realizarán a la palmera, cuantas veces a la semana, así como las fechas de inicio y fin del periodo en el que se realizarán las actividades.
5. El sistema registra las asignaciones de las actividades a la palmera seleccionada.

Extensiones (o Flujos Alternativos):

a. En cualquier momento el sistema falla:

1. El especialista reinicia el sistema, inicia sesión y vuelve a empezar la asignación de actividades.

2a. La clave de la palmera ingresada no existe:

3. El sistema alerta al especialista que la clave no corresponde a ninguna palmera existente.
4. El sistema regresa al paso 1 del flujo básico.

Requisitos especiales:

- El servicio de asignación de actividades deberá poder completarse en un minuto o menos el 90% de las veces.

Frecuencia:

En cualquier momento de todo el año.

Glosario de términos

Dátil: El dátil es el fruto obtenido de las especies de palmeras Phoenix, principalmente de la especie Phoenix dactylifera, llamada popularmente palmera datilera, dicho fruto es el producto que cultiva y comercializa la compañía.

Predio: Finca, tierra o posesión inmueble, en la donde la compañía cultivas el dátil.

Variedad de dátil: Las diferentes variantes de dátiles que la empresa produce y comercializa.

pH: Concentración de iones de hidrógeno presentes en el suelo de los predios en los que se cultiva el dátil, importante para la capacidad del predio de producir dátil orgánico.

Horas luz al año: La cantidad de horas al año de rayos o luz solar que reciben las palmeras.

Temperatura promedio anual: Un promedio de temperatura a lo largo del año en cada predio.

Asignación: Fecha en la que tiene agendada una determinada palmera la realización de una determinada actividad necesaria para su cuidado.

Actividad: Acción que se requiere realizar a una palmera de algún determinado predio.

Dátil orgánico: un dátil de cualquier variedad que cumpla con ciertos criterios de calidad

Palmera Orgánico: Palmeras que cumplen con los criterios para producir dátil orgánico.

Predio Orgánico: Predio que contiene palmeras catalogadas como orgánicas que pueden producir dátil orgánico.

Contenedor: Recipiente metálico en el cual se almacena a la producción de dátiles tras en cultivo.

Localidad: Pueblo o comunidad a la que pertenece el predio.

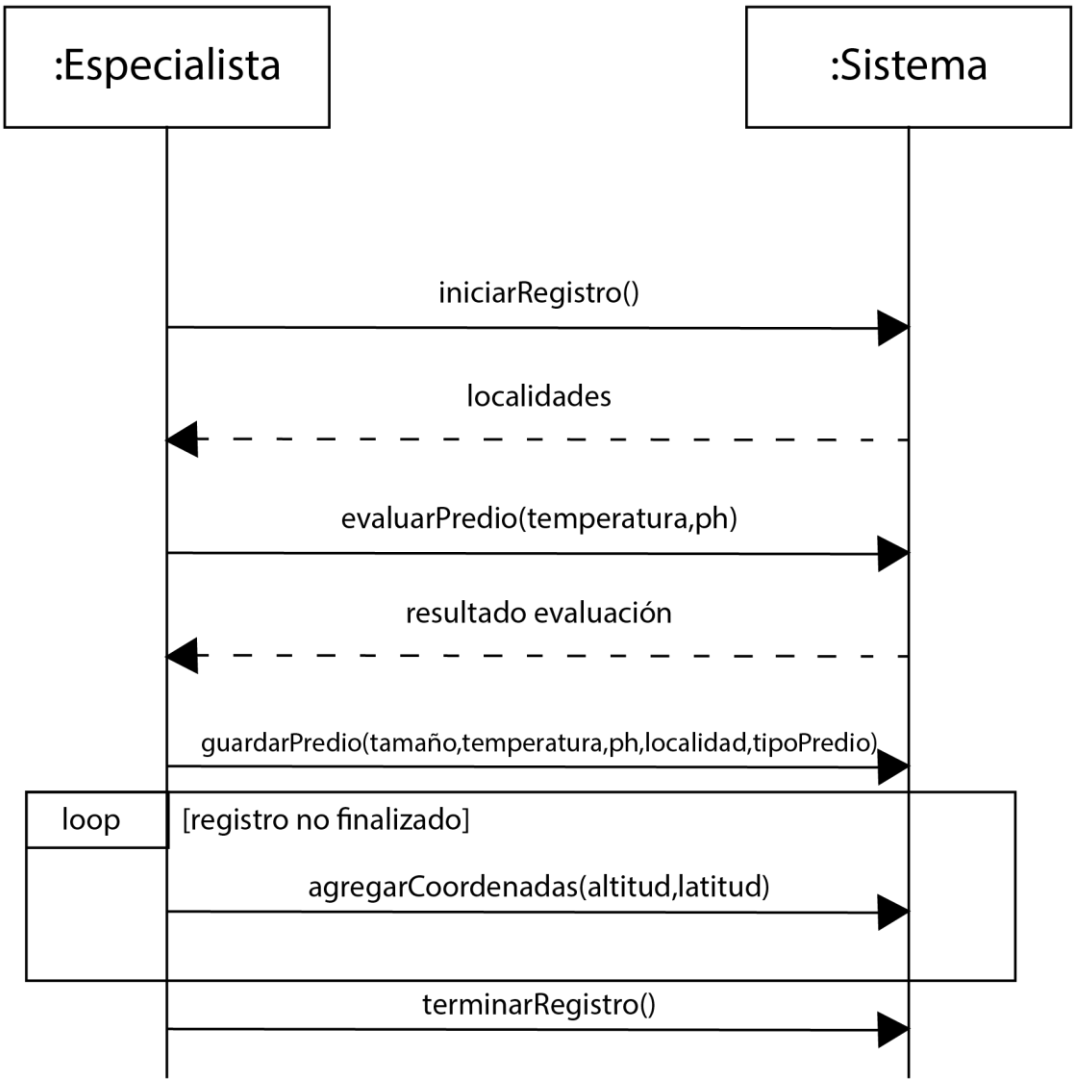
Servicio de validación: Servicio el cual determina si un predio es orgánico o no, mediante el análisis de ciertos criterios.

Fase de elaboración

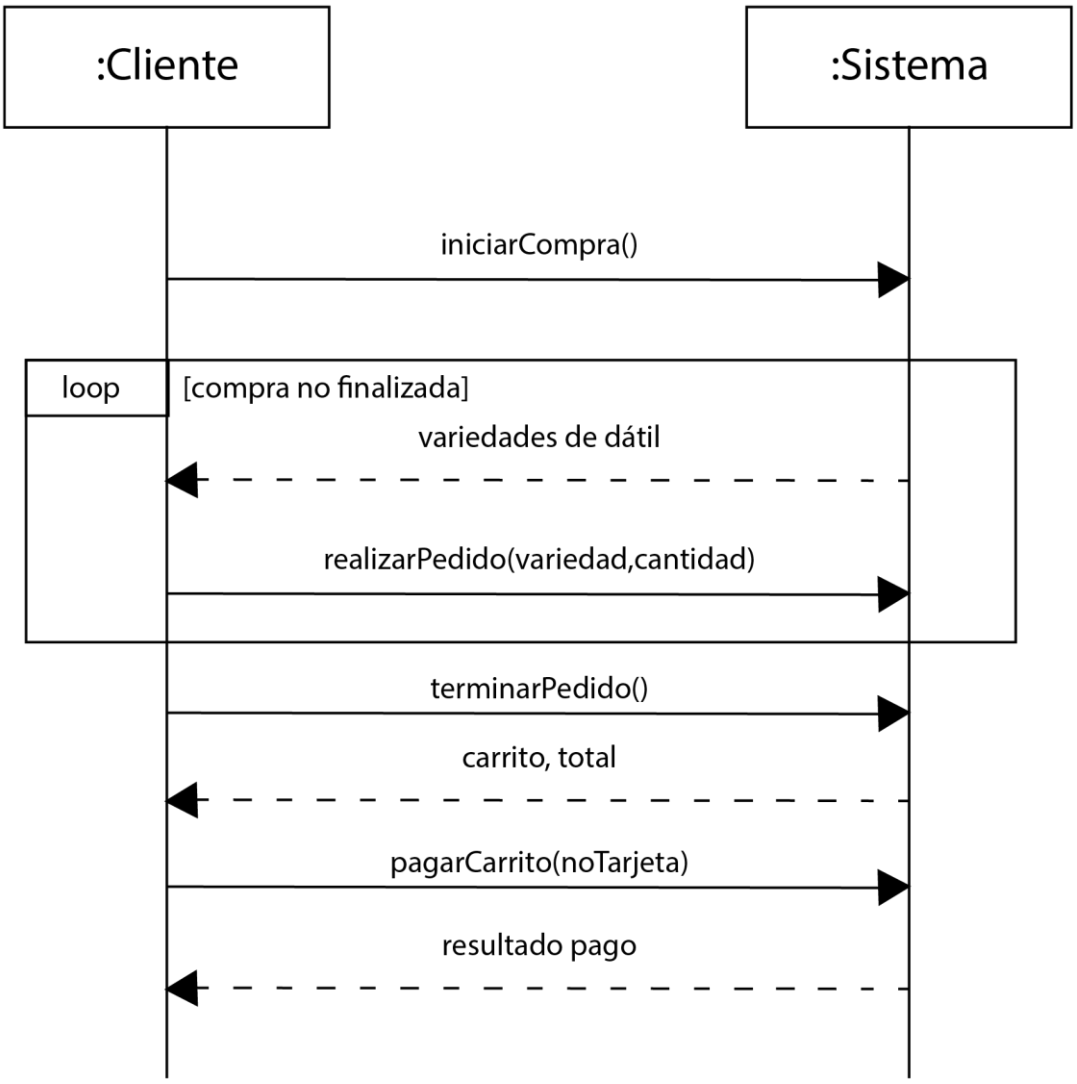
Análisis

Diagramas de secuencia del sistema

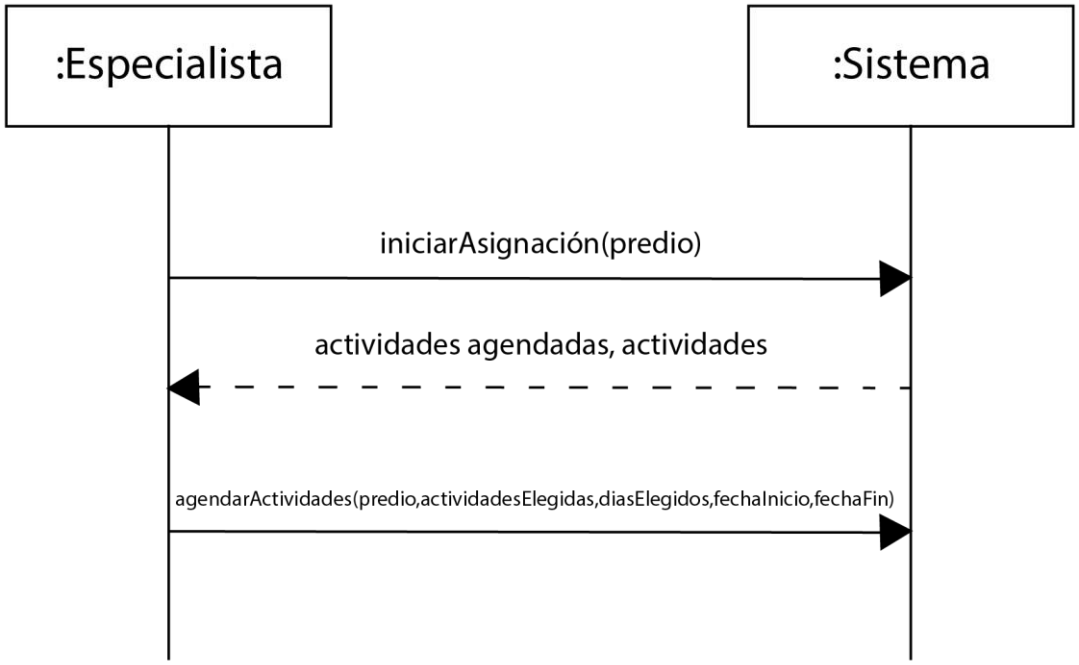
CU01 – Registrar predio



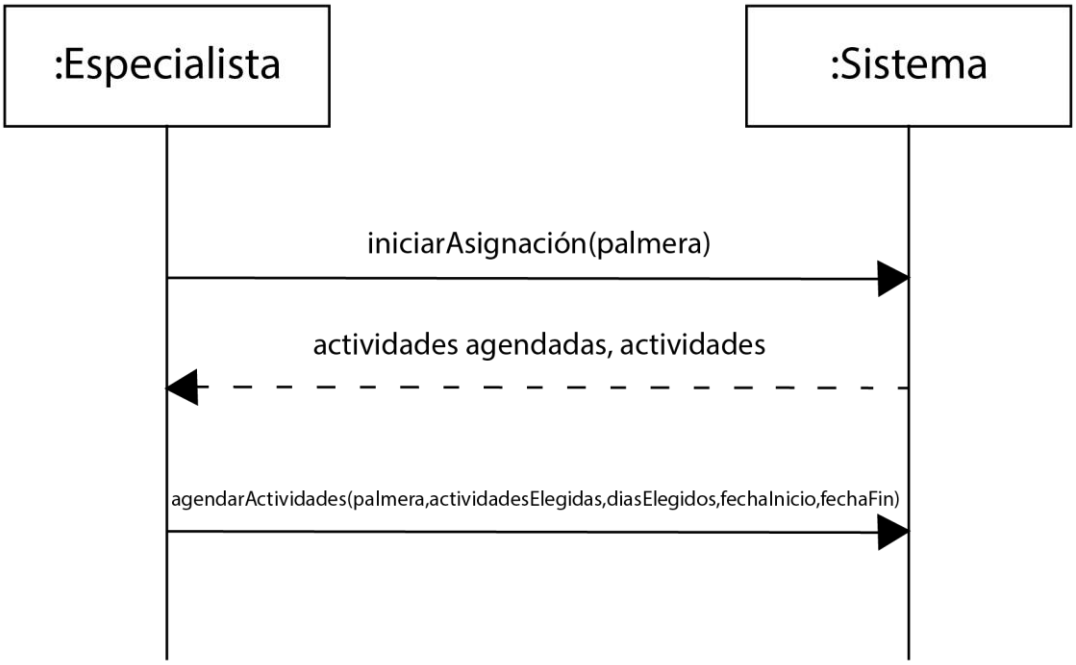
CU09 – Realizar pedido



CU05 - Asignar actividades a predios



CU06 - Asignar actividades a palmeras orgánicas en predios no orgánicos



Diagramas de clases conceptuales

Lista categorizada de clases

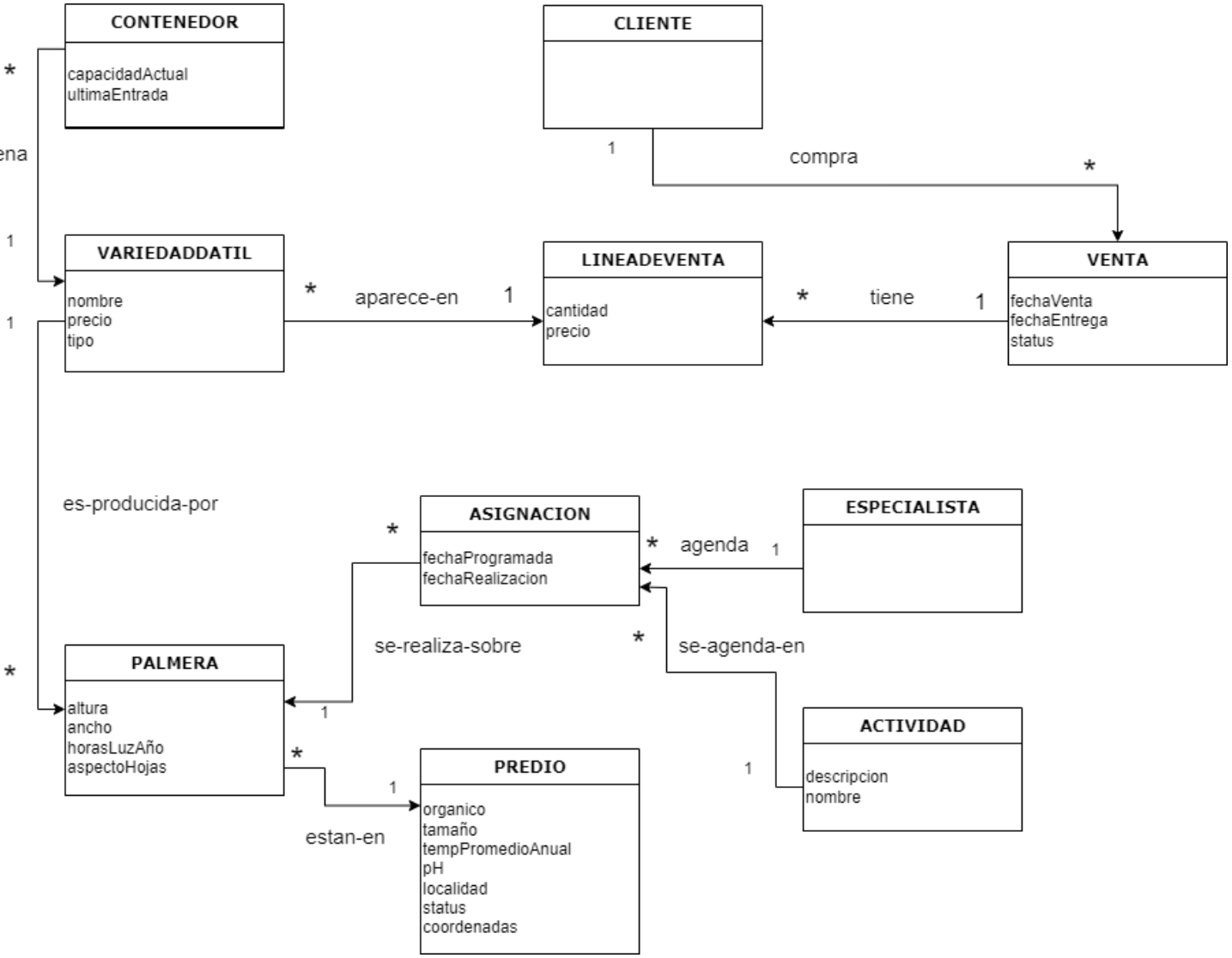
Objetos tangibles o físicos	Predio, Dátil, Tarjeta, Producto, Palmeras
Especificaciones, diseños	Calendario
Lugares	Almacén, Empresa
Transacciones	Registro, Pago, Compra, Agenda, Venta, Pedido, Asignación, Actividades
Roles de la gente	Especialista, Cliente
Otros sistemas informáticos	Sistema bancario
Contenedores	Carrito, Inventario

Clases conceptuales candidatas

- Predio
- Dátil - Atributo, convertido en variedad de dátil
- Tarjeta - Irrelevante para el sistema
- Producto - Atributo, convertido en variedadDeDatil
- Palmera
- Calendario - Irrelevante para el sistema
- Almacén - Convertido en Contenedor
- Empresa - Irrelevante para el sistema
- Registro - Descartada por ser una operación
- Pago - Convertido en Venta
- Compra - Convertido en Venta
- Agenda - Descartada por ser una operación
- Venta
- Pedido - Convertido en Venta
- Asignación
- Actividades - Atributo, convertido en actividad
- Especialista
- Cliente
- Sistema bancario - Sistema externo
- Carrito - Convertido en Venta
- Inventario - Convertido en Contenedor
- Humedad del suelo - Atributo
- Tamaño - Atributo
- Localización - Atributo
- Temperatura - Atributo
- pH del suelo - Atributo
- Estatus - Atributo
- Variedad - Atributo, convertido en variedadDeDatil

- Cantidad - Atributo
- Existencia - Atributo, convertido en Cantidad
- Precio del producto - Atributo
- Precio del envío - Atributo
- Cantidad del producto - Atributo
- Precio total - Atributo
- Dirección - Atributo
- Fecha de llegada - Atributo
- Clave del predio - Atributo
- Cuantas veces al día - Atributo
- Cuantas veces a la semana - Atributo
- Fecha inicio - Atributo
- Fecha fin - Atributo
- Clave de la palmera - Atributo

Diagrama de clases conceptuales



Contratos

C01: iniciarRegistro
Operación: iniciarRegistro()

Referencia cruzada: CU01– Registrar predio

Precondiciones: El especialista debe haberse identificado y autenticado

Postcondiciones:

-

C02: evaluarPredio
Operación: evaluarPredio(temperatura,ph)

Referencia cruzada: CU01– Registrar predio

Precondiciones: Hay un registro de predio en curso.

Postcondiciones:

-

C03: guardarPredio
Operación: guardarPredio(tamaño,temperatura,ph,localidad,tipoPredio)

Referencia cruzada: CU01– Registrar predio

Precondiciones: Hay un registro de predio en curso.

Postcondiciones:

- Se creo una instancia de Predio predio y se inicializaron sus atributos.

C04: agregarCoordenada
Operación: agregarCoordenada(altitud,latitud)

Referencia cruzada: CU01– Registrar predio

Precondiciones: Hay un registro de predio en curso, se creó una instancia de Predio predio.

Postcondiciones:

- Se creo una instancia de Coordenada coordina, y se inicializaron sus atributos.
- Se asocio coordenada a predio.

C05: terminarRegistro
Operación: terminarRegistro()

Referencia cruzada: CU01– Registrar predio

Precondiciones: Hay un registro de predio en curso.

Postcondiciones:

-

C06: iniciarCompra
Operación: iniciarCompra()

Referencia cruzada: CU09– Realizar

Precondiciones:

Postcondiciones:

-

C07: realizarPedido
Operación: realizarPedido(variedad:VariedadDatil,cantidad:integer,cliente)

Referencia cruzada: CU09– Realizar pedido

Precondiciones: El cliente debe haberse identificado y autenticado.

Postcondiciones:

- Se creó una instancia de ArticuloCarrito artículo.
- Se asocio variedad con artículo.

C08: terminarPedido
Operación: verCarrito(cliente)

Referencia cruzada: CU09– Realizar pedido

Precondiciones: Hay un pedido en curso.

Postcondiciones:

C09: pagarCarrito
Operación: pagarCarrito(noTarjeta:String)

Referencia cruzada: CU09– Realizar pedido

Precondiciones: Hay un pedido en curso. Hay por lo menos una instancia de ArticuloCarrito.

Postcondiciones:

- Se creo una instancia de Venta venta.
- Se creo al menos una instancia de LineaDeVenta lineaVenta.
- Se asocio lineaVenta con venta

C10: iniciarAsignacion
Operación: iniciarAsignacion(predio:Predio)

Referencia cruzada: CU05- Asignar actividades a predios

Precondiciones: El especialista debe haberse identificado y autenticado.

Postcondiciones:

-

C11: agendarActividades
Operación: agendarActividades(predio:Predio, fechaFin:Fecha, actividad:Actividad)

Referencia cruzada: CU05- Asignar actividades a predios

Precondiciones: Hay una asignación de actividades en curso.

Postcondiciones:

- Se creo al menos una instancia de Asignación asignación.
- Se asocio actividad con asignación.

C12: iniciarAsignacion
Operación: iniciarAsignacion(palmera:Palmera)

Referencia cruzada: CU06- Asignar actividades a palmeras orgánicas en predios no orgánicos.

Precondiciones: El especialista debe haberse identificado y autenticado.

Postcondiciones:

-

C13: agendarActividades
Operación: agendarActividades(predio:Predio, fechaFin:Fecha, actividad:Actividad)

Referencia cruzada: CU06- Asignar actividades a palmeras orgánicas en predios no orgánicos.

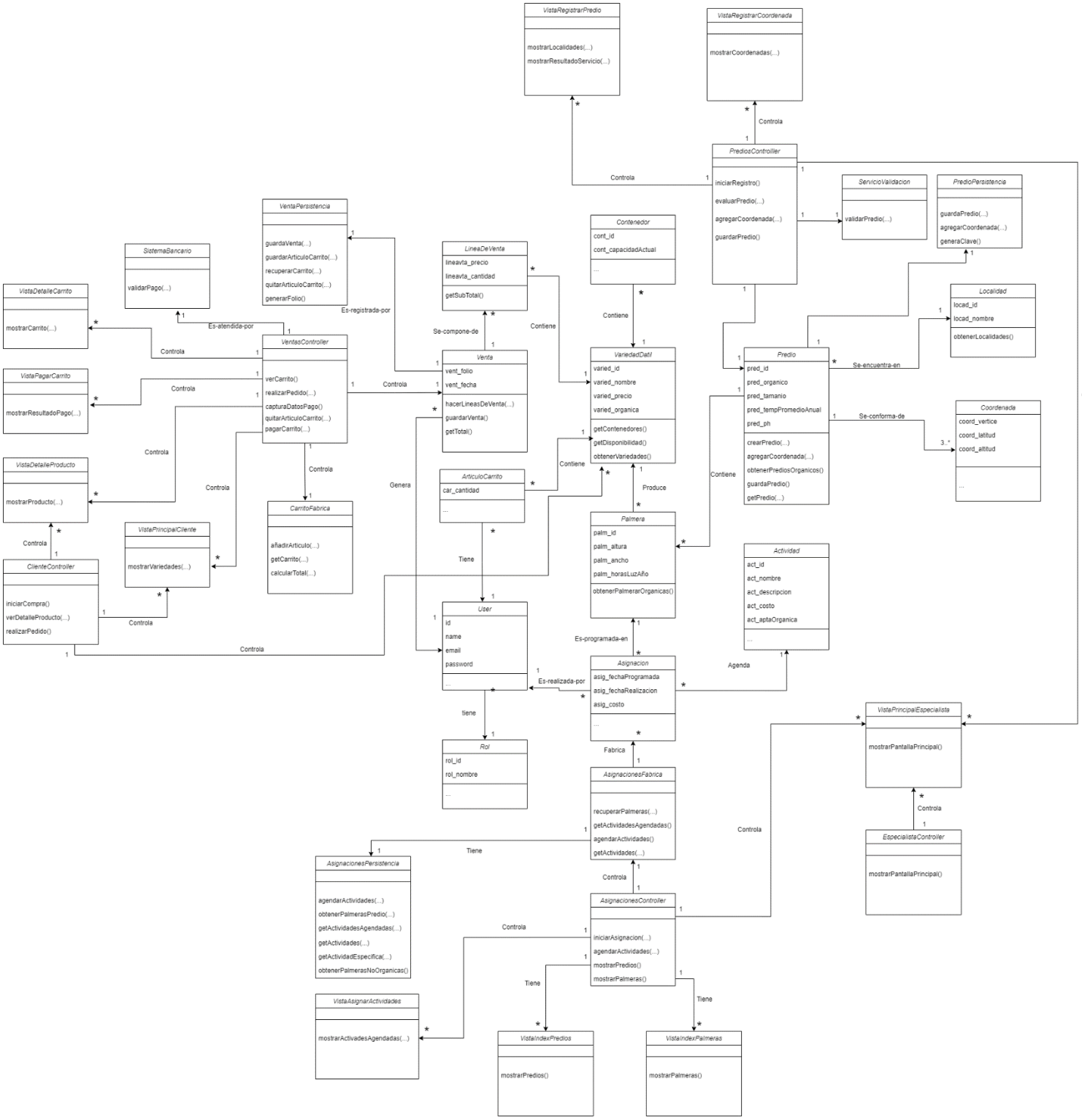
Precondiciones: Hay una asignación de actividades en curso.

Postcondiciones:

- Se creo al menos una instancia de Asignación asignación.
- Se asocio actividad con asignación.

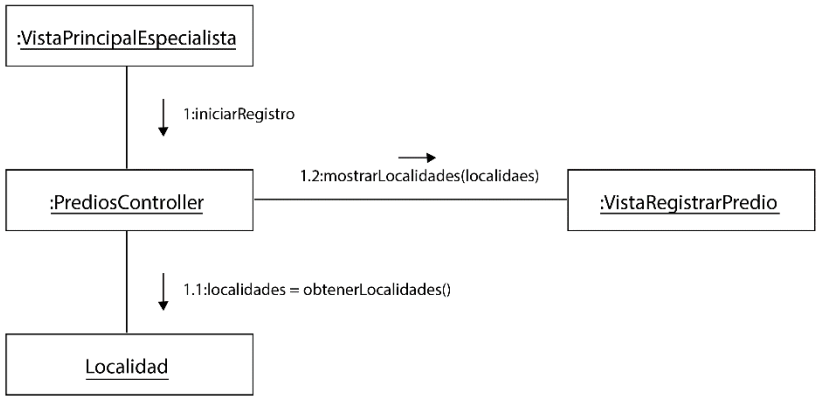
Diseño

Diagrama de clases de diseño

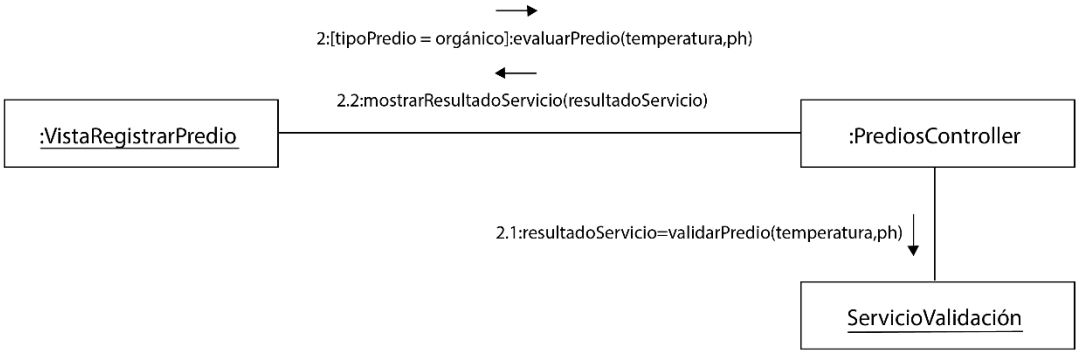


Diagramas de interacción

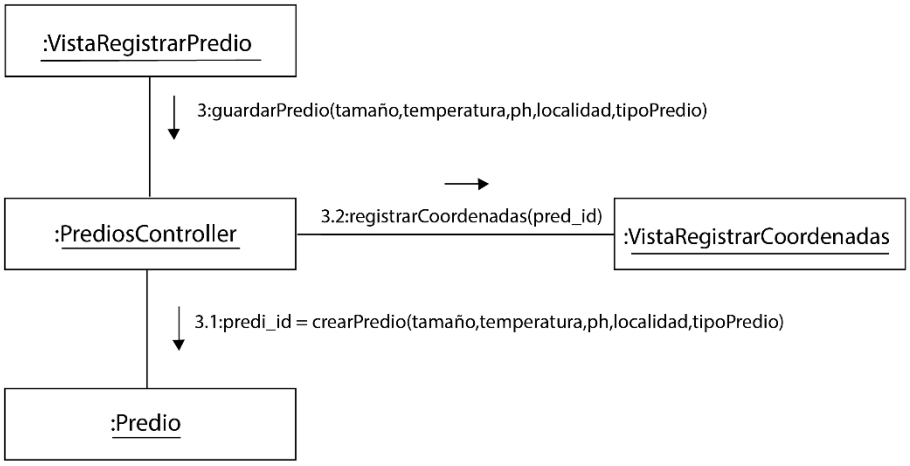
DI-C01



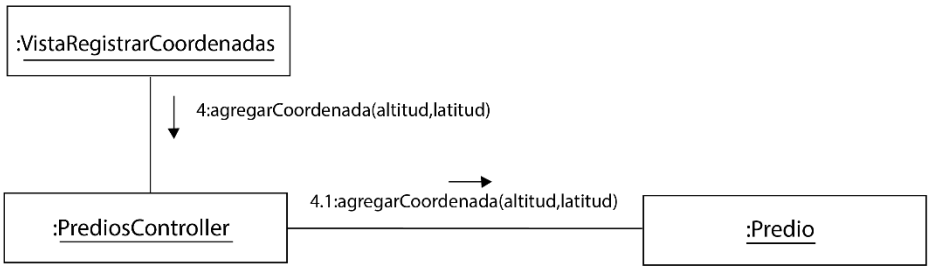
DI-C02



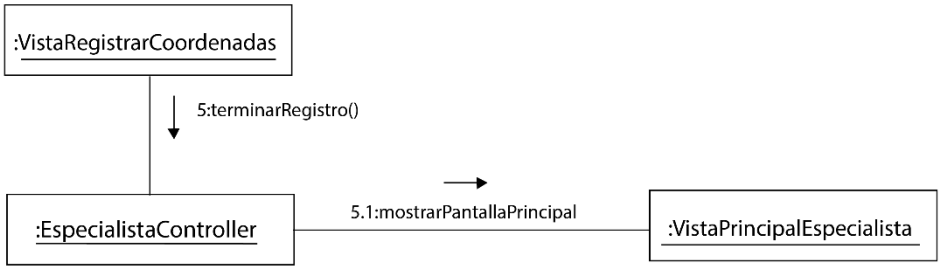
DI-C03



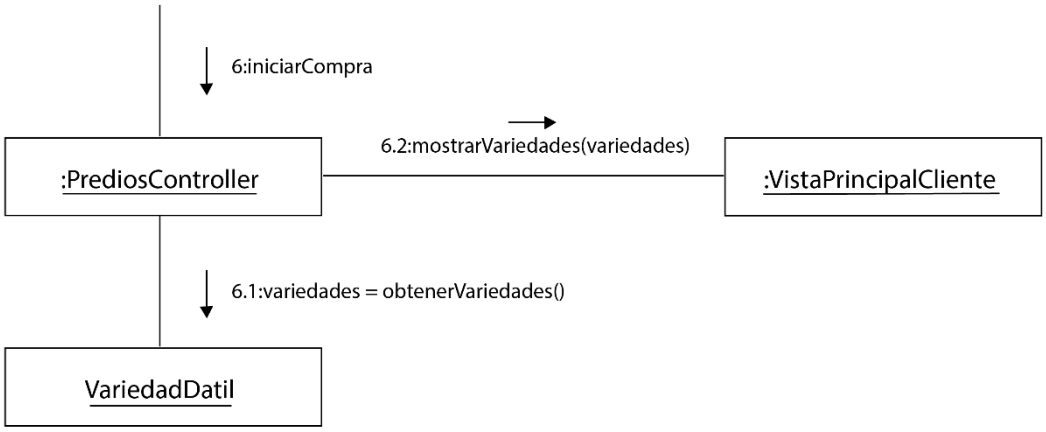
DI-C04



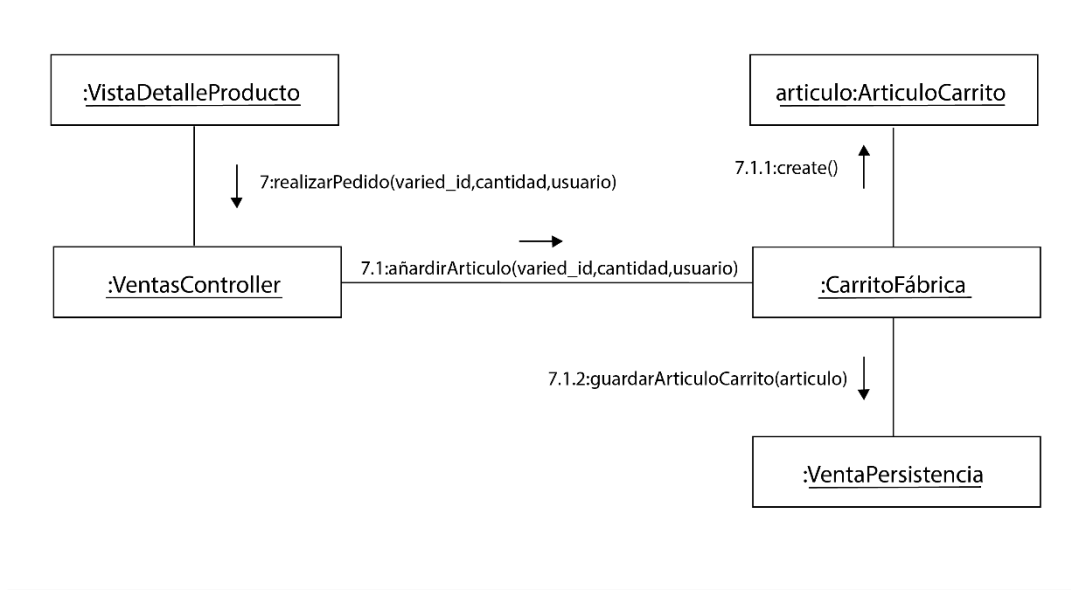
DI-C05



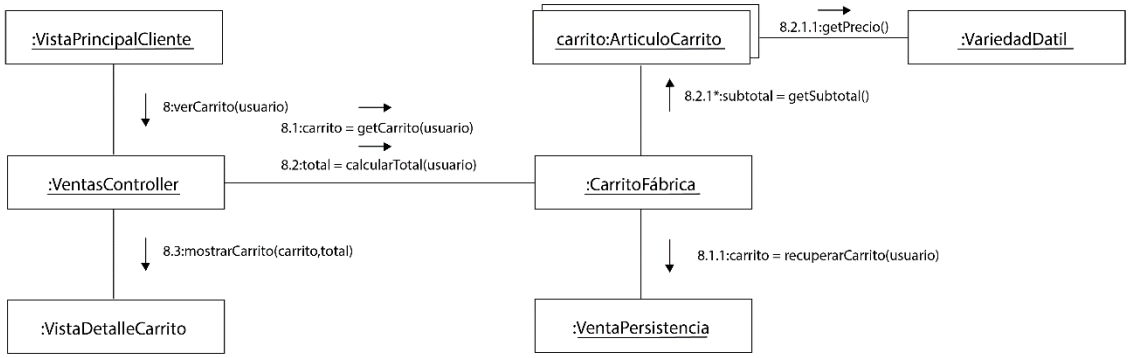
DI-C06


6:iniciarCompra
6.2:mostrarVariedades(variedades)
:PrediosController
:VistaPrincipalCliente
6.1:variedades = obtenerVariedades()
VariedadDatil

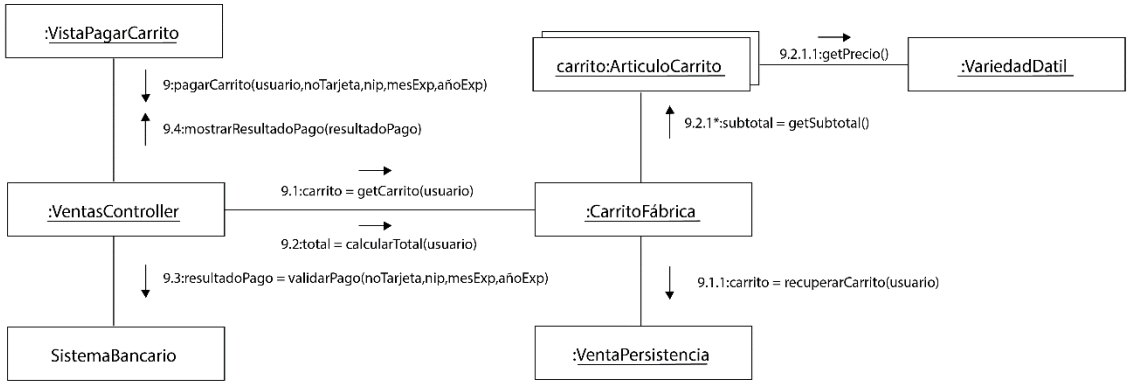
DI-C07


:VistaDetalleProducto
articulo:ArticuloCarrito
7.1.1:create()
7:realizarPedido(varied_id,cantidad,usuario)
:VentasController
7.1:añardirArticulo(varied_id,cantidad,usuario)
:CarritoFábrica
7.1.2:guardarArticuloCarrito(articulo)
:VentaPersistencia

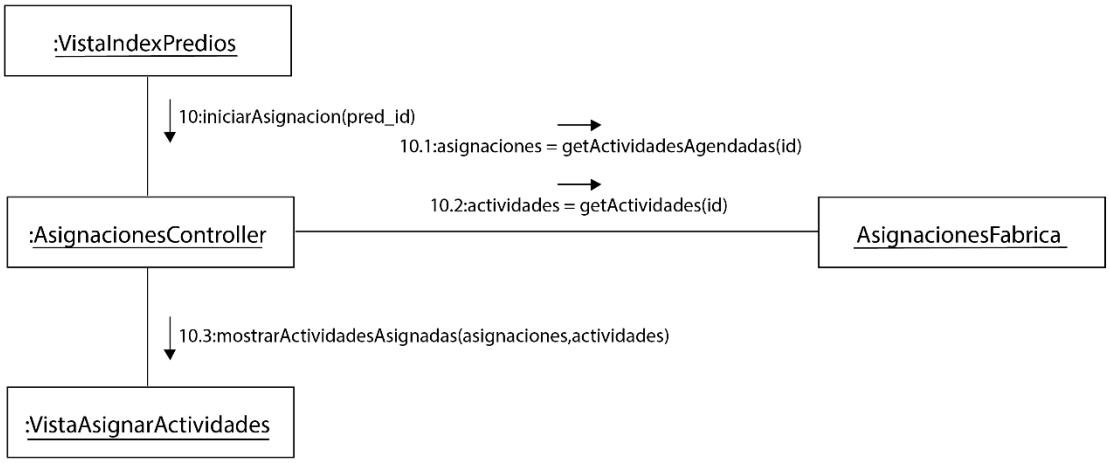
DI-C08



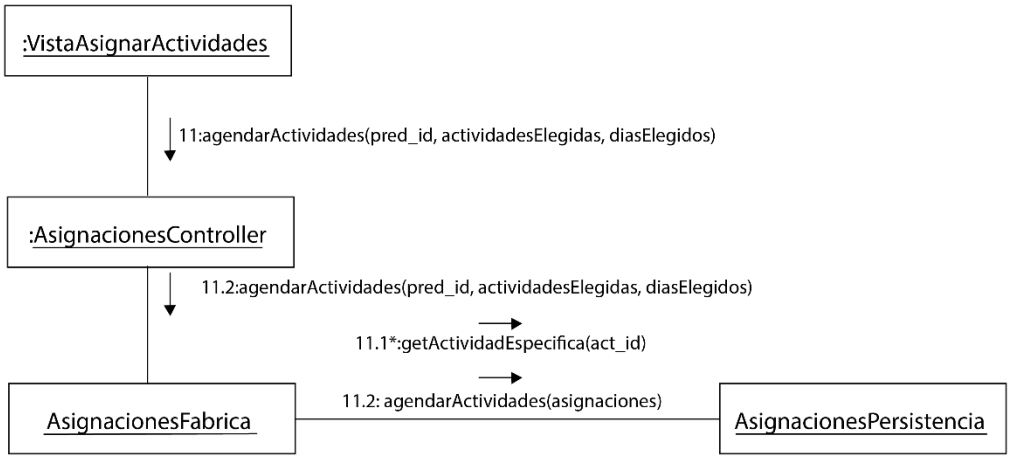
DI-C09



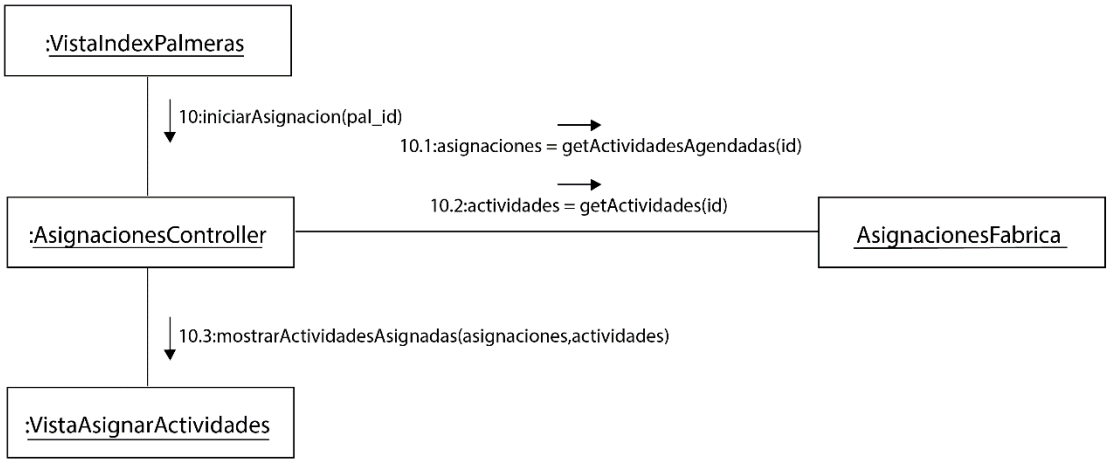
DI-C10



DI-C11


:VistaAsignarActividades
11:agendarActividades(pred_id, actividadesElegidas, diasElegidos)
:AsignacionesController
11.2:agendarActividades(pred_id, actividadesElegidas, diasElegidos)
11.1*:getActividadEspecifica(act_id)
11.2: agendarActividades(asignaciones)
AsignacionesFabrica
AsignacionesPersistencia

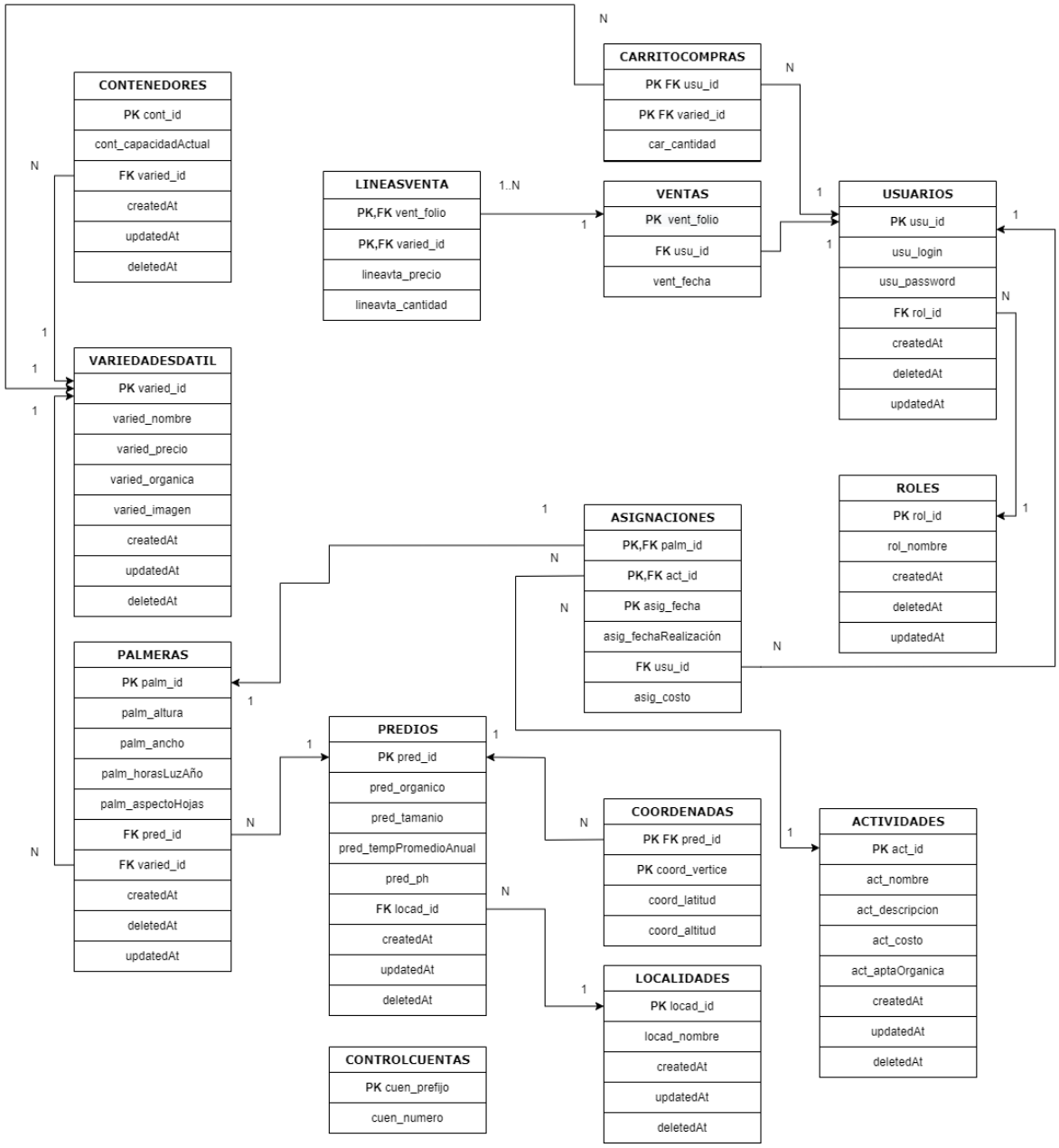
DI-C12


:VistaIndexPalmeras
10:iniciarAsignacion(pal_id)
10.1:asignaciones = getActividadesAgendadas(id)
10.2:actividades = getActividades(id)
:AsignacionesController
AsignacionesFabrica
10.3:mostrarActividadesAsignadas(asignaciones,actividades)
:VistaAsignarActividades

DI-C13


:VistaAsignarActividades
11:agendarActividades(pal_id, actividadesElegidas, diasElegidos)
:AsignacionesController
11.2:agendarActividades(pred_id, actividadesElegidas, diasElegidos)
11.1*:getActividadEspecifica(act_id)
11.2: agendarActividades(asignaciones)
AsignacionesFabrica
AsignacionesPersistencia

Modelo de la base de datos



Implementación

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\SoftDeletes;
use Illuminate\Database\Eloquent\Model;

class Actividad extends Model
{
    use SoftDeletes;
    protected $fillable = ["act_id", "act_nombre", "act_descripcion", "act_costo", "act_aptaOrganica"];
    protected $primaryKey = "act_id";
    protected $table = 'actividades';
    protected $casts = [];
    public $incrementing = true;
    public $timestamps = true;
    protected $guarded = [];
    protected $hidden = [];
    protected $dates = ['created_at','updated_at', 'deleted_at'];
```

```php
    public function __construct($actividad = array()) {···
    }
    public function getID() {···
    }
    public function setNombre($act_nombre) {···
    }
    public function getNombre() {···
    }
    public function setDescripcion($act_descripcion) {···
    }
    public function getDescripcion() {···
    }
    public function setCosto($act_costo) {···
    }
    public function getCosto() {···
    }
    public function setAptaOrganica($act_aptaOrganica) {···
    }
    public function getAptaOrganica() {···
    }
}
```

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class ArticuloCarrito extends Model
{
    protected $fillable = ["usu_id", "varied_id", "car_cantidad"];
    protected $primaryKey = array('usu_id', 'varied_id');
    protected $table = 'articulos_carrito';
    protected $casts = [];
    public $incrementing = false;
    public $timestamps = false;
    protected $guarded = [];
    protected $hidden = [];
    protected $dates = [];
```

```php
    public function __construct($articuloCarrito = array()) {…
    }

    public function getUsuario() {…
    }
    public function getVariedadDatil() {…
    }
    public function setUsuarioID($usu_id) {…
    }
    public function getUsuarioID() {…
    }
    public function setVariedadID($varied_id) {…
    }
    public function getVariedadID() {…
    }
    public function setCantidad($car_cantidad) {…
    }
    public function getCantidad() {…
    }
}
```

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Asignacion extends Model
{
    protected $fillable = ["palm_id", "act_id", "asig_fechaProgramada", "asig_fechaRealizacion", "usu_id", "asig_costo"];
    protected $primaryKey = array('palm_id', 'act_id', 'asig_fechaProgramada');
    protected $table = 'asignaciones';
    protected $casts = [];
    public $incrementing = false;
    public $timestamps = false;
    protected $guarded = [];
    protected $hidden = [];
    protected $dates = [];
```

```php
    public function __construct($asignaciones = array()) {···
    }
    public function getPalmera() {···
    }
    public function getActividad() {···
    }
    public function getUsuario() {···
    }
    public function setPalmeraID($palm_id) {···
    }
    public function getPalmeraID() {···
    }
    public function setActividadID($act_id) {···
    }
    public function getActividadID() {···
    }
    public function setFechaProgramada($asig_fechaProgramada) {···
    }
    public function getFechaProgramada() {···
    }
    public function setFechaRealizacion($asig_fechaRealizacion) {···
    }
    public function getFechaRealizacion() {···
    }
    public function setUsuarioID($usu_id) {···
    }
    public function getUsuarioID() {···
    }
    public function setActividad($asig_costo) {···
    }
    public function getCosto() {···
    }
}
```

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\SoftDeletes;
use Illuminate\Database\Eloquent\Model;

class Contenedor extends Model
{
    use SoftDeletes;
    protected $fillable = ["cont_id", "cont_capacidadActual", "varied_id"];
    protected $primaryKey = "cont_id";
    protected $table = 'contenedores';
    protected $casts = [];
    public $incrementing = true;
    public $timestamps = true;
    protected $guarded = [];
    protected $hidden = [];
    protected $dates = ['created_at','updated_at', 'deleted_at'];
```

```php
    public function __construct($contenedores = array()) {···
    }
    public function getVariedadDatil() {···
    }
    public function getID() {···
    }
    public function setCapacidadActual($cont_capacidadActual) {···
    }
    public function getCapacidadActual() {···
    }
    public function setVariedadID($varied_id) {···
    }
    public function getVariedadID() {···
    }
}
```

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\SoftDeletes;
use Illuminate\Database\Eloquent\Model;

class Coordenada extends Model
{
    use SoftDeletes;
    protected $fillable = ["pred_id", "coord_vertice", "coord_latitud", "coord_altitud"];
    protected $primaryKey = array('pred_id', 'coord_vertice');
    protected $table = 'coordenadas';
    protected $casts = [];
    public $incrementing = false;
    public $timestamps = true;
    protected $guarded = [];
    protected $hidden = [];
    protected $dates = ['created_at','updated_at', 'deleted_at'];
```

```php
    public function __construct($coordenadas = array()) {···
    }
    public function setPredioID($pred_id) {···
    }
    public function getPredioID() {···
    }
    public function setVertice($coord_vertice) {···
    }
    public function getVertice() {···
    }
    public function setLatitud($coord_latitud) {···
    }
    public function getLatitud() {···
    }
    public function setAltitud($coord_altitud) {···
    }
    public function getAltitud() {···
    }
}
```

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class LineaDeVenta extends Model
{
    protected $fillable = ["vent_folio", "varied_id", "lineavta_precio", "lineavta_cantidad"];
    protected $primaryKey = array('vent_folio', 'varied_id');
    protected $table = 'lineas_de_venta';
    protected $casts = [];
    public $incrementing = false;
    public $timestamps = false;
    protected $guarded = [];
    protected $hidden = [];
    protected $dates = [];
```

```php
    public function __construct($lineasDeVenta = array()) {···
    }
    public function getVariedadDatil() {···
    }
    public function getFolio() {···
    }
    public function setVariedadID($varied_id) {···
    }
    public function getVariedadID() {···
    }
    public function setPrecio($lineavta_precio) {···
    }
    public function getPrecio() {···
    }
    public function setCantidad($lineavta_cantidad) {···
    }
    public function getCantidad() {···
    }
}
```

```php
namespace App\Models;

use Illuminate\Database\Eloquent\SoftDeletes;
use Illuminate\Database\Eloquent\Model;

class Localidad extends Model
{
    use SoftDeletes;
    protected $fillable = ["locad_id", "locad_nombre"];
    protected $primaryKey = "locad_id";
    protected $table = 'localidades';
    protected $casts = [];
    public $incrementing = true;
    public $timestamps = true;
    protected $guarded = [];
    protected $hidden = [];
    protected $dates = ['created_at','updated_at', 'deleted_at'];

    public function __construct($localidades = array()) {···
    }
    public static function obtenerLocalidades() {···
    }
    public function setID($locad_id) {···
    }
    public function getID() {···
    }
    public function setNombre($locad_nombre) {···
    }
    public function getNombre() {···
    }
}
```

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\SoftDeletes;
use Illuminate\Database\Eloquent\Model;

class Palmera extends Model
{
    use SoftDeletes;
    protected $fillable = ["palm_id", "palm_altura", "palm_ancho", "palm_horasLuzAnio", "palm_aspectoHojas", "pred_id", "varied_id"];
    protected $primaryKey = "palm_id";
    protected $table = 'palmeras';
    protected $casts = [];
    public $incrementing = false;
    public $timestamps = true;
    protected $guarded = [];
    protected $hidden = [];
    protected $dates = ['created_at','updated_at', 'deleted_at'];
```

```php
    public function __construct($palmeras = array()) {···
    }
    public function obtenerPalmerasOrganicas() {···
    }
    public function getVariedadDatil() {···
    }
    public function getPredio() {···
    }
    public function getID() {···
    }
    public function setAltura($palm_altura) {···
    }
    public function getAltura() {···
    }
    public function setAncho($palm_ancho) {···
    }
    public function getAncho() {···
    }
    public function setHorasLuzAnio($palm_horasLuzAnio) {···
    }
    public function getHorasLuzAnio() {···
    }
    public function setAspectoHojas($palm_aspectoHojas) {···
    }
    public function getAspectoHojas() {···
    }
    public function setPredioID($pred_id) {···
    }
    public function getPredioID() {···
    }
}
```

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\SoftDeletes;
use Illuminate\Database\Eloquent\Model;
use App\Http\Persistencias\PredioPersistencia;
use App\Http\ServiciosExternos\ServicioValidacion;
use Illuminate\Support\Facades\Auth;

class Predio extends Model
{
    use SoftDeletes;
    protected $fillable = ["pred_id", "pred_organico", "pred_tamanio", "pred_tempPromedioAnual", "pred_ph", "locad_id"];
    protected $primaryKey = "pred_id";
    protected $table = 'predios';
    protected $casts = [];
    public $incrementing = false;
    public $timestamps = true;
    protected $guarded = [];
    protected $hidden = [];
    protected $dates = ['created_at','updated_at', 'deleted_at'];
```

```php
    public function __construct($predios = array()) {···
    }
    public static function crearPredio($datos) {···
    }
    public static function obtenerPredios() {···
    }
    public static function getPredio($pred_id) {···
    }
    public function agregarCoordenada($altitud, $latitud) {···
    }
    public function getLocalidad() {···
    }
    public function getPalmeras() {···
    }
    public function getCoordenadas() {···
    }
    public function getID() {···
    }
    public function setOrganico($pred_organico) {···
    }
    public function getOrganico() {···
    }
    public function setTamanio($pred_tamanio) {···
    }
    public function getTamanio() {···
    }
    public function setTemperaturaPromedioAnual($pred_tempPromedioAnual) {···
    }
    public function getTemperaturaPromedioAnual() {···
    }
    public function setPh($pred_ph) {···
    }
    public function getPh() {···
    }
    public function setLocalidadID($locad_id) {···
    }
    public function getLocalidadID() {···
    }
}
```

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\SoftDeletes;
use Illuminate\Database\Eloquent\Model;

class Rol extends Model
{
    use SoftDeletes;
    protected $fillable = ["rol_id", "rol_nombre"];
    protected $primaryKey = "rol_id";
    protected $table = 'roles';
    protected $casts = [];
    public $incrementing = true;
    public $timestamps = true;
    protected $guarded = [];
    protected $hidden = [];
    protected $dates = ['created_at','updated_at', 'deleted_at'];

    public function __construct($roles = array()) {...
    }
    public function setID($rol_id) {...
    }
    public function getID() {...
    }
    public function setNombre($rol_nombre) {...
    }
    public function getNombre() {...
    }
}
```

```php
namespace App\Models;

use Illuminate\Contracts\Auth\MustVerifyEmail;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Foundation\Auth\User as Authenticatable;
use Illuminate\Notifications\Notifiable;
use Laravel\Sanctum\HasApiTokens;

class User extends Authenticatable
{
    use HasApiTokens, HasFactory, Notifiable;

    /**
     * The attributes that are mass assignable.
     *
     * @var array<int, string>
     */
    protected $fillable = [
        'name',
        'email',
        'password',
        'rol_id'
    ];

    /**
     * The attributes that should be hidden for serialization.
     *
     * @var array<int, string>
     */
    protected $hidden = [
        'password',
        'remember_token',
    ];

    /**
     * The attributes that should be cast.
     *
     * @var array<string, string>
     */
    protected $casts = [
        'email_verified_at' => 'datetime',
    ];
}
```

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\SoftDeletes;

class VariedadDatil extends Model
{
    use SoftDeletes;
    protected $fillable = ["varied_id", "varied_nombre", "varied_precio", "varied_organica", "varied_imagen"];
    protected $primaryKey = "varied_id";
    protected $table = 'variedades_datil';
    protected $casts = [];
    public $incrementing = false;
    public $timestamps = true;
    protected $guarded = [];
    protected $hidden = [];
    protected $dates = ['created_at','updated_at', 'deleted_at'];
```

```php
    public function __construct($variedades = array()) {···
    }
    public function getContenedores() {···
    }
    public function getDisponibilidad() {···
    }
    public static function obtenerVariedades() {···
    }
    public function setID($varied_id) {···
    }
    public function getID() {···
    }
    public function setNombre($varied_nombre) {···
    }
    public function getNombre() {···
    }
    public function setPrecio($varied_precio) {···
    }
    public function getPrecio() {···
    }
    public function setOrganica($varied_organica) {···
    }
    public function getOrganica() {···
    }
    public function setRutaImagen($varied_imagen) {···
    }
    public function getRutaImagen() {···
    }
}
```

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\SoftDeletes;
use Illuminate\Database\Eloquent\Model;
use App\Http\Persistencias\VentaPersistencia;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\DB;

class Venta extends Model
{
    use SoftDeletes;
    protected $fillable = ["vent_folio", "usu_id", "vent_fecha"];
    protected $primaryKey = "vent_folio";
    protected $table = 'ventas';
    protected $casts = [];
    public $incrementing = false;
    public $timestamps = false;
    protected $guarded = [];
    protected $hidden = [];
    private $ventaPersistencia;
    private $lineasDeVenta = array();
```

```php
    public function __construct($venta = array()) {···
    }
    public function getUsuario() {···
    }
    public function getLineasDeVenta() {···
    }
    public function hacerLineasDeVenta($carrito) {···
    }
    public function setFolio($vent_folio) {···
    }
    public function getFolio() {···
    }
    public function setUsuarioID($usu_id) {···
    }
    public function getUsuarioID() {···
    }
    public function setFecha($vent_fecha) {···
    }
    public function getFecha() {···
    }
}
```

```php
<?php

namespace App\Http\Controllers;

use App\Http\Fabricas\AsignacionesFabrica;
use App\Models\Predio;
use App\Models\Palmera;
use App\Http\Requests\asignacionesRequest;

class AsignacionesController extends Controller
{
    public function iniciarAsignacion($id) {…
    }

    public function mostrarPredios() {…
    }

    public function mostrarPalmeras() {…
    }

    public function agendarActividades(asignacionesRequest $request) {…
    }
}
```

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Http\Fabricas\CarritoFabrica;
use App\Models\VariedadDatil;
use Illuminate\Support\Facades\Auth;

class ClienteController extends Controller
{
    private $carritoFabrica;

    public function __construct()
    {…
    }

    public function iniciarCompra() {…
    }

    public function verDetalleProducto($id) {…
    }

    public function realizarPedido() {…
    }
}
```

```php
<?php

namespace App\Http\Controllers;
use Illuminate\Support\Facades\Auth;

class EspecialistaController extends Controller
{
    public function __construct()
    {…
    }

    public function mostrarPantallaPrincipal()
    {…
    }
}
```

```php
<?php

namespace App\Http\Controllers;

use App\Models\Predio;
use App\Models\Localidad;
use App\Http\Requests\coordenadasRequest;
use App\Http\Requests\prediosRequest;
use App\Http\ServiciosExternos\ServicioValidacion;
use Illuminate\Support\Facades\Auth;
use Symfony\Component\Console\Input\Input;

class PrediosController extends Controller
{
    public function iniciarRegistro()
    {…
    }

    public function guardarPredio(prediosRequest $request) {…
    }

    public function agregarCoordenada(coordenadasRequest $request) {…
    }
}
```

```php
<?php

namespace App\Http\Controllers;

use App\Http\Fabricas\CarritoFabrica;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use App\Http\Requests\datosPagoRequest;
use App\Http\ServiciosExternos\SistemaBancario;
use App\Models\Venta;

class VentasController extends Controller
{
    private $carritoFabrica;

    public function __construct()
    {...
    }

    public function realizarPedido(Request $request) {...
    }

    public function verCarrito() {h...
    }

    public function quitarArticuloCarrito($varied_id) {...
    }

    public function capturaDatosPago() {...
    }

    public function pagarCarrito(datosPagoRequest $request) {...
    }
}
```

```php
<?php

namespace App\Http\Fabricas;

use App\Http\Persistencias\AsignacionesPersistencia;
use App\Models\Predio;
use App\Models\Palmera;
use App\Models\Actividad;
use App\Models\Asignacion;
use DateInterval;
use DatePeriod;
use DateTime;
use Illuminate\Support\Facades\Auth;

class AsignacionesFabrica
{
    public static function getActividadesAgendadas($id) {...
    }

    public static function recuperarPalmeras($id) {...
    }

    public static function agendarActividades($datosAsignacion) {...
    }

    public static function getActividades($id) {...
    }
}
```

```php
<?php

namespace App\Http\Fabricas;

use App\Http\Persistencias\VentaPersistencia;
use App\Models\ArticuloCarrito;

class CarritoFabrica
{
    private $ventaPersistencia;

    public function __construct()
    {...
    }

    public function añadirArticulo($varied_id, $usu_id, $car_cantidad) {...
    }

    public function getCarrito($usu_id) {...
    }

    public function calcularTotal($usu_id) {...
    }

    public function quitarArticuloCarrito($varied_id, $usu_id) {...
    }
}
```

```php
<?php

namespace App\Http\Persistencias;

use App\Models\Asignacion;
use App\Models\Palmera;
use App\Models\Predio;
use App\Models\VariedadDatil;
use App\Models\Actividad;

class AsignacionesPersistencia
{
    public static function agendarActividades($asignaciones) {…
    }

    public static function obtenerPalmerasPredio($pred_id) {…
    }

    public static function getActividadesAgendadas($palm_id) {…
    }

    public static function getActividades($aptaOrganica) {…
    }

    public static function getActividadEspecifica($act_id) {…
    }
}
```

```php
<?php

namespace App\Http\Persistencias;
use App\Models\Predio;
use Illuminate\Support\Facades\DB;

class PredioPersistencia
{
    public static function agregarCoordenada($coordenada) {…
    }

    public static function generaClave() {…
    }

    public static function guardarPredio($predio) {…
    }
}
```

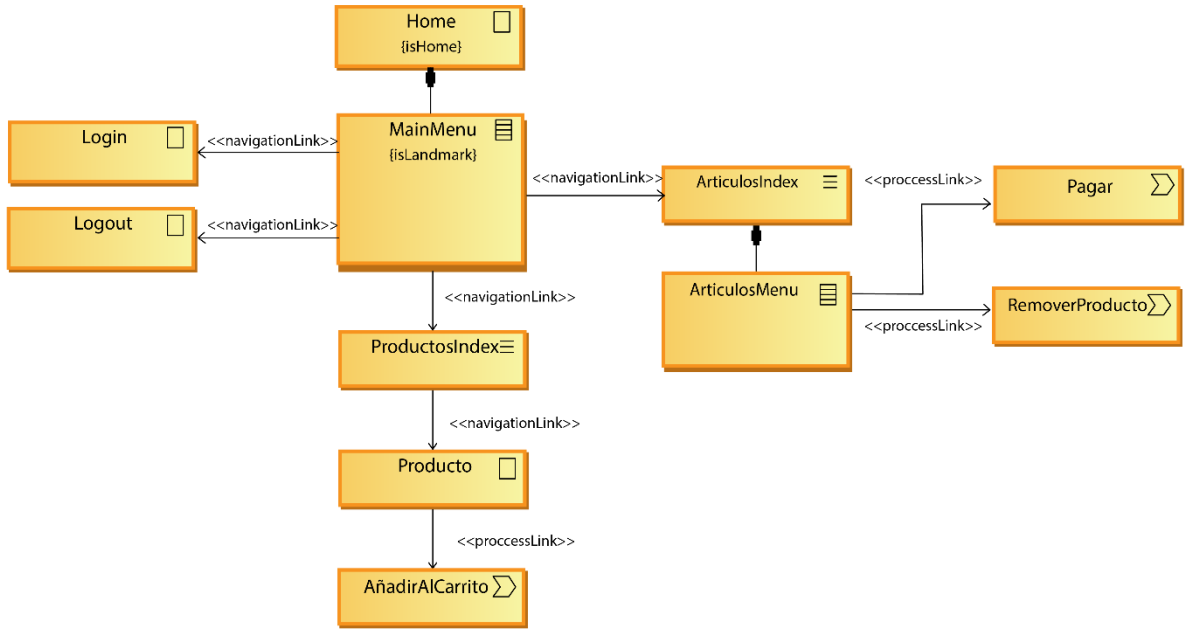
```php
namespace App\Http\Persistencias;

use App\Models\VariedadDatil;
use App\Models\ArticuloCarrito;
use Illuminate\Support\Facades\DB;

class VentaPersistencia
{
    public static function guardarArticuloCarrito($articulo) {…
    }

    public static function recuperarCarrito($usu_id) {…
    }

    public function quitarArticuloCarrito($varied_id, $usu_id) {…
    }

    public function generaFolio() {…
    }

    public function guardarVenta($lineasDeVenta, $venta) {…
    }
}
```

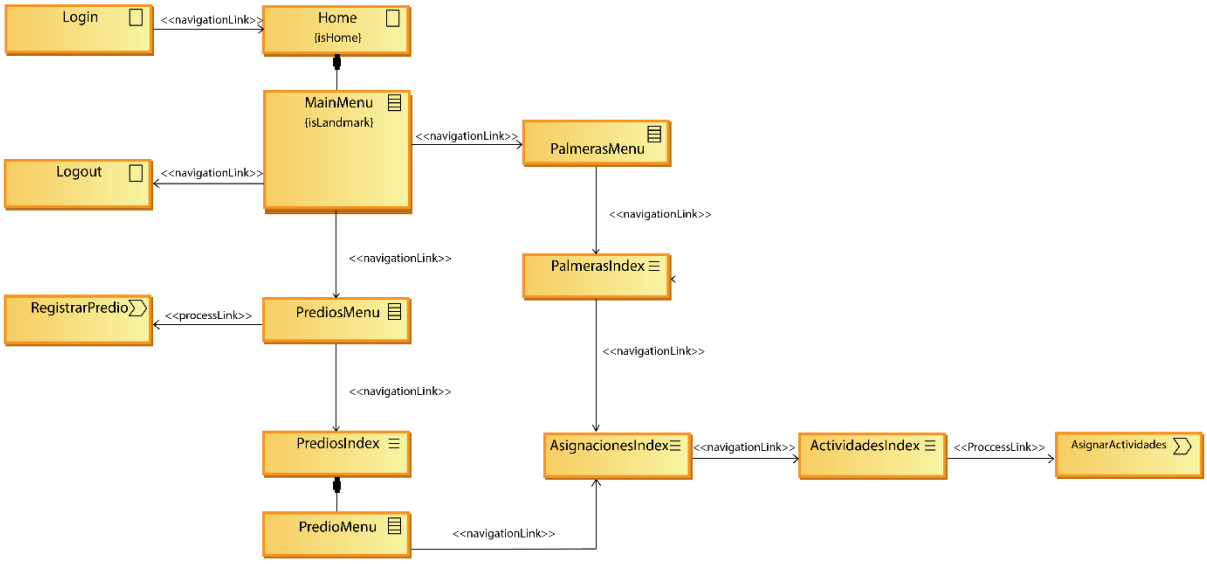
```php
namespace App\Http\ServiciosExternos;

use \GuzzleHttp\Client;

class ServicioValidacion
{
    public static function evaluarPredio($pred_temperatura, $pred_ph) {…
}
```

```php
namespace App\Http\ServiciosExternos;


class SistemaBancario
{
    public static function validarPago($datosTarjetaCredito, $totalPedido) {…
    }
}
```

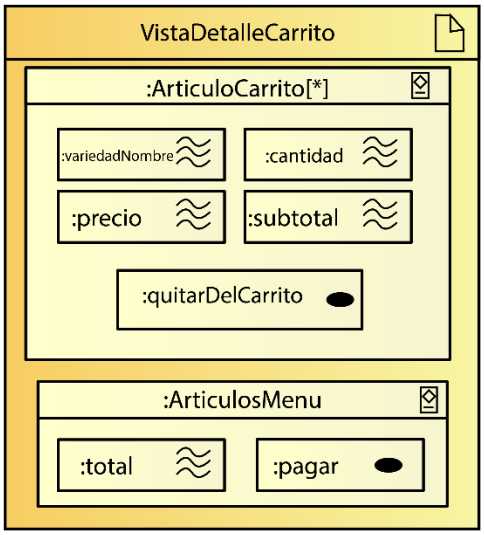
Artefactos de Ingeniería Web

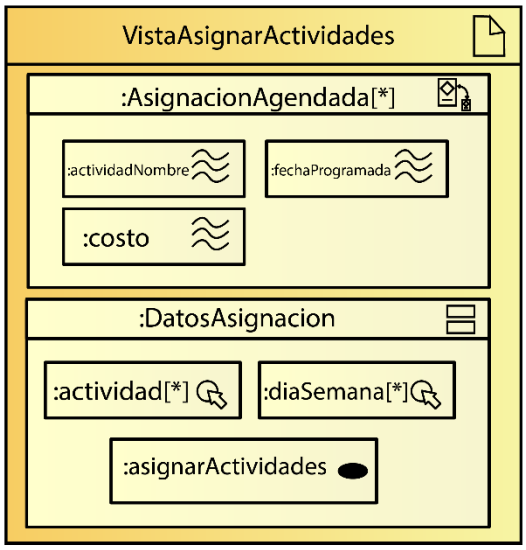
Modelo de navegación

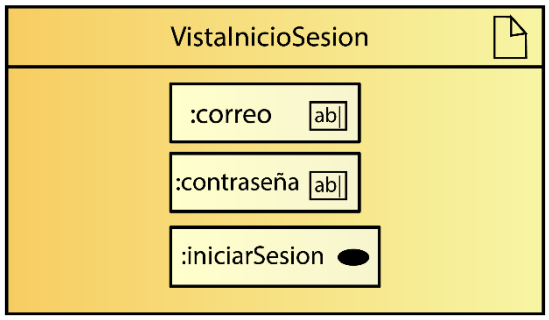


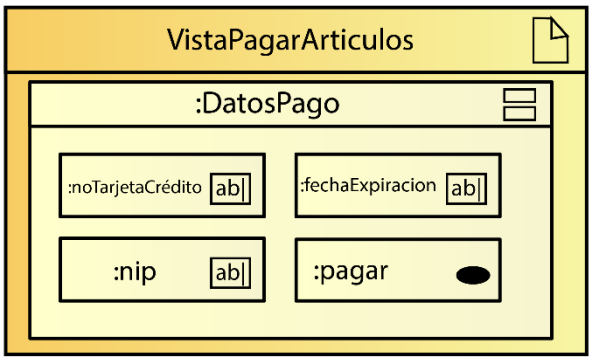


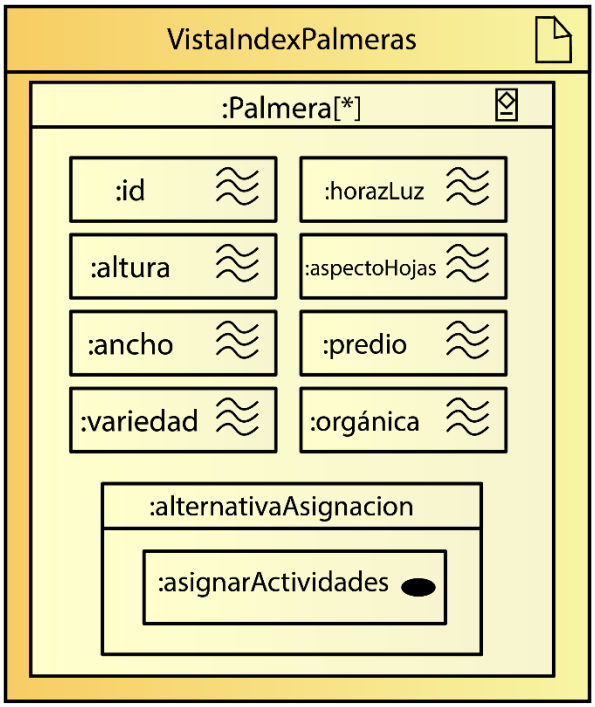
Modelo de presentación

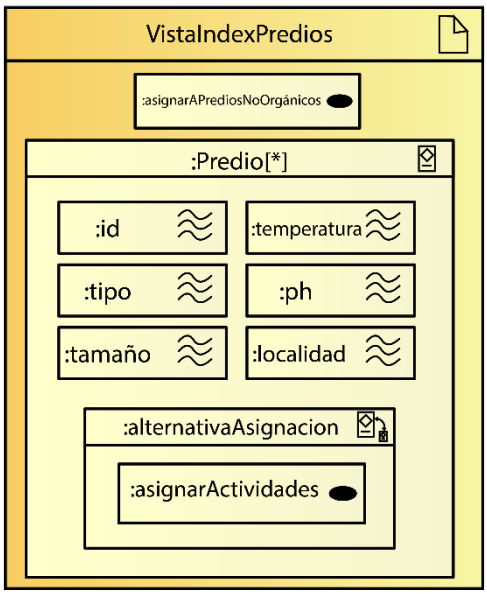


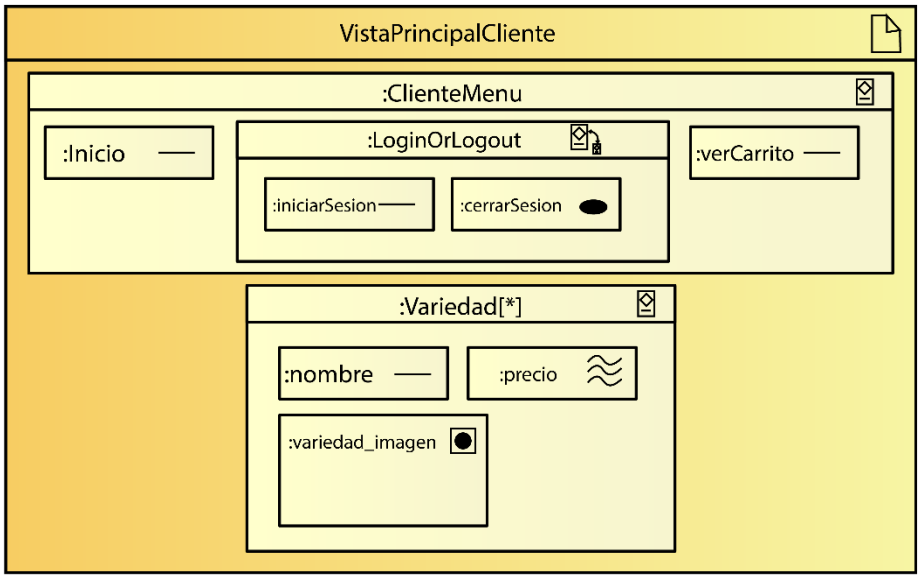


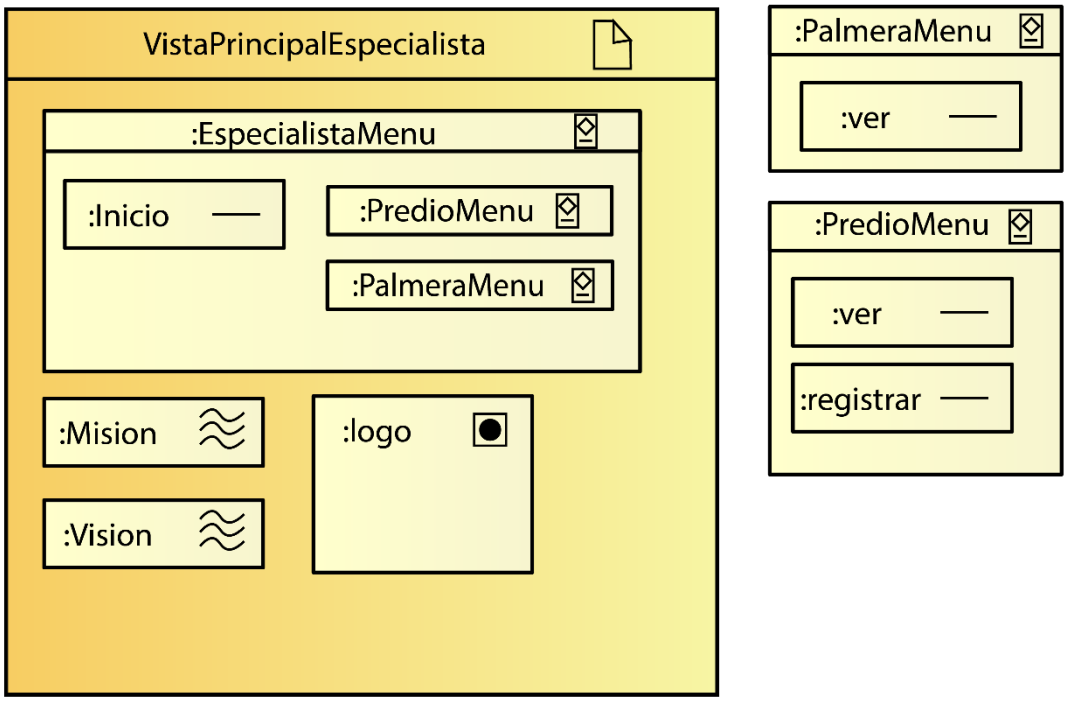

VistaInicioSesion
:correo
:contraseña
:iniciarSesion

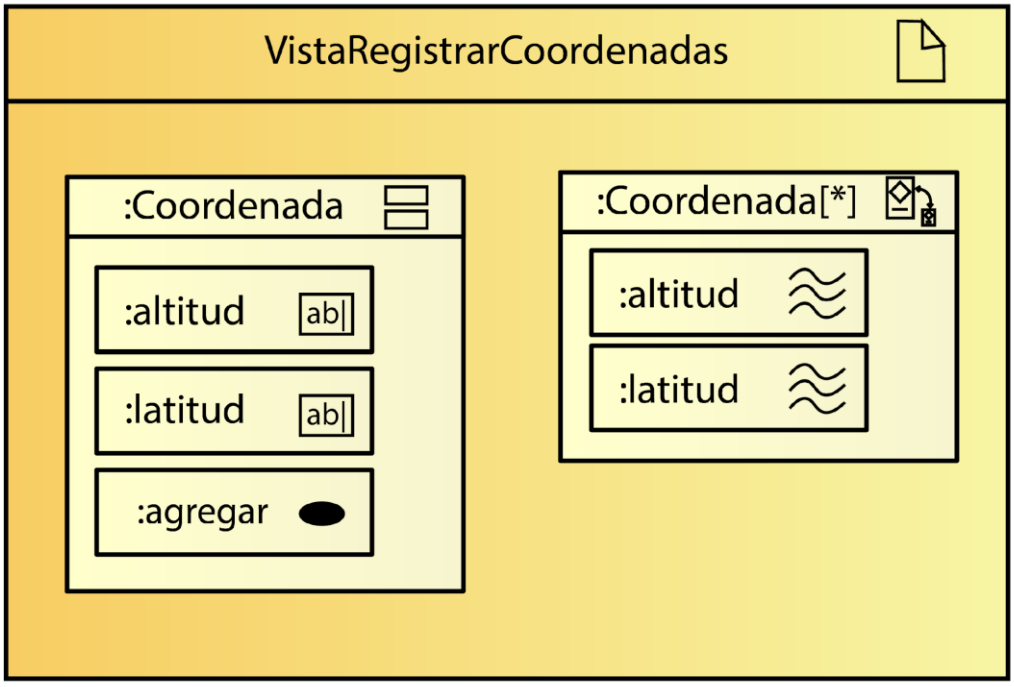

VistaPagarArticulos
:DatosPago
:noTarjetaCrédito
:fechaExpiracion
:nip
:pagar

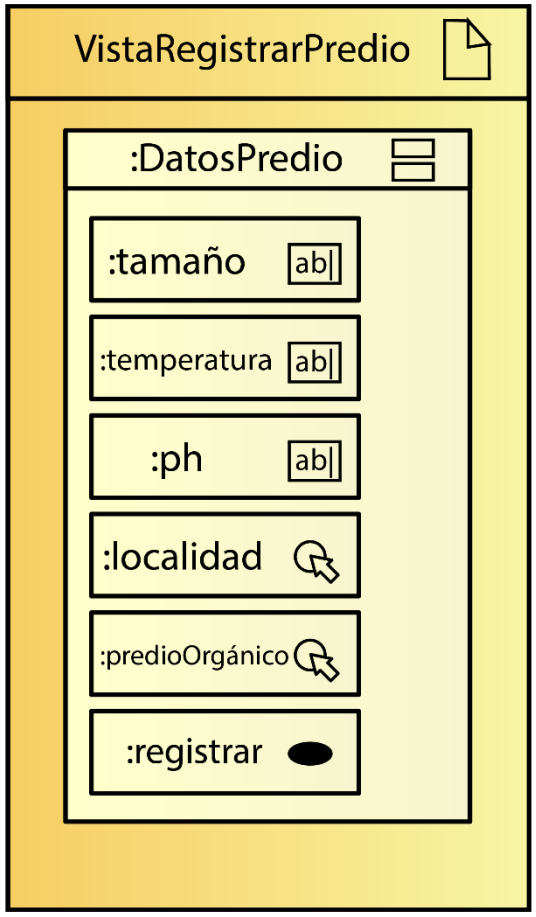

VistaIndexPalmeras
:Palmera[*]
:id
:horazLuz
:altura
:aspectoHojas
:ancho
:predio
:variedad
:orgánica
:alternativaAsignacion
:asignarActividades

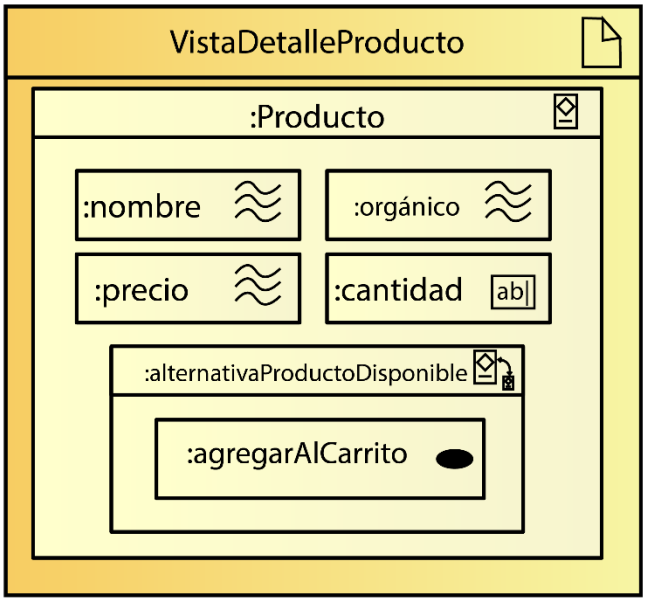

VistaIndexPredios
:asignarAPrediosNoOrgánicos
:Predio[*]
:id
:temperatura
:tipo
:ph
:tamaño
:localidad
:alternativaAsignacion
:asignarActividades


VistaPrincipalCliente
:ClienteMenu
:Inicio
:LoginOrLogout
:iniciarSesion
:cerrarSesion
:verCarrito
:Variedad[*]
:nombre
:precio
:variedad_imagen


VistaPrincipalEspecialista
:EspecialistaMenu
:Inicio
:PredioMenu
:PalmeraMenu
:Mision
:Vision
:logo
:PalmeraMenu
:ver
:PredioMenu
:ver
:registrar


VistaRegistrarCoordenadas
:Coordenada
:altitud
:latitud
:agregar
:Coordenada[*]
:altitud
:latitud


VistaRegistrarPredio
:DatosPredio
:tamaño
ab|
:temperatura
ab|
:ph
ab|
:localidad
:predioOrgánico
:registrar


VistaDetalleProducto
:Producto
:nombre
:orgánico
:precio
:cantidad
ab|
:alternativaProductoDisponible
:agregarAlCarrito

Anexos

Anexo 1: Creación de instancias de Asignación recuperando atributos de instancias de Palmera, Actividad y Usuario

```
foreach ($periodo  as $dia) {
    if (in_array($dia->format("w"), $diasElegidos)) {
        foreach ($actividadesElegidas as $actividad) {
            foreach ($palmeras as $palmera) {
                $asignacion["palm_id"] = $palmera->getID();
                $asignacion["act_id"] = $actividad->getID();
                $asignacion["asig_fechaProgramada"] = $dia;
                $asignacion["asig_fechaRealizacion"] = '1999-01-01';
                $asignacion["usu_id"] = Auth::user()->id;
                $asignacion["asig_costo"] = $actividad->getCosto();
                $asignaciones[] = new Asignacion($asignacion);
            }
        }
    }
}
```

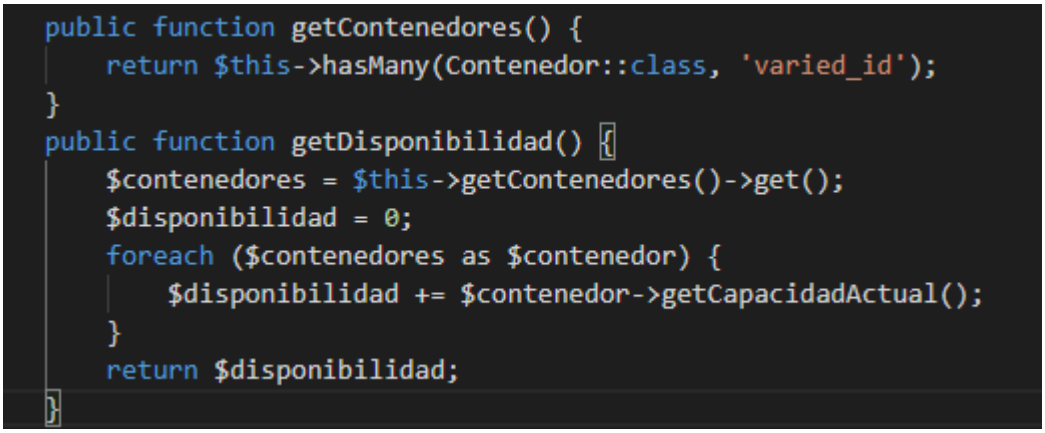
Anexo 2:Cálculo del subtotal en una instancia de ArticuloCarrito recuperando el atributo precio de su instancia de VariedadDatil agregada.

```
    public function getSubTotal(){
        return this->$cantidad * getVariedadDatil()->get()[0]->getPrecio();
    }
```

Anexo 3: Creación de instancias de LineaDeVenta a partir de atributos de instancias de ArticuloCarrito y Venta.

```
public function hacerLineasDeVenta($carrito) {
    $this->vent_folio = $this->ventaPersistencia->generaFolio();
    $this->usu_id = Auth::user()->id;
    $this->vent_fecha = date('Y-m-d h:i:s a', time());
    foreach ($carrito as $articulo) {
        $datosVenta['varied_id'] = $articulo->getVariedadID();
        $datosVenta['lineavta_precio'] = $articulo->getVariedadDatil()->get()[0]->getPrecio();
        $datosVenta['lineavta_cantidad'] = $articulo->getCantidad();
        $datosVenta['vent_folio'] = $this->vent_folio;
        $aux = new LineaDeVenta($datosVenta);
        $this->lineasDeVenta[] = $aux;
        DB::table('articulos_carrito')->where('usu_id', $articulo->getUsuarioID())->where('varied_id', $articulo->getVariedadID())->delete();
    }
    $this->ventaPersistencia->guardarVenta($this->lineasDeVenta, $this);
}
```

Anexo 4: Cálculo de la cantidad disponible de una instancia de VariedadDatil a partir de la capacidad actual de las instancias de Contenedor en las que se encuentra agregada.



```php
public function getContenedores() {
    return $this->hasMany(Contenedor::class, 'varied_id');
}
public function getDisponibilidad() {
    $contenedores = $this->getContenedores()->get();
    $disponibilidad = 0;
    foreach ($contenedores as $contenedor) {
        $disponibilidad += $contenedor->getCapacidadActual();
    }
    return $disponibilidad;
}
```