

2024



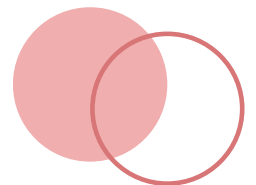
ANÁLISIS, DISEÑO E IMPLEMENTACIÓN

Porky Cakes

Materia: AyDS2

Grupo: 1

Integrantes: Civalero, Santiago;
Melchor, Leo;
Oronás, Melani.



Índice

1. Porky Cakes	2
a. Narrativa organización	2
2. Modelo de casos de uso	3
3. Modelo relacional	4
4. Primera iteración	5
a. Caso de uso agregar receta derivada	5
i. Especificación del c.u y prototipos de la interfaz	5
ii. Modelo de diseño estático	8
iii. Modelo de diseño dinámico	8
b. Caso de uso buscar producto	9
i. Especificación del c.u y prototipos de la interfaz	9
ii. Modelo de diseño estático	11
iii. Modelo de diseño dinámico	11
5. Segunda iteración	12
a. Caso de uso agregar producto	12
i. Especificación del c.u y prototipos de la interfaz	13
ii. Modelo de diseño estático	14
iii. Modelo de diseño dinámico	14
b. Caso de uso consultar lista de recetas	15
i. Especificación del c.u y prototipos de la interfaz	16
ii. Modelo de diseño estático	17
iii. Modelo de diseño dinámico	17
6. Explicación patrón DAO	18
7. Explicación patrón reflexivo	18
8. Explicación arquitectura del sistema	18
9. Link a repositorio en Github	18

1. Porky Cakes

a. Narrativa organización

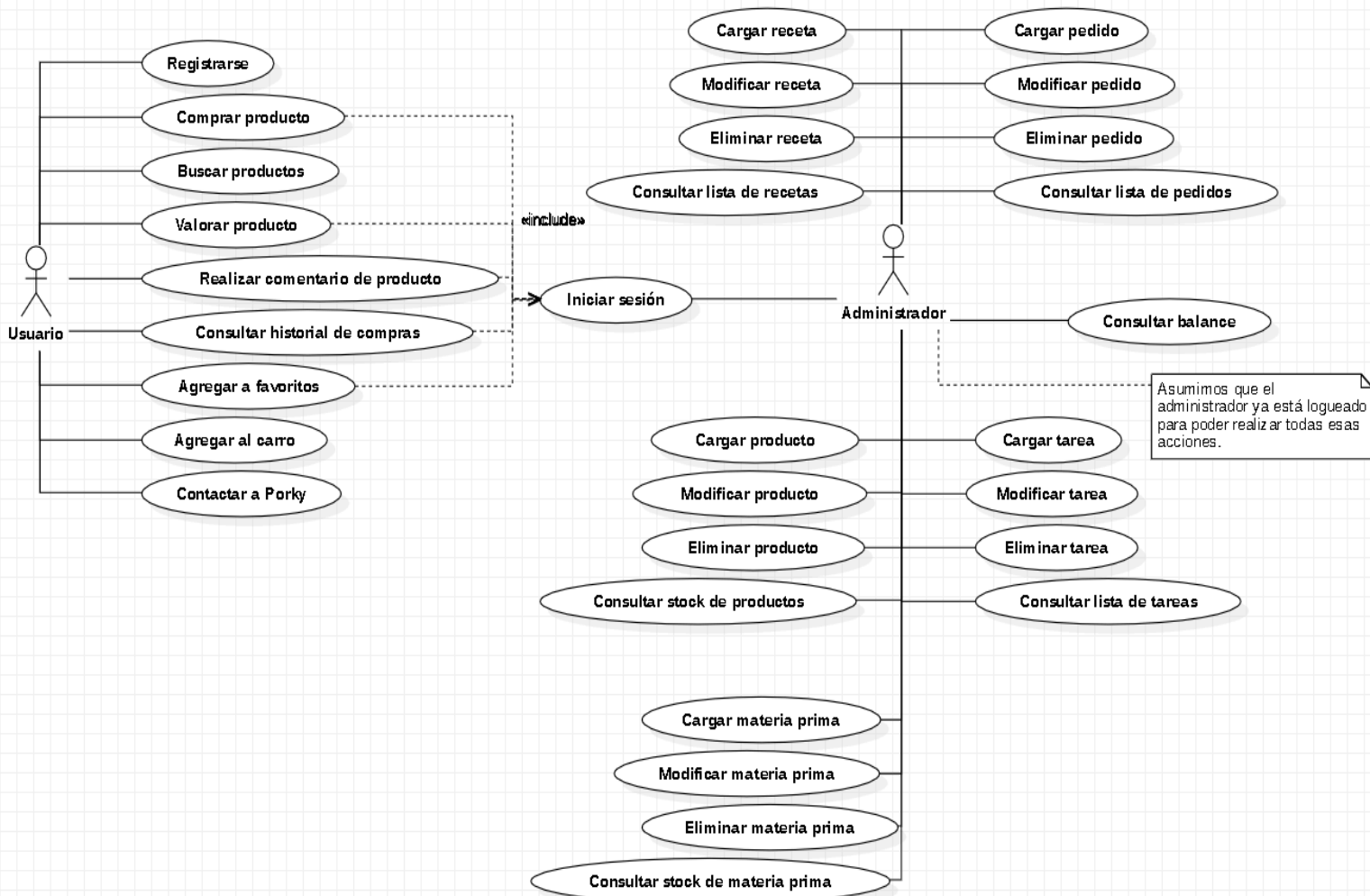
Porky Cakes es un emprendimiento de productos de pastelería a pedido, ubicada en General Pico, dirigida por Agustina Cuadrelli, quien recibe pedidos mediante Instagram y WhatsApp, la cual ya cuenta con algunos clientes fijos. Agustina acepta pedidos mediante algunas condiciones, como revisar que ella esté disponible, que tenga cupo disponible y que el pedido sea hecho con 72 horas de anticipación, cuando cumple estas condiciones el estado del pedido queda como "Aceptado" y cuando no está disponible, ella recomienda a una colega quedando el estado como "No aceptado". Los domingos no acepta pedidos. Además, Agustina realiza pedidos personalizados: agregando o quitando ingredientes, productos sin TACC y presupuestos al por mayor, ya sea para locales o eventos especiales. Cada producto tiene un nombre que lo identifica, un tamaño, un precio y el tipo de producto que es (Torta, masa fina, cupcake, tarta, etc). Del cliente que pide el producto le interesa saber su nombre, apellido y número de teléfono. Luego de registrar el pedido en la agenda junto con la fecha de pedido, cantidad pedida y el monto total, procede a chequear su stock manualmente, en sus planillas de excel ya armadas previamente con su catálogo de productos, recetas y costo de insumos. El stock se basa en todos los ingredientes que ya tiene, porque generalmente ella compra de más cuando hay ofertas o por mayor en algunos insumos, como es la harina o las galletitas. También chequea el stock de empaquetados. En caso de no tener stock, realiza una lista de compras, en la cual prioriza comprar determinados insumos porque los que necesitan conservación son los últimos que compra. Tiene algunos proveedores fijos en determinados insumos, como en el caso de los empaquetados.

Seguido a este proceso, comienza la producción del pedido, la cual tiene pasos extensos, en primer lugar revisa los pedidos, cada producto tiene su receta, la cual contiene su descripción, cuando tiene pedidos sin TACC realiza una limpieza detallada de la zona de trabajo, teniendo el máximo cuidado. Prosigue con el armado de la lista de quehaceres, la búsqueda de ingredientes y utensilios, y la medición de cantidades. Cada receta puede contener muchos ingredientes identificados con un nombre y stock, y muchas preparaciones

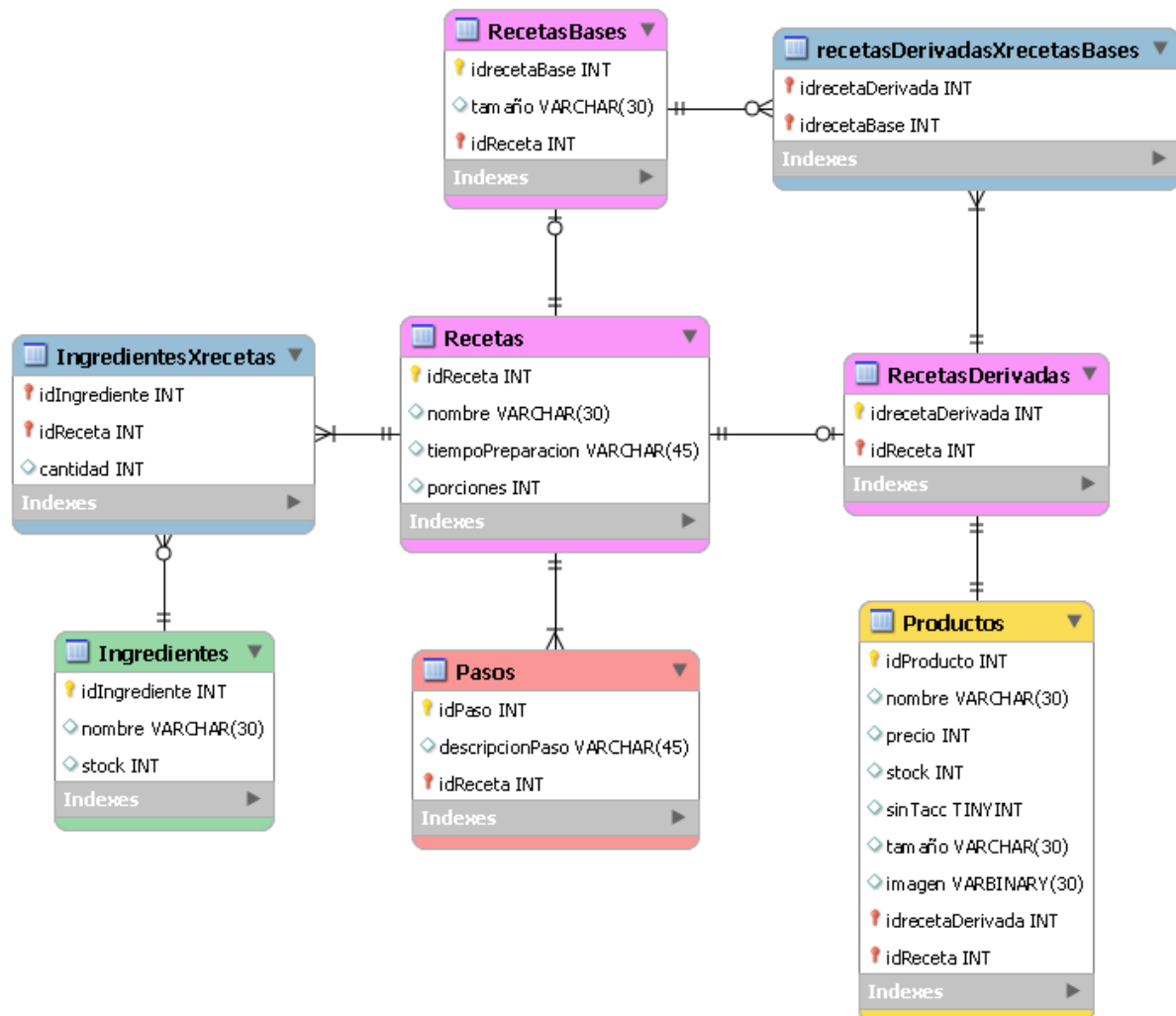
dependiendo del tipo de producto. A su vez, la preparación puede requerir de muchas tareas, las cuales tienen un tiempo estimado y descripción de la misma. De aquí, sigue en base al tipo de producto que debe hacer, ya que si debe realizar bases solo debe prepararlas y hornearlas, distinto es si debe realizar masas porque éstas necesitan de preparación, enfriamiento adecuado y horneado. Como continua, depende de si debe realizar rellenos, el cual se basa en elaborar el relleno elegido y dejarlo enfriar el tiempo adecuado. Uno de los últimos pasos a realizar es el armado del producto, el cual difiere si necesita decoración ya que debe esperar hasta el día antes o mismo día de la entrega. Más tarde procede a realizar el empaquetado, hermético cuando es sin TACC, sino tradicional.

Finalmente coordina con el cliente la entrega, la cual puede ser retirada por su casa o su lugar de trabajo, y la forma de pago, la cual es efectuada en efectivo.

2. Modelo de casos de uso




3. Modelo relacional





4. Primera iteración

a. Caso de uso agregar receta derivada

i. Especificación del c.u y prototipos de la interfaz



[Inicio](#)
[Productos](#)
[Preguntas frecuentes](#)


Agregar receta derivada



NOMBRE

TIEMPO DE PREPARACIÓN


PORCIONES



Recetas base

 Borrar
 [+ Nueva receta base](#)


 Nombre	Tamaño
 Seleccione una opción	▼ Seleccione una opción



Ingredientes adicionales

 Borrar
 [+ Nuevo ingrediente](#)

 Nombre	Cantidad
 Escriba aquí	0

Preparación adicional

 Borrar
 [+ Nuevo paso](#)

 Paso	Descripción
 1	Escriba aquí

[GUARDAR](#)

Caso de uso: Agregar receta derivada

1 Breve descripción

El actor puede agregar recetas al sistema, ya sean recetas base o recetas derivadas. Las recetas base son aquellas que se utilizan como componentes para la elaboración de otros productos, pero que no se comercializan de forma independiente, como masas, rellenos o decoraciones. Por otro lado, las recetas derivadas corresponden a los productos finales que se venden directamente, como tortas, tartas, entre otros.

1.1 Actor Principal: Administrador.

1.2 Actor Secundario: No tiene.

2 Precondiciones

El actor debe previamente haber iniciado sesión para poder utilizar este caso de uso.

3 Flujo de evento básico

1. El **actor** selecciona la opción "Agregar receta".
2. El **sistema** muestra la opción de agregar "Receta base" o "Receta derivada".
3. El **actor** elige "Receta Derivada".
4. El **sistema** muestra un formulario para la creación de una nueva receta derivada, con los campos:
 - nombre.
 - tiempo de preparación.
 - porciones.

También permite agregar recetas bases, con los campos:

- nombre.
- tamaño.

Agregar ingredientes adicionales, con los campos:

- nombre.
- cantidad.

Y agregar preparación, con los campos:

- paso.
- descripción.

5. El **actor** completa los campos de la receta, y agrega los ingredientes, pasos y recetas bases que necesite. Luego envía la información, confirmando su operación.
6. El **sistema** almacena la receta en un medio de persistencia e indica que la receta fue agregada exitosamente.

4 Flujos alternativos

Alternativa Paso 3: 3.1 El **actor** elige "Receta Base".

Alternativa Paso 4: 4.1 El **sistema** muestra un formulario para la creación de una nueva receta base, con los campos:

- nombre.
- tiempo de preparación.
- tamaño.

- porciones.

También permite agregar ingredientes, con los campos:

- nombre.
- cantidad.

Y agregar preparación, con los campos:

- paso.
- descripción.

Alternativa Paso 5:

- ☐ 5.1 El **actor** intenta guardar una receta, ingrediente o paso con un nombre que ya existe y el sistema muestra un mensaje de error avisando que ese nombre ya existe, y solicitando que cambie el nombre por otro.
- ☐ 5.2 El **actor** no completa todos los campos y el **sistema** muestra un mensaje de error solicitando que se completen los datos.

5 Post-condiciones

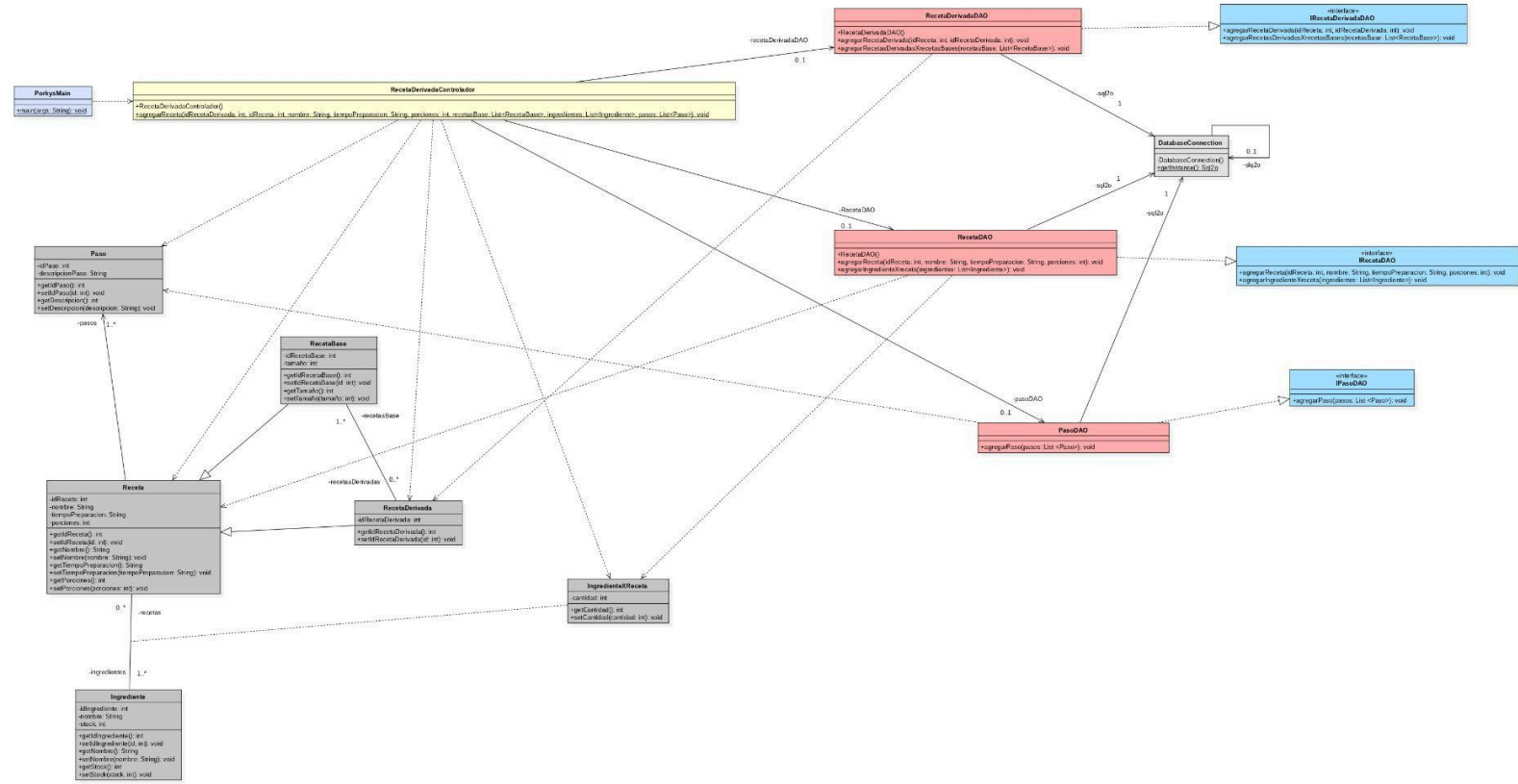
5.1 Se añadió al **sistema** de manera exitosa una receta, por ende, está disponible para su uso.

5.2 El **actor** puede seguir agregando más recetas.

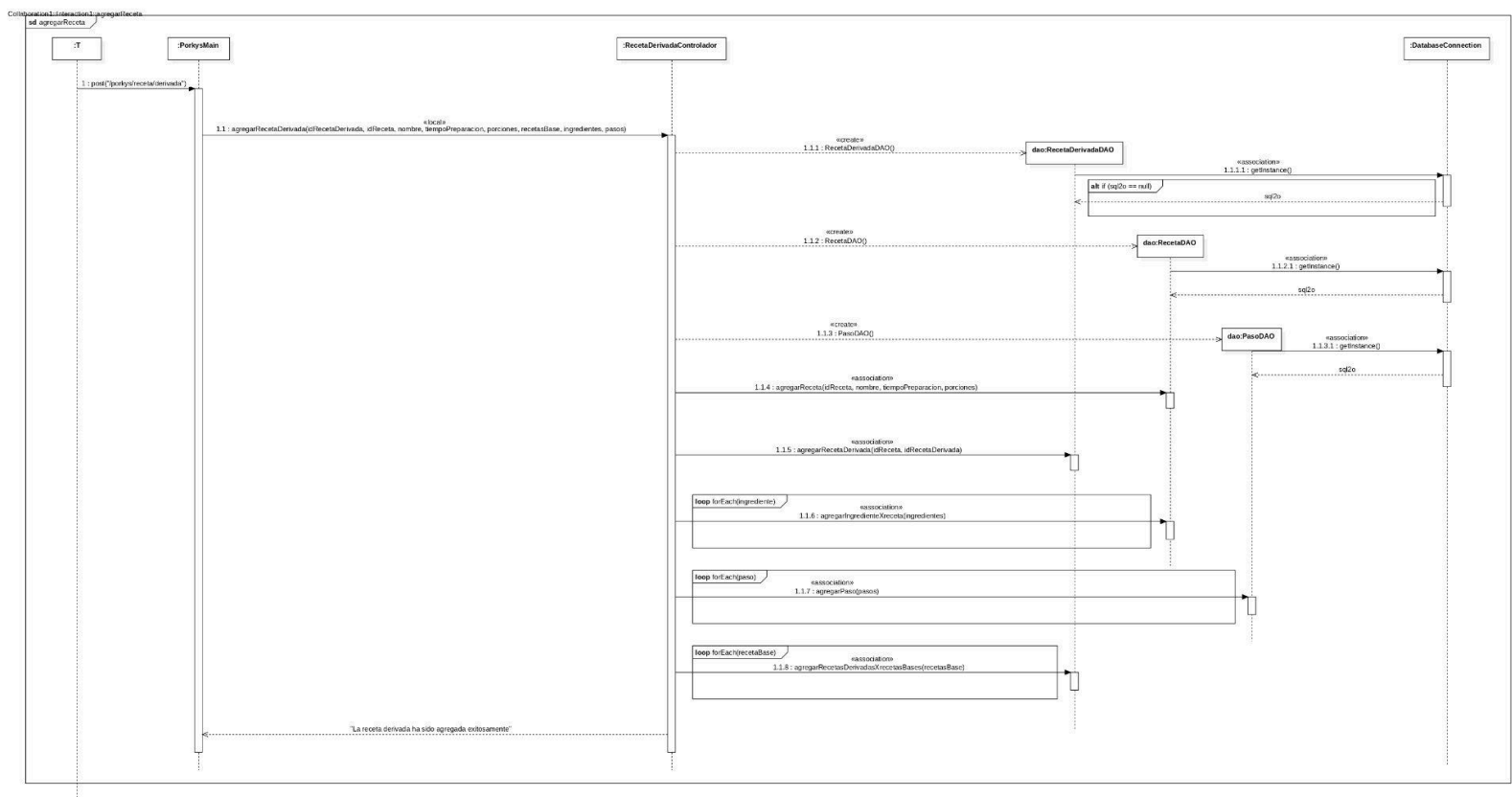
6 Requerimientos especiales

- Marcar la diferencia entre recetas base y derivadas.
- Validar que se eviten los nombres duplicados.
- Asegurar que todos los campos de formulario sean completados.

ii. Modelo de diseño estático

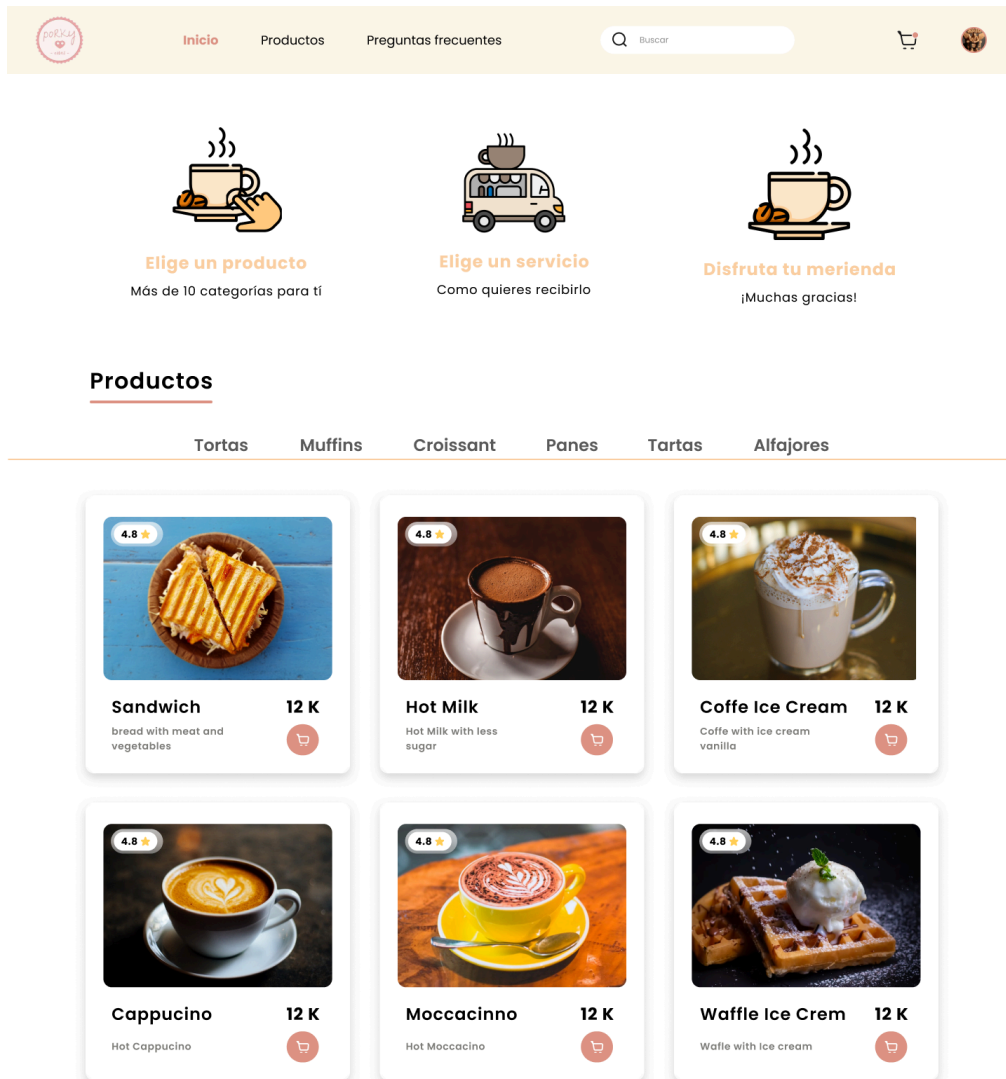


iii. Modelo de diseño dinámico



b. Caso de uso buscar producto

i. Especificación del c.u y prototipos de la interfaz



Caso de uso: Buscar producto

1 Breve descripción

Los actores pueden buscar un producto específico que quieren ver en ese momento determinado con solo escribir el nombre del producto, esto permite ahorrar tiempo en la búsqueda porque sin un buscador se debería ubicar el producto en una amplia lista de todos los productos. Es una manera de filtrar rápido lo que se necesita.

1.1 Actor Principal: Usuario web

1.2 Actor Secundario: Administrador.

2 Precondiciones

No debe cumplirse ninguna condición antes de que se inicie el caso de uso ya que cualquier usuario web, registrado o no, puede utilizar el buscador.

3 Flujo de evento básico

1. El **actor** ingresa el nombre del producto que quiere buscar.
2. El **sistema** busca por productos que coinciden con ese criterio de búsqueda. Y devuelve como resultado los siguientes datos: nombre, tamaño, precio, descripción e imagen.
3. El **actor** visualiza todos los resultados.

4 Flujos alternativos

- ☐ Alternativa Paso 4: 4.1 El actor introduce un nombre en el campo de búsqueda pero el sistema muestra un mensaje indicando que no se encontró un producto que coincida con ese criterio de búsqueda.

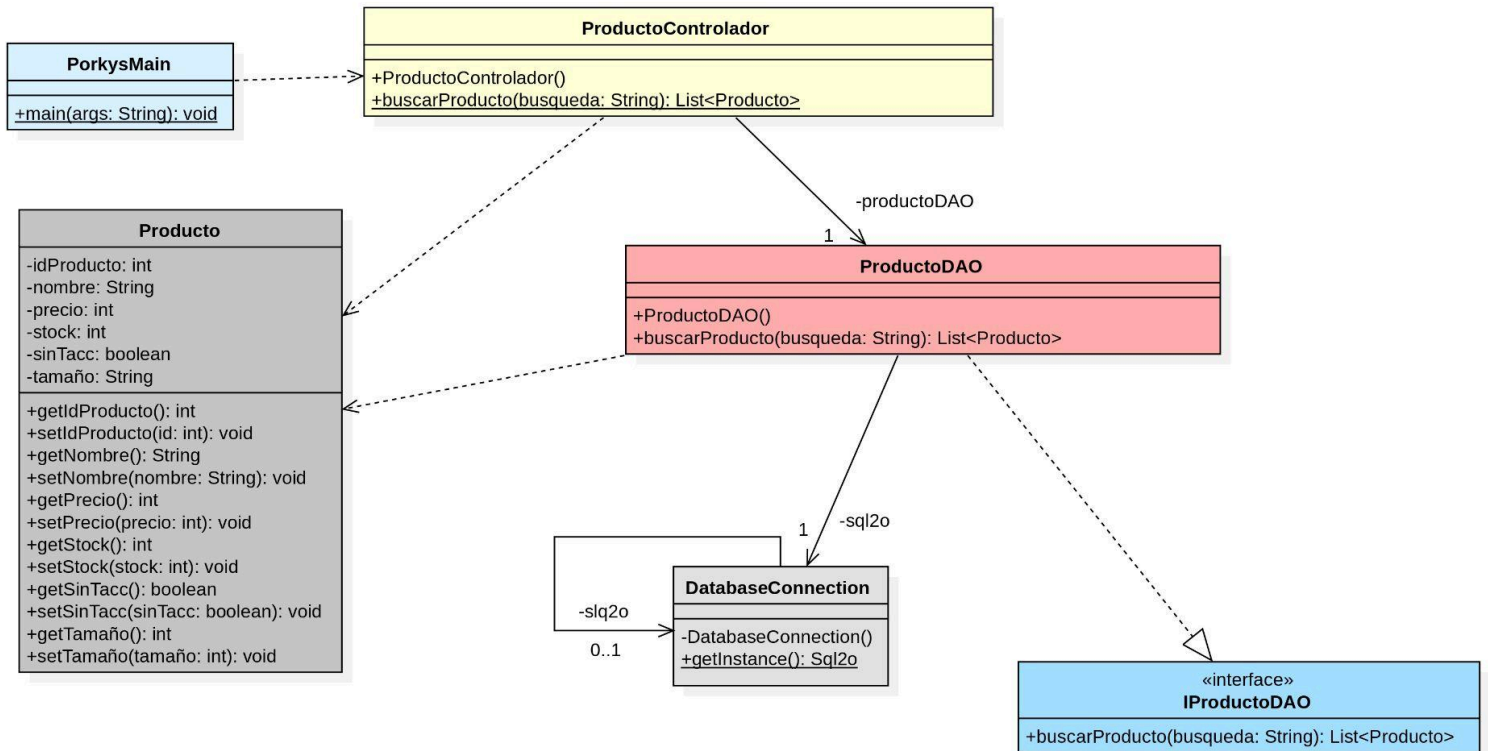
5 Post-condiciones

- 5.1 El sistema devuelve al actor el resultado de su búsqueda.

6 Requerimientos especiales

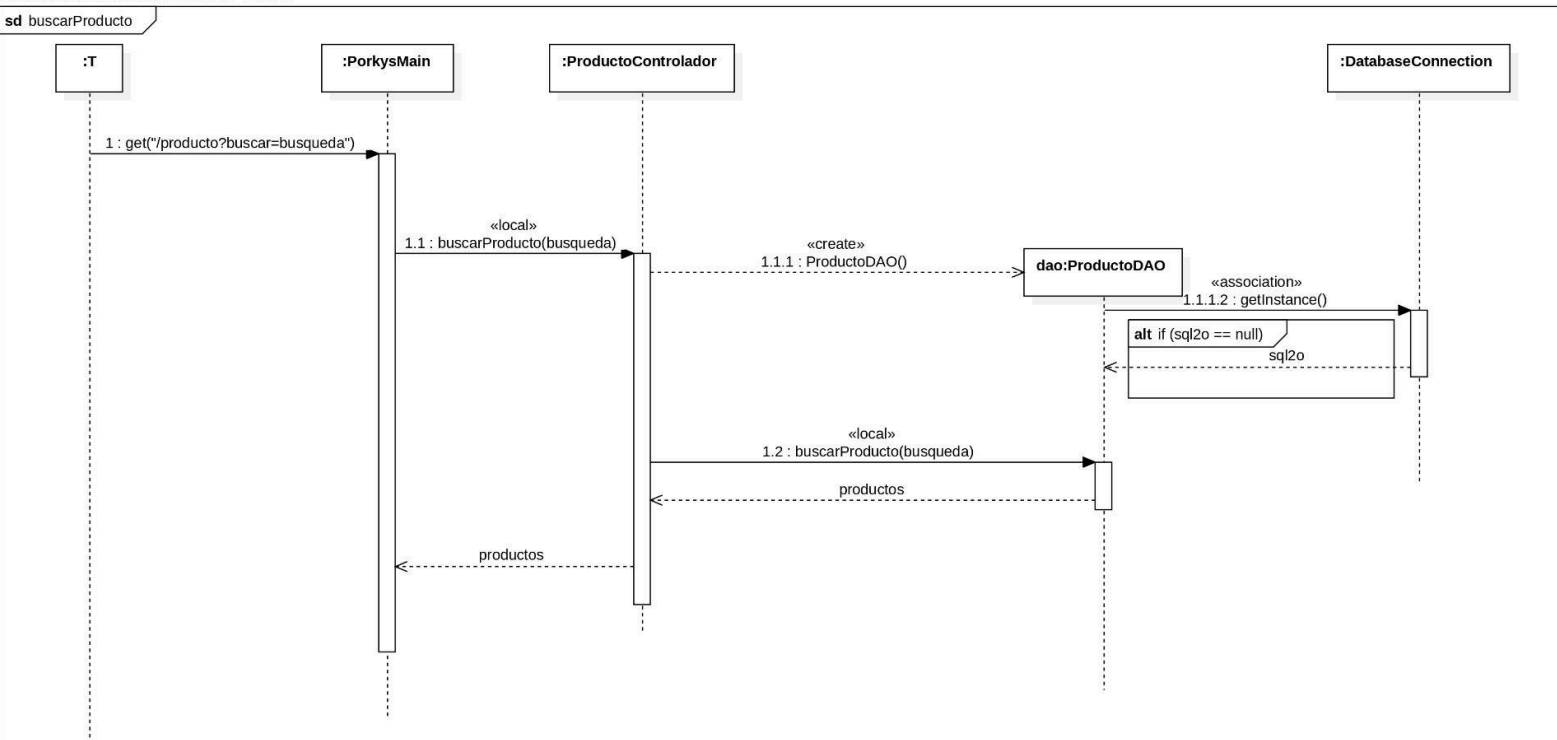
- La interfaz debe ser responsiva y ser visible fácilmente.
- La búsqueda debe ser rápida.
- El sistema debe ofrecer sugerencias de autocompletado mientras el actor escribe en el campo de búsqueda.

ii. Modelo de diseño estático



iii. Modelo de diseño dinámico

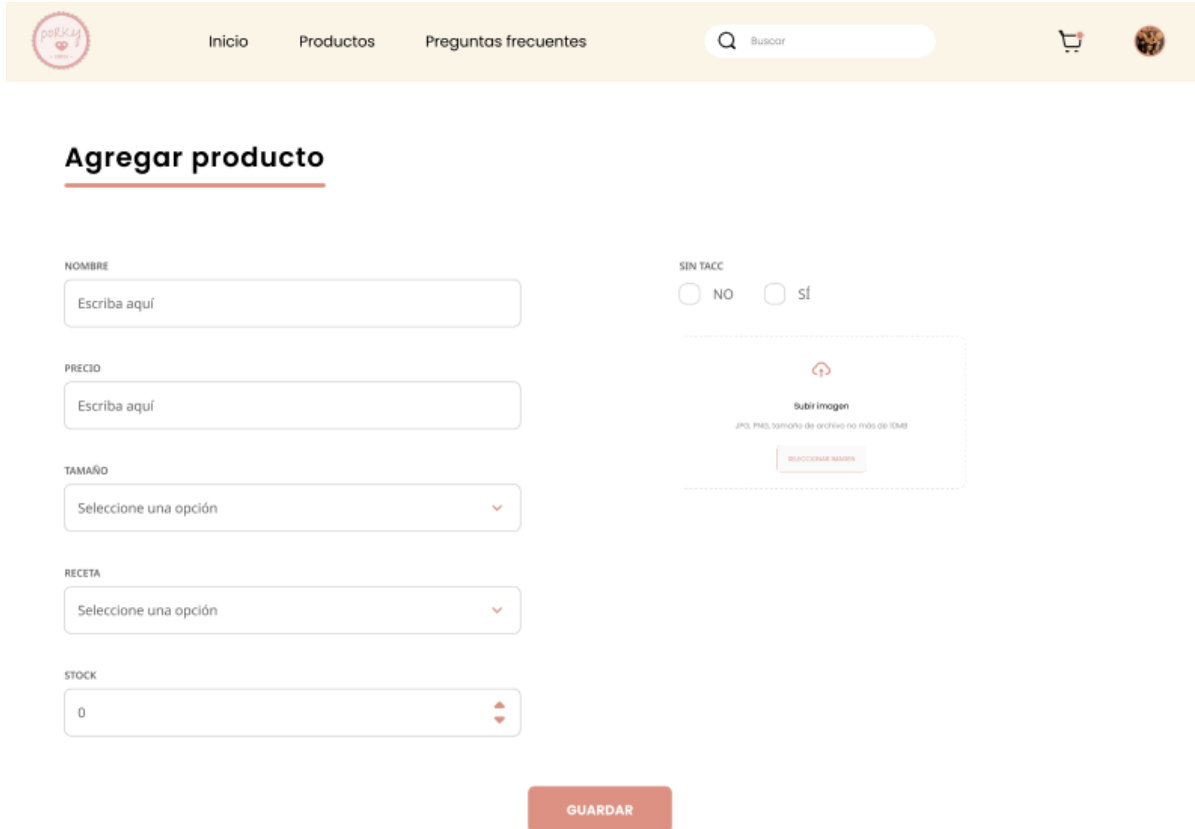
Collaboration1::Interaction1::buscarProducto



5. Segunda iteración

a. Caso de uso agregar producto

i. Especificación del c.u y prototipos de la interfaz



Agregar producto

NOMBRE
Escriba aquí

PRECIO
Escriba aquí

TAMAÑO
Seleccione una opción

RECETA
Seleccione una opción

STOCK
0

SIN TACC
☐ NO ☐ SI

Subir imagen
JPG, PNG, tamaño de archivo no más de 10MB
SELECCIONAR IMAGEN

GUARDAR

Caso de uso: Agregar producto

1 Breve descripción

El actor puede agregar productos al sistema, que son los que luego va a mostrar para vender.

1.1 Actor Principal: Administrador.

1.2 Actor Secundario: No tiene.

2 Precondiciones

El actor debe previamente haber iniciado sesión para poder utilizar este caso de uso.

3 Flujo de evento básico

1. El **actor** selecciona la opción "Agregar producto".

2. El **sistema** muestra un formulario para la creación de un nuevo producto, con los campos:
 - nombre.
 - precio.
 - tamaño.
 - receta asociada.
 - stock.
 - si se trata de un producto sin tacc o no.
 - imagen.
3. El **actor** completa los campos del producto. Luego envía la información, confirmando su operación.
4. El **sistema** almacena el producto en un medio de persistencia e indica que el producto fue agregado exitosamente.

4 Flujos alternativos

Alternativa Paso 3:

3.1 El **actor** intenta guardar un producto con un nombre que ya existe y el sistema muestra un mensaje de error avisando que ese nombre ya existe, solicitando que cambie el nombre por otro.

3.2 El **actor** no completa todos los campos y el **sistema** muestra un mensaje de error solicitando que se completen los datos.

5 Post-Condiciones

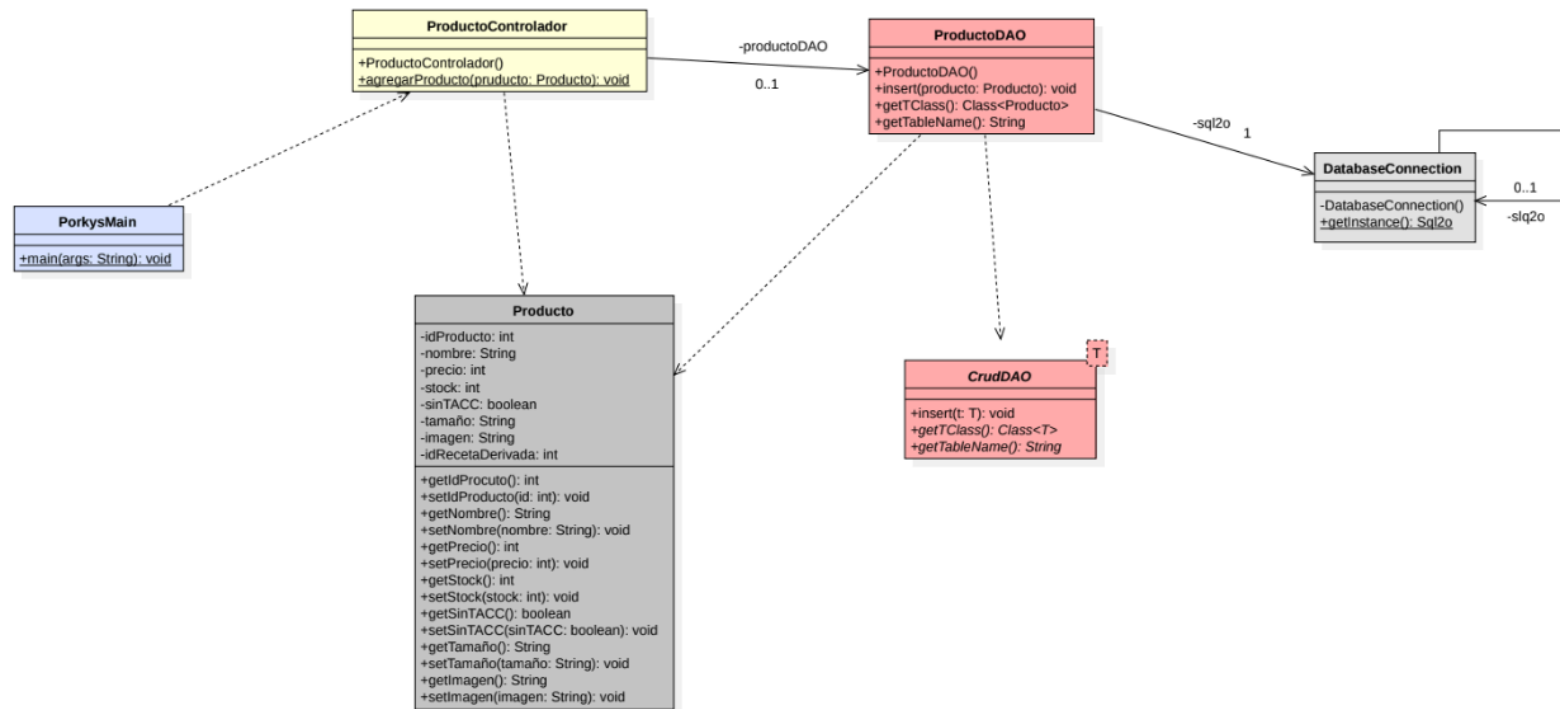
5.1 Se añadió al **sistema** de manera exitosa un producto, por ende, está disponible para su uso.

5.2 El **actor** puede seguir agregando más productos.

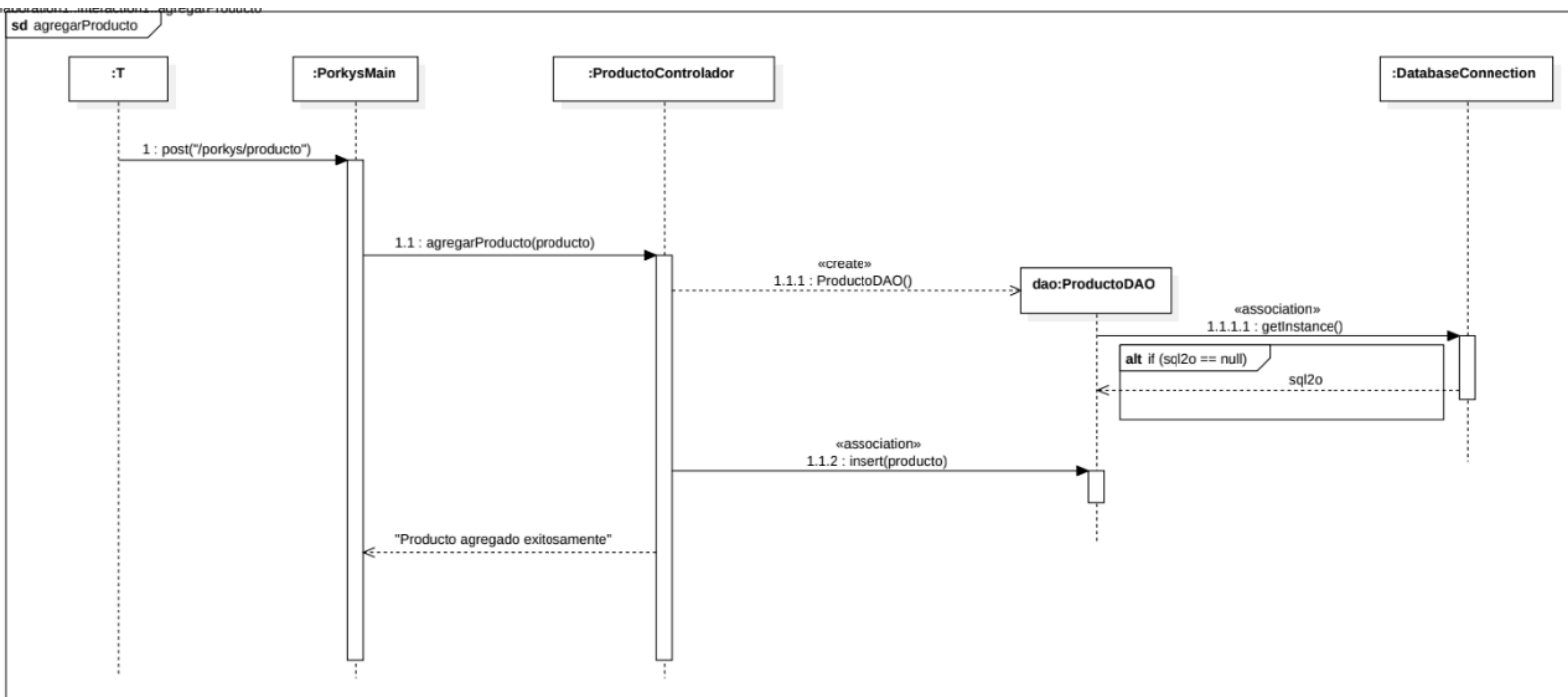
6 Requerimientos especiales

- Validar que se eviten los nombres duplicados.
- Asegurar que todos los campos de formulario sean completados.

ii. Modelo de diseño estático

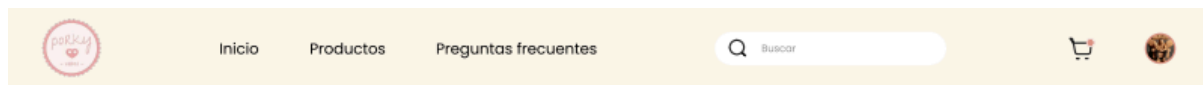


iii. Modelo de diseño dinámico



b. Caso de uso consultar lista de recetas

i. Especificación del c.u y prototipos de la interfaz



DERIVADAS

BASES

Recetas derivadas

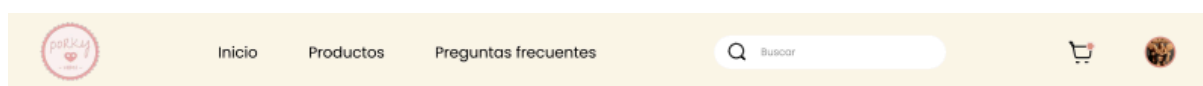
Delete

Filters

Export

Nueva receta

	Nombre	Tiempo de preparación	Porciones	Recetas bases
<input type="checkbox"/>	Pastel de chocolate	1.5 horas	8 a 10	Ver
<input type="checkbox"/>	Cheesecake de frutos rojos	2 horas	10 a 12	Ver
<input type="checkbox"/>	Croissants	3 a 4 horas	8 a 12	Ver
<input type="checkbox"/>	Tarta de limón y merengue	1.5 horas	8 a 10	Ver
<input type="checkbox"/>	Galletas de mantequilla	45 minutos	20 a 24 galletas	Ver
<input type="checkbox"/>	Cupcakes de vainilla	1 hora	12 cupcakes	Ver
<input type="checkbox"/>	Macarons	2 a 3 horas	20 a 24 macarons	Ver
<input type="checkbox"/>	Bizcocho de zanahoria	1.5 horas	8 a 10	Ver
<input type="checkbox"/>	Tarta de manzana	1.5 a 2 horas	8 a 10	Ver
<input type="checkbox"/>	Brownies	1 hora	12 a 16	Ver
<input type="checkbox"/>	Tarta Sacher	2 a 2.5 horas	12 a 14	Ver
<input type="checkbox"/>	Rollo de crema	1.5 horas	8 a 10	Ver
<input type="checkbox"/>	Tarta de queso	1.5 horas	8 a 10	Ver
<input type="checkbox"/>	Mini tartaletas de frutas	1 hora	12 a 16	Ver



DERIVADAS

BASES

Recetas bases

Delete

Filters

Export

Nueva receta

<div></div>	Nombre	Tiempo de preparación	Porciones	Tamaño	
<div></div>	Masa quebrada	30 minutos	8 a 10	Grande	
<div></div>	Masa de hojaldre	4 a 5 horas	10 a 12	Mediano	
<div></div>	Crema pastelera	20 a 30 minutos	8 a 12	Pequeño	
<div></div>	Masa de bizcocho genovés	1 hora	8 a 10	Grande	
<div></div>	Merengue	30 minutos	8 a 10	Mediano	
<div></div>	Ganache de chocolate	15 minutos	10 a 12	Pequeño	
<div></div>	Almibar simple	10 minutos	12 a 16	Pequeño	
<div></div>	Crema de mantequilla	20 minutos	8 a 10	Mediano	
<div></div>	Masa de galleta	15 minutos	8 a 10	Grande	
<div></div>	Pasta de almendra	30 minutos	12 a 16	Mediano	
<div></div>	Crema chantilly	10 minutos	12 a 14	Pequeño	
<div></div>	Masa sablé	30 minutos	8 a 10	Pequeño	
<div></div>	Crema inglesa	20 minutos	8 a 10	Mediano	
<div></div>	Frangipane	20 minutos	12 a 16	Pequeño	

Caso de uso: Consultar lista de recetas

1 Breve descripción

El actor visualiza todas las recetas almacenadas en el sistema, las cuales se agrupan en dos categorías: *Recetas Base* y *Recetas Derivadas*.

1.1 Actor Principal: Administrador.

1.2 Actor Secundario: No tiene.

2 Precondiciones

El actor debe previamente haber iniciado sesión para poder utilizar este caso de uso.

3 Flujo de evento básico.

1. El **actor** selecciona la opción "Ver recetas".
2. El **sistema** muestra dos pestañas:
 - **Recetas Derivadas** (por defecto), muestra una tabla con las columnas:
 - Nombre.
 - Tiempo de preparación.
 - Porciones.
 - Recetas Base (listando las bases relacionadas).
 - **Recetas Base**, muestra una tabla con las columnas:
 - Nombre.
 - Tiempo de preparación.
 - Porciones.
 - Tamaño.
3. El **actor** elige entre las pestañas para visualizar la categoría de receta deseada.

4 Flujos alternativos

Alternativa Paso 2:

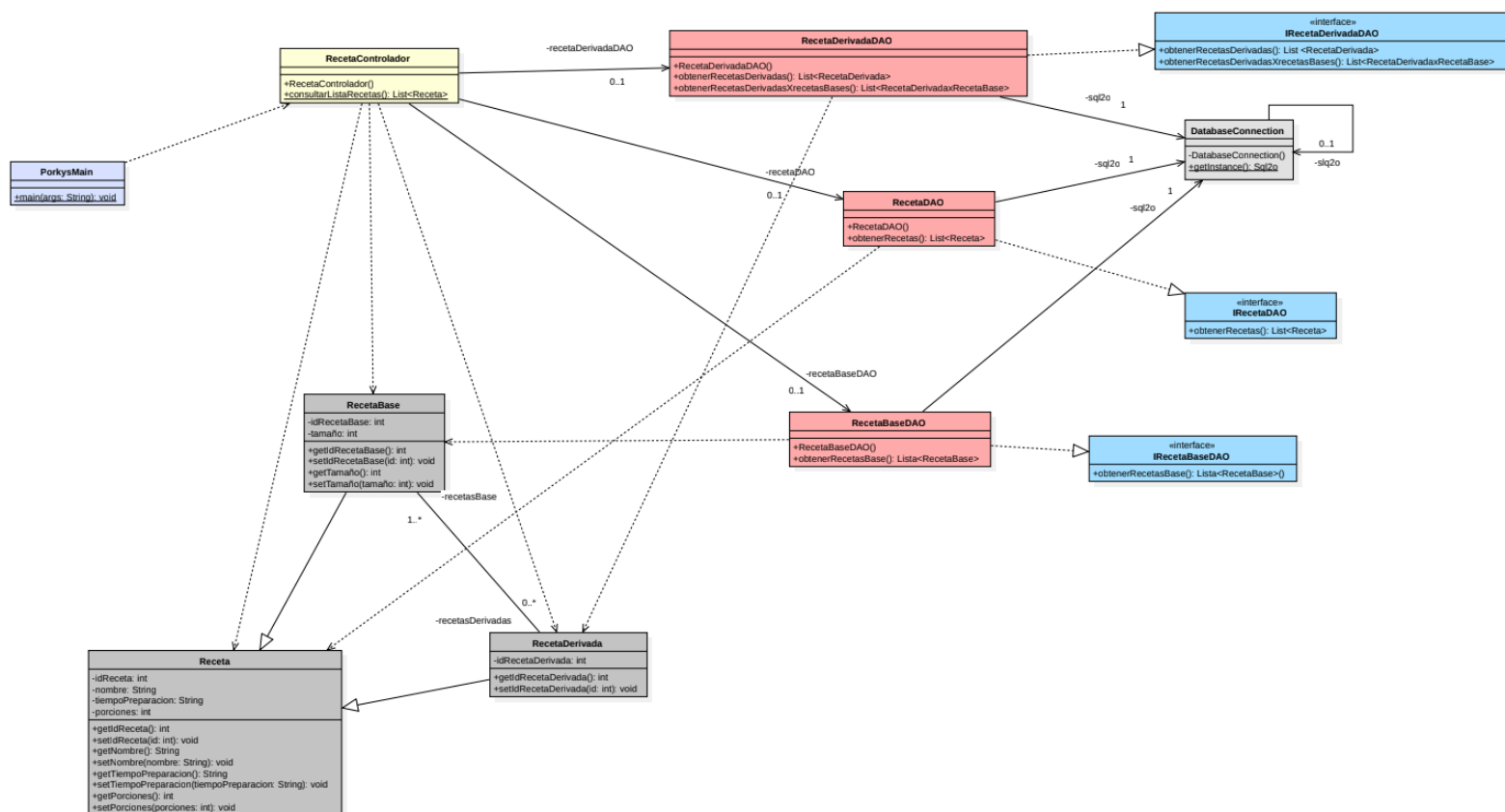
- ☐ 2.1 Si no existen recetas almacenadas, el **sistema** muestra un mensaje indicando que no hay recetas disponibles, junto con un botón para agregar una receta.

5 Poscondiciones

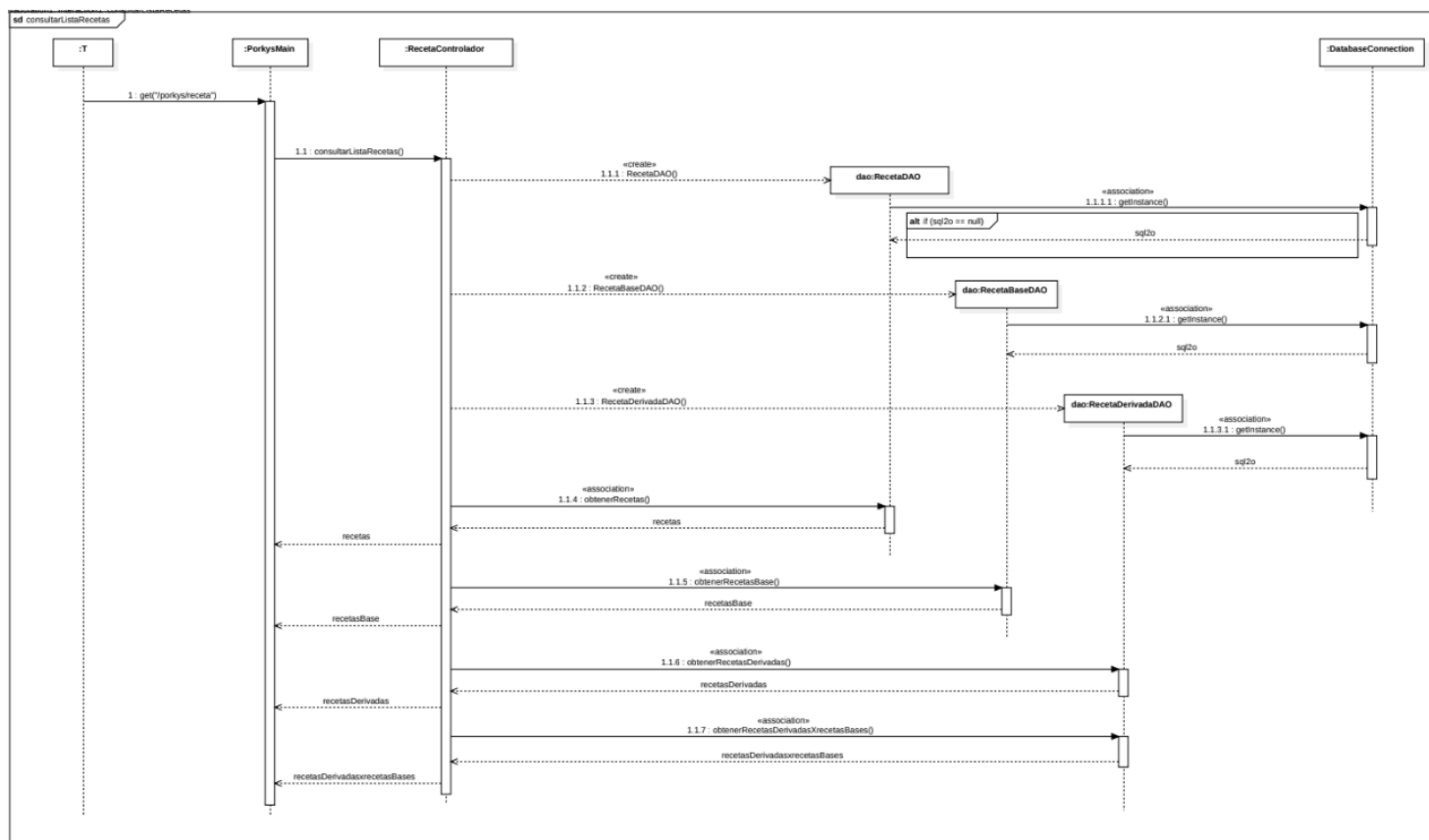
5.1 Se muestra en pantalla la lista de recetas almacenadas, organizada en las categorías correspondientes.



ii. Modelo de diseño estático



iii. Modelo de diseño dinámico



6. Explicación patrón DAO

En este proyecto se implementó el patrón DAO (Data Access Object) para gestionar la interacción con la base de datos de manera estructurada, desacoplando la lógica de acceso a datos del resto de la aplicación. Realizamos esto para centralizar el acceso a la base de datos y reducir la duplicación de código. Este patrón fue implementado siguiendo los principios de la interfaz DAO, con una implementación concreta por cada entidad del sistema.

7. Explicación patrón reflexivo

En este proyecto se implementó el patrón reflexivo ya que nos permitió construir instancias de clases y asignar sus valores dinámicamente a partir de datos externos (como JSON o formularios) sin necesidad de escribir código específico para cada clase o propiedad. Realizamos esto para reducir el código repetitivo al mapear datos de entrada a los objetos del modelo, adaptarse fácilmente a cambios en la estructura de las entidades sin necesidad de modificar la lógica de carga y proveer una solución genérica y reutilizable para instanciar objetos dinámicamente.

8. Explicación arquitectura del sistema

En el desarrollo del proyecto se adoptaron 2 arquitecturas:

- **Arquitectura REST:**

Utilizamos los principios del protocolo HTTP para construir servicios web, representando cada recurso por una URI única y usando los métodos estándar de HTTP: GET Y POST. Se utilizó JSON como formato principal de intercambio de datos debido a su simplicidad y a que las respuestas siguen un formato consistente, para facilitar la integración a futuro con frontend.

- **Arquitectura MVC:**

Lo utilizamos para separar las responsabilidades de una aplicación en tres componentes principales:

- Modelo (Model): Representa los datos y la lógica de negocio.
- Vista (View): Es la interfaz de usuario que muestra los datos y recibe la interacción del usuario. Dado que este proyecto se centró en el backend no fue una interfaz gráfica, en su lugar, se representó mediante respuestas JSON.
- Controlador (Controller): Gestiona las solicitudes del usuario, procesa la lógica de negocio a través del modelo, y actualiza la vista.

9. Link a repositorio en Github

https://github.com/mmmmel16/Grupo1_Porkys