# Introducción

## Propósito

El propósito del presente documento es establecer el alcance y la funcionalidad requerida por el sistema, el mismo reúne además el detalle de las restricciones a las que el sistema se debe adecuar. Será tomado como base para las actividades que siguen en el proceso de desarrollo.

Este documento está destinado a los desarrolladores, administradores y clientes del sistema.

## Alcance

El proyecto pretende ofrecer una solución para el abastecimiento de ingredientes y la gestión de inventario a las áreas o clientes que necesiten servicios desde el área de Compras en el sistema General de Empanadas Hornito Rico.

## Definiciones, acrónimos y abreviaturas

Sistema General de Empanadas Hornito Rico: es una referencia al sistema que permite la creación, venta, y gestión de empanadas en una empresa culinaria.

## Referencias

URL - Analisis principal: <https://docs.google.com/document/d/15nimoMkW-VupQgWLz0O_upoqDxCFxJGJqI_lm0XzK7k/edit#heading=h.2nqe8jhd7m7>

Diagramas de Procesos: <https://cawemo.com/share/e84b05ca-23f2-4a79-af9a-4d256d86b46d>

## Visión general del documento

En la Sección 2 del documento se describen los factores generales que afectan al producto, sus requerimientos, el ambiente en el cual será usado y todas las limitaciones y consideraciones supuestas.

En la Sección 3 se describen detalladamente los requerimientos del sistema y las restricciones que afectarán a dichas funciones.

Todo documento destinado a ampliar o aclarar cualquier concepto o especificación se ubica en la Sección Apéndices.

# Descripción general

## Perspectiva del producto

El producto a desarrollar funcionará de manera independiente por lo que no dependerá de otros sistemas existentes para poder operar. Existen productos similares en el mercado que ofrecen características similares al producto a desarrollar.

## Funciones del producto

El sistema debe poder generar solicitudes de ingredientes para evitar quedarse sin y tener un stock del inventario.

## Características de los usuarios

Los usuarios que deberán interactuar con el sistema son:

Administradores del área de Compras: tendrán conocimiento pleno del dominio.

Administradores del área de Producto: tendrán conocimiento limitado del dominio. Solamente podrán dar de alta o ver productos, y generar una solicitud de ingredientes.

## Restricciones generales

El producto deberá comunicarse con proveedores externos para obtener información sobre nuevos ingredientes.

## Suposiciones y dependencias

El producto será desarrollado exclusivamente para la empresa Hornito Rico destinada a la venta de Empanadas.

# Requisitos específicos

# Interfaces externas

Se describirán los requisitos que afecten a la interfaz de usuario, interfaz con otros sistemas (hardware y software) e interfaces de comunicaciones. DTO

DTO para generar una solicitud de ingredientes:

| {  “id”: 1,  "fecha\_solicitud": "2023-10-25T08:00:00", // Fecha y hora de la solicitud  "ingredientes": [  {  "id": "1",  "cantidad": 500 // Cantidad de ingredientes solicitados  },  {  "id": "2",  "cantidad": 50  }  ],  "descripcion": "Necesitamos estos ingredientes para un evento especial el fin de semana."  } |
| --- |

DTO de ingredientes:

| {  “id”: 1,  "nombre": "Aceitunas verdes",  "cantidad": 500 // Cantidad de ingredientes solicitados  "precio": 3.5  } |
| --- |

DTO de producto:

| {  “id”: 1,  "nombre": "Empanada de carne",  "descripcion": "Deliciosa empanada rellena de carne de res",  "precio": 2.5, // Precio del producto  "ingredientes": [  "Carne de res",  "Cebolla",  "Comino",  "Masa de empanada"  ],  "calorias": 300 // Valor calórico del producto  } |
| --- |

DTO de contabilidad:

| {  Fecha de emisión de factura/presupuesto,  Tipo de documento (factura o presupuesto),  Monto,  Medio de pago (indefinido, efectivo, transferencia, crédito, débito),  Cantidad de cuotas,  Porcentaje de descuento,  Porcentaje de interés,  Imagen del documento  } |
| --- |

## Funciones

RF1 - El sistema debe permitir dar de alta productos.

RF2 - El sistema debe permitir dar ver los productos.

RF3 - El sistema debe permitir actualizar productos.

RF4 - El sistema debe permitir generar solicitudes de ingredientes.

RF5 - El sistema debe poder enviar una solicitud de ingredientes a proveedores externos.

RF6 - El sistema debe permitir generar una orden de compra.

RF7 - El sistema debe notificar al área de contabilidad al generar una orden de compra.

RF8 - El sistema debe notificar al área de calidad al recibir ingredientes por parte del proveedor.

RF9 - El sistema debe permitir generar informes.

## Requerimientos no funcionales

RNF1 - Rendimiento: Especificar tiempos de respuesta aceptables para las operaciones, como la búsqueda de productos o la generación de informes. Esto garantiza que el software sea eficiente incluso con grandes volúmenes de datos.

RNF2 - Disponibilidad: Definir el tiempo de disponibilidad requerido del sistema para garantizar que esté siempre disponible para los usuarios, lo que puede implicar la implementación de redundancia y copias de seguridad.

RNF3 - Seguridad: Establecer requisitos de seguridad, como la encriptación de datos, la autenticación de usuarios y el control de acceso para proteger la información del inventario.

RNF4 - Escalabilidad: Especificar cómo el software debe escalar para manejar un aumento en la cantidad de productos o usuarios sin una degradación significativa del rendimiento.

RNF5 - Fiabilidad: Definir la tolerancia a fallos y los procedimientos de recuperación en caso de problemas inesperados, como caídas del sistema.

RNF6 - Usabilidad: Establecer criterios de usabilidad para garantizar que el software sea fácil de aprender y utilizar, lo que puede incluir pautas de diseño de interfaz de usuario.

RNF7 - Interoperabilidad: Si el software debe integrarse con otros sistemas, se deben definir los estándares de interoperabilidad necesarios.

RNF8 - Cumplimiento normativo: Especificar los requisitos legales y regulatorios que el software debe cumplir, como regulaciones de control de inventario o normativas fiscales.

RNF9 - Rendimiento de informes: Definir los tiempos de generación de informes, especialmente si se trata de informes complejos o extensos.

RNF10 - Mantenimiento y soporte: Especificar los requisitos de mantenimiento del software, incluyendo actualizaciones y soporte técnico.

## Arquitectura

### Descripción del Diseño Arquitectónico:

#### Componentes Principales:

* Interfaz de Usuario (UI):
  + La interfaz de usuario proporciona una forma intuitiva y amigable para que los usuarios interactúen con el sistema de abastecimiento de ingredientes y gestión de inventario.
  + Debe ser fácil de navegar y permitir a los usuarios realizar tareas como agregar productos, modificar cantidades, realizar búsquedas y generar informes y solicitudes de ingredientes.
* Base de Datos:
  + Almacena la información sobre los productos, sus cantidades, ingredientes y otros detalles relevantes.
  + Debe ser eficiente y estar diseñada para manejar grandes volúmenes de datos, permitiendo consultas rápidas y actualizaciones.
* Lógica de Negocio:
  + Contiene las reglas y algoritmos que gobiernan el comportamiento del sistema. Esto incluye la lógica para gestionar inventario, manejar pedidos, generar alertas de inventario bajo, generar solicitud de ingredientes, entre otras funcionalidades.
* Gestión de Usuarios y Autenticación:
  + Controla la autenticación y autorización de usuarios. Garantiza que solo usuarios autorizados tengan acceso a las funcionalidades del gestor de stock y abastecimiento de ingredientes.
* Integraciones Externas:
  + Puede incluir integraciones con sistemas de producción, contabilidad, etc. Esto permite que el gestor de stock y abastecimiento de ingredientes se comunique con otros sistemas dentro de la organización.
* API (Interfaz de Programación de Aplicaciones):
  + Proporciona una forma estandarizada para que otras aplicaciones o servicios se comuniquen con el gestor de inventario y abastecimiento de ingredientes. Esto puede ser útil para la integración con sistemas externos.
* Módulos de Reportes y Análisis:
  + Permite a los usuarios generar informes sobre el estado del inventario, tendencias de ventas, pronósticos de demanda, entre otros datos relevantes.

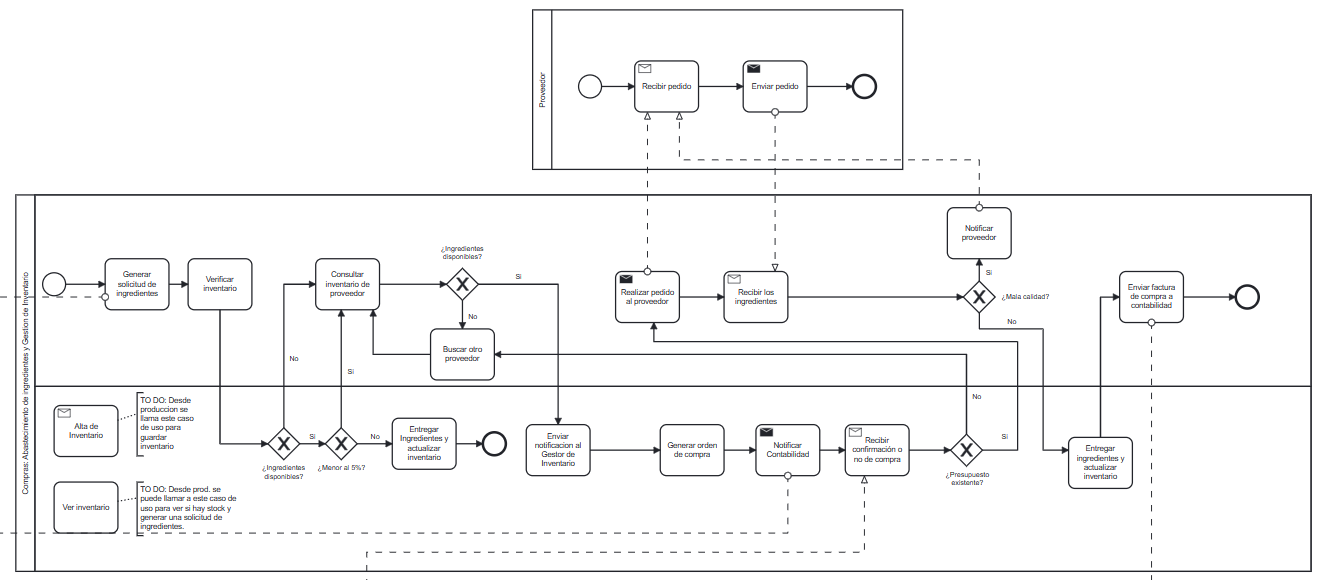
#### Justificación del Diseño:

* Escalabilidad:
  + El diseño arquitectónico propuesto permite que el sistema sea escalable. Esto significa que puede manejar un crecimiento en la cantidad de productos, transacciones y usuarios sin comprometer el rendimiento.
* Mantenibilidad:
  + El diseño se basa en una arquitectura modular, lo que facilita la incorporación de nuevas funcionalidades o la realización de actualizaciones sin afectar otras partes del sistema.
* Eficiencia y Rendimiento:
  + La separación de la lógica de negocio y la base de datos, así como la optimización de consultas, contribuye a un rendimiento eficiente, permitiendo respuestas rápidas a las solicitudes de los usuarios.
* Seguridad:
  + La gestión de usuarios y autenticación garantiza que solo personal autorizado tenga acceso al sistema, protegiendo la información crítica sobre el inventario.
* Flexibilidad e Integración:
  + El sistema está diseñado para poder integrarse con otros sistemas dentro de la organización, lo que facilita la colaboración entre diferentes departamentos.
* Usabilidad:
  + La interfaz de usuario se centra en proporcionar una experiencia intuitiva y fácil de usar, lo que reduce la curva de aprendizaje para los usuarios y aumenta la eficiencia en la gestión de inventario.

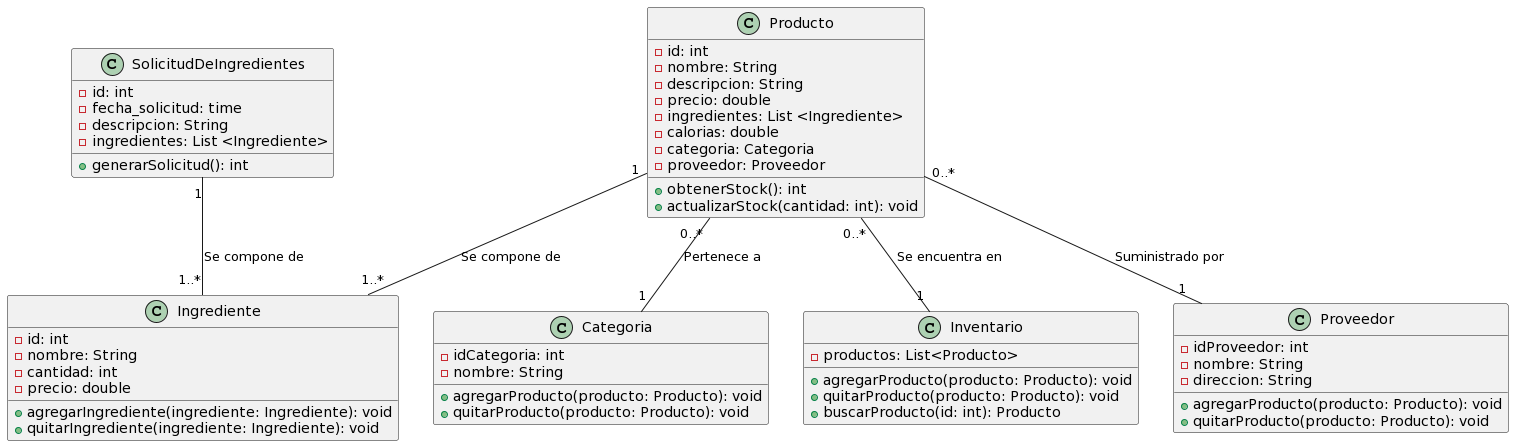
En resumen, el diseño arquitectónico propuesto ofrece un sistema robusto, escalable, seguro y eficiente para la gestión de inventario y abastecimiento de ingredientes, que se adapta a las necesidades actuales y futuras de la organización.

## Apéndices

BPMN:



**DC:**



**Código UML (Diagrama previo):**

@startuml

class SolicitudDeIngredientes {

-id: int

-fecha\_solicitud: time

-descripcion: String

-ingredientes: List <Ingrediente>

+generarSolicitud(): int

}

class Producto {

-id: int

-nombre: String

-descripcion: String

-precio: double

-ingredientes: List <Ingrediente>

-calorias: double

-categoria: Categoria

-proveedor: Proveedor

+obtenerStock(): int

+actualizarStock(cantidad: int): void

}

class Ingrediente {

-id: int

-nombre: String

-cantidad: int

-precio: double

+agregarIngrediente(ingrediente: Ingrediente): void

+quitarIngrediente(ingrediente: Ingrediente): void

}

class Categoria {

-idCategoria: int

-nombre: String

+agregarProducto(producto: Producto): void

+quitarProducto(producto: Producto): void

}

class Inventario {

-productos: List<Producto>

+agregarProducto(producto: Producto): void

+quitarProducto(producto: Producto): void

+buscarProducto(id: int): Producto

}

class Proveedor {

-idProveedor: int

-nombre: String

-direccion: String

+agregarProducto(producto: Producto): void

+quitarProducto(producto: Producto): void

}

Producto "0..\*" -- "1" Categoria : Pertenece a

Producto "1" -- "1..\*" Ingrediente : Se compone de

SolicitudDeIngredientes "1" -- "1..\*" Ingrediente : Se compone de

Producto "0..\*" -- "1" Inventario : Se encuentra en

Producto "0..\*" -- "1" Proveedor : Suministrado por

@enduml

**Prototipo interfaz:**

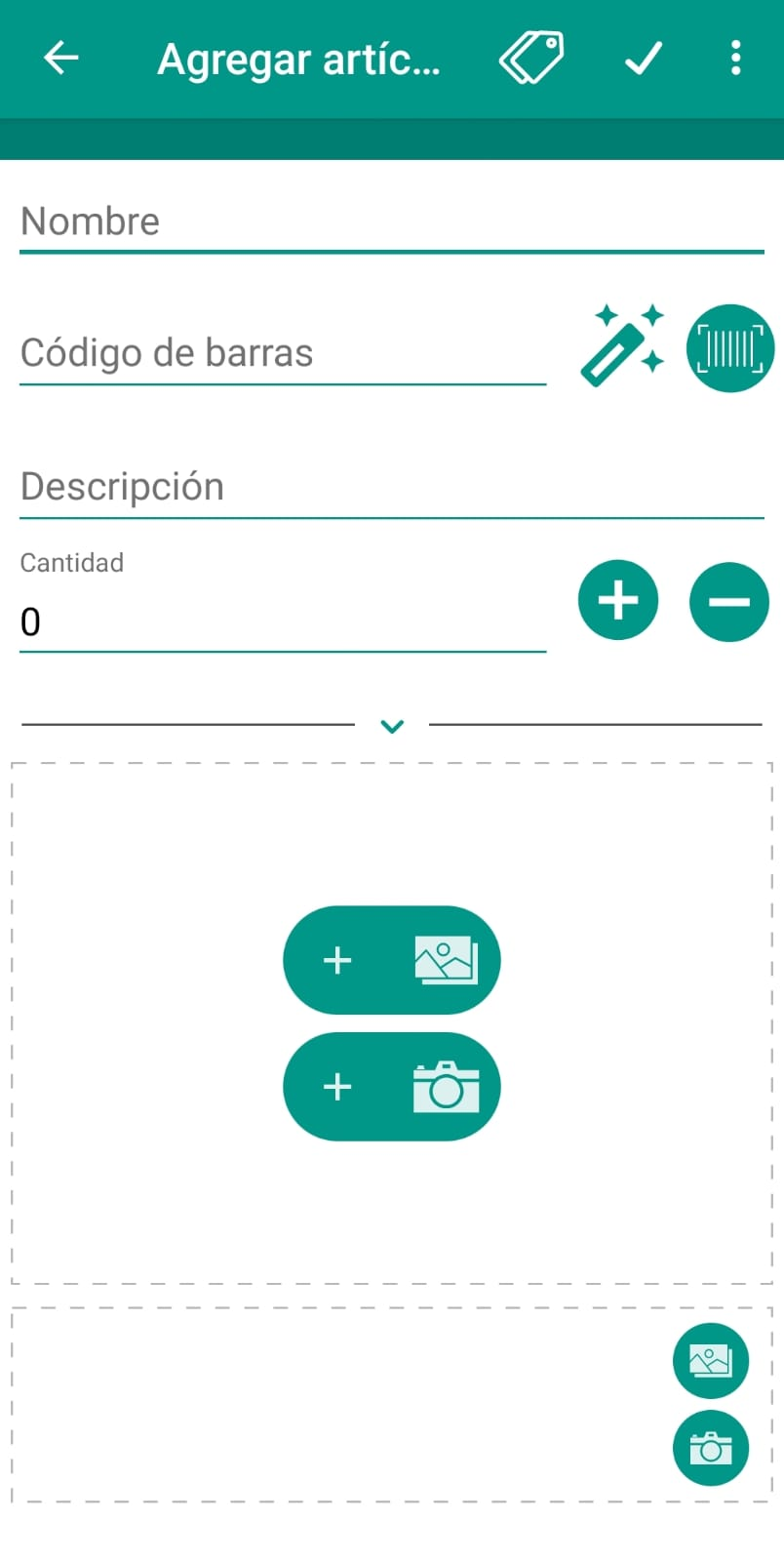
**Pantalla Inicial:**

****

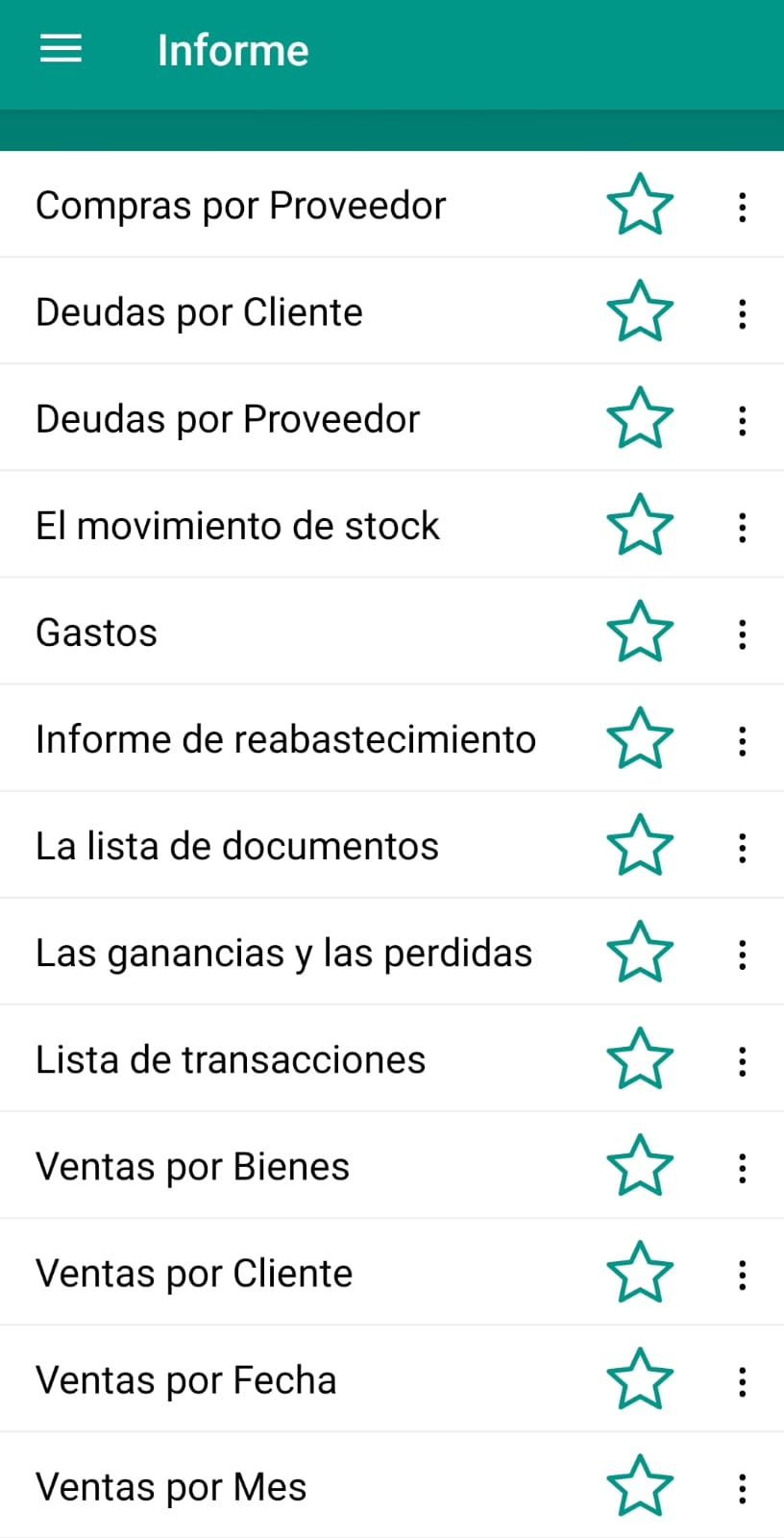
**Pantalla Productos:**

****

**Pantalla Alta Productos:**

****

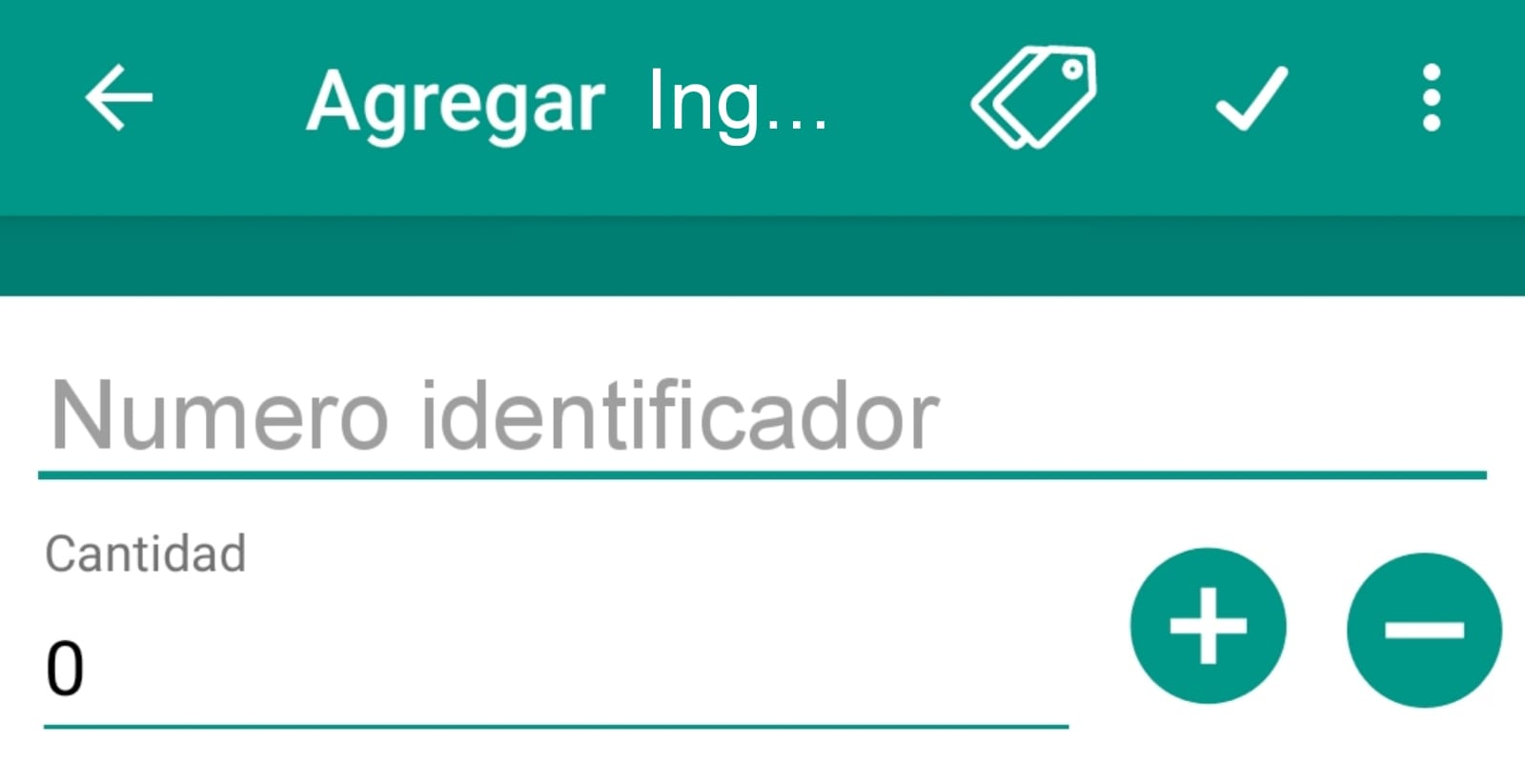
**Pantalla Informes:**

****

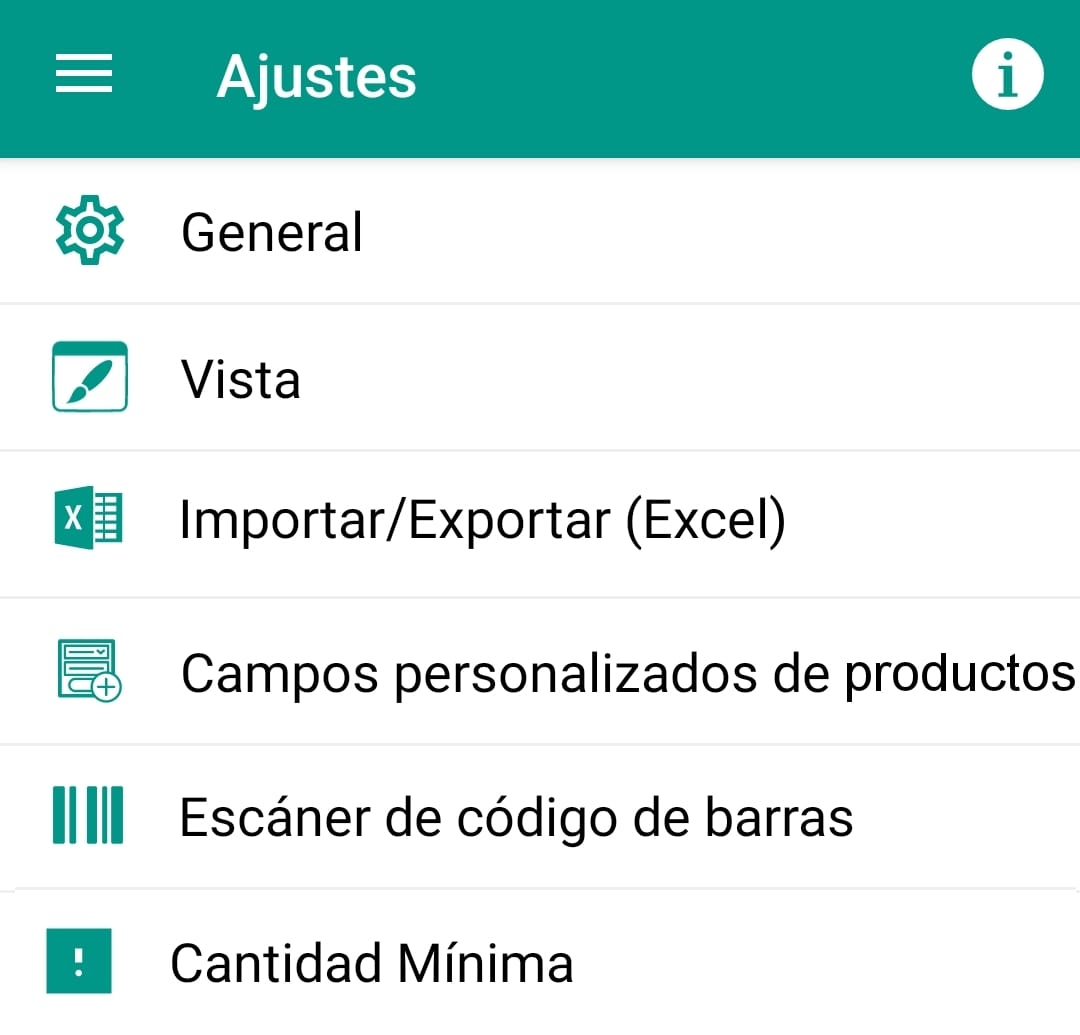
**Pantalla Solicitud de Ingredientes:**

****

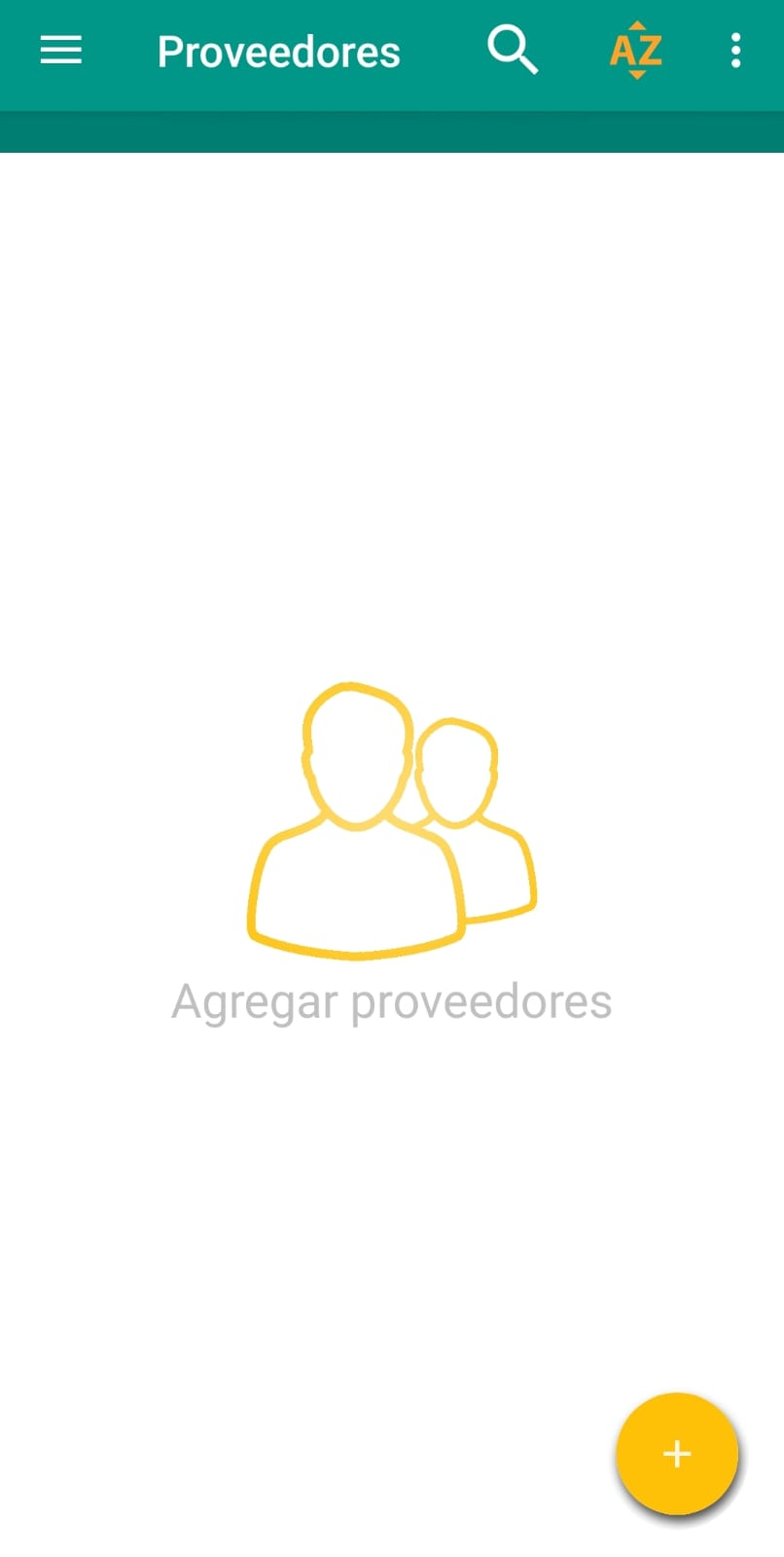
**Pantalla Alta Ingrediente:**

****

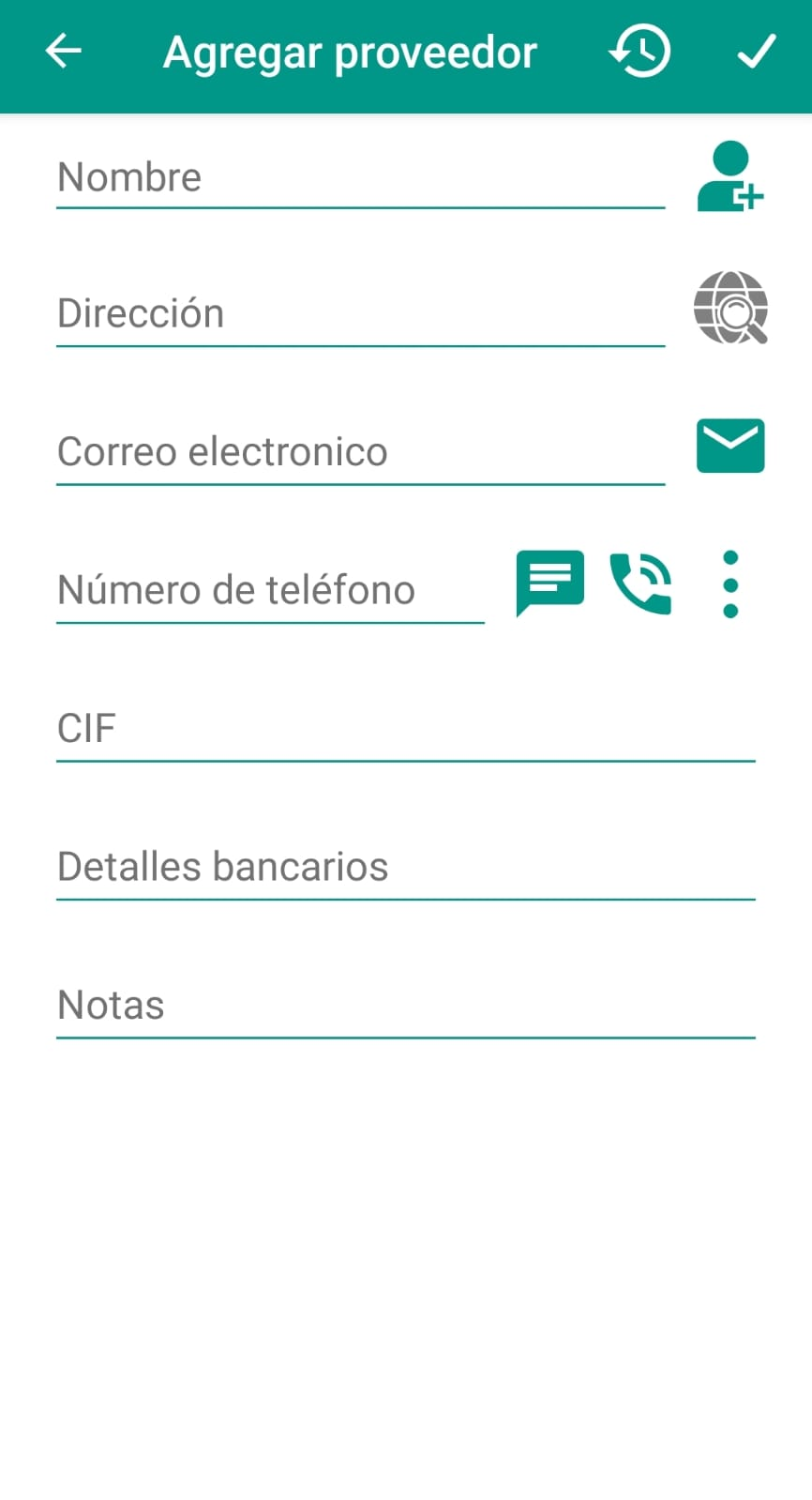
**Pantalla Ajustes:**

****

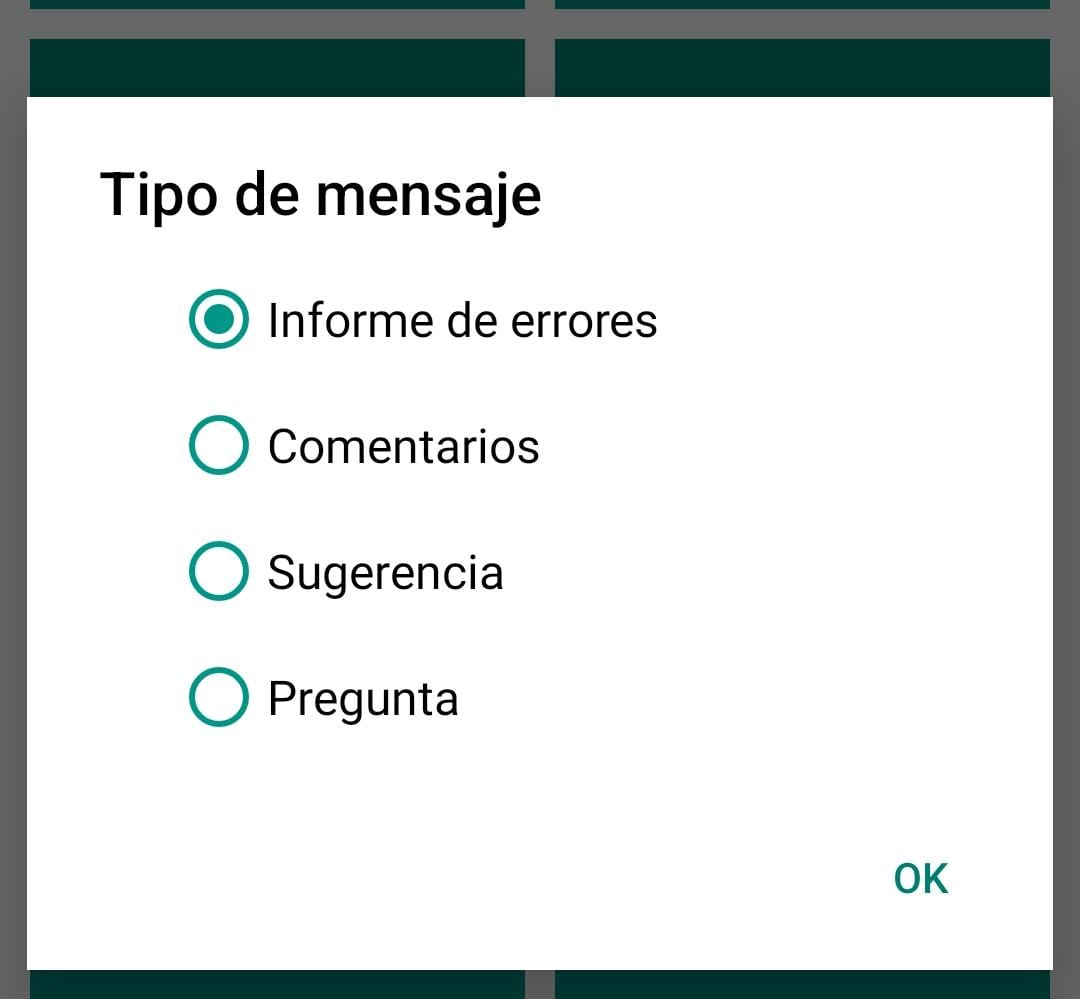
**Pantalla (botón) Proveedores:**

****

**Pantalla Alta Proveedor:**

****

**Pantalla (botón) Soporte:**

****

**Pantalla (botón) signos de pregunta y exclamación: Modal con información de la aplicación y modal con novedades de la aplicación.**