Casos de uso Extendido
KairosMix

Fecha	Versión	Autor	Verificado dep. calidad.
10/12/2025	1	Matías Lugmaña, Camilo Orrico, Denise Rea, Julio Viche	Mgt. Jenny Alexandra Ruiz Robalino

1. Introducción

El presente documento detalla la especificación de los Casos de Uso Extendidos para el módulo de Clientes dentro de la arquitectura del sistema **Kairoz Mix**.

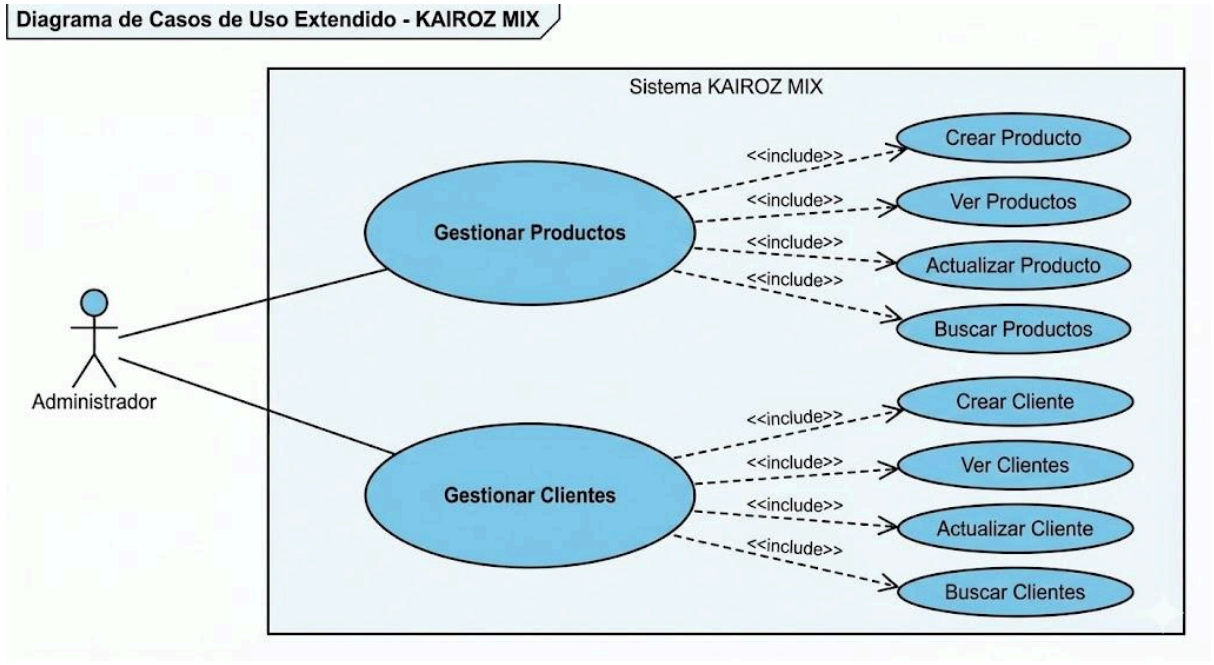
El objetivo de este módulo es permitir al usuario administrador gestionar la información base de los clientes (CRUD), asegurando la integridad de los datos mediante validaciones en la capa de negocio y presentación. Este diseño funcional está alineado con la arquitectura técnica previamente definida (React + Vite en Frontend y Node/Express + Mongo en Backend), garantizando que acciones como "Buscar", "Crear" o "Editar" se correspondan directamente con los controladores y servicios del sistema.

A continuación, se presentan los diagramas UML y la descripción narrativa de las interacciones, incluyendo las relaciones de inclusión (`<<include>>`) y extensión (`<<extend>>`).

---

2. Diagrama de Casos de Uso (Nivel General)

Este diagrama representa la interacción completa del actor principal con las funcionalidades de gestión de clientes. Se evidencian las dependencias obligatorias (validaciones) y las funcionalidades opcionales (filtros de búsqueda).



---

3. Desarrollo y Especificación de Casos de Uso

A continuación, se describen los flujos detallados representados en el diagrama anterior.

3.1. Caso de Uso Principal: Gestionar Clientes (CRUD)

Actor: Administrador / Usuario de Ventas **Descripción:** El actor accede al módulo para realizar operaciones de mantenimiento sobre la base de datos de clientes.

- **Flujo Básico:**
    1. El actor ingresa a la sección "Clientes".
    2. El sistema (Frontend) solicita la lista de clientes al Backend (`ClientController.getClients`).
    3. El sistema muestra la tabla de datos (`ClientTable`).

3.2. Caso de Uso: Registrar Nuevo Cliente

Tipo: `<<include>>` (Incluye Validación) **Descripción:** Proceso para dar de alta un nuevo cliente en el sistema Kairoz Mix.

- **Flujo Principal:**
    - El actor selecciona "Nuevo Cliente".
    - El sistema despliega el formulario modal (`ClientModal`).
    - El actor ingresa los datos (Nombre, Cédula/RUC, Dirección, Teléfono).
    - El actor presiona "Guardar".
    - **Relación `<<include>>` - Validar Datos:** El sistema verifica obligatoriamente que:
        - Los campos requeridos no estén vacíos.
        - El formato del correo sea correcto.
        - La Cédula/RUC no esté duplicada en la base de datos.
    - Si la validación es exitosa, se envía la petición al repositorio (`MongoClientRepository`) y se persiste el dato.
- **Flujo Alternativo (Error de Validación):**
    - Si la validación falla (ej. RUC duplicado), el sistema extiende el caso de uso mostrando una alerta de error y no permite guardar.

3.3. Caso de Uso: Buscar Cliente

Tipo: `<<extend>>` (Filtro por Criterio) **Descripción:** Permite localizar un cliente específico dentro del gran volumen de datos.

- **Flujo Principal:**
    3. El actor visualiza la barra de búsqueda (`ClientSearch`).
    4. El actor ingresa un término de búsqueda.
- **Relación `<<extend>>` - Filtrar Resultados:**
    3. El caso de uso base "Buscar" se extiende si el usuario decide aplicar filtros específicos (ej. buscar solo por "Cédula" o solo por "Nombre").
    4. El sistema realiza la consulta dinámica (`searchClients`) y actualiza la tabla en tiempo real.

3.4. Caso de Uso: Editar Cliente

Descripción: Modificación de datos de un cliente existente.

- **Flujo Principal:**
    1. El actor selecciona un cliente de la tabla.
    2. El sistema carga los datos actuales en el `ClientModal`.
    3. El actor modifica la información (ej. actualiza el número de teléfono).
    4. **Relación `<<include>>` - Validar Existencia:** El sistema verifica que el ID del cliente siga existiendo antes de intentar el *update*.
    5. Se guardan los cambios.

---

4. Conclusión Técnica

La implementación de estos casos de uso se soporta directamente en la arquitectura de capas diseñada para **Kairoz Mix**:

1. La **Interfaz (React)** maneja las excepciones de usuario y validaciones visuales.
2. El **Controlador (ClientController)** orquesta la lógica de negocio recibiendo las peticiones de creación y edición.
3. El **Repositorio (MongoClientRepository)** ejecuta las operaciones físicas sobre la base de datos MongoDB.

Este diseño desacoplado asegura que, si en el futuro se requieren nuevos datos para el cliente (ej. geolocalización), solo sea necesario extender el modelo y el formulario, manteniendo intactos los flujos de casos de uso principales.