Campus Santa Fe

Ingeniería en Tecnologías Computacionales (ITC)

Semestre: 5°

Clase:

Integración de seguridad informática en redes y sistemas de software (Gpo 402)

Título:

SRS: Análisis de Requerimientos

Equipo 6:

Fernando Adrián Fuentes - A01028796

Pedro Mauri Martínez - A01029143

Ricardo Alfredo Calvo - A01028889

Salvador Gilberto Vaquero - A01027920

Profesores:

Carlos Enrique Vega Álvarez

Edith Carolina Arias Serna

Lizbeth Peralta Malváez **(Coordinadora)**

Osvaldo Cecilia Martínez

SRS

Introducción:

1. Contexto del Cliente:

La Fundación Sanders es una fundación mexicana la cual se dedica por medio de donaciones y eventos; apoyar a grupos necesitados por medio de los siguientes programas de ayuda:

- **Salud reproductiva:** Intervención preventiva en embarazo precoz y enfermedades de transmisión.
- **Agua para todos:** Mediante capacitación de agua pluvial y abasto de agua potable.
- **Nutrición:** Promover la salud y la sana alimentación a través de un modelo de intervención preventiva.

Página web de la fundación: https://sanders.com.mx

Actualmente, la fundación no tiene una base de datos acerca de sus donadores, ellos llegan de manera presencial o por medio de plataformas como PayPal, lo que impide recopilar información estructurada de los donantes y donaciones. Lo cual hace a la fundación ineficiente a la hora de tomar decisiones que les permita crecer.

2. Oportunidad de negocio:

La fundación necesita una forma para tener un control de sus donantes y sus donaciones, una interfaz que les permita visualizar la información importante para prescindir de procesos obsoletos y esfuerzo manual para que la fundación pueda llegar a más gente. La fundación requiere un CRM (Customer Relationship Management) o un Software de Gestión de Relación con los Clientes para impulsar su causa.

El sistema propuesto no solo automatizará las respuestas de donativos, sino que también permitirá una gestión integral de los donantes, lo que fortalecerá la gestión, toma de decisiones y transparencia de la Fundación Sanders, permitirá la visualización de datos críticos para la toma de decisiones estratégicas y facilitará la promoción del impacto social logrado a través de las donaciones. A largo plazo, el sistema podrá escalar para cubrir otros programas de la fundación o otras funciones de gestión, ampliando su impacto social.

3. Descripción de Solución:

El CRM será desarrollado utilizando React-Admin como framework para la interfaz de usuario y MongoDB como base de datos para garantizar flexibilidad y escalabilidad. Únicamente habrá dos roles de usuario para diferenciar entre “Las Sanders” y los empleados de la fundación; mientras que los donadores solo tendrán la capacidad de crear una donación.

Historias de Usuario:

Yo, como Sanders me gustaría que el sistema…

- Guarde suficiente información del usuario para identificarlo (Nombre, apellido, mail, teléfono y número de donaciones).
- Guarde suficiente información de las donaciones para relacionarlas con su donante (Información del donante, comentario, monto y fecha).
- Ver la información de las donaciones y donantes para utilizar esta información para tomar decisiones financieras.
- Me dejará ver en gráficas la información de las donaciones y donantes en relación a fechas o montos para que sea de una manera más eficiente y eficaz.
- Me permita crear, ver, editar y eliminar las cuentas de mis empleados para tener control sobre ellos.
- Tenga un manual para las funciones que yo pueda usar en la aplicación para que sea fácil de aprender la aplicación.
- No tenga vulnerabilidades que puedan comprometer los datos guardados para proteger la identidad de los donantes.
- Sea entendible en cada acción (botones con función clara) para que no me pierda.
- Que la interfaz sea intuitiva y agradable a la vista para que sea fácil de usar.
- Que tenga como un segundo como tiempo de respuesta para no perder el tiempo.

Yo, como empleado de la fundación me gustaría que el sistema…

- Guarde suficiente información del usuario para identificarlo (Nombre, apellido, mail, teléfono y número de donaciones).
- Guarde suficiente información de las donaciones para relacionarlas con su donante (Información del donante, comentario, monto y fecha).
- Ver la información de las donaciones y donantes para utilizar esta información para tomar decisiones financieras.
- Me dejará ver en gráficas la información de las donaciones y donantes en relación a fechas o montos para que sea de una manera más eficiente y eficaz.

- Tenga un manual para las funciones que yo pueda usar en la aplicación para que sea fácil de aprender la aplicación.

Yo, como donante me gustaría que el sistema…

- No pidiera mucha de mi información personal para mantener mi privacidad.
- Me gustaría que mis datos sean tratados de forma confidencial y segura para no exponer información personal.
- Que la interfaz sea intuitiva y agradable a la vista para realizar mis donaciones.
- Sea entendible en cada acción (botones con función clara) para no perderme.

Requerimientos:

Interfaz en React - Admin:

Funcionales:

1. Las Sanders y los empleados pueden iniciar sesión para ingresar a la interfaz.
2. Las Sanders tienen un CRUD completo del recurso de “Cuentas de empleados”.
3. Los empleados únicamente pueden ver las cuentas de los empleados.
4. Las Sanders y los empleados pueden crear y ver las cuentas de donantes y todas las donaciones.
5. Las Sanders y los empleados pueden ver estadísticas de las cuentas de donadores y de donaciones.
6. Los donantes únicamente pueden crear donaciones y cuentas de donantes, no será necesario que inicie sesión para realizar donaciones.
7. Para crear una donación, se pedirá la siguiente información; Nombre, apellido, mail, teléfono, comentario, monto y fecha (se llena automáticamente).
8. Notificar de manera clara que una acción importante como borrar, fue realizada.
9. Poder restaurar alguna acción hecha por accidente.
10. Que muestre mensajes de error si faltan campos que llenar en inputs.

No Funcionales:

1. Que tenga documentación adecuada para los administradores.
2. Realizar una interfaz que sea intuitiva y agradable a la vista.
3. Utilizar HTTPS para el front y backend para asegurar la información.

Base de datos en Mongodb:

Funcionales:

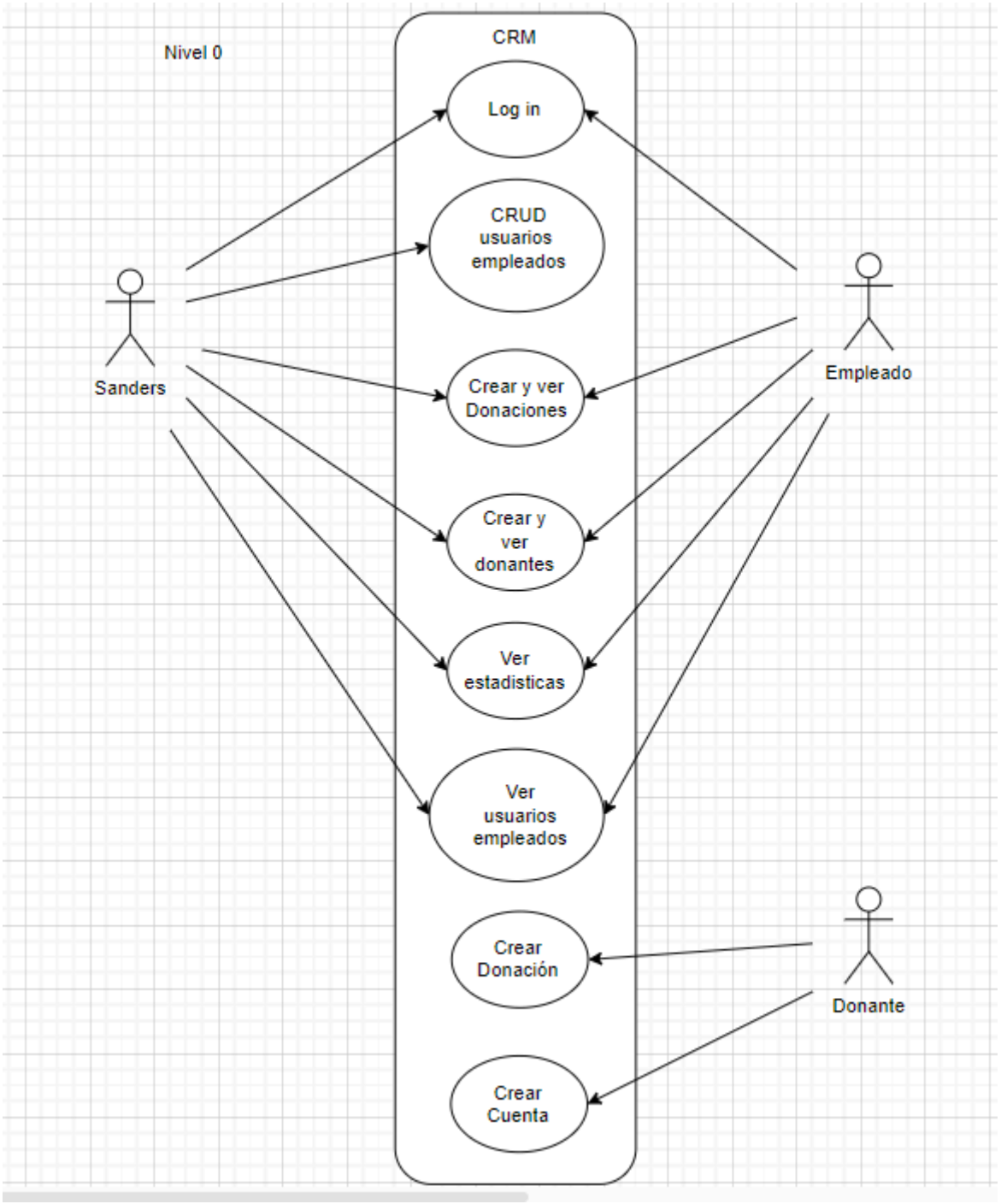
1. Guardar la información proporcionada de los donantes (nombre, correo, teléfono y monto donado).
2. Guardar la información de los empleados (nombre, rol, correo, contraseña.)
3. Tenga conexión con el proyecto React a través de la api.

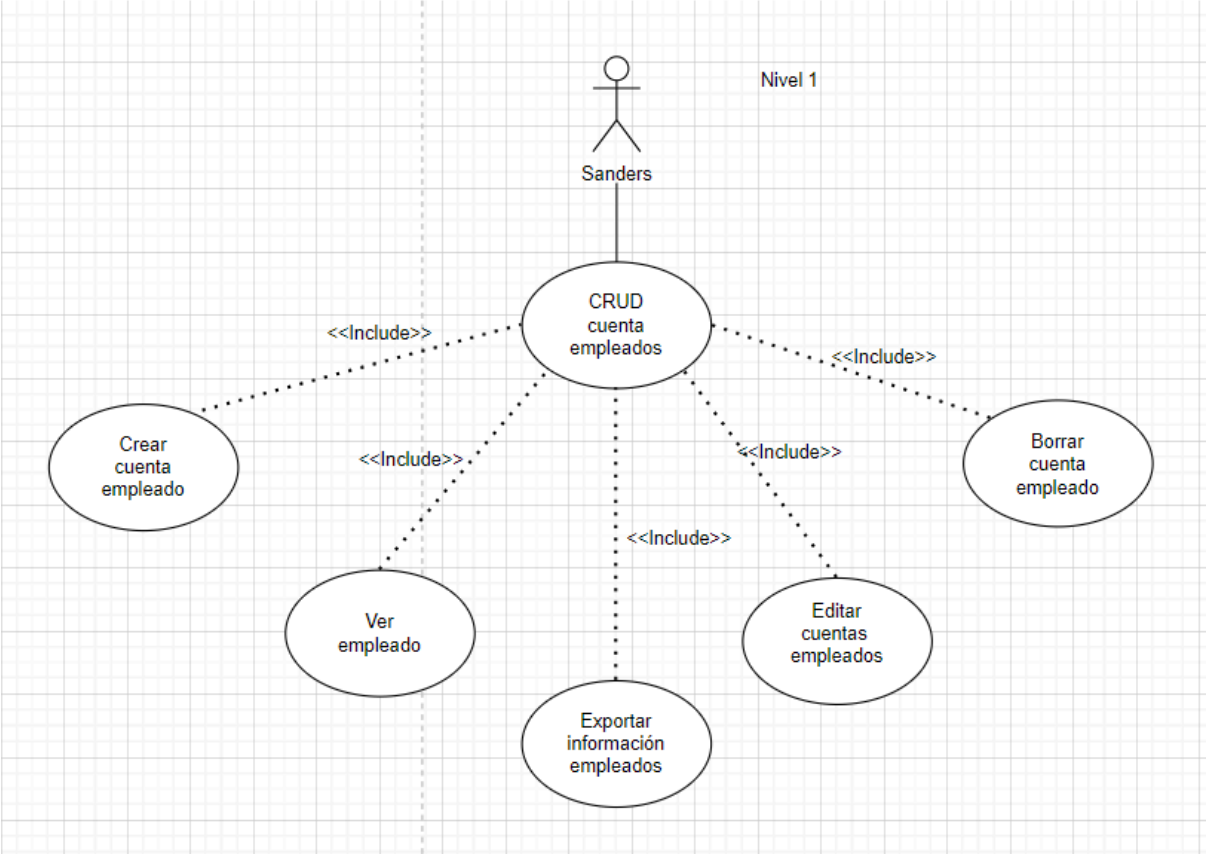
No Funcionales:

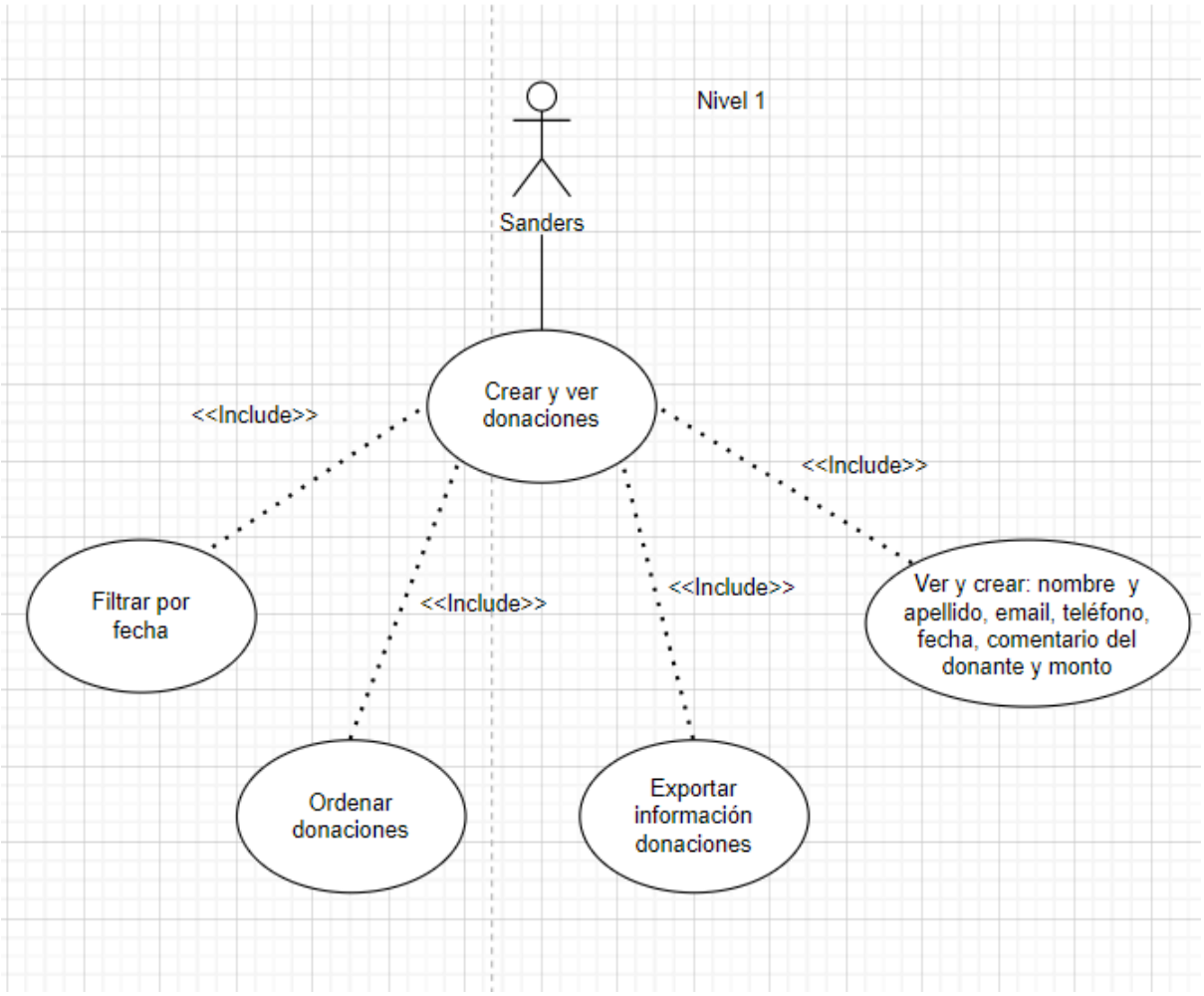
1. Encriptar la información de la base de datos.
2. Que el proyecto tenga buen rendimiento. Esto significa que la respuesta del sistema debería ser menor a 1.5 segundos, especialmente durante la edición o modificación sobre la información. Además debe responder el 95% de las veces.
3. Que tenga documentación adecuada para los administradores.
4. Que la base de datos tenga la capacidad de escalabilidad.

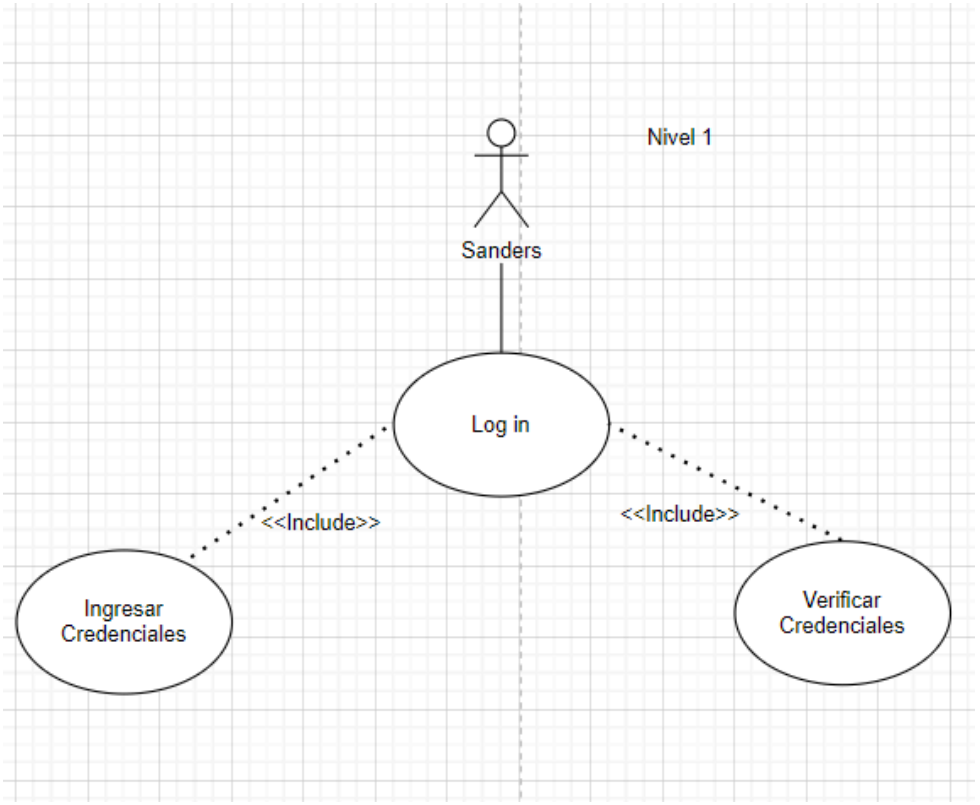
Diagramas:

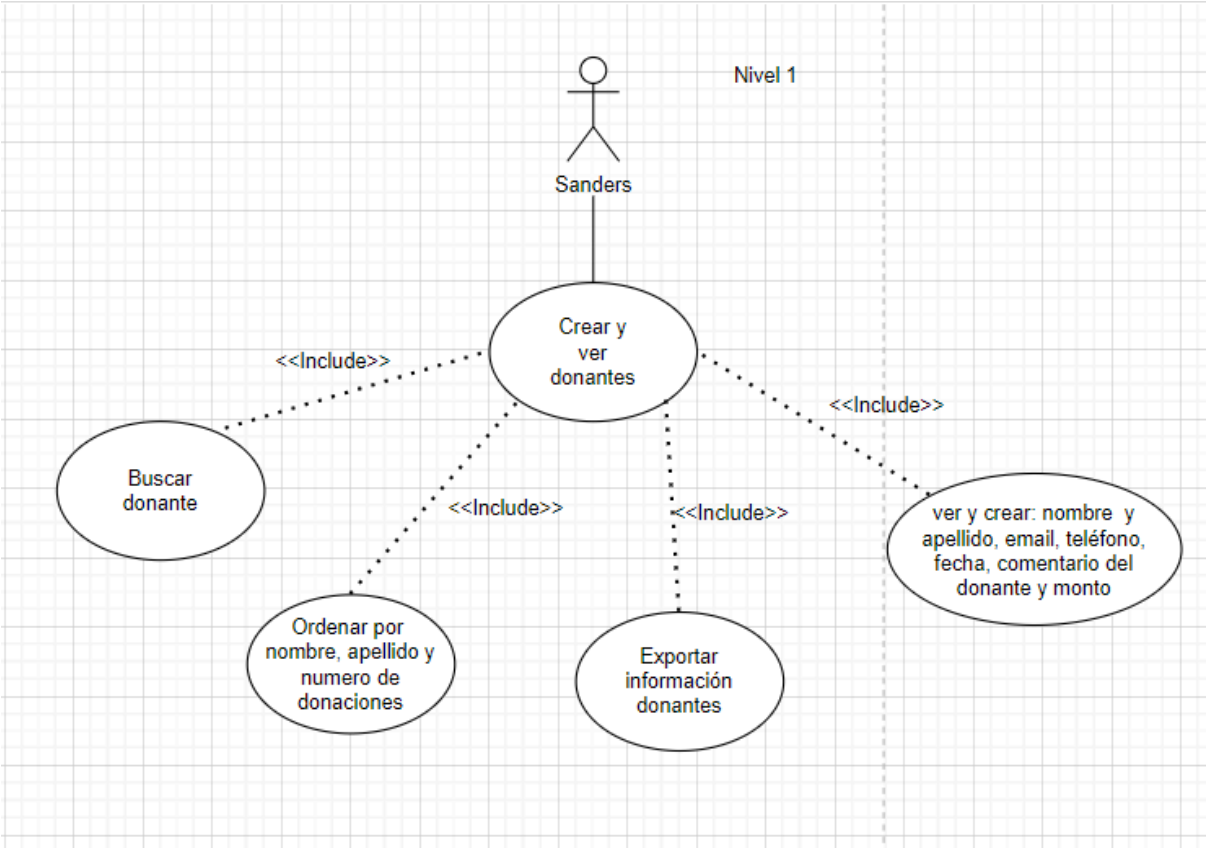
- ## Diagramas UML:
    1. **Diagrama de Casos de Uso:**

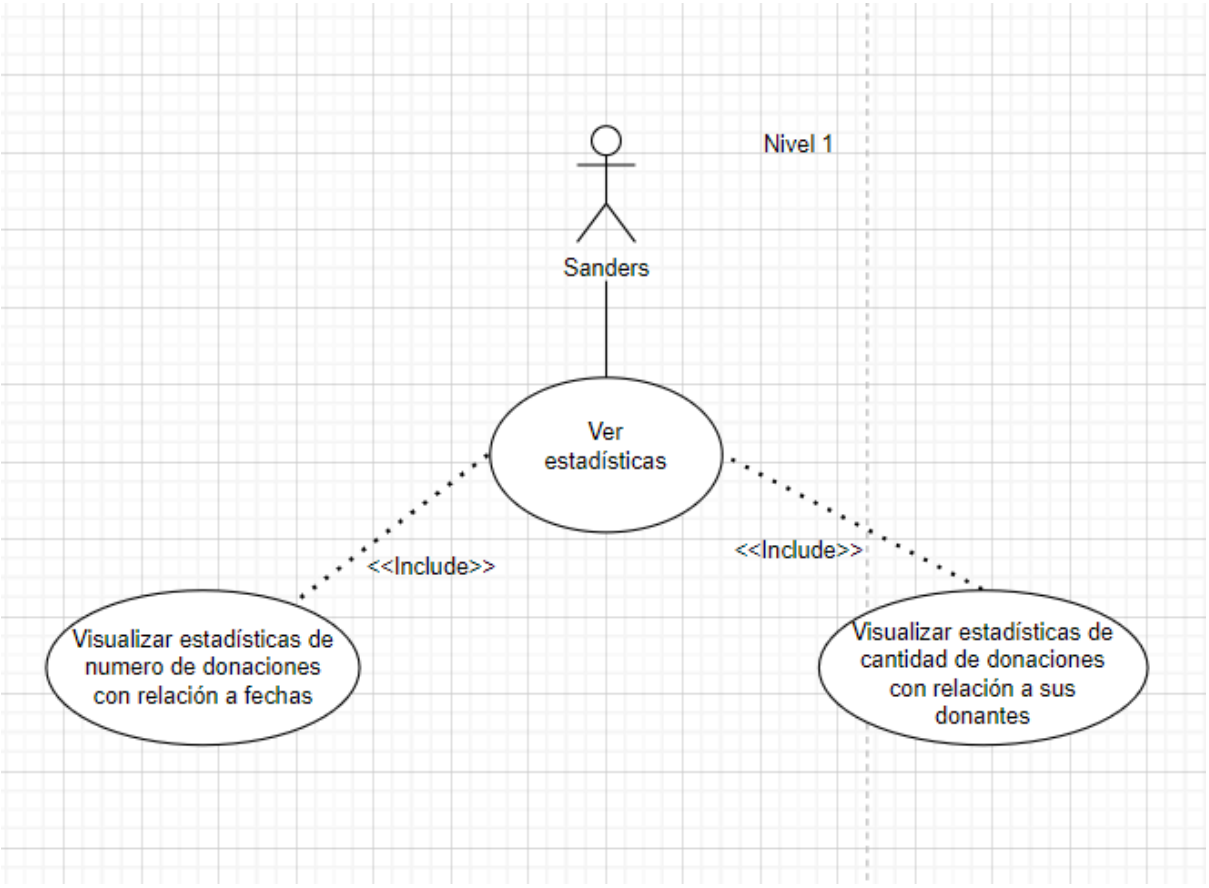

Nivel 0
CRM
Log in
CRUD usuarios empleados
Crear y ver Donaciones
Crear y ver donantes
Ver estadisticas
Ver usuarios empleados
Crear Donación
Crear Cuenta
Sanders
Empleado
Donante

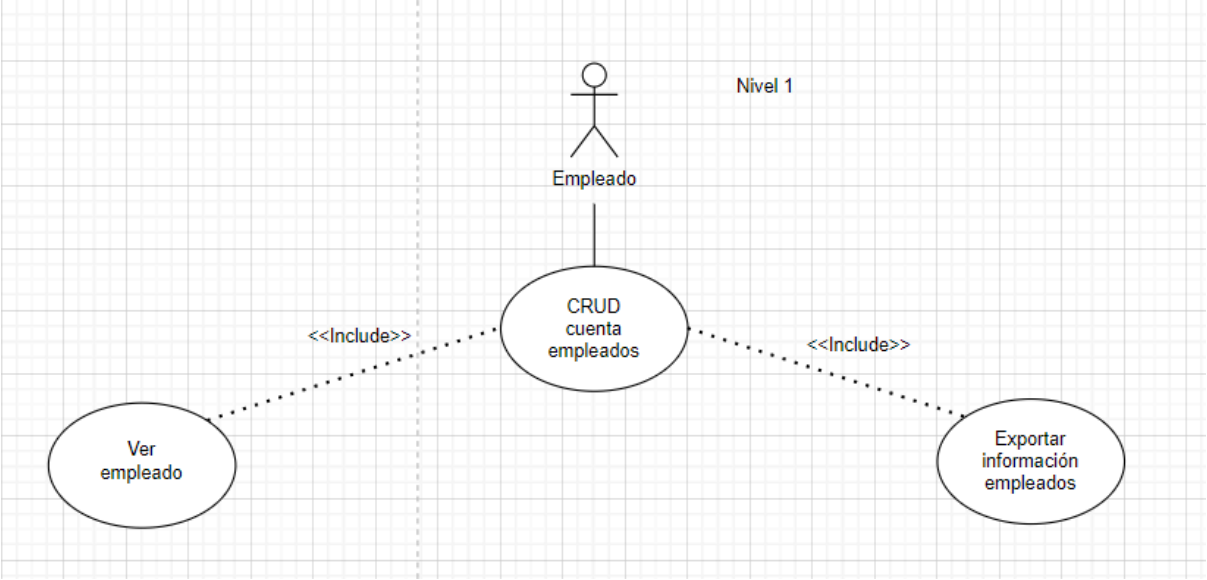

Nivel 1
Sanders
CRUD
cuenta
empleados
<<Include>>
<<Include>>
<<Include>>
<<Include>>
<<Include>>
Crear
cuenta
empleado
Ver
empleado
Exportar
información
empleados
Editar
cuentas
empleados
Borrar
cuenta
empleado

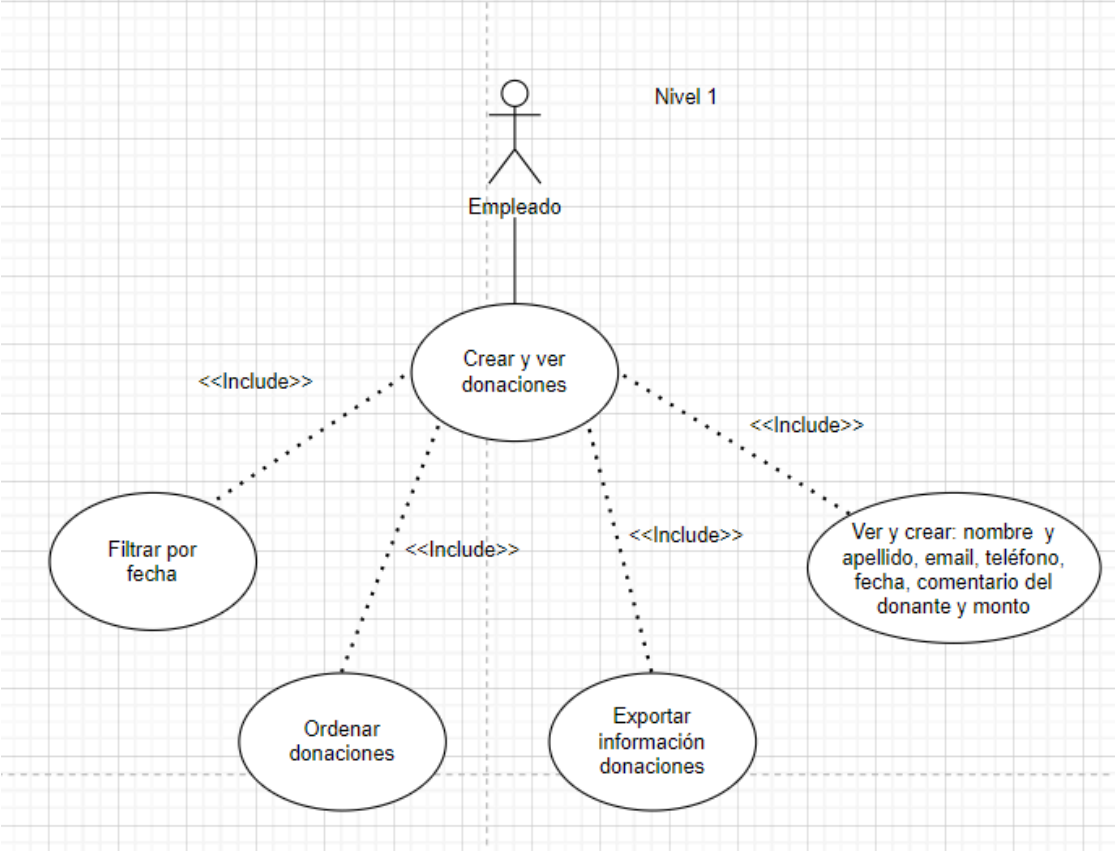

Nivel 1
Sanders
Crear y ver donaciones
<<Include>>
<<Include>>
<<Include>>
<<Include>>
Filtrar por fecha
Ordenar donaciones
Exportar información donaciones
Ver y crear: nombre y apellido, email, teléfono, fecha, comentario del donante y monto

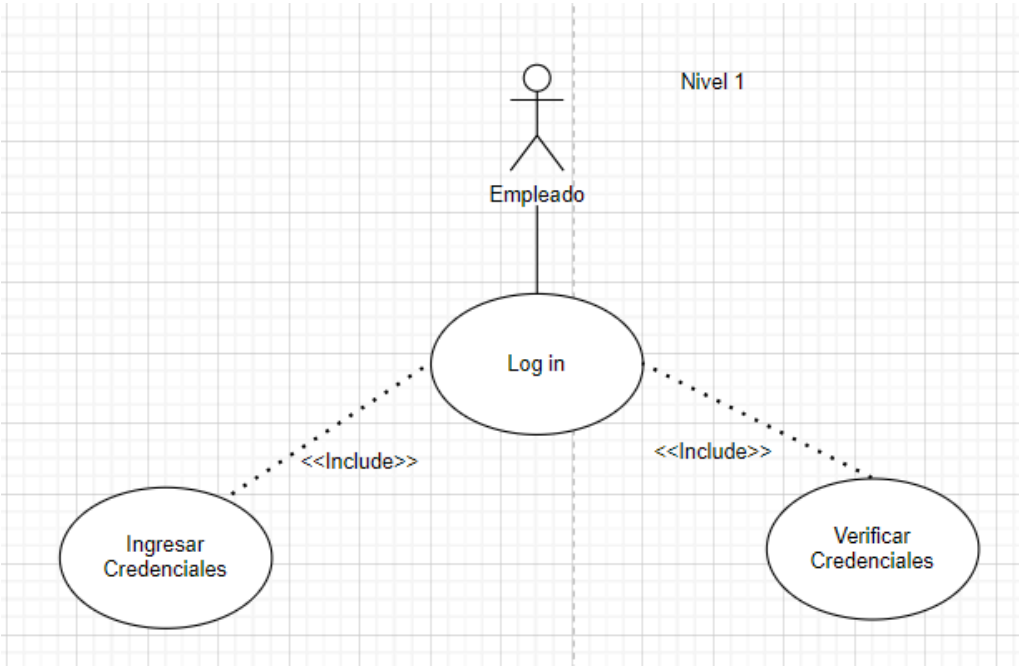

Nivel 1
Sanders
Log in
<<Include>>
<<Include>>
Ingresar Credenciales
Verificar Credenciales

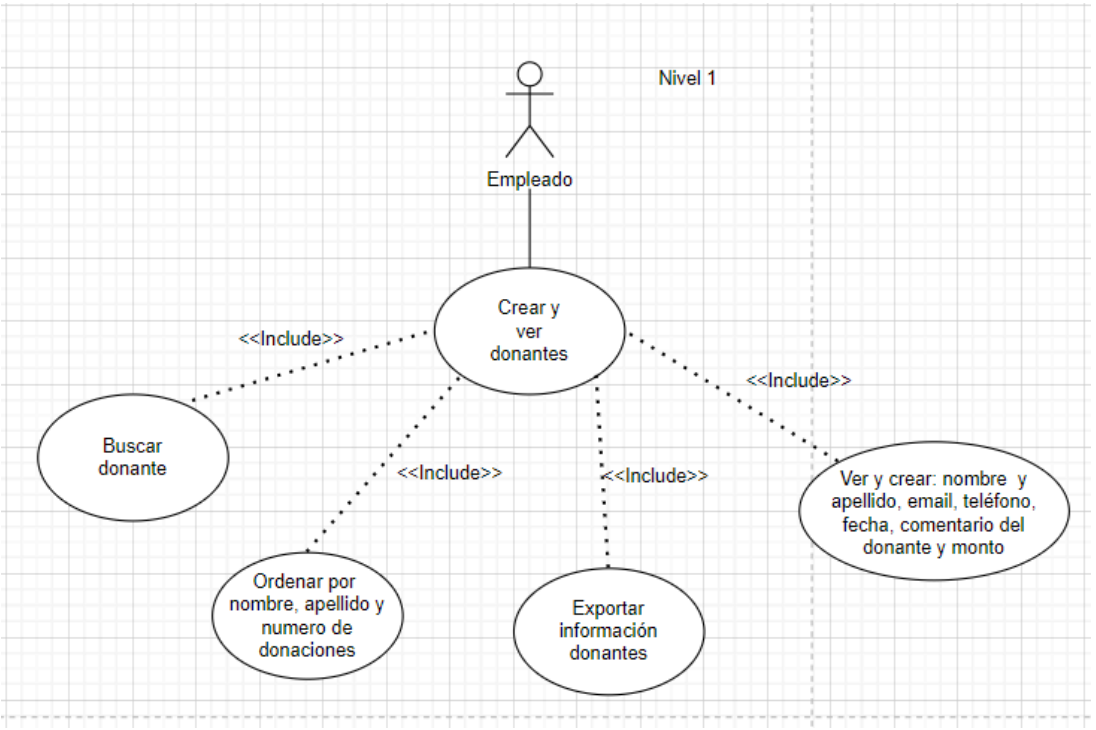

Nivel 1
Sanders
Crear y ver donantes
<<Include>>
<<Include>>
<<Include>>
<<Include>>
Buscar donante
Ordenar por nombre, apellido y numero de donaciones
Exportar información donantes
ver y crear: nombre y apellido, email, teléfono, fecha, comentario del donante y monto

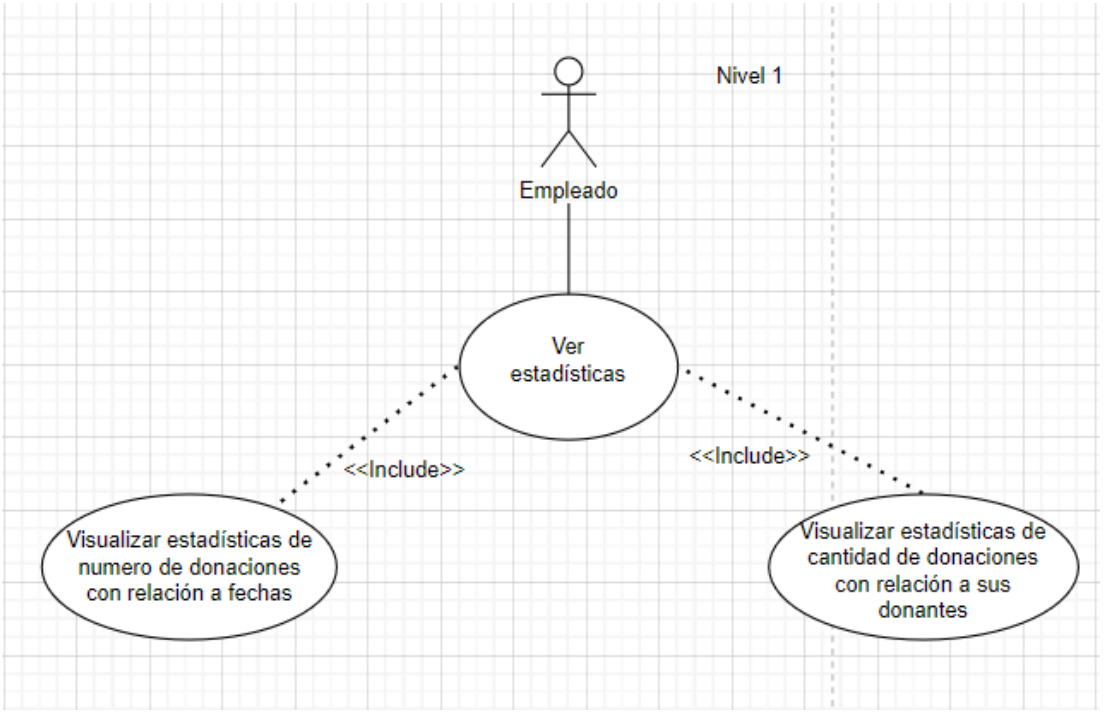

Nivel 1
Sanders
Ver estadísticas
<<Include>>
<<Include>>
Visualizar estadísticas de numero de donaciones con relación a fechas
Visualizar estadísticas de cantidad de donaciones con relación a sus donantes

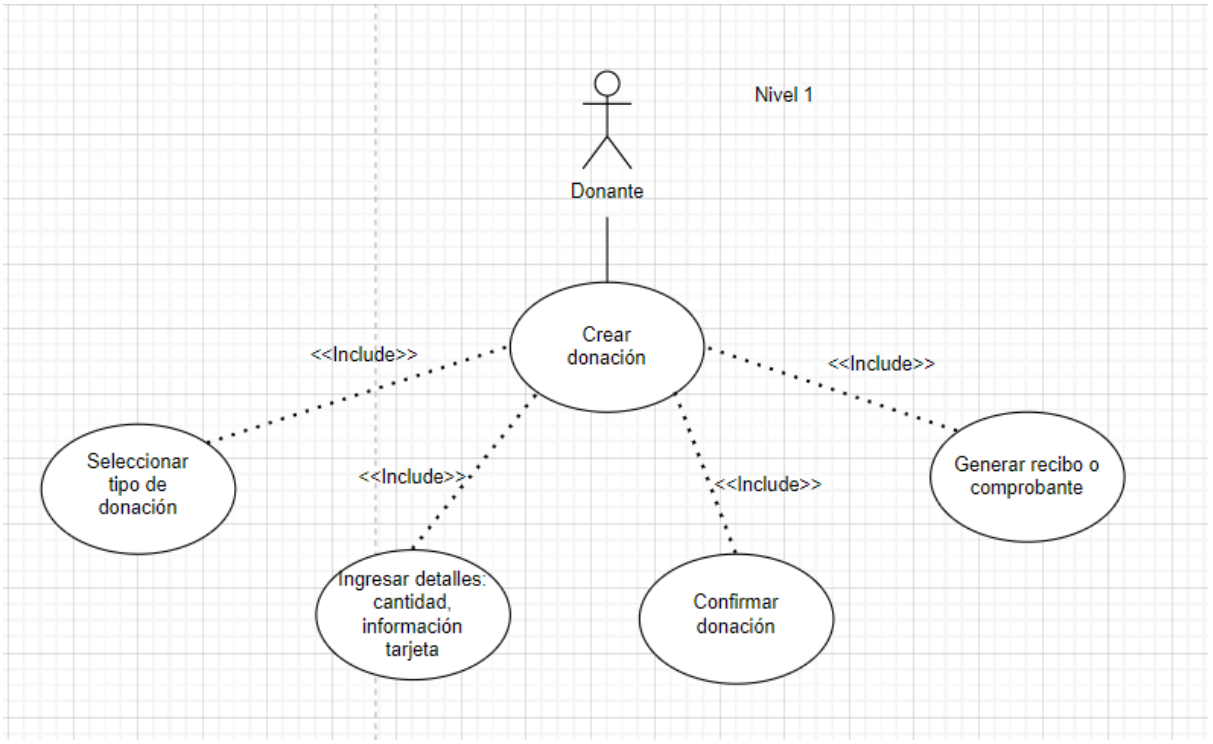

Nivel 1
Empleado
CRUD cuenta empleados
<<Include>>
<<Include>>
Ver empleado
Exportar información empleados

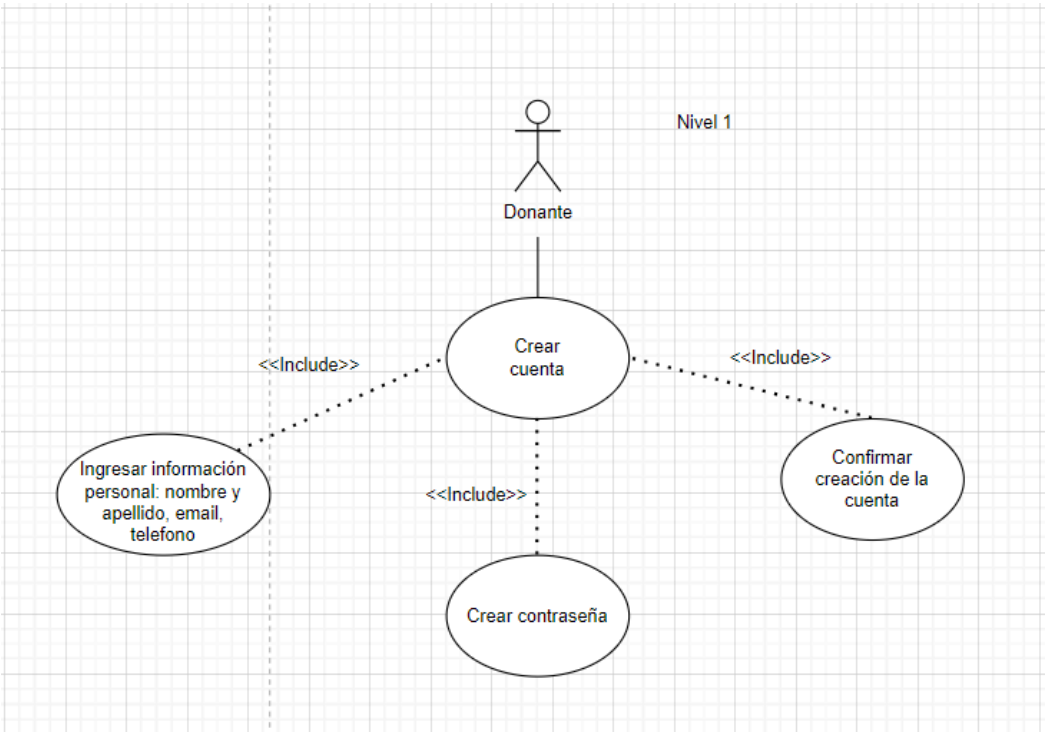

Nivel 1
Empleado
Crear y ver donaciones
<<Include>>
<<Include>>
<<Include>>
<<Include>>
Filtrar por fecha
Ver y crear: nombre y apellido, email, teléfono, fecha, comentario del donante y monto
Ordenar donaciones
Exportar información donaciones


Nivel 1
Empleado
Log in
<<Include>>
<<Include>>
Ingresar Credenciales
Verificar Credenciales

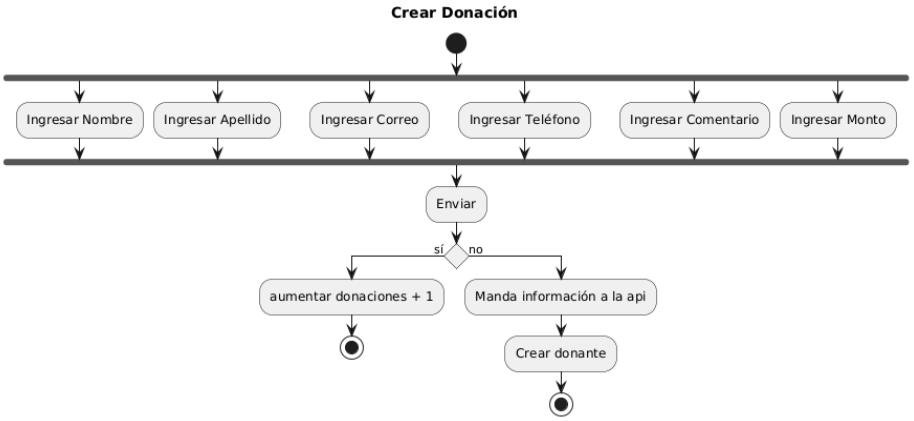

Nivel 1
Empleado
Crear y ver donantes
<<Include>>
<<Include>>
<<Include>>
<<Include>>
Buscar donante
Ordenar por nombre, apellido y numero de donaciones
Exportar información donantes
Ver y crear: nombre y apellido, email, teléfono, fecha, comentario del donante y monto

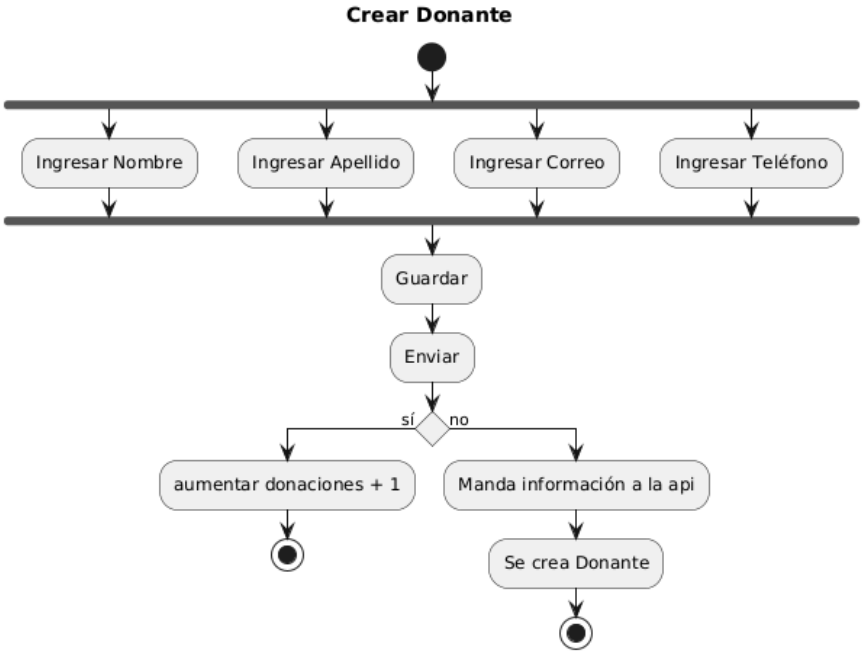

Nivel 1
Empleado
Ver estadísticas
<<Include>>
<<Include>>
Visualizar estadísticas de numero de donaciones con relación a fechas
Visualizar estadísticas de cantidad de donaciones con relación a sus donantes

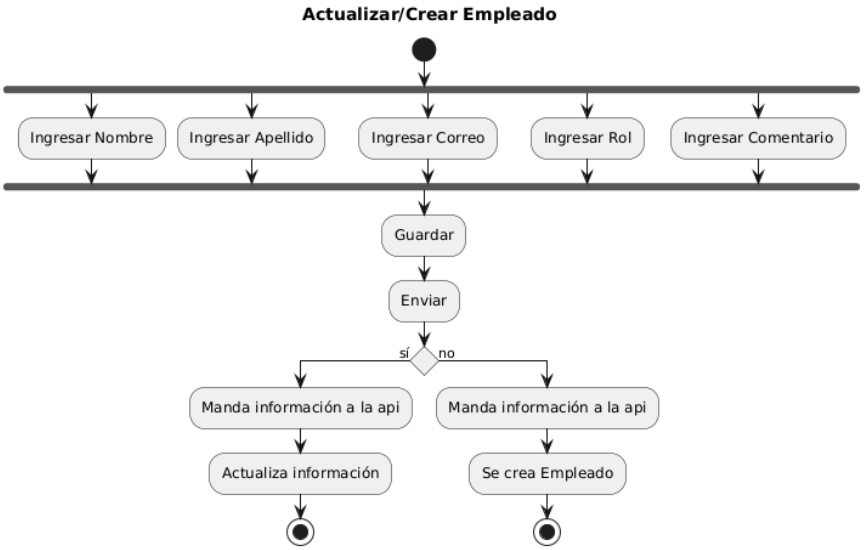

Nivel 1
Donante
Crear donación
<<Include>>
<<Include>>
<<Include>>
<<Include>>
Seleccionar tipo de donación
Ingresar detalles: cantidad, información tarjeta
Confirmar donación
Generar recibo o comprobante



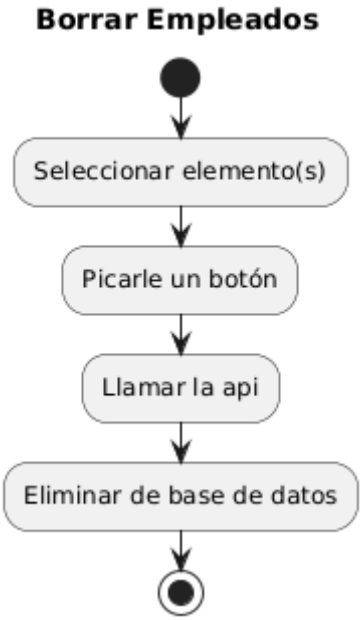
2. Diagrama de Actividades:

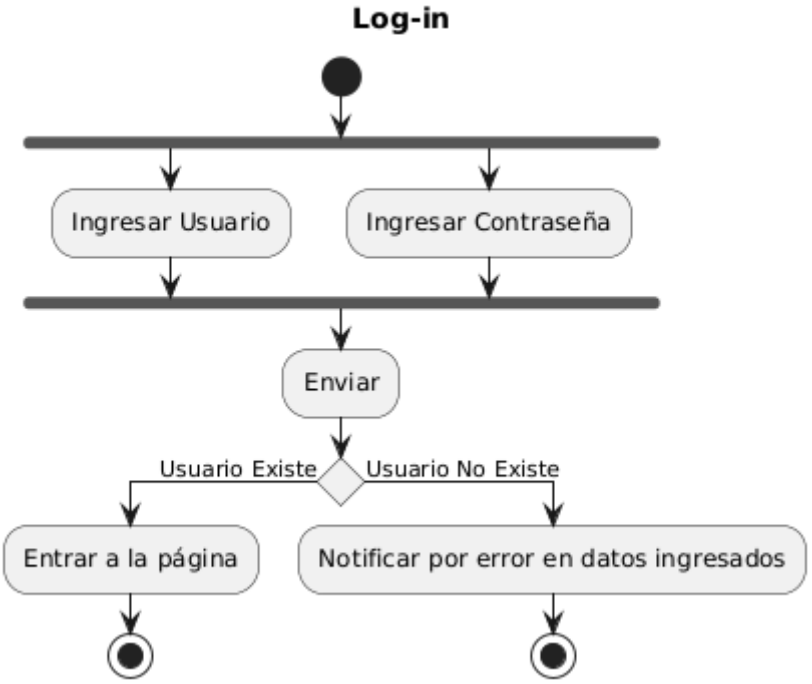



Crear Donante
Ingresar Nombre
Ingresar Apellido
Ingresar Correo
Ingresar Teléfono
Guardar
Enviar
sí
no
aumentar donaciones + 1
Manda información a la api
Se crea Donante

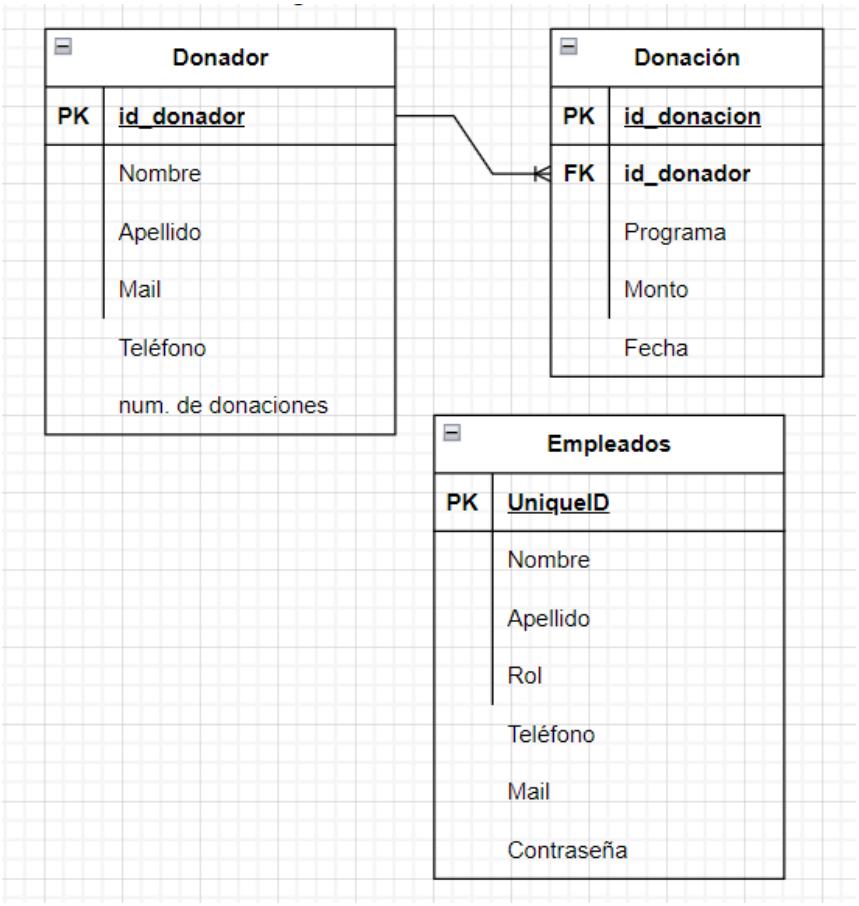

Actualizar/Crear Empleado
Ingresar Nombre
Ingresar Apellido
Ingresar Correo
Ingresar Rol
Ingresar Comentario
Guardar
Enviar
sí
no
Manda información a la api
Actualiza información
Manda información a la api
Se crea Empleado


Ver información
Seleccionar información a consultar
Llamar a la api
Desplegar información a la aplicación

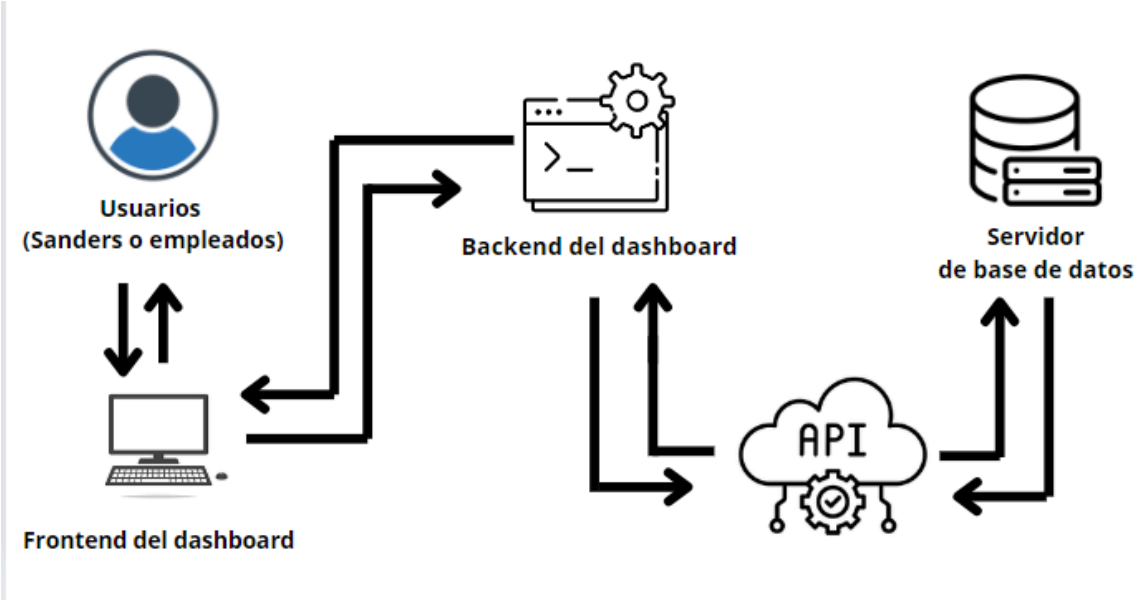




- **Diagramas Estructurales:**
    1. **Diagrama de Entidad Relación:**



2. **Diagrama de Arquitectura:**



Pruebas:

- **Estrategia de pruebas:**

Alpha Testing:
Se realizarán pruebas una vez que el sistema esté completo para identificar posibles errores antes de la entrega final.

Verificar que todas las funciones del sistema trabajen como se espera. Por ejemplo, que las operaciones de CRUD (Crear, Leer, Actualizar, Eliminar) funcionen correctamente en el dashboard.

Unit Testing:
Se probarán individualmente los componentes del dashboard para asegurar que funcionen de manera correcta de forma independiente. Estas pruebas serán documentadas para ayudar a encontrar y corregir errores antes de la entrega final.

Prueba cada componente del dashboard por separado. Por ejemplo, verifica que el componente de lista de donantes muestre correctamente los datos cuando se le pasen props.

Integration Testing:
Se comprobará si los componentes del dashboard logran trabajar correctamente juntos, se revisará la interacción entre los diferentes módulos del sistema para asegurar que funcionen correctamente.

Probar que el módulo de entrada de datos (formulario de donaciones) se comunique correctamente con el módulo de almacenamiento (base de datos).

Usability Testing:
Se evaluará qué tan bien los usuarios (Sanders y empleados) pueden utilizar el dashboard para lograr visualizar toda la información sobre donantes y donaciones de manera eficiente y sencilla.

Comprobar que los usuarios pueden navegar por el dashboard de manera intuitiva. Realiza tareas específicas y mide el tiempo que les toma completarlas.

Estrategia de Evolución del Sistema:

El sistema propuesto está diseñado para ser flexible y escalable, lo que permitirá adaptarse a las necesidades futuras de la Fundación Sanders a medida que evolucionen sus programas y alcance. La estrategia de evolución del sistema contempla las siguientes fases:

a. **Escalabilidad a otros programas de la fundación:**

A medida que la fundación expanda sus actividades, el sistema podrá integrarse con otros programas como "Salud reproductiva" y "Nutrición". Para esto, se agregaron nuevas funcionalidades para gestionar diferentes tipos de beneficiarios y datos específicos de cada programa. El diseño de la base de datos en MongoDB está estructurado para soportar el crecimiento de datos y la adición de nuevas colecciones relacionadas con otros programas.

b. **Integración con nuevas plataformas de pago:**

Con el tiempo, la Fundación podría integrar otras plataformas de pago además de PayPal, como transferencias bancarias directas. Estas integraciones permitirán un mejor seguimiento de las donaciones y diversificar las opciones de pago para los donantes. El sistema se construirá con APIs modulares para facilitar futuras integraciones.

c. **Mejoras en la seguridad:**

Con el incremento en la cantidad de donantes y datos almacenados, se implementarán actualizaciones continuas de seguridad, incluyendo la adopción de nuevas prácticas de encriptación de datos y mejores prácticas de autenticación para proteger tanto los datos de los donantes como la infraestructura del sistema.

d. **Optimización del rendimiento:**

A medida que aumente el número de usuarios, la infraestructura subyacente se ajustará para mantener un alto rendimiento. Esto incluye optimizar consultas a la base de datos, balancear la carga del servidor y mejorar la latencia en la interacción con la interfaz de usuario. El sistema se diseñará para poder migrar a una arquitectura basada en microservicios en caso de ser necesario.

e. **Personalización para los usuarios:**

En el futuro, los roles del sistema podrán expandirse para incluir nuevas categorías de usuarios, como coordinadores de programas o voluntarios. Estas nuevas funcionalidades permitirán una mayor granularidad en los permisos de acceso y en la visualización de datos relevantes para cada grupo de usuarios.

Apéndice:

Tecnologías empleadas:

- **React-Admin:** Framework utilizado para la construcción de la interfaz de usuario.
- **MongoDB:** Base de datos NoSQL que permite flexibilidad en la estructura de datos y alta escalabilidad.

- **Node.js:** Backend basado en JavaScript para manejar las peticiones y gestionar la lógica del servidor.
- **HTTPS:** Protocolo de seguridad para garantizar que las comunicaciones entre cliente y servidor estén cifradas.

Glosario

- **CRM (Customer Relationship Management):** Sistema de gestión de relaciones con los clientes que permite a las organizaciones gestionar interacciones y datos con sus usuarios.

- **MongoDB:** Base de datos NoSQL que almacena datos en formato JSON, lo que permite una mayor flexibilidad en el diseño de las estructuras de datos.

- **React-Admin:** Framework basado en React.js que facilita la creación de interfaces de administración, especialmente útil para construir dashboards y sistemas de gestión de datos.

- **API (Application Programming Interface):** Conjunto de funciones y procedimientos que permiten la interacción entre diferentes sistemas de software.

- **HTTPS (HyperText Transfer Protocol Secure):** Protocolo que asegura la comunicación a través de la red mediante el cifrado de datos, protegiendo la integridad y confidencialidad de la información.

- **Escalabilidad:** Capacidad del sistema de manejar un aumento en la carga de trabajo mediante la adición de recursos, como más datos o usuarios, sin comprometer el rendimiento.

- **Interfaz de usuario (UI):** Parte del sistema con la que interactúan directamente los usuarios, incluyendo todos los elementos gráficos y botones que permiten a los usuarios realizar acciones.