Lemon Sensors

Sistema de adquisición de datos sobre diferentes sensores eléctricos

Documento de arquitectura de software

Revisión: 1.2

Noviembre 2020

Histórico de Revisiones

Fecha	Revisión	Descripción	Autor
03-11-2020	1.0	Inicio del documento	José Luis Cuevas
04-11-2020	1.1	Revisión general del documento	Luis Fierro
04-11-2020	1.2	Se agregaron definiciones, conceptos, ejemplos	José Luis Cuevas

Tabla de Contenidos

Introducción

1. Propósito

El presente documento expone la arquitectura de software del sistema Lemon Sensors a través de las diferentes vistas propuestas por Kruchten (1999) que ilustra un aspecto particular del software desarrollado. El objetivo es brindar una visión global y comprensible del sistema.

2. Alcance

El documento presenta el modelo de vistas arquitectónicas 4+1 propuesto y diseñado por Philippe Kruchten:

- Vista lógica
- Vista de desarrollo
- Vista de proceso
- Vista física
- Vista de escenarios

Cada una de las vistas es representadas por algunos de los siguientes diagramas:

- Diagrama de casos de uso
- Diagrama de secuencia
- Diagrama de clases
- Diagrama de estado
- Diagrama de componentes
- Diagrama de distribución
- Diagrama de actividades

Representación de la arquitectura

El modelo propuesto por Kruchten para representar la arquitectura utiliza el siguiente conjunto de vistas:

- **Vista lógica:** Apoya principalmente los requisitos no funcionales –lo que el sistema debe brindar en términos de servicios a sus usuarios. El sistema se descompone en una serie de abstracciones clave, tomadas (principalmente) del dominio del problema en la forma de objetos o clases de objetos. Aquí se aplican los principios de abstracción, encapsulamiento y herencia. Esta descomposición no sólo se hace para potenciar el análisis funcional, sino también sirve para identificar mecanismos y elementos de diseño comunes a diversas partes del sistema.
- **Vista de desarrollo:** La vista de desarrollo se centra en la organización real de los módulos de software en el ambiente de desarrollo del software. El software se empaqueta en partes pequeñas –bibliotecas de programas o subsistemas– que pueden ser desarrollados por uno o un grupo pequeño de desarrolladores. Los subsistemas se organizan en una jerarquía de capas, cada una de las cuales brinda una interfaz estrecha y bien definida hacia las capas superiores.
- **Vista de proceso:** La arquitectura de procesos toma en cuenta algunos requisitos no funcionales tales como la performance y la disponibilidad. Se enfoca en asuntos de concurrencia y distribución, integridad del sistema, de tolerancia a fallas. La vista de procesos también especifica en cuál hilo de control se ejecuta efectivamente una operación de una clase identificada en la vista lógica.
- **Vista física:** La arquitectura física toma en cuenta primeramente los requisitos no funcionales del sistema tales como la disponibilidad, confiabilidad (tolerancia a fallas), performance (throughput), y escalabilidad. El software se ejecuta sobre una red de computadores o nodos de procesamiento (o tan solo nodos). Los variados elementos identificados –redes, procesos, tareas y objetos– requieren ser mapeados sobre los variados nodos.
- **Vista de escenarios:** Esta vista es redundante con las otras (y por lo tanto “+1”), pero sirve a dos propósitos principales: como una guía para descubrir elementos arquitectónicos durante el diseño de arquitectura tal como lo describiremos más adelante, como un rol de validación e ilustración después de completar el diseño de arquitectura, en el papel y como punto de partido de las pruebas de un prototipo de la arquitectura.

Objetivos y restricciones

El objetivo del sistema es recolectar las lecturas de los sensores y guardarlas en una base de datos para su posterior procesamiento y presentación. Sin embargo, presenta algunos requisitos esenciales que afectan el diseño arquitectónico:

- Enviar alertas de lecturas fuera de un rango de parámetros
- Conexión de múltiples sensores a un IoT Gateway.
- Facilidad para agregar sensores futuros.
- Disponibilidad de los servicios 24/7.
- Réplica de datos.

1. Requerimientos especiales

Seguridad - JSON Web Token
El sistema deberá autenticar a los usuarios por medio de un Json Web Token que tendrá un tiempo de expiración de 24 hrs. Cada vez que expire, se le pedirá al usuario reiniciar su sesión. Cada vez que el usuario inicie sesión se le entregará un nuevo token de autenticación.

Portabilidad - Progressive Web Application
La parte del cliente debería ser una aplicación web progresiva (PWA) para poder utilizarla en todas las plataformas con acceso a internet que soporten web. La aplicación web podrá ser instalada en dispositivos móviles como si de una aplicación nativa se tratase.

Interoperabilidad - Application Programming Interface
El sistema deberá tener una interfaz de programación de aplicaciones (API) para establecer conexión con el servidor y realizar peticiones HTTP. Los tipos de peticiones serán **GET**, **POST**, **PUT**, **DELETE**.

Disponibilidad - Hosting en Firebase
El sistema deberá ser alojado en un servicio de hosting web (en este caso Firebase) para poder ser accedido desde cualquier lugar y en cualquier momento.

Mantenibilidad - Componentes
El sistema deberá tener una arquitectura que permita mantener componentes reutilizables y fáciles de modificar.

Modelo de vistas arquitectónicas 4+1

1. Vista lógica

Un diagrama de clases muestra un conjunto de clases y sus relaciones lógicas: asociaciones, uso, composición, herencia y similares. Si es necesario definir el comportamiento interno de un objeto, esto se realiza con un diagrama de transición de estados o diagrama de estados.

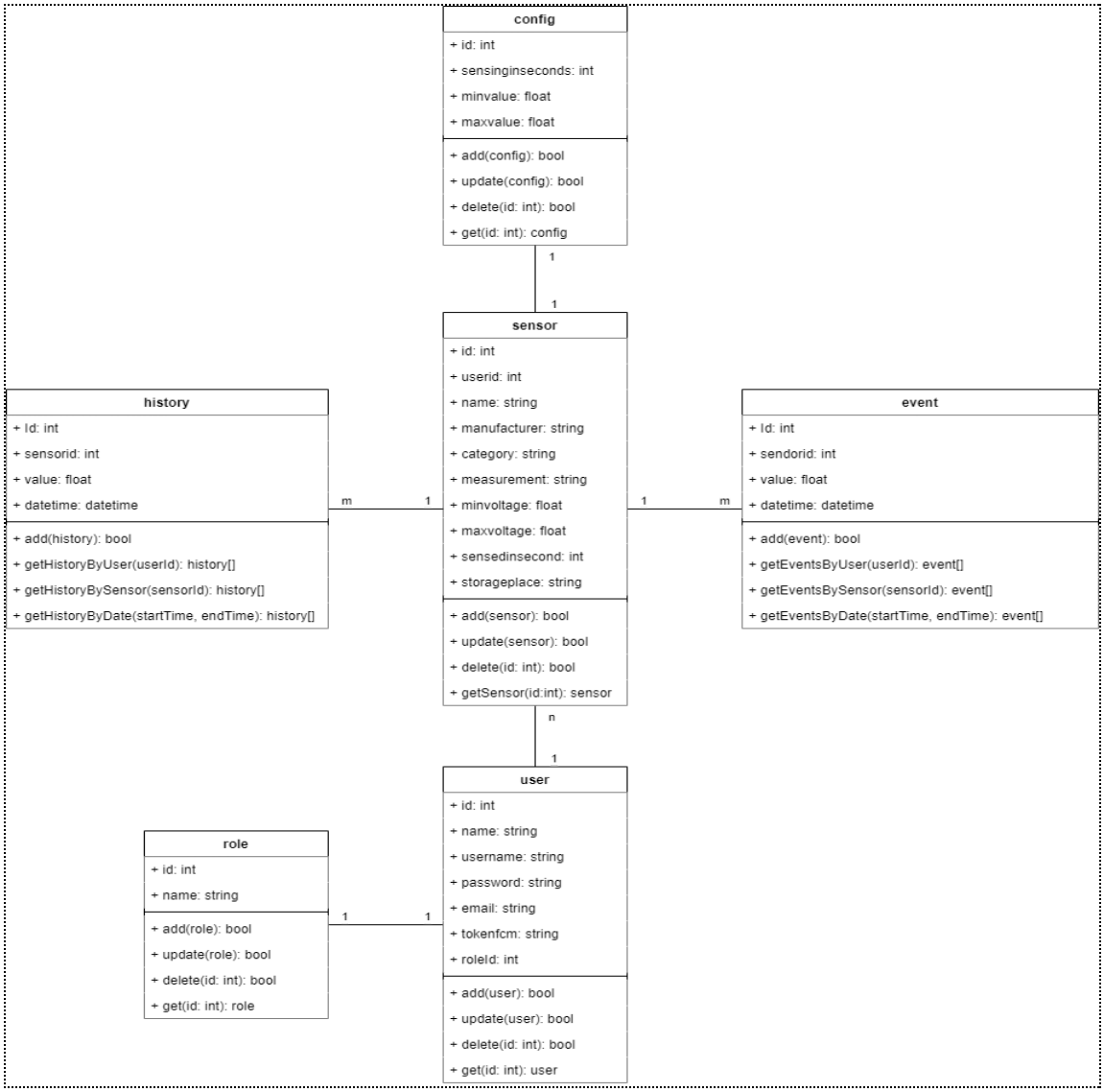
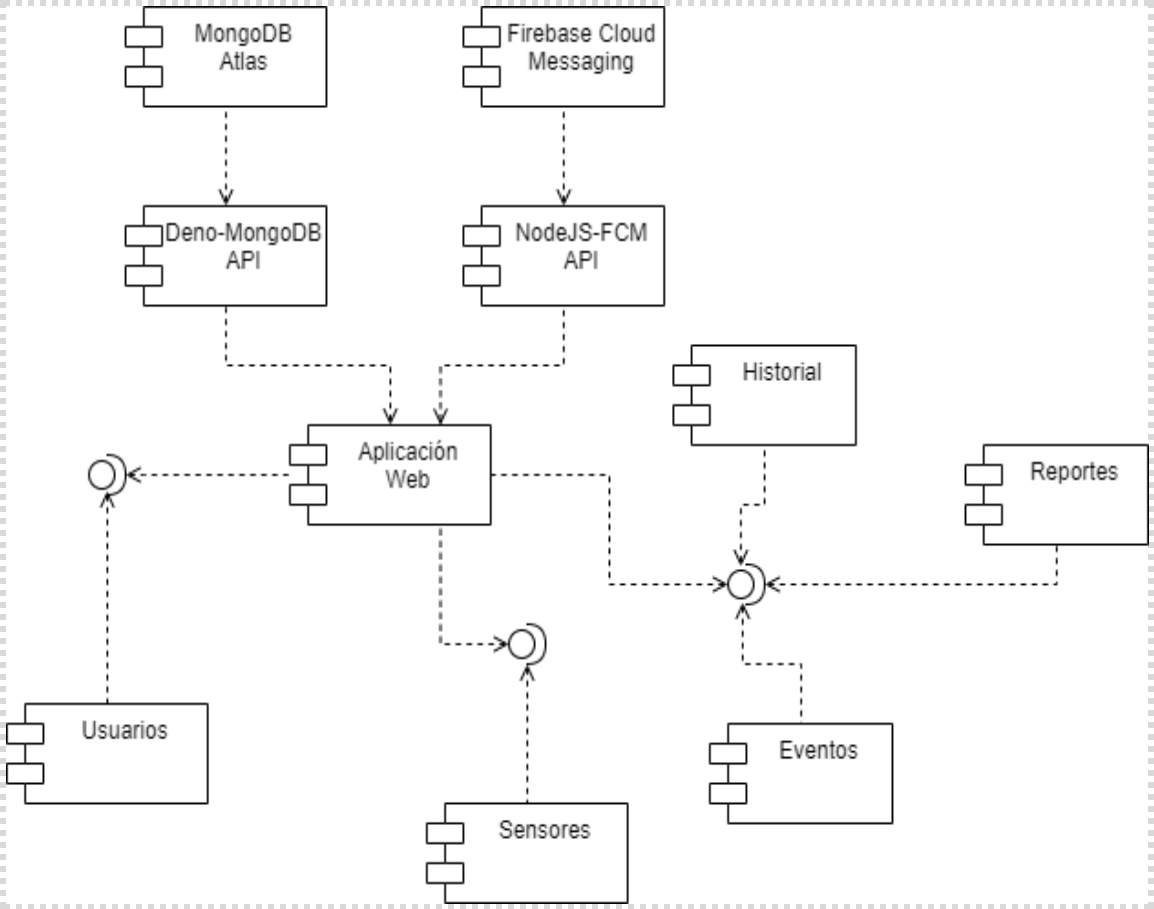


Ilustración 1. Diagrama de clases

2. Vista de desarrollo

Un diagrama de componentes proporciona una vista de alto nivel de los componentes dentro de un sistema. Los componentes pueden ser un componente de *software*, como una base de datos o una interfaz de usuario; o un componente de *hardware* como un circuito, microchip o dispositivo; o una unidad de negocio como un proveedor, nómina o envío.



3. Vista de proceso

Los diagramas de secuencia se centran específicamente en *líneas de vida* o en los procesos y objetos que coexisten simultáneamente, y los mensajes intercambiados entre ellos para ejecutar una función antes de que la línea de vida termine.

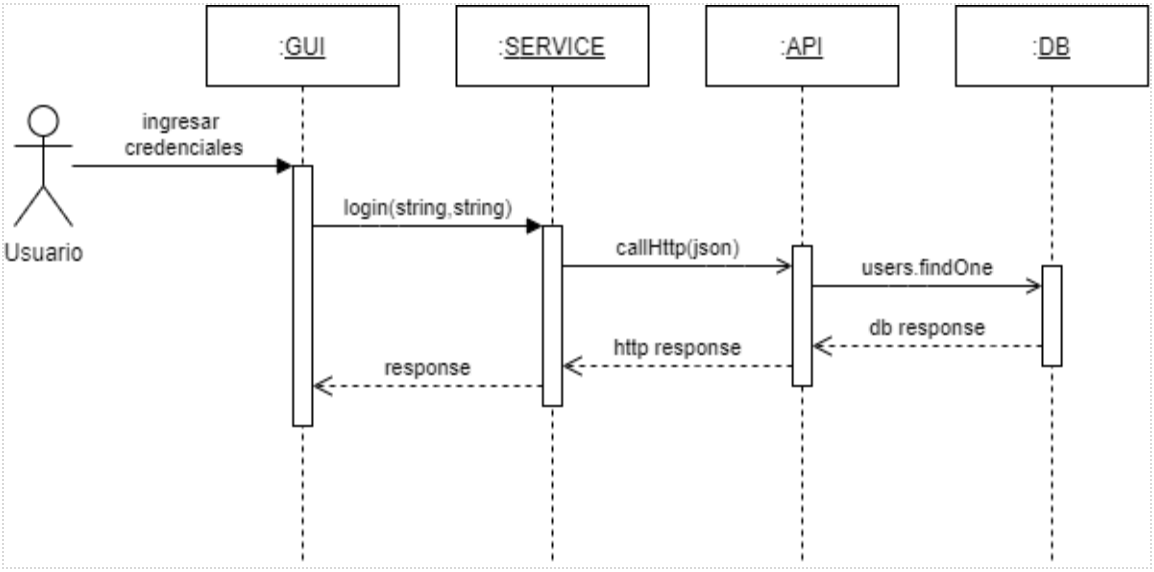


Ilustración 3. Diagrama secuencia Inicio de sesión

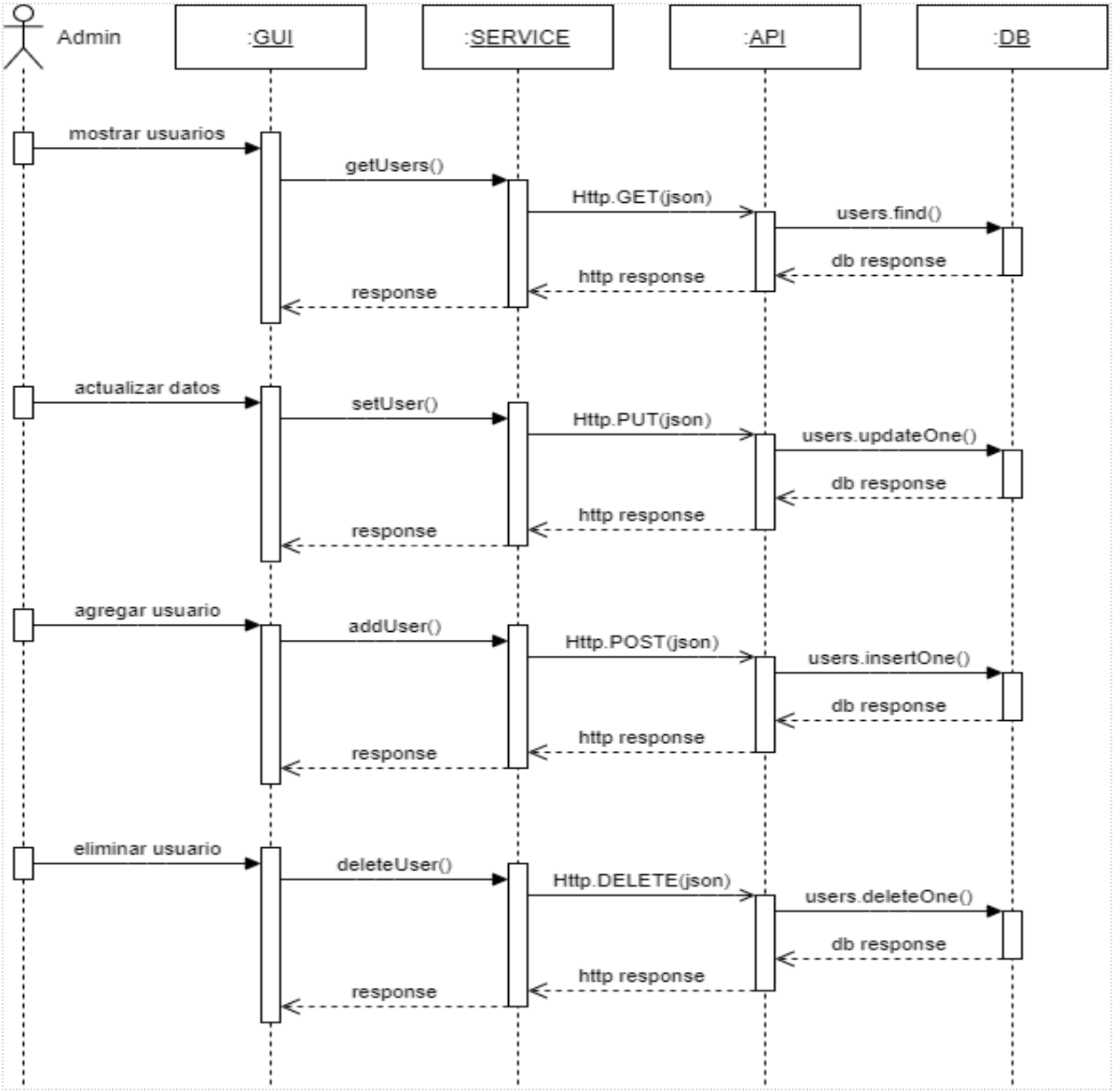


Ilustración 4. Diagrama secuencia Inicio de sesión

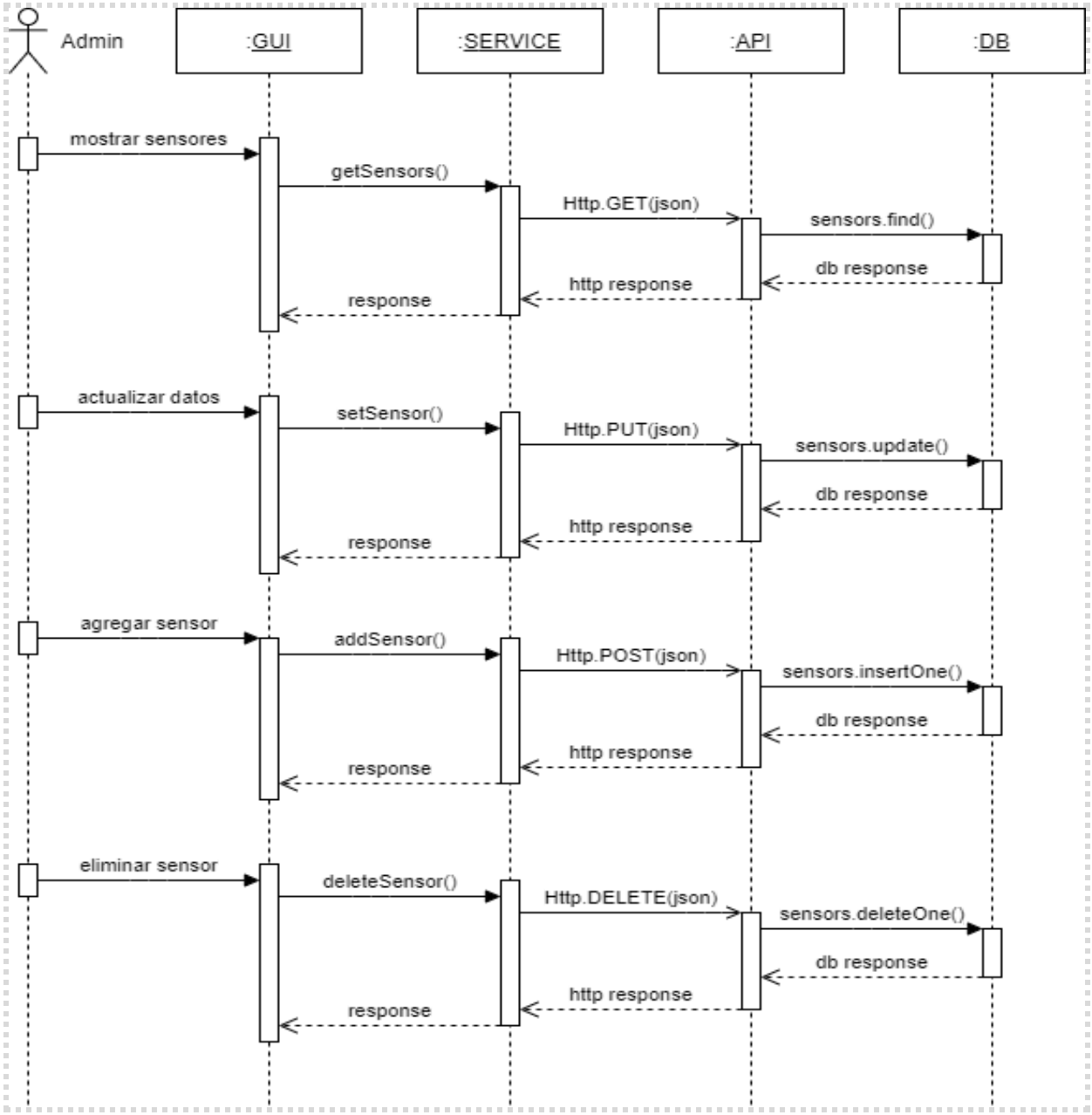


Ilustración 5. Diagrama secuencia Gestión de Sensores

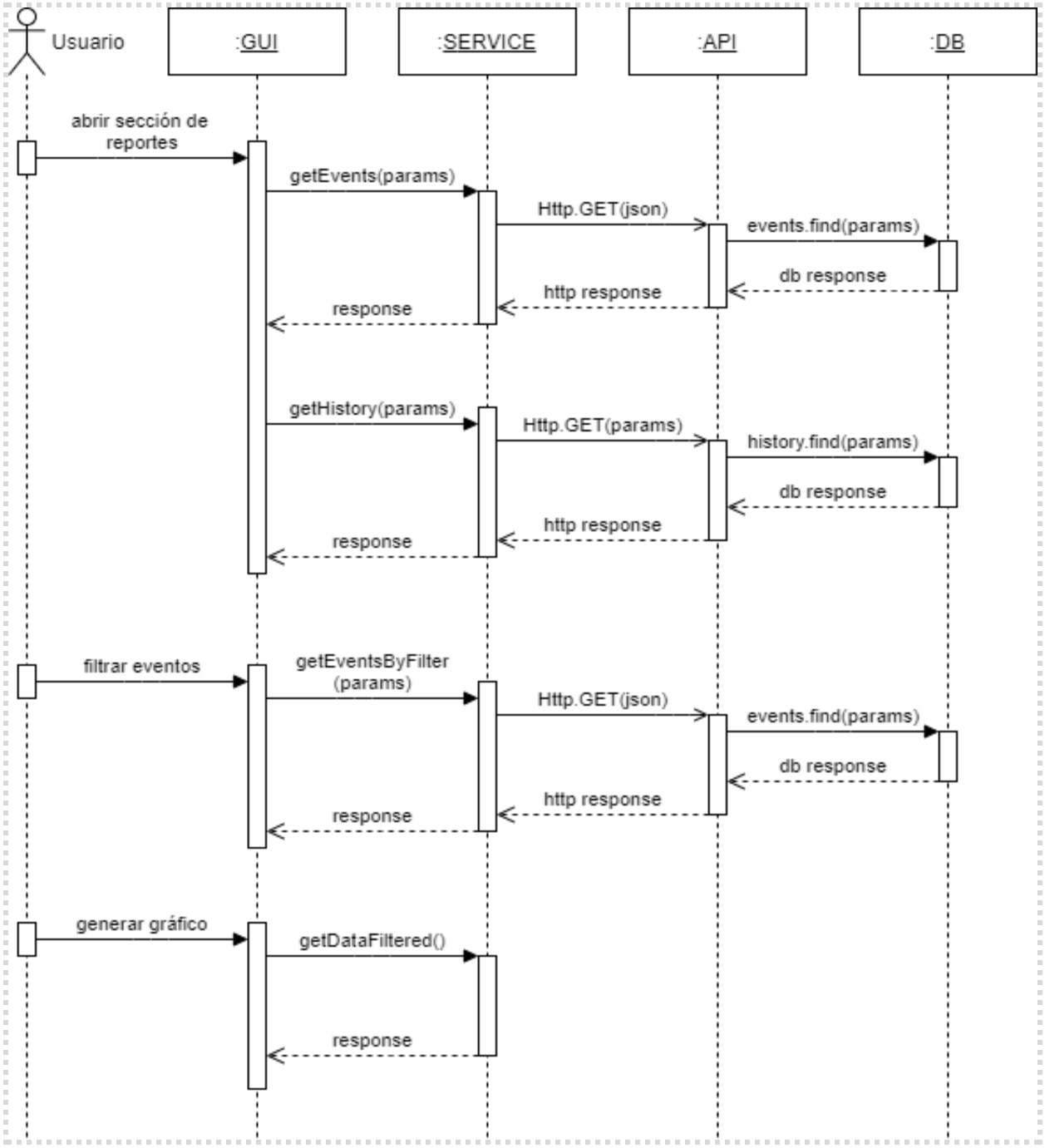


Ilustración 6. Diagrama secuencia Generar reportes

Los diagramas de estado describen el comportamiento de los sistemas describiendo todos los estados posibles en los que puede entrar un objeto en particular y la manera que cambia el estado del objeto a través de las diversas acciones o transiciones que toma para pasar de un estado a otro.

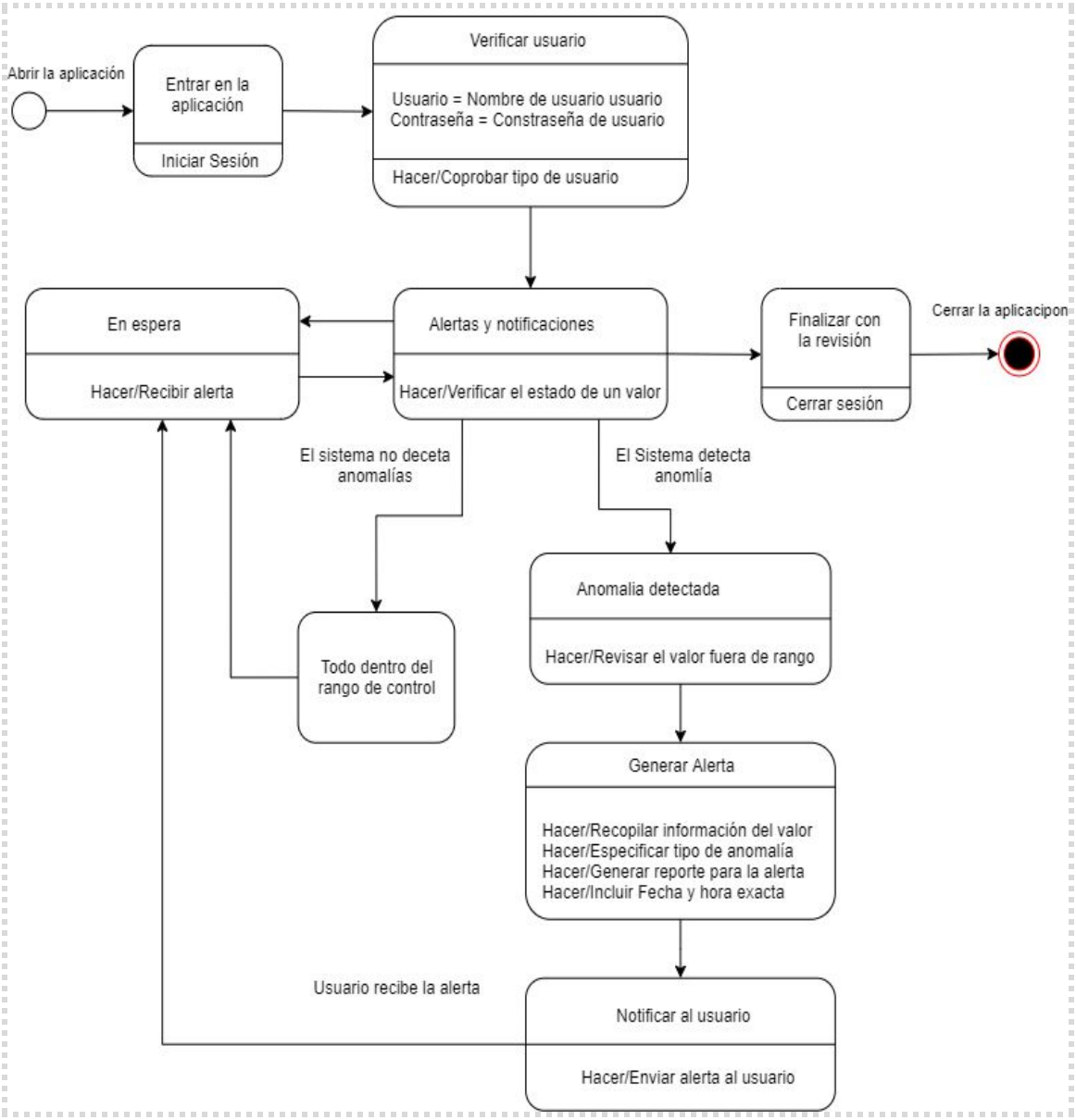


Ilustración 6. Diagrama de estado de Alertas.

4. Vista física

Un Diagrama de distribución dibuja pares relacionados de variables para presentar un patrón de relación o de correlación. Cada conjunto de datos representa un factor diferente que puede ser cuantificado. Un conjunto de datos es dibujado en un eje horizontal (eje x) y el otro conjunto de datos se dibuja en el eje vertical (eje y). El resultado es un número de puntos que pueden ser analizados para determinar si existe una relación significativa (también conocida como “correlación”) entre los dos conjuntos de datos. esto nos ayuda para comprender mejor el programa o sistema que en este caso estamos diseñando como a continuación se detalla:

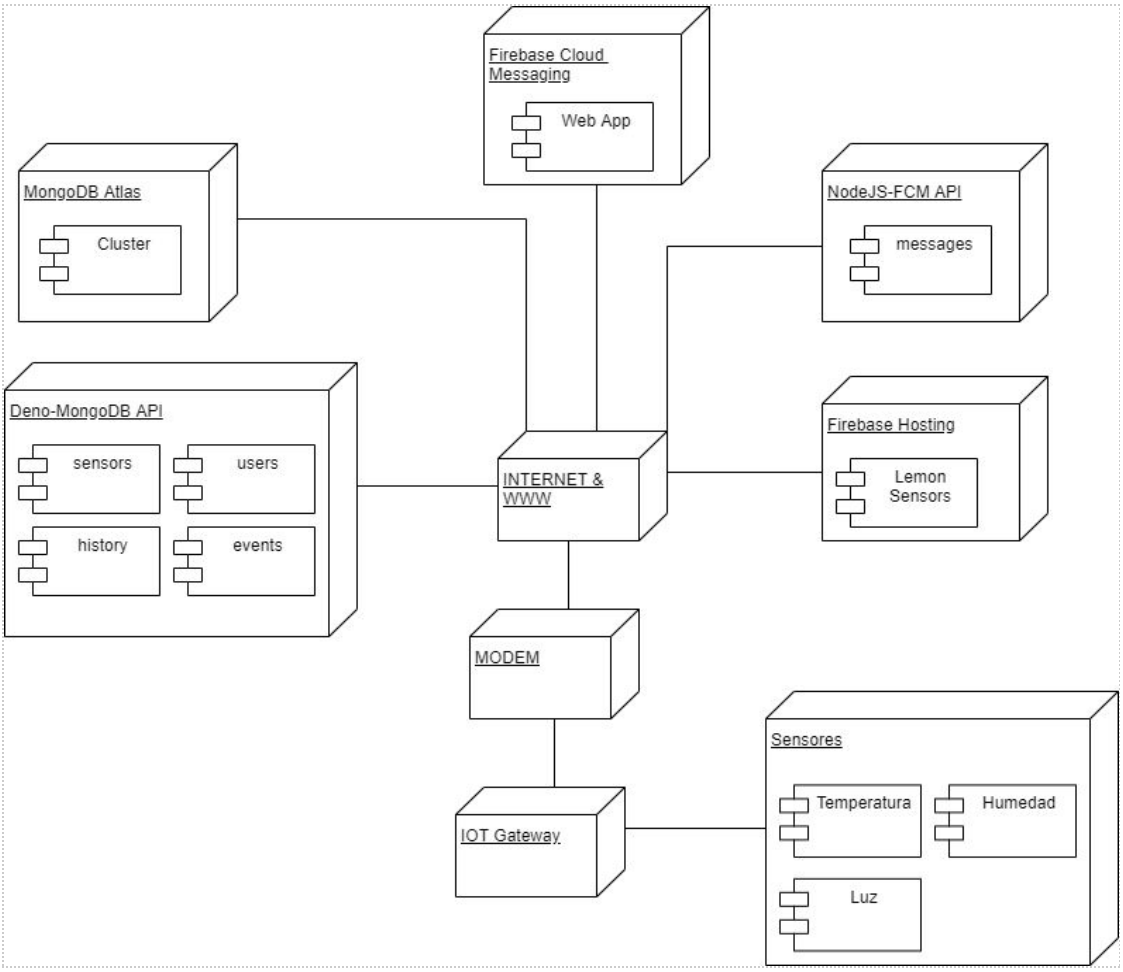


Ilustración 7. Diagrama de distribución.

5. Vista de escenarios +1

Los diagramas de caso de uso muestran la comunicación y el comportamiento entre los usuarios involucrados mostrando la relación que tiene cada uno con los casos de uso como se muestra a continuación.

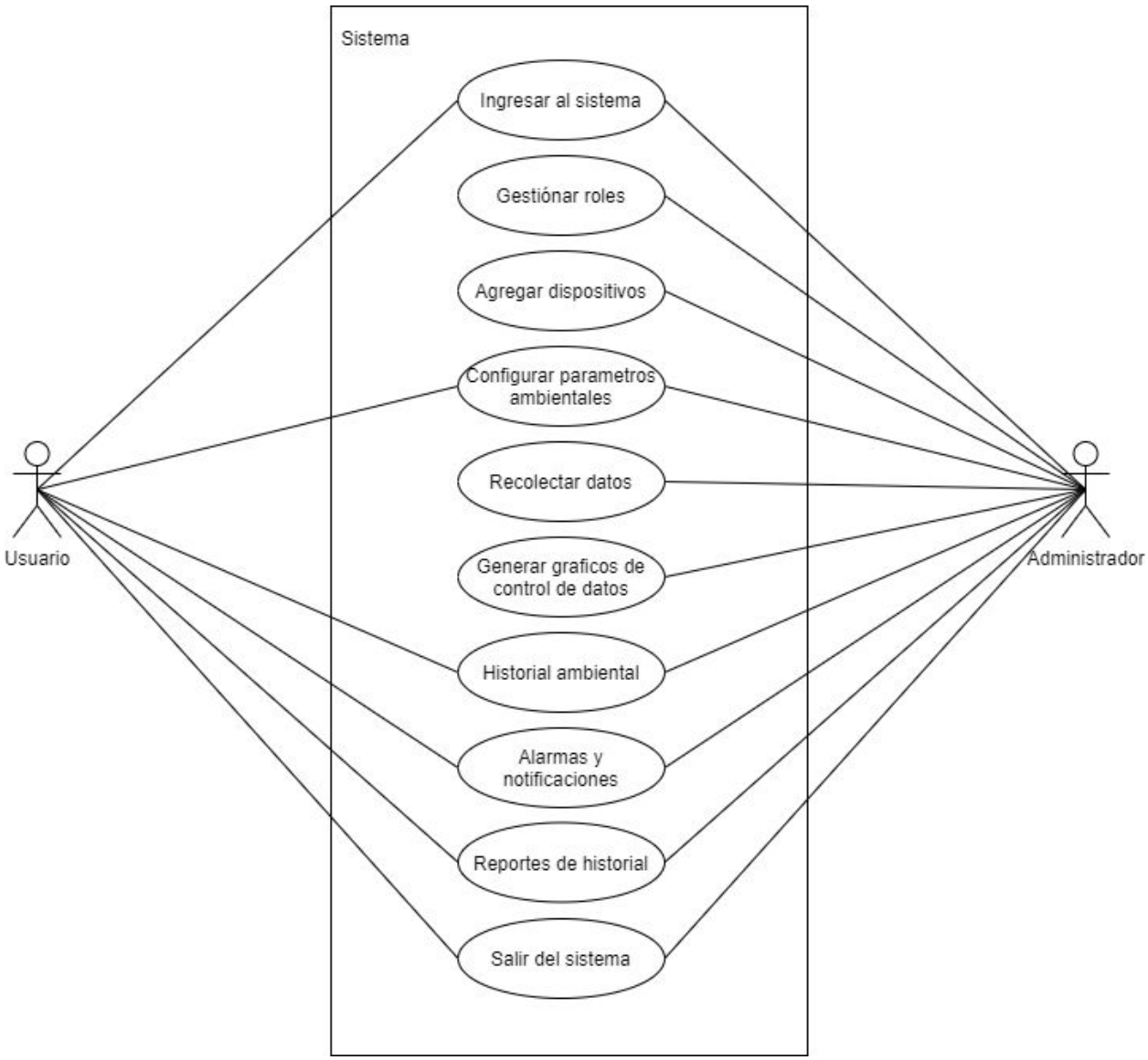


Ilustración 8. Diagrama de casos de uso (General).

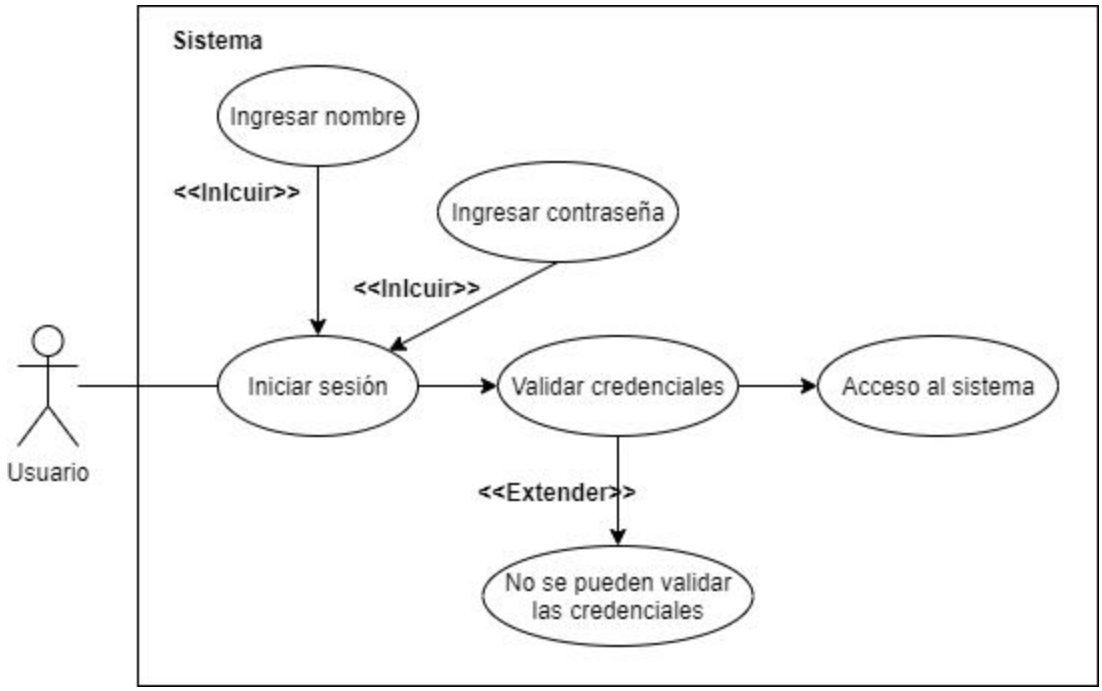


Ilustración 9. Diagrama de caso de uso, Acceso al sistema.

Conclusiones

José Luis Cuevas

El modelo 4+1 permite presentar el sistema de acuerdo al tipo de usuario que lo requiera. Así se puede dar una idea de las funcionalidades o estructura sin tener que entender componentes que no le infieren.

Luis Eduardo Fierro

La vistas a las que hace referencia este documento nos sirven para modelar y ejemplificar la estructura del sistema, logrando con ello dar una mejor representación gráfica de dicha estructura y con eso, lograr entender cómo funcionan y se relaciona ciertos aspectos del sistema.

Jonathan Navarro

Este documento nos ayudará a Entender y comprender cómo se estructura el sistema,logrando con esto una mayor eficiencia en el proyecto y que se entienda lo que se pretende realizar con el.