UNMDP - Facultad de Ingeniería
Departamento de Informática

Poseidón CMMS: Sistema para la gestión del mantenimiento de dispositivos IoT

Alumnos:

- Lima, Mauricio Alberto <limamauricio303@gmail.com>
- Navarro, Fernando Daniel <fernavarro2607@gmail.com>

Director: Hinojal, Hernán

Co-director: Finocchietto, Mariano

Proyecto final para optar al grado de Ingeniero en Informática

Mar del Plata, mayo de 2022

UNMDP - Facultad de Ingeniería
Departamento de Informática

Poseidón CMMS: Sistema para la gestión del mantenimiento de dispositivos IoT

Alumnos:

- Lima, Mauricio Alberto <limamauricio303@gmail.com>
- Navarro, Fernando Daniel <fernavarro2607@gmail.com>

Director: Hinojal, Hernán

Co-director: Finocchietto, Mariano

Proyecto final para optar al grado de Ingeniero en Informática

Mar del Plata, mayo de 2022

Agradecimientos

A nuestras familias y amigos por el apoyo.

Al cuerpo docente por el conocimiento transmitido.

Índice

Resumen

En el presente informe se describe el proyecto Poseidón CMMS llevado a cabo por los alumnos Fernando Daniel Navarro y Mauricio Alberto Lima. El principal objetivo del proyecto consiste en analizar el problema y diseñar una solución para la gestión del mantenimiento de dispositivos IoT instalados sobre maquinaria agrícola distribuida geográficamente. Además, se busca resolver las problemáticas de gestión de reparaciones, seguimientos de stock de dispositivos, asignación de fallas a técnicos para su reparación y visualización de dashboards con métricas relacionadas al mantenimiento de dispositivos.

El proyecto se llevó a cabo de manera interdisciplinaria con un proyecto paralelo de la carrera de Ingeniería Industrial sobre la misma temática. El trabajo en conjunto fue una gran ventaja durante todas las etapas.

El documento incluye la descripción de todo el análisis, diseño e implementación realizado por los alumnos mediante la comunicación con la empresa demandante Ponce AgTech. Además, se incluyen las estimaciones de tiempos planificadas previamente y una comparación con la duración real del proyecto.

Capítulo 1: Introducción

Una de las principales actividades económicas desarrolladas en la Argentina es la agricultura. Para su realización se deben administrar eficientemente los recursos naturales para maximizar la rentabilidad y minimizar el impacto ambiental. En este contexto, surgen empresas como Ponce AgTech cuyo objetivo es asistir a los productores agrícolas para que hagan un eficiente uso de los recursos. Para poder proveer este servicio, empresas de esta índole hacen uso de dispositivos IoT (del inglés *Internet of Things*, internet de las cosas) para monitorear el rendimiento de maquinaría agrícola como son los equipos de riego.

Sin embargo, los dispositivos se encuentran a la intemperie, por lo que aunque se los protege mediante carcasas y sellados herméticos, es frecuente que dejen de funcionar. La detección temprana de estas fallas resulta vital para proveer un buen servicio a los productores agrícolas. La solución a las fallas siempre reside en enviar un técnico capacitado a la ubicación del dispositivo roto con el objetivo de reemplazarlo o repararlo.

La empresa demandante cuenta con una herramienta destinada a satisfacer esta necesidad. Es un conjunto de aplicaciones montadas sobre la plataforma AppSheet que permite una gestión manual de estas fallas. Sin embargo, esta herramienta presenta varias limitaciones en cuanto a la organización de la información y la flexibilidad a la hora de agregar funcionalidad.

Debido a estas limitaciones y la necesidad de resolver el problema de la gestión del mantenimiento en forma rápida y automatizada, surge la idea para el proyecto Poseidón CMMS. Uno de sus principales objetivos es satisfacer esta necesidad mediante un producto diseñado específicamente para el problema en cuestión.

Capítulo 2: Objetivos del proyecto

2.1 Objetivos

- **Obtener información y entender el dominio del problema, mediante el análisis de los flujos de trabajo pertinentes en la empresa demandante.**
  El primer paso para el diseño de una buena solución es familiarizarse con el problema, su contexto y las soluciones existentes en la empresa demandante.

- **Trabajar de manera conjunta con un alumno de Ingeniería Industrial mediante proyectos finales relacionados.**
  El proceso de análisis será realizado en conjunto con un estudiante de la carrera de Ingeniería Industrial que trabaja actualmente para la empresa demandante.

- **Diseñar una solución de gestión del mantenimiento que facilite el trabajo del personal operativo y técnico de la empresa que utilice el sistema.**
  Con el conocimiento obtenido se diseñará una plataforma que ofrezca una mejora a la operatoria de la empresa con respecto a sus procesos actuales relacionados al mantenimiento de dispositivos y su gestión.

- **Construir una solución que permita una eficiente gestión de los problemas que se presenten en los equipos**.
  Desarrollar una aplicación web que permita la visualización y administración tanto de los activos de la empresa que la utilice como de los problemas de cada uno.

- **Facilitar el almacenamiento de información sobre la ubicación de los dispositivos de la empresa.**

Tener un buen seguimiento de los dispositivos mediante la utilización del producto minimizará pérdidas y proveerá mejor información para la toma de decisiones de la empresa.

- **Integrar la detección automática de fallas en los dispositivos para minimizar su tiempo no operativo.**
  Mediante la integración, los usuarios del sistema serán notificados rápidamente ante la aparición de fallas en algún equipo. Así, se logrará enviar un técnico a repararlo lo antes posible. De este modo, mediante la utilización del producto se minimizarán los tiempos no operativos de los sensores instalados en los equipos de riego.

2.2 Alcance

El proyecto tiene como objetivo analizar el problema del mantenimiento de dispositivos electrónicos instalados sobre equipos de riego. A partir de este análisis se busca diseñar una solución que permita su monitoreo con el objetivo de detectar en forma rápida discontinuidades en la transmisión. No es objetivo del proyecto que el sistema permita observar la información de los sensores en tiempo real. Solo se debe monitorear si están transmitiendo correctamente. Adicionalmente, los sensores ya existen y están ubicados en los equipos correspondientes, por lo que sólo forma parte de este proyecto su monitoreo y gestión de reemplazo en caso de falla.

El proyecto tendrá los siguientes entregables:

- Requerimientos funcionales y no funcionales a cumplir.
- Casos de uso y diagramas relacionados detallando los flujos normales de evento y los casos alternativos.
- Diagramas de diseño ideados a partir de los requerimientos y las restricciones del dominio.
- El sistema de gestión del mantenimiento en sí, incluyendo sus componentes descritos en la documentación previamente mencionada.

- Una aplicación web que se ejecuta en un navegador. No es una aplicación nativa para celulares pero contará con una interfaz adaptable a pantallas pequeñas.
- Un servidor backend que provea las capacidades de almacenamiento y persistencia de los datos para el correcto funcionamiento de la aplicación web.

Capítulo 3: Proyecto interdisciplinario

Una parte integral del proyecto consiste en el análisis del problema enfrentado por la empresa demandante. Para obtener información se recurre a un referente funcional que trabaja en la empresa. Además de su participación, el proyecto cuenta con la asistencia de un estudiante de la carrera de Ingeniería Industrial. El estudiante trabaja en la empresa demandante por lo que inherentemente conoce acerca de los procesos cotidianos de la empresa que están dentro del alcance del proyecto.

Además, el estudiante está desarrollando su proyecto final sobre la gestión del mantenimiento de los activos en la empresa demandante. Es por ello que los proyectos finales comparten un dominio. El proyecto de Ingeniería Industrial busca analizar el problema con el objetivo de proponer mejoras en los procesos. Mientras tanto, el proyecto de Ingeniería Informática busca cumplir con los objetivos detallados anteriormente, para lo cual la información provista por el estudiante resulta de gran utilidad.

La oportunidad de colaborar en dos proyectos finales concurrentes sobre la misma temática facilita la investigación sobre el tema y permite una verdadera colaboración interdisciplinaria.

3.1 Proyecto paralelo de Ingeniería Industrial

El proyecto de Ingeniería Industrial tiene como objetivo relevar los procesos actuales de la empresa demandante relativos a la gestión del mantenimiento. Adicionalmente, busca proponer mejoras a partir del conocimiento obtenido con el fin de optimizar sus procesos.

Ambos proyectos comparten un dominio. Se deben analizar los mismos procesos con objetivos diferentes, sin embargo la colaboración entre los equipos es de beneficio mutuo.

3.2 Impacto en el proyecto

3.2.1 Información sobre el dominio

El estudiante de Ingeniería Industrial trabaja como Analista de datos en la empresa demandante. Es una clara ventaja para el proyecto debido a que permitió conocer mejor los flujos de trabajo que existen actualmente en la empresa.

3.2.2 Dashboard de métricas a monitorear

Uno de los objetivos del proyecto de Ingeniería Industrial es la designación de métricas claves del problema que podrán ser monitoreadas para mejorar la toma de decisiones dentro de la empresa.

En el proyecto de Informática, se tomarán estas métricas para integrarlas dentro del sistema desarrollado. Esta integración se apoya en la información almacenada por el sistema sobre los dispositivos instalados en los equipos de riego.

Capítulo 4: Gestión

4.1 Planificación original del proyecto

Previo a la entrega del protocolo de proyecto final, fue necesario un proceso de análisis del problema para contar con el conocimiento del dominio suficiente para su escritura. Este proceso de análisis no aparece formalmente dentro de las planificaciones. Sin embargo, fue una parte vital del proyecto ya que permitió conocer el problema y poder establecer el alcance.

Al comenzar el proyecto, se planteó un cronograma pensando en una dedicación de 14 horas semanales por miembro del equipo. Adicionalmente, se dividió el proceso de desarrollo en una serie de etapas delimitando módulos de funcionalidad independientes. Para cada módulo se planteó realizar el análisis, el diseño, la implementación y las pruebas en forma consecutiva. La validación con la empresa demandante debía realizarse cada intervalos regulares para garantizar que la problemática estaba siendo descrita correctamente.

Los módulos de funcionalidad propuestos fueron los siguientes:

- Etapa 1: Relevamiento inicial del dominio, requerimientos y elaboración del protocolo.
- Etapa 2: Análisis, diseño, implementación y testeo del MVP (*Minimum Viable Product*, del inglés Producto Mínimo Viable) que incluya seguimiento del estado de los *assets* en tiempo real.
- Etapa 3: Análisis, diseño, implementación y testeo del stock de dispositivos.
- Etapa 4: Análisis, diseño, implementación y testeo de funcionalidad de archivos de registro/carga de documentos.
- Etapa 5: Análisis, diseño, implementación y testeo del módulo de gestión de problemas de hardware. Incluyendo el estado en el que se encuentra (Diagnóstico, Pericia, Análisis forense)
- Etapa 6: Análisis, diseño, implementación y testeo del seguimiento de órdenes de trabajo y *service management*.

- Etapa 7: Análisis, diseño, implementación y testeo del módulo de gestión de contratos de servicio con los clientes. Incluye nivel de servicio contratado y relación con equipos.
- Etapa 8: Integración final y pruebas de sistema.

4.2 Estimaciones previas

Las etapas previamente mencionadas tenían plazos de ejecución que se pueden ver en el siguiente diagrama de Gantt, que fue entregado con el protocolo de Proyecto final.

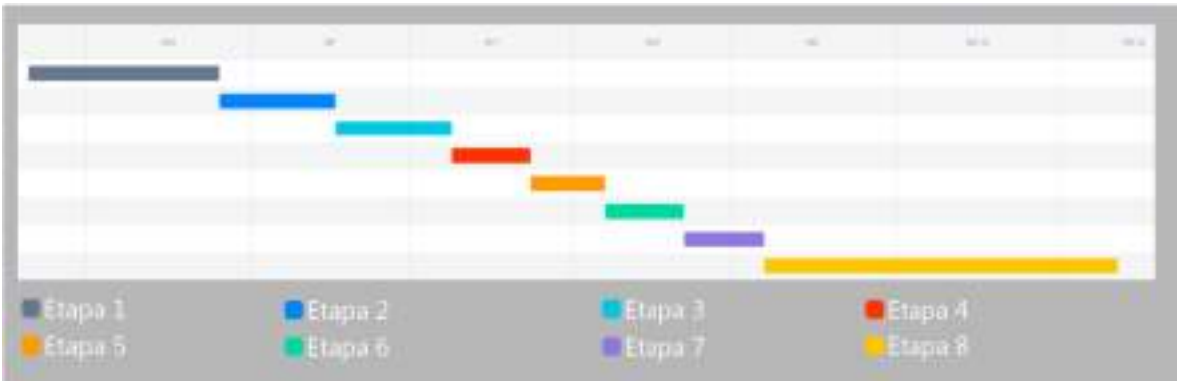


Figura 1. Estimación de plazos de cada etapa.

Se estimó que la etapa 1 tendría una duración más prolongada (8 semanas). Esta decisión fue tomada debido a que incluye el relevamiento inicial del problema y la estructura inicial de la implementación, que normalmente incluye tareas tales como la configuración de entornos de desarrollo y todas las decisiones relacionadas a las tecnologías a utilizar.

A las etapas 2 y 3 se les asignó una duración mayor a la de las demás etapas intermedias (3 semanas cada una) debido a que en ellas se analizaron los dos principales procesos del proyecto.

Por último, se estimó una duración más prolongada para la etapa final, similar a la de la etapa inicial. Esto se debe a que incluye evaluaciones intermodulares y validaciones finales con la empresa demandante.

4.3 FODA del proyecto

En esta sección se presentarán las Fortalezas, Oportunidades, Debilidades y Amenazas del proyecto.

4.3.1 Fortalezas

- Uno de los objetivos del proyecto es la construcción de un sistema que se adapte a las necesidades de la empresa demandante. El producto del proyecto se adaptará perfectamente al control del mantenimiento de dispositivos IoT ubicados sobre equipos de riego. Esta característica no se encuentra disponible en ningún producto del mercado.

- Los integrantes del equipo cuentan con experiencia previa trabajando en conjunto. En consecuencia, cada uno conoce la forma de trabajar del otro. El equipo comparte una dinámica de trabajo y esto es un beneficio a la hora de resolver los conflictos propios del desarrollo de un proyecto.

- La ciudad donde se realiza el proyecto se encuentra en una zona fuertemente agropecuaria. Otras empresas que cuenten con sensores ubicados sobre equipos de riego pueden ser potenciales clientes.

4.3.2 Oportunidades

- La realización del proyecto suplirá una demanda insatisfecha en el ambiente de las empresas de monitoreo de dispositivos IoT agrícolas.

- El proyecto presenta la oportunidad de trabajar en conjunto con un alumno de la carrera de Ingeniería Industrial, lo cual permite compartir conocimientos entre ambas disciplinas.

- La colaboración con la empresa demandante permite aprovechar todo su conocimiento sobre el problema a resolver. Esto significa que la elicitación de los requerimientos del sistema resultará más simple debido a que la empresa ya ha hecho una elicitación previa que puede ser generalizada a otras empresas del rubro.

4.3.3 Debilidades

- Como la mayoría de la información del dominio es obtenida del referente funcional, es posible que los resultados del proyecto también terminen siendo específicos a la empresa demandante.

- Los miembros del equipo trabajan 8 horas diarias. En consecuencia, la cantidad de tiempo disponible para avanzar con el proyecto es limitada.

- El segmento de mercado al que apunta el proyecto es pequeño y especializado (empresas con dispositivos IoT que mantener).

4.3.4 Amenazas

- Al tratarse de un proyecto interdisciplinario, existe la posibilidad de que el proyecto paralelo de Ingeniería Industrial se frene por algún motivo. Esta situación estaría fuera del control del equipo y causaría demoras en la ejecución del proyecto dado que la información que se esperaba obtener no estaría disponible.

- El demandante del proyecto es una empresa joven y de rápido crecimiento. Por ello, los requerimientos pueden cambiar drásticamente acorde a las necesidades del negocio.

Capítulo 5: Metodología de trabajo

Al momento de seleccionar una metodología que sea acorde al equipo de trabajo se decidió descartar la metodología tipo cascada debido a su principal desventaja para realizar proyectos con requerimientos cambiantes: la falta de oportunidades para obtener retroalimentación por parte del demandante.

En contraste con la metodología de cascada, existen las metodologías ágiles. Su principal diferencia radica en que evitan separar la realización de un proyecto en grandes etapas de análisis, diseño, implementación, y pruebas. En las metodologías ágiles se busca entregar valor al demandante en cantidades pequeñas pero con mucha frecuencia. En consecuencia, se obtiene una retroalimentación constante lo que garantiza un buen cumplimiento de sus necesidades.

Es por ello que se decidió la utilización de una metodología ágil para la ejecución del proyecto. La metodología seleccionada fue Kanban. Kanban se basa en la división del trabajo en tareas pequeñas. Cada tarea tiene un estado posible: Pendiente; En curso; En pruebas; Completada. De esta manera, los resultados son observables por todo el equipo. La subdivisión del trabajo permite la constante validación con el demandante. Por lo tanto, se evitan costos de retrabajo asociados con la acumulación de errores.

En el proyecto, las tareas representaban labores de análisis de un proceso particular, ajustes en el diseño necesarios para su implementación en el producto, implementación propiamente dicha y pruebas.

5.1 Metodología de análisis

El análisis del dominio se basó en dos grandes fuentes de información:

- El referente funcional del proyecto.
- El estudiante de Ingeniería Industrial que trabaja en la empresa demandante.

La información se obtuvo principalmente durante videollamadas realizadas en forma semanal. Durante estas llamadas se realizaron consultas sobre el dominio del problema. Además se pudo discutir y discernir acerca de requerimientos para el sistema.

Al planificar las etapas del proyecto, se decidió que haya un proceso de análisis en cada una. Por ejemplo, la etapa 2 (*MVP* de seguimiento de los *assets*) comenzaría con la obtención de los requerimientos para esta funcionalidad en particular. Del mismo modo, cada etapa subsiguiente debía tener su parte de análisis respectiva.

Una vez realizado el análisis, los entregables serían validados con el referente funcional. De esta forma, se asegura una rápida detección de errores.

5.1.1 Entregables esperados

5.1.1.1 Requerimientos funcionales y no funcionales

Los requerimientos son el conjunto de funcionalidades y restricciones no funcionales que el sistema debe cumplir. Son información invaluable a la hora de diseñar una solución que cumpla con las necesidades de la empresa demandante y el nicho de mercado.

5.1.1.2 Casos de uso

Los casos de uso aportan información adicional para tener una mejor comprensión de la funcionalidad presentada por los requerimientos y permiten contemplar todos los casos alternativos que pueden surgir.

5.1.1.3 Entidades del dominio

Su entendimiento es crucial para poder desarrollar el resto de los diagramas de análisis.

5.1.1.4 Flujos de trabajo

Los flujos de trabajo reflejan el funcionamiento actual de la empresa demandante y son de vital importancia para adaptar la solución a sus procesos.

5.2 Metodología de diseño

El diseño del sistema fue realizado a partir de los requerimientos planteados en cada etapa, y siempre pensando en lograr una arquitectura escalable que pueda adaptarse a los requerimientos del demandante.

5.2.1 Entregables esperados

5.2.1.1 Diagrama Entidad-Relación

El diagrama entidad-relación permite plasmar las entidades reales del dominio en un modelo simplificado según las necesidades del demandante. En el sistema solo se almacenarán los atributos de cada entidad que suplan los requerimientos. Adicionalmente, se conocerán las relaciones entre las entidades y su cardinalidad.

5.2.1.2 Arquitectura del sistema

La arquitectura está compuesta por todos los componentes que forman parte del sistema y que interactúan entre sí para cumplir con las funcionalidades propuestas. En el capítulo de diseño se describen las características de cada uno.

5.2.1.3 Tecnologías a utilizar

El producto utilizará diversas tecnologías, la especificación de cada una de ellas es un entregable del diseño. El producto debe ser desarrollado utilizando tecnologías que deben ser seleccionadas teniendo en cuenta varios criterios, entre los que se encuentran la afinidad del equipo con las mismas, el soporte ofrecido por sus respectivas comunidades y su grado de madurez.

5.2.1.4 Directivas de seguridad informática

Es sabido que el producto manejará información sensible tal como ubicaciones geográficas de los equipos, datos personales de sus usuarios, entre otras. Es por ello que se debe pensar en el manejo seguro de los datos.

5.3 Metodología de desarrollo y pruebas

La metodología de desarrollo y testing a utilizar fue Kanban según lo descrito en la metodología de trabajo general. Además, se propone la creación de un proceso de integración continua de los cambios. El resultado del mismo será un entorno de pruebas accesible por la empresa demandante con el fin de validar los avances en la implementación del producto.

Capítulo 6: Análisis del problema

6.1 Dominio del problema

6.1.1 Sistema de riego

Se denomina sistema de riego al conjunto de estructuras que hace posible que una determinada área pueda ser cultivada con la aplicación del agua que las plantas necesitan.

6.1.1.1 Sistemas de riego circulares

Su función consiste en llevar el agua de riego hasta los cultivos mediante una tubería metálica, que es montada sobre torres de metal que se mueven sobre conjuntos de ruedas, de modo que el pivote gira en círculos manteniendo uno de sus extremos fijos en el centro del campo. A lo largo de toda la tubería cuelgan aspersores, cuyas cabezas de riego pueden ser ubicadas a distancias variables del suelo [1].


Figura 2. Campos circulares con sistema de riego por pivot. [2] [3]

6.1.2 *Assets*

Para comprender el alcance del proyecto es necesario entender la disposición actual de los sensores y dispositivos de medición existentes en la empresa demandante. La arquitectura de sensores se basa en tres grandes dispositivos posicionados en cada equipo de riego, cada uno compuesto por subcomponentes.

Estos son:

6.1.2.1 *Gateway* (GTW)

Todo sistema de sensores debe tener alguna manera de comunicarse con un servidor para transmitir los datos que ha obtenido. Esta responsabilidad es aceptada por el dispositivo conocido como ***Gateway***, el cual se encarga de transformar los datos a un formato apto para su transmisión.

Figura 3. *Gateway* de la empresa instalado en un equipo de riego.[4]

Por definición cumple la función de *gateway*: permite el flujo de datos de un tipo de red hacia otra. Esto puede verse en el caso de la empresa demandante como la obtención de los datos mediante el protocolo LoRa para su posterior envío mediante red satelital. Su ubicación dentro del equipo de riego es en el centro, donde tiene una conexión cableada con el **sensor de presión** como puede verse en la Figura 3.

Adicionalmente, esta posición resulta ventajosa debido a que tiene acceso a la energía eléctrica provista por el pivote de riego.

Dentro del *gateway* se pueden encontrar los siguientes componentes:

- **Placa *gateway*:** placa base donde se conectan los distintos componentes.
- **Housing:** carcasa del dispositivo, lo protege de daños y factores ambientales.
- **Batería:** provee energía al dispositivo.
- **Módem:** permite la transmisión satelital.
- **Antena LoRa**: necesaria para la comunicación con el **Sensor GPS.**

6.1.2.2 Sensor GPS (SGPS)

En el contexto del pivote de riego, resulta necesario saber en qué ángulo se encuentra para validar que esté operando correctamente (un régimen de funcionamiento normal es aquel en el cual el pivote gira sin dificultades, no se encaja y riega en forma uniforme). Este ángulo es calculado mediante el **sensor GPS** (del inglés *Global Positioning System,* Sistema de Posicionamiento Global), ubicado en el extremo del pivote. La información de posición obtenida es transmitida al ***Gateway*** para su posterior transmisión.

Figura 4. Sensor GPS de la empresa demandante.

Dentro del **Sensor GPS** se pueden encontrar los siguientes componentes:

- **Antena GPS:** antena para recibir la señal satelital.
- **Placa SGPS:** placa base donde se conectan los distintos componentes.
- **Housing:** carcasa del dispositivo, lo protege de daños y factores ambientales.
- **Batería:** provee energía al dispositivo.
- **Antena LoRa**: necesaria para la comunicación con el ***Gateway*.**

- **Panel solar:** se utiliza para cargar la batería durante el día, utilizando energía solar.

6.1.2.3 Sensor de presión (SPRES)

Un **sensor de presión** es un instrumento compuesto por un elemento detector de presión con el que se determina la presión real aplicada al sensor (utilizando distintos principios de funcionamiento) y otros componentes que convierten esta información en una señal de salida [5].

Figura 5. Sensor de presión utilizado en la empresa.

Los sensores están instalados en equipos de riego y tienen un rol importante en el control del correcto funcionamiento de este sistema. Su función es medir la presión del agua, antes de ser expelida hacia los cultivos. El sensor de presión está conectado mediante un cable con el ***Gateway*** (Figura 5), proporcionándole la información medida.

6.1.3 Comunicación entre *assets*

La comunicación entre el *gateway* y el sistema informático existente de la empresa demandante es realizada mediante transmisión satelital utilizando servicios provistos por otra empresa.

La comunicación entre el gateway y el sensor GPS es realizada mediante el protocolo LoRa.

6.1.3.1 LoRa

LoRa (abreviación de *Long Range*, largo alcance en inglés) es un protocolo de comunicación inalámbrica muy utilizado entre dispositivos IoT [6]. Sus principales cualidades que lo destacan para este uso son un bajo consumo energético y optimización de las transmisiones para largo alcance.

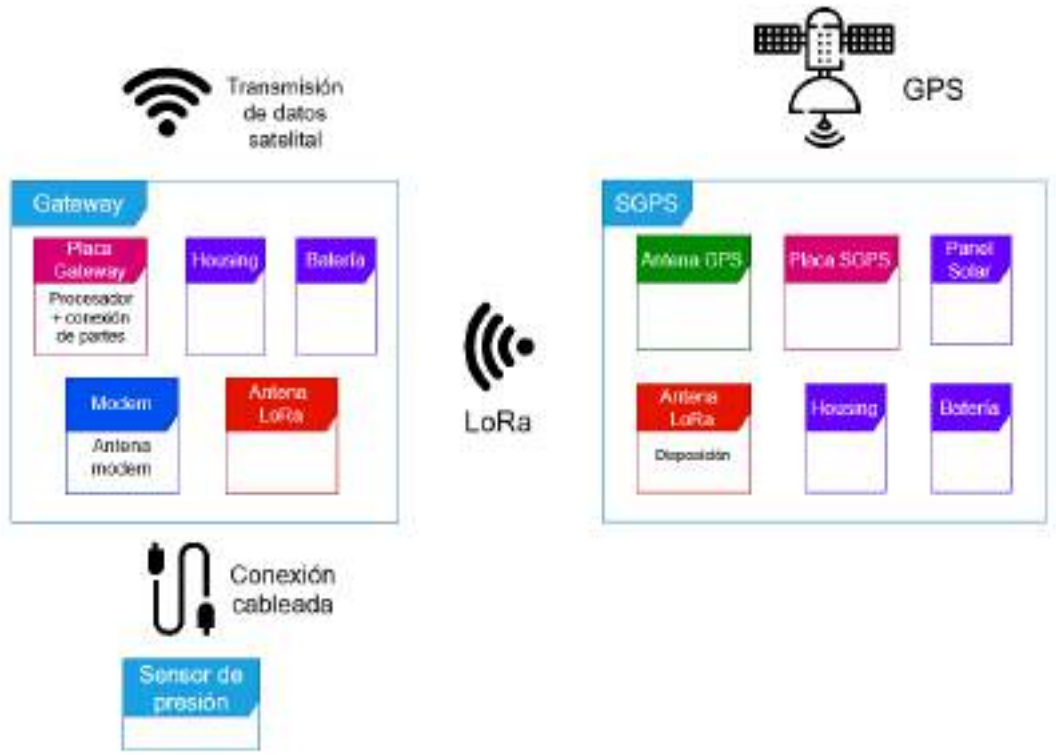


Figura 6. Diagrama de componentes de dispositivos utilizados por la empresa demandante.

6.1.4 Órdenes de trabajo

Las órdenes de trabajo involucran dos entidades principales que serán descritas a continuación.

6.1.4.1 *Hardware issue*

Un *hardware issue* es la representación de un problema real ocurrido sobre un equipo de riego. Son creados cuando algún dispositivo presenta una falla. Los

estados posibles por los que un *hardware issue* puede transitar son, según la empresa demandante:

1. **In field:** Los *issues* recién creados se encuadran en este estado, esperando que se les asigne un técnico.
2. **Asignado:** Una vez que un técnico es asignado, pasa a este estado y el técnico puede comenzar a realizar pericias y reparaciones.
3. **Reparado:** Los *issues* que tengan este estado, tendrán por lo menos una reparación vinculada y ya fueron visitados por el técnico.
4. **Out of field:** Cuando en la reparación se hizo algún cambio de dispositivo, el *hardware issue* pasa a este estado, pudiéndose realizar autopsias a los *assets* dañados.
5. **Cerrado:** Una vez que se realizan las autopsias correspondientes o se realiza una reparación sin cambio de dispositivo, el issue finaliza en estado cerrado.

6.1.4.2 Pericia

Consiste en una inspección del estado del equipo de riego y sus *assets* instalados al llegar. Incluye imágenes de los distintos *assets* y logs obtenidos desde los mismos mediante el uso de un teléfono celular.

6.1.4.3 Reparación

El técnico revisa el problema y lo soluciona, actividad que posteriormente documenta en el sistema. La reparación puede ser de dos tipos:

1. Reemplazo de *assets*: Se detalla cuales fueron cambiados. Por ejemplo, el reemplazo del gateway instalado en el equipo por otro que el técnico traía como repuesto.
2. Solución en campo: Cuando no fue necesario el reemplazo de ningún *asset*, se documenta qué cambios tuvieron que hacerse para reparar el dispositivo. Por ejemplo, si el técnico encuentra un cable desconectado y lo conecta.

6.1.4.4 Solicitud de instalación

Una solicitud de instalación representa una necesidad por parte de un cliente de comenzar el monitoreo de un equipo de riego. Para cumplirla es necesario que

personal técnico de la empresa viaje hasta la ubicación e instale en el equipo varios *assets* encargados de monitorear y transmitir los datos en forma satelital.

La solicitud pasará por varios estados hasta completarse exitosamente. Estos son:

1. **Abierta**: La solicitud está esperando ser asignada a algún técnico de campo de la empresa.
2. **Asignada**: La solicitud ya fue asignada y se espera la instalación por parte del técnico.
3. **Realizada**: El técnico ya visitó el equipo e instaló los *assets*. La empresa debe corroborar que el equipo esté transmitiendo correctamente para pasar al siguiente estado.
4. **Completada**: Se confirma que el equipo ya está transmitiendo datos y la solicitud finaliza exitosamente.

Puede que una vez instalado el equipo no transmita correctamente. En este caso la solicitud será rechazada y se creará una nueva para el mismo equipo.

6.1.4.5 Solicitud de desinstalación

Una solicitud de desinstalación representa la necesidad de finalizar el monitoreo de un equipo de riego. Para cumplirla es necesario que personal técnico de la empresa vaya a recuperar los *assets* instalados en el equipo. Los estados son análogos a los de una solicitud de instalación.

6.2 Elicitación de requerimientos

Para la elicitación de los requerimientos se mantuvo un trato constante con la empresa demandante. Adicionalmente, el trabajo interdisciplinario con el proyecto paralelo de Ingeniería Industrial resultó muy útil para su obtención.

6.2.1 Requerimientos no funcionales

A continuación se detallan los requerimientos no funcionales impuestos por la empresa demandante:

- La interfaz de usuarios debe ser desarrollada con la tecnología Vue.js versión 3.
- El producto debe ser accesible tanto desde dispositivos móviles como computadoras.
- El producto debe contar con control de acceso para los distintos tipos de usuarios requeridos.

6.2.2 Requerimientos funcionales principales

A continuación se describen los principales requerimientos funcionales solicitados por el demandante. No se incluyen en esta sección los requerimientos de Alta, Baja, Modificación y Consulta de entidades del dominio para facilitar la lectura del informe. Estos se encuentran en el apéndice C: requerimientos completos

Identificación del requerimiento:	RF01
Nombre del Requerimiento:	Monitoreo en tiempo real de sensores de cada equipo de riego.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario debe contar con permisos de administrador.
Características:	El sistema debe permitir visualizar el estado de transmisión de los sensores de un equipo de riego, detallando qué equipos están transmitiendo normalmente y cuáles están teniendo fallas.

Identificación del requerimiento:	RF02
Nombre del Requerimiento:	Seguimiento de los equipos de riego.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario debe contar con permisos de administrador.
Características:	El sistema debe permitir visualizar todos los sensores y clientes asociados a un determinado equipo de riego.

Identificación del requerimiento:	RF03
Nombre del Requerimiento:	Cálculo de métricas de desempeño.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario debe contar con permisos de administrador.
Características:	El sistema debe calcular métricas de desempeño, propias del contexto

	del funcionamiento y mostrarlas al usuario.

Identificación del requerimiento:	RF04
Nombre del Requerimiento:	Control de stock de *assets*.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario debe contar con permisos de administrador.
Características:	El sistema debe permitir visualizar la ubicación de los *assets* de la empresa. Además, el sistema debe permitir conocer la cantidad de *assets* de cada ubicación, discriminando por tipo.

Identificación del requerimiento:	RF05
Nombre del Requerimiento:	Seguimiento de fallas y órdenes de trabajo.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario debe contar con permisos de administrador.
Características:	El sistema debe permitir la detección de fallas de transmisión de los *assets* y proveer los medios para realizar un seguimiento del problema, que incluye su asignación a un técnico y su resolución.

Identificación del requerimiento:	RF06
Nombre del Requerimiento:	Control de nivel de servicio para cada cliente.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe dar la posibilidad de asociar a cada equipo un nivel de servicio contratado.

6.3 Casos de uso abreviados

6.3.1 Login al sistema

Un usuario ingresa con sus credenciales al sistema y se encontrará logueado. La autenticación es por email y contraseña. El usuario podrá ser de dos tipos:

- Administrador: cuenta con permisos para utilizar todo el sistema

- Técnico: solo cuenta con permisos para modificar el estado de los *issues* y solicitudes que tenga asignados.

6.3.2 Monitoreo general de equipos de riego

Un usuario **administrador** utiliza el sistema para conocer el estado de todos los equipos de riego. Puede visualizar su ubicación además de ver el estado de transmisión actual de los mismos.

6.3.3 Monitoreo de sensores de un equipo

Un usuario **administrador** utiliza el sistema para conocer el estado de un equipo de riego en particular. Además de la información provista por el 6.3.2, puede ver que *assets* de la empresa se encuentran instalados en el equipo.

6.3.4 Detección automática de *hardware issues*

El sistema recibe notificaciones de sistemas externos ante la falla de *assets* instalados y genera los *hardware issues* correspondientes para su tratamiento por parte de los usuarios.

6.3.5 Consulta de estado de *hardware issues* de un equipo

El usuario accede al módulo de *hardware issues* y selecciona uno para ver sus detalles. Si el usuario es **administrador** puede acceder a todos los listados (discriminado por estado de *hardware issue*) disponibles. En cambio, si el usuario es un **técnico** puede visualizar los *issues* que le hayan asignado o en los que haya realizado trabajos.

6.3.6 Diagnóstico de un *hardware issue* válido

Ante la detección automática de un *hardware issue*, lo único sabido es que el dispositivo no funciona. Es tarea del usuario **administrador** ingresar al sistema y asignarle un diagnóstico. En el diagnóstico se detalla el tipo de falla que está teniendo el dispositivo. Esta información se obtiene revisando los logs de transmisión y los últimos datos enviados a la plataforma de monitoreo de riego de la empresa demandante. Por ejemplo, un diagnóstico puede ser “Presión de agua en 0mA”.

6.3.7 Asignación de un *hardware issue*

Luego de haber sido creado, un usuario **administrador** puede asignarle al problema un **técnico**. El nuevo estado del *hardware issue* será ”asignado”.

6.3.8 Carga de peritaje y reparación

Este caso comienza cuando un usuario técnico tiene un *hardware issue* asignado. Su trabajo consiste en ir hasta la ubicación del equipo de riego con fallas con el objetivo de documentar el estado de los *assets* y las reparaciones realizadas.

6.3.9 Carga de autopsia

Luego de que un técnico reemplaza un dispositivo, se deben realizar autopsias sobre los *assets* dañados. Cuando un **administrador** realiza una autopsia, éste la carga al sistema. El nuevo estado del *issue* es “*out-of-field”*.

6.3.10 Seguimiento del stock de *assets*

Es necesario conocer la ubicación de los activos del sistema. Para ello, el usuario **administrador** ingresa al módulo de stock. Luego, puede visualizar por qué almacenamientos de stock transitó el *issue*.

6.3.11 Vista del panel de control

El usuario **administrador** ingresa al panel de control donde puede observar métricas importantes sobre el sistema, como por ejemplo: cantidad de equipos con *assets* instalados comparado con cantidad total de equipos, promedio de tiempo transcurrido desde que se diagnosticó un *issue* hasta que fue resuelto.

6.4 Flujos de trabajo

En esta sección se describirán los principales flujos de trabajo existentes en la empresa demandante relacionados a la gestión del mantenimiento de sus activos. Los flujos aportan información valiosa en el desarrollo del proyecto.

6.4.1 Flujo de *hardware issues*

El estado inicial de un *hardware issue* es ***in-field***, indicando que un equipo que está ubicado en determinado campo tiene un problema. Una vez creado, un usuario administrador puede asignar un técnico para que lo resuelva y el nuevo estado del *issue* será **asignado.**

Una vez asignado, el técnico puede cargar pericias (para indicar qué fue lo sucedido) y reparaciones vinculadas al *issue*. De esta forma, el nuevo estado será **reparado**.

Un usuario administrador debe corroborar la reparación. Si no la aprueba, el *issue* vuelve al estado ***in-field***. Si la aprueba, en caso de que en la reparación se haya realizado algún cambio de *asset*, el nuevo estado será ***out of field***, quedando a la espera de autopsias y luego pasando al estado **cerrado**. En caso contrario, es decir si se reparó sin cambiar *assets,* se cerrará el *issue*.

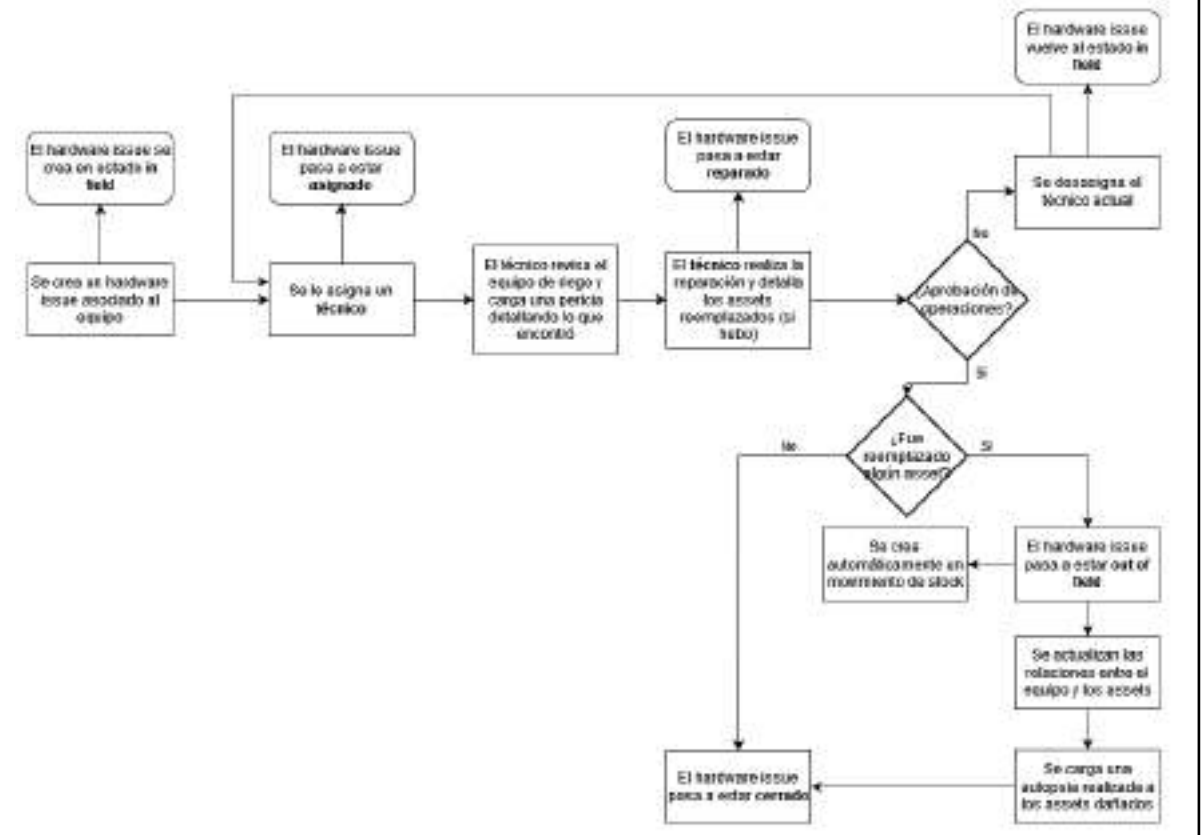


Figura 7. Diagrama de flujo para un *hardware issue*.

6.4.2 Flujo de solicitudes de instalación

El objetivo de este flujo es instalar en un equipo de riego los componentes necesarios para permitir su monitoreo en forma remota. Las etapas son las siguientes:

1. Se parte desde un equipo que no cuenta con ningún dispositivo de la empresa instalado. La solicitud es creada y queda vacante, en estado **abierta**.
2. Un usuario administrador se encarga de asignarle la solicitud a un técnico cuya zona geográfica de acción incluya a este equipo. La solicitud ahora está **asignada.**
3. El técnico se encargará de viajar hasta el equipo e instalarle los *assets* necesarios. El técnico utilizará su celular para cargar en el sistema detalles de la instalación. Estos son los identificadores de los *assets* que instaló, además de los logs e imágenes del trabajo realizado. La solicitud se encuentra **realizada.**
4. Un usuario administrador verifica que el equipo se encuentre transmitiendo correctamente y da la aprobación final. La solicitud pasa a estar **completada** y en el sistema se almacenan las nuevas ubicaciones de los *assets* instalados.

También es posible que un técnico complete una instalación pero el equipo no comience a transmitir correctamente. En este caso la solicitud será **rechazada** y se creará una nueva solicitud para el mismo equipo. Esta podrá ser asignada a un nuevo técnico para que el proceso pueda completarse exitosamente.

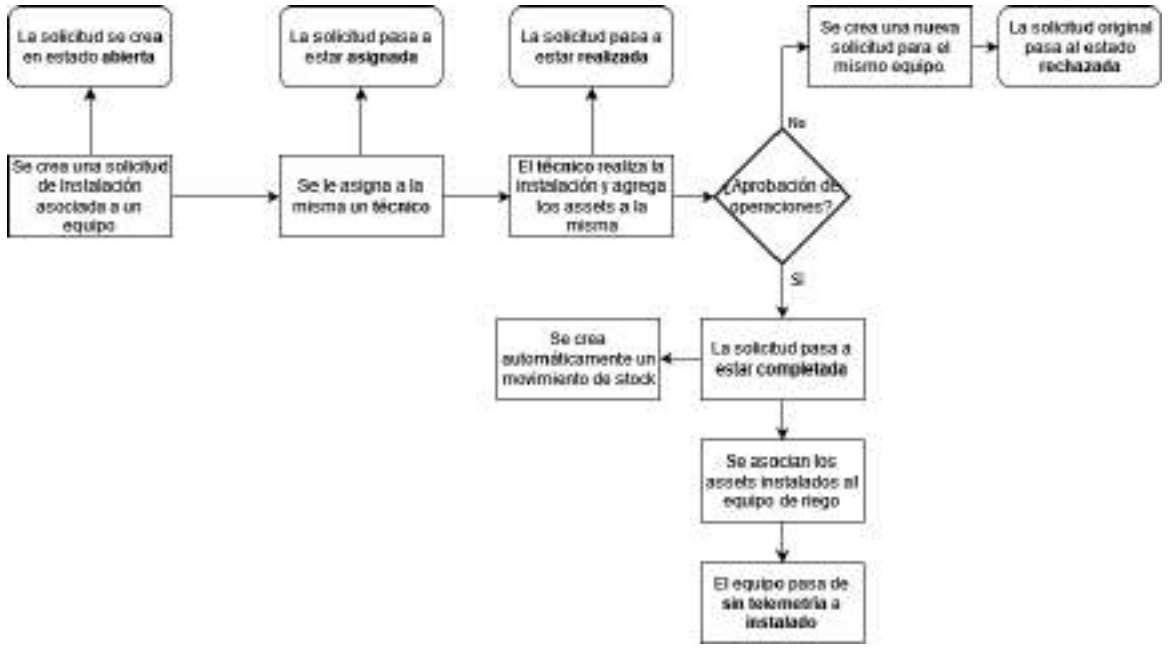


Figura 8. Diagrama de flujo para una solicitud de instalación en un equipo de riego.

6.4.3 Flujo de solicitudes de desinstalación

El objetivo de este flujo es finalizar el monitoreo de un equipo de riego, el cual posee *assets* instalados que deben ser recuperados por la empresa. Las etapas son las siguientes:

1. Se parte desde un equipo que posee dispositivos instalados por parte de la empresa. La solicitud es creada, en estado **abierta**.
2. Un usuario administrador se encarga de asignársela a un técnico cuya zona geográfica de acción incluya a este equipo. La solicitud ahora está **asignada.**
3. El técnico se encargará de viajar hasta la ubicación del equipo y recuperar los *assets* instalados. Posteriormente detallará en el sistema la fecha en la cual realizó la desinstalación. La solicitud se encuentra **realizada.**
4. Un usuario administrador verifica que el equipo haya cesado su transmisión. La solicitud pasa a estar **completada** y en el sistema se mueven los *assets* removidos al inventario del técnico.

También es posible que un técnico complete una desinstalación pero esta no sea considerada satisfactoria. El caso más común es el caso en el que no todos los *assets* fueron removidos. En este caso la solicitud será **rechazada** y se creará una nueva solicitud para el mismo equipo. Esta podrá ser asignada a un nuevo técnico para intentar nuevamente.

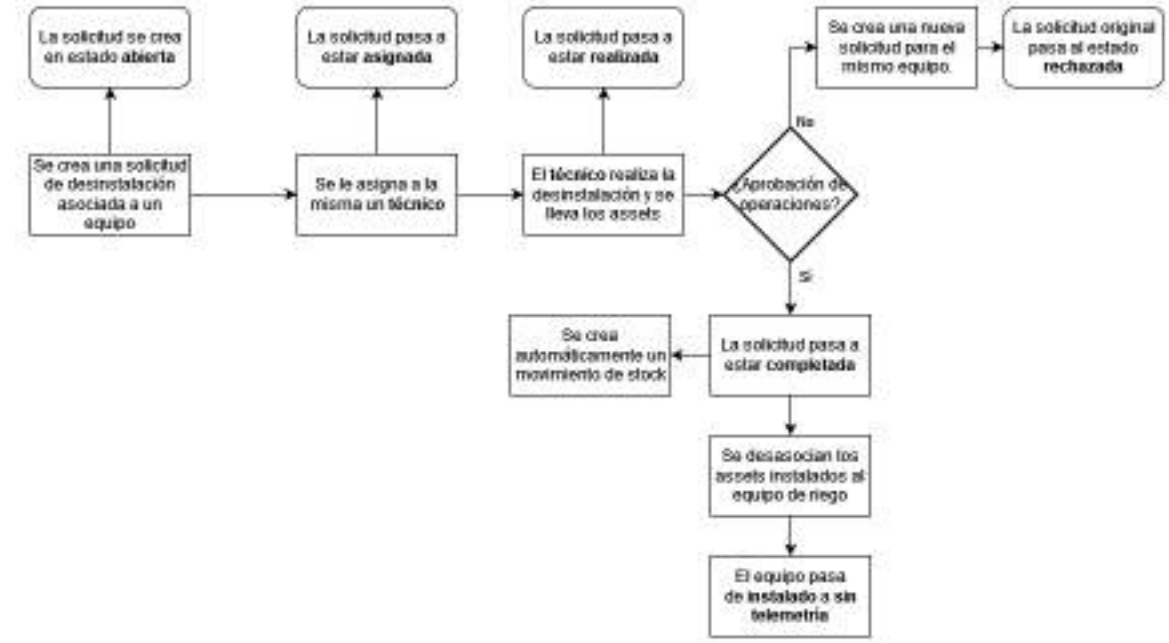


Figura 9. Diagrama de flujo para una solicitud de desinstalación en un equipo de riego.

Capítulo 7: Diseño de la solución

En este capítulo se detallarán todas las decisiones de diseño que fueron tomadas a lo largo del proyecto. Estas incluyen la elección de las tecnologías a utilizar, la arquitectura del sistema, sus componentes y sus características de seguridad informática.

7.1 Arquitectura

El diseño se basa en el modelo cliente-servidor centralizado tradicional. La alternativa sería pensar en una arquitectura de sistema descentralizada. Esta opción es rápidamente descartada debido a que sus ventajas no son pertinentes al proyecto y sus desventajas la vuelven poco práctica.

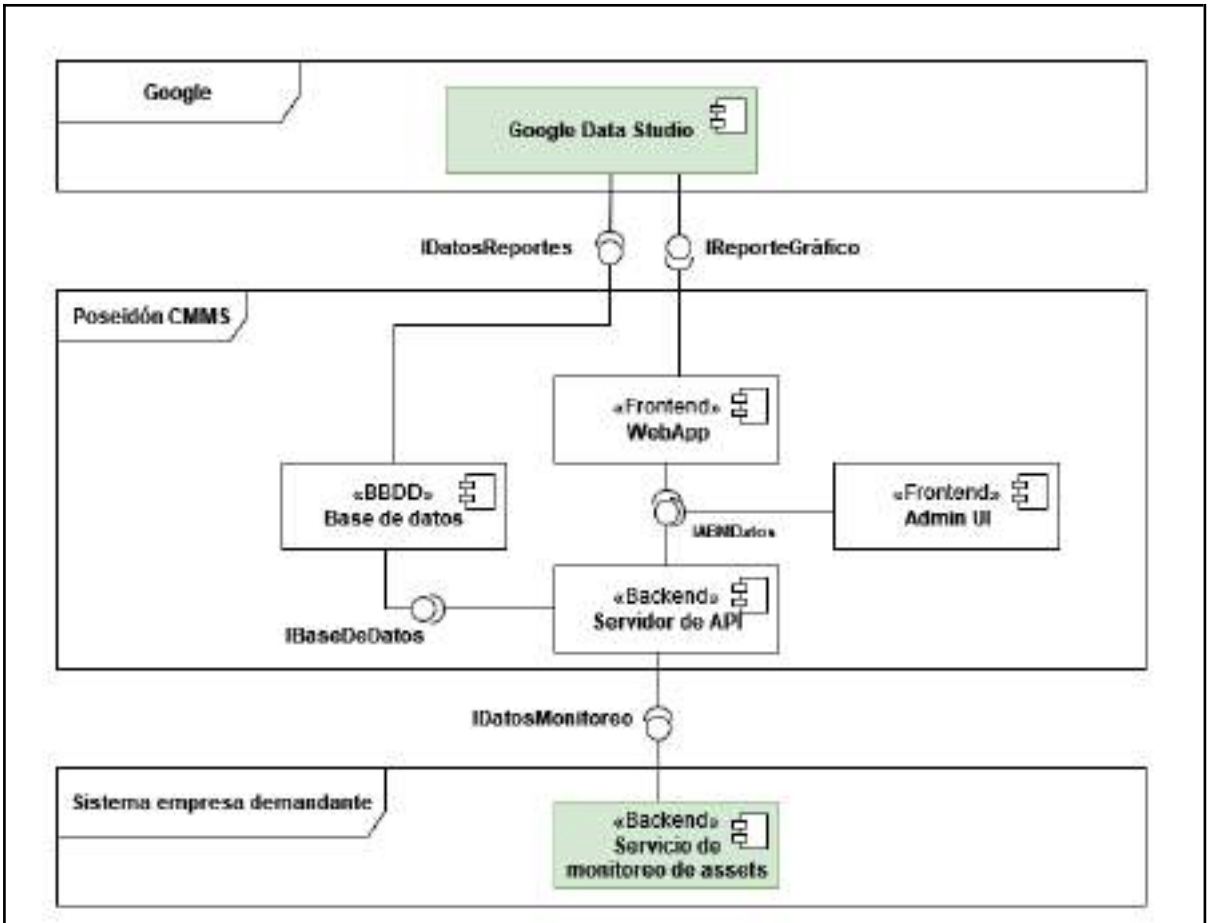


Figura 10. Diagrama de componentes. Los servicios externos están destacados en verde.

7.1.1 Componentes internos

7.1.1.1 Admin UI

El Admin UI es una interfaz gráfica que permite a un usuario administrador hacer los cambios que considere necesarios sobre cualquier entidad almacenada. Esta interfaz es provista por el framework KeystoneJS, cuya elección se fundamenta en el subtítulo siguiente.

Su única conexión es con el servidor de API donde se ven reflejados todos los cambios solicitados por el administrador.

7.1.1.2 WebApp

La WebApp es la interfaz gráfica principal del sistema. Desde ella se ejecutan las acciones principales de gestión del mantenimiento (por ejemplo, carga y actualización de *hardware issues* y órdenes de trabajo) y control de stock. Es una interfaz que se comunica con el servidor de API para la obtención y persistencia de datos.

Los usuarios de tipo técnico se verán limitados a realizar algunas acciones, mientras que los usuarios administradores podrán realizar todas las operaciones.

La existencia de la WebApp se debe a que el Admin UI provisto por KeystoneJS presenta una usabilidad insuficiente de cara a las necesidades de los usuarios de la empresa demandante. En consecuencia, se diseñó la WebApp para suplir esta necesidad de mayor usabilidad en los flujos de trabajo principales.

7.1.1.3 Servidor de API

El servidor de API es el encargado de implementar toda la lógica del negocio necesaria para garantizar que la base de datos se mantenga en un estado consistente. Implementa todas las validaciones y restricciones del dominio necesarias para cumplir este objetivo.

Servicios que ofrece:

- API GraphQL consumida por ambos clientes frontend.
- Un endpoint tipo REST expuesto para que sistemas de la empresa demandante puedan notificar sobre fallas en los dispositivos.

Servicios que consume:

- La base de datos que funciona como su capa de persistencia.

7.1.1.4 Base de datos

La base de datos elegida es de tipo relacional. Entre las bases de datos relacionales, se seleccionó a PostgreSQL. En la base de datos se almacenan los datos utilizados por el sistema mediante tablas. En el siguiente apartado de este capítulo (7.2.2.4) se detallarán los motivos de esta elección.

Sus dos conexiones son:

- Conexión con el servidor de API: se encarga de proveer todos los datos para su persistencia.
- Conexión con Google Data Studio: suministra los datos necesarios para la generación de reportes. La selección de métricas de los reportes provienen del proyecto paralelo de Ingeniería Industrial mientras que los datos utilizados son provistos por la base de datos.

7.1.2 Integraciones

7.1.2.1 Integración con Google Data Studio

Los datos de entrada para Google Data Studio son obtenidos mediante consultas a la base de datos del sistema. Como resultado, Google Data Studio genera un reporte que se muestra dentro de la WebApp. Para la selección de métricas y diseño de dashboards se trabajó en conjunto con el proyecto de Ingeniería Industrial paralelo.

7.1.2.2 Integración con el servicio de monitoreo de *assets* de la empresa demandante

Poseidón CMMS debe contar con la capacidad de recibir información sobre el estado de transmisión de los *assets* y en consecuencia, crear y almacenar los *hardware issues* correspondientes en la base de datos. Para ello se realizó una integración con el sistema de monitoreo de sensores existente en la empresa demandante. Ante una falla en un sensor, el sistema de monitoreo existente en la empresa notifica a Poseidón para que sea creado el *hardware issue* correspondiente.

7.2 Elección de tecnologías

7.2.1 Interfaz gráfica web

Para la elección de las tecnologías a utilizar para la implementación de la interfaz web, se deben ponderar varios factores:

- Experiencia del equipo en su utilización.
- Soporte de la comunidad.
- Madurez de la tecnología.

Considerando al tiempo de implementación como métrica principal a optimizar, se hubiera preferido la utilización de ReactJS como framework para las interfaces web. Esto se debe a que ambos miembros del equipo tienen mucha experiencia con su utilización y prácticamente no hubiera existido curva de aprendizaje.

Sin embargo, existen restricciones impuestas por la empresa demandante. Se solicitó que cualquier interfaz web construida sea implementada utilizando el framework VueJS. En la empresa, las páginas existentes se encuentran implementadas utilizando este framework, por lo que les resulta conveniente que se siga con esta elección para facilitar la mantenibilidad del producto. Finalmente, se decidió utilizar VueJS.

El equipo contaba con experiencia en el lenguaje JavaScript, pero no tenía experiencia utilizando VueJS. Los conocimientos requeridos para hacer un correcto uso de este framework debieron ser adquiridos dentro del tiempo del proyecto. En consecuencia, las etapas de desarrollo correspondientes tuvieron una duración mayor a la esperada dado que incluyeron el aprendizaje de la tecnología.

7.2.2 Servidor de aplicación

7.2.2.1 Lenguaje de programación

Las opciones a la hora de decidir el lenguaje de programación para desarrollar el sistema se vieron influenciadas por los siguientes factores:

- Conocimientos previos del equipo.
- Comunidad y soporte.
- Existencia de librerías.
- Condiciones del demandante.

Para determinar el lenguaje a utilizar, se partió de 3 alternativas conocidas por al menos uno de los miembros del equipo. Estas fueron:

- JavaScript
- C#
- PHP

Estos tres lenguajes cuentan con grandes comunidades de usuarios, por lo que existe mucha documentación y soporte en cada uno de ellos. Además, todos cuentan con muchas librerías para desempeñar tareas comunes. En consecuencia, estos factores no fueron prioritarios en la decisión dado que ninguno presenta una ventaja clara.

El demandante del proyecto no impuso condiciones sobre el lenguaje a utilizar en el servidor de aplicación, por lo que este factor no resultó importante a la hora de decidir. Sin embargo, JavaScript sí fue obligatorio para el desarrollo de la Interfaz web y el desarrollo de toda la implementación en un solo lenguaje es una característica deseable.

Con respecto a los conocimientos previos, ambos miembros del equipo cuentan con experiencia en la utilización de JavaScript y C#. Además uno de los miembros cuenta con mucha experiencia en la utilización de PHP. Utilizando este criterio, es preferible un lenguaje conocido por ambos miembros. Entre JavaScript y C# el equipo contaba con mayor experiencia en JavaScript por lo que este resulta favorecido.

Otro factor importante en la decisión fue una sugerencia por parte del co-director del proyecto. Se recomendó la utilización de un framework para servidores llamado KeystoneJS. Este provee facilidades para la implementación de servidores y está desarrollado en JavaScript. Este factor contribuye a que la elección del lenguaje sea JavaScript.

En conclusión, se optó por JavaScript por los siguientes motivos:

- Ambos miembros del equipo cuentan con experiencia en su utilización, lo cual evita una curva de aprendizaje causada por un lenguaje poco conocido.
- Al elegir JavaScript para la implementación del servidor, todos los componentes del proyecto se desarrollan utilizando el mismo lenguaje lo cual resulta conveniente.
- El amplio soporte provisto por la comunidad en la forma de consultas resueltas y librerías hace que el desarrollo sea más rápido comparado con otros lenguajes menos populares.
- Para la utilización del framework KeystoneJS necesariamente se debe utilizar Javascript.

7.2.2.2 REST Puro vs GraphQL

Para la implementación se decidió desarrollar un servidor de API del tipo GraphQL por sus significativas ventajas sobre las APIs REST tradicionales. Las APIs de tipo GraphQL presentan varias ventajas:

- Permiten solicitar al servidor exactamente las entidades que se necesitan, evitando la transferencia innecesaria de datos.

- Permiten acceder a varias entidades relacionadas en una sola consulta, lo cual disminuye los tiempos de respuesta del sistema.
- Permiten creaciones y modificaciones de entidades en cascada, lo cual facilita el proceso de desarrollo.

Las ventajas mencionadas, además de su presencia en el mercado apoyada por varios gigantes tecnológicos, generaron curiosidad en el equipo por lo que finalmente se decidió su adopción. Adicionalmente, el framework utilizado en el servidor de aplicación (KeystoneJS) presenta soporte nativo para GraphQL por lo que su elección resultó natural.

Es importante aclarar que GraphQL puede no ser el tipo de API óptimo para arquitecturas altamente escalables. Esto se debe a que la creación de queries complejas puede ralentizar el sistema, sumado a la dificultad técnica de cachear resultados comparado con una API REST. Sin embargo, estas desventajas no resultan relevantes para satisfacer los requerimientos del proyecto dado que el producto es de uso interno y no deberá procesar gran cantidad de solicitudes.

7.2.2.3 Elección de framework

Una vez decidido el lenguaje se contempló la posibilidad de utilizar un framework para facilitar el proceso de desarrollo y poder cumplir con las estimaciones previstas. Para ello se evaluó la utilización de KeystoneJS. Se encontró que cuenta con varias características útiles:

- Generación de *resolvers* GraphQL en forma automática. Estas funciones son las encargadas de mapear queries GraphQL a llamados a la base de datos, y su creación automática implica un ahorro de tiempo significativo.
- Provisión de una sintaxis para la descripción de las entidades del dominio y sus relaciones.
- Como todo framework, permite el agregado de código mediante *hooks* para lograr una aplicación específica a partir de un patrón genérico.
- Generación automática de una interfaz de administrador que permite visualizar y actualizar la información almacenada en el sistema.

Considerando estas ventajas, se decidió utilizar el framework KeystoneJS. La única desventaja de su utilización es la curva de aprendizaje asociada. Sin embargo, esto no fue un obstáculo debido a la buena documentación existente.

7.2.2.4 Base de datos

Al haber decidido utilizar KeystoneJS, las opciones sobre qué motor de base de datos utilizar se limitaron a las siguientes:

- PostgreSQL
- MongoDB

PostgreSQL es una base de datos relacional de código abierto con buena documentación y amplio soporte de la comunidad. Ambos miembros del equipo contaban con experiencia en la utilización de bases de datos relacionales por lo que esto favorece la utilización de PostgreSQL.

Por su parte, MongoDB es una base de datos no relacional de código abierto. Su estructura permite almacenar documentos y no impone restricciones estrictas sobre el formato de los datos. El equipo no contaba con experiencia en la utilización de este tipo de bases de datos.

En conclusión, se decidió utilizar PostgreSQL debido a que el equipo ya tenía experiencia en la utilización de la tecnología.

7.3 Entidades del dominio

7.3.1 Principales *assets* y sus relaciones

Uno de los objetivos principales del proyecto es: “Construir una solución que permita una eficiente gestión de los problemas que se presenten en los equipos”. En consecuencia, las entidades del dominio más importantes son los equipos de riego y los assets instalados para su monitoreo. En el siguiente diagrama se detallan los 3 assets principales (nodo GPS, *gateway* y sensor de presión). Además se incluyen

las cardinalidades correspondientes y participaciones de cada entidad en las relaciones.

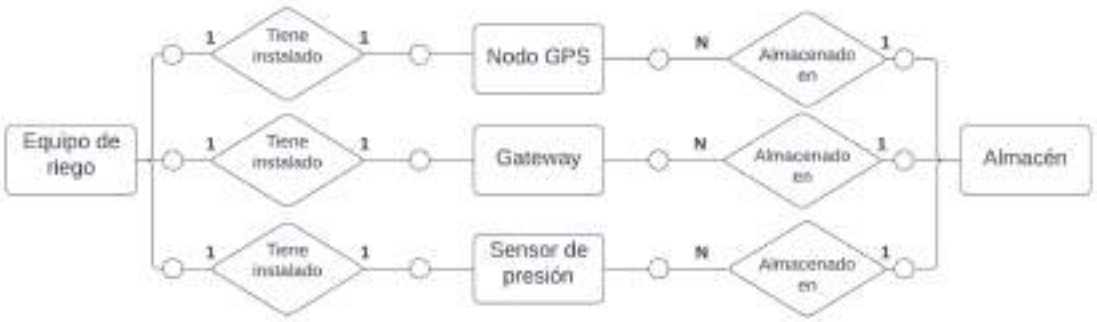


Figura 11. DER simplificado de *assets*.

Como se puede ver en el diagrama, cada equipo de riego puede tener instalado un *asset* de cada tipo. Esta situación se cumple en el caso de que el equipo sea monitoreado por la empresa. Al finalizar el contrato con el cliente, la empresa demandante debe desinstalar los *assets* del equipo de riego. En esta situación el equipo de riego queda desinstalado, por lo que ya no posee relación con ningún *asset*.

Asimismo, cada uno de los *assets* no necesariamente debe encontrarse instalado en algún equipo de riego. Por ejemplo, un *gateway* puede estar almacenado esperando a que se lo requiera para su instalación en un equipo nuevo. El inventario de un usuario técnico también es considerado un almacén.

7.3.2 Diagrama entidad relación simplificado de *issues* y solicitudes de trabajo

Es necesario destacar las entidades relacionadas a las órdenes de trabajo. Esto se debe a que las órdenes de trabajo representan tanto la reparación de los problemas que ocurren en los equipos de riego (*hardware issues*) ó la realización de solicitudes de instalación/desinstalación.

A continuación se presenta un diagrama con las principales entidades de esta categoría. El DER (Diagrama Entidad - Relación) completo se encuentra disponible en el apéndice B, al final del documento.

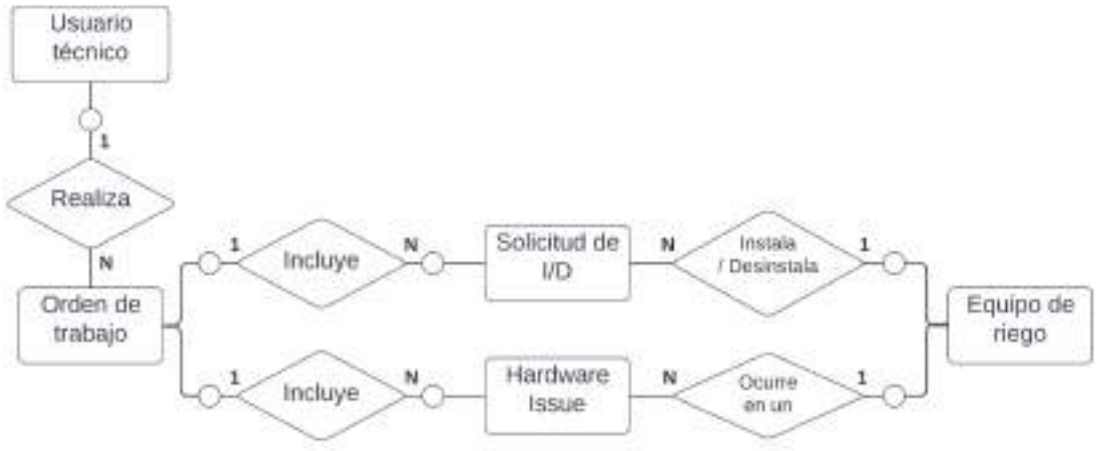


Figura 12. DER simplificado de issues y solicitudes de I/D (instalación / desinstalación).

Los usuarios tipo técnico son los encargados de llevar a cabo las órdenes de trabajo. Dentro de su jornada laboral llevan a cabo tareas que les asignaron y lo documentan junto con su recorrido en una orden de trabajo. La orden contiene datos importantes para la empresa como los kilómetros recorridos por el técnico y sus comentarios.

Por otra parte, las solicitudes de instalación o desinstalación almacenan información sobre el trabajo realizado. Se almacena qué *assets* fueron instalados para completar la solicitud o cuales fueron quitados del equipo, si fue una desinstalación. Esta información facilita el seguimiento de stock por parte de la empresa demandante.

En el caso de los *hardware issues*, se persiste un historial detallado de la falla y sus reparaciones incluyendo:

- Diagnóstico del usuario administrador que reportó la falla.
- Pericias en campo realizadas por el técnico asignado.
- Información sobre las reparaciones realizadas por el técnico.

- Evaluación de los equipos que fallaron por parte de la empresa demandante.

7.4 Seguridad informática

7.4.1 Autenticación y gestión de las sesiones de usuario

Para la autenticación de todos los usuarios se utiliza un esquema de correo electrónico y contraseña. Al momento de crear un nuevo usuario, un usuario administrador se encargará de registrarlo en el sistema. Si un usuario no recuerda su contraseña, deberá contactar a un administrador. Se considera que esta estrategia es adecuada considerando los requerimientos del demandante en cuanto a cantidad de usuarios activos. Las contraseñas no serán almacenadas sino que se almacenará su hash para evitar la exposición de las mismas ante una falla de seguridad.

La gestión de sesiones es realizada mediante *tokens* encriptados mediante una clave simétrica, por lo que no se almacena información de sesiones en el servidor. Cuando un usuario inicia sesión, su navegador obtiene un *token* válido por un período de tiempo determinado. El *token* será enviado con cada solicitud del usuario para autenticarse ante el servidor.

7.4.2 Recomendación de protocolo HTTPS

La comunicación cliente-servidor en el sistema involucra la transmisión del *token* de autenticación/autorización con cada solicitud enviada. Si se utiliza HTTP plano, el sistema es vulnerable a la intercepción del *token* por parte de agentes maliciosos. Si el agente obtuviese el *token* de un usuario legítimo, podría ejecutar acciones en su nombre (un ataque de tipo repetición).

Para evitar problemas de seguridad, se recomienda la utilización del protocolo HTTPS con encriptación TLS en la capa de transporte para la comunicación cliente-servidor. Al utilizar este protocolo, el ataque descrito anteriormente no resulta posible debido a que las solicitudes estarán encriptadas.

7.4.3 Autorización, roles y permisos

El *token* que el usuario obtiene al iniciar sesión incluye su rol dentro del sistema. Este puede ser:

- **Administrador**: tiene acceso a todo el sistema.
- **Técnico de campo**: solo tiene acceso a los *hardware issues* y solicitudes que le han asignado los administradores.

Capítulo 8: El producto

La solución se basa en una aplicación web que se ejecuta en un navegador. Permite administrar y gestionar el mantenimiento de dispositivos instalados en equipos de riego. Su principal característica es realizar el monitoreo de los dispositivos y llevar a cabo un seguimiento acerca de la resolución de sus conflictos. Los usuarios administradores tienen la capacidad de asignarle a un técnico la resolución de algún conflicto.

8.1 Características principales

8.1.1 Seguimiento del estado de transmisión de los *assets* en tiempo real

Toda empresa que cuenta con *assets* electrónicos desea reparar los que fallen en el menor tiempo posible. Para ello, Poseidón CMMS permite el seguimiento en tiempo real del estado de transmisión de los *assets*. La rápida reparación de fallas asegura una calidad del servicio superior para la empresa que utilice el producto.

En la siguiente figura se observa el panel de control de los equipos de riego, en el cual se puede visualizar el estado de transmisión de todos aquellos que se encuentran en estado **instalado**.

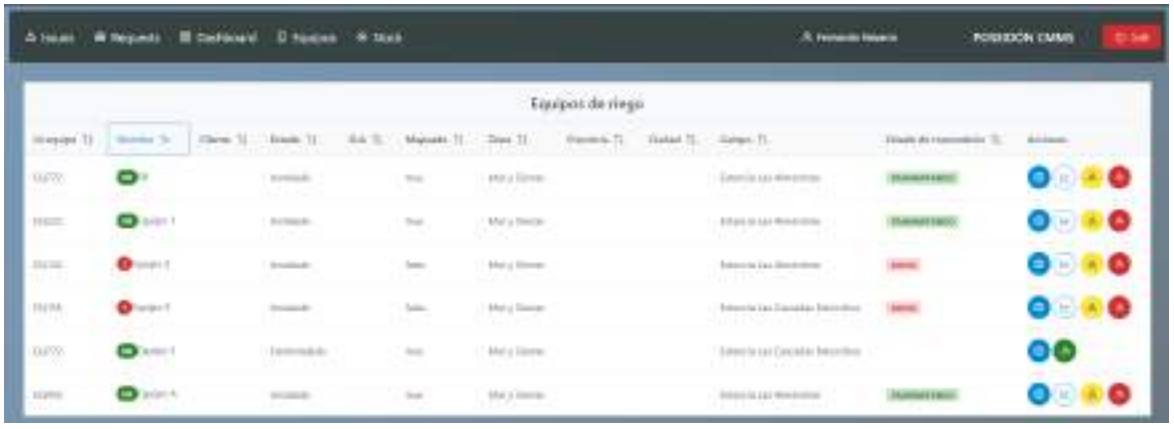


Figura 13. Monitoreo de equipos.

8.1.2 Automatización de creación de incidencias al reportarse un problema

Luego de un trabajo coordinado con desarrolladores de la empresa demandante, se integró la funcionalidad de Poseidón CMMS con su sistema de detección de errores. Las empresas con *assets* dispersos geográficamente deben monitorearlos para asegurarse de que estén operando correctamente. En caso de fallas, es necesario comenzar el proceso de asignación de técnicos y reparación lo antes posible.

Con este fin, Poseidón CMMS permite la creación automática de un *hardware issue* en el sistema ante una falla. En consecuencia, el tiempo que pasa entre que ocurre una falla hasta que es reportada en el sistema disminuye notablemente.

8.1.3 Trazabilidad mediante movimientos de stock

Las empresas que entregan equipos a sus clientes para poder prestar su servicio requieren tener un control riguroso de la ubicación de sus activos. Es por ello que Poseidón CMMS posee un módulo de control de stock.

Cada vez que se realiza una instalación o desinstalación de *assets* sobre un equipo de riego, estos cambios se ven reflejados en la ubicación geográfica almacenada de cada *asset*. Asimismo, cuando un técnico repara una falla en algún *asset* o lo reemplaza por otro, esto queda registrado en el sistema lo cual permite tener un mejor control sobre el personal técnico de la empresa.

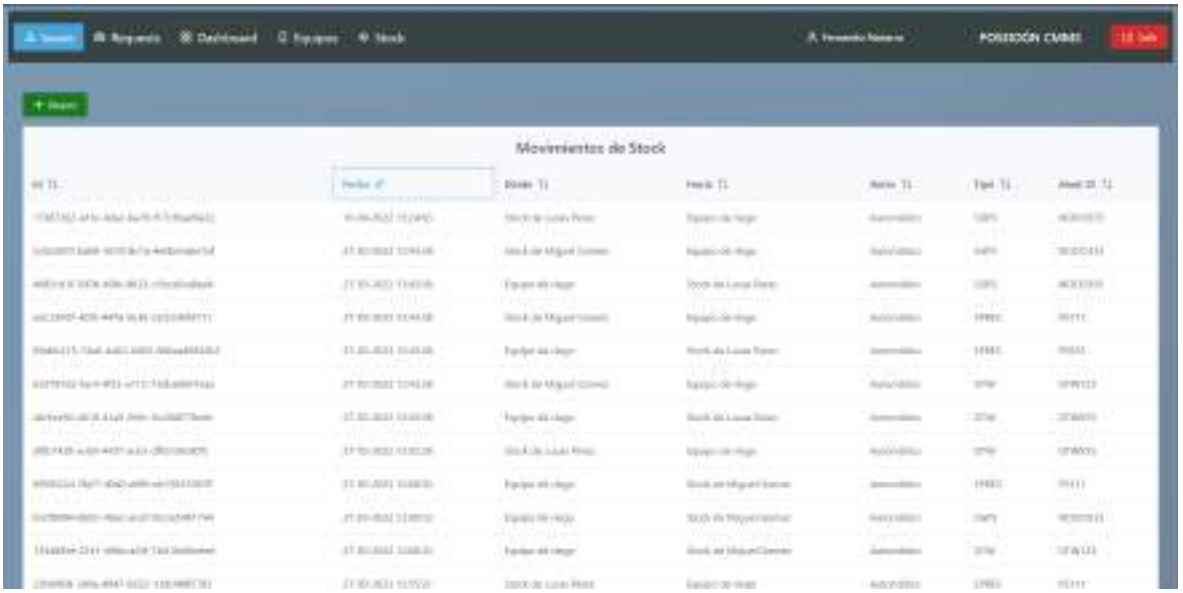


Figura 14. Movimientos de stock.

Adicionalmente, es sabido que no todos los movimientos de Stock se deben a instalaciones o reparaciones. A veces es necesario abastecer a un técnico de *assets* para que pueda cumplir con sus labores de reparación. Para cumplir con esta necesidad Poseidón CMMS cuenta con una vista de movimientos de stock desde la cual se pueden detallar envíos de *assets* y/o correcciones de stock.

8.1.4 Interfaz web para técnicos incluida dentro del sistema.

Los usuarios encargados de solucionar los problemas que se presentan en el campo son los que tienen funcionalidades restringidas. Solo pueden acceder a los módulos necesarios para realizar y reportar su trabajo diario. La información sobre su trabajo queda almacenada en el mismo sistema que gestiona las fallas, lo cual provee una observabilidad punta a punta del proceso de mantenimiento.

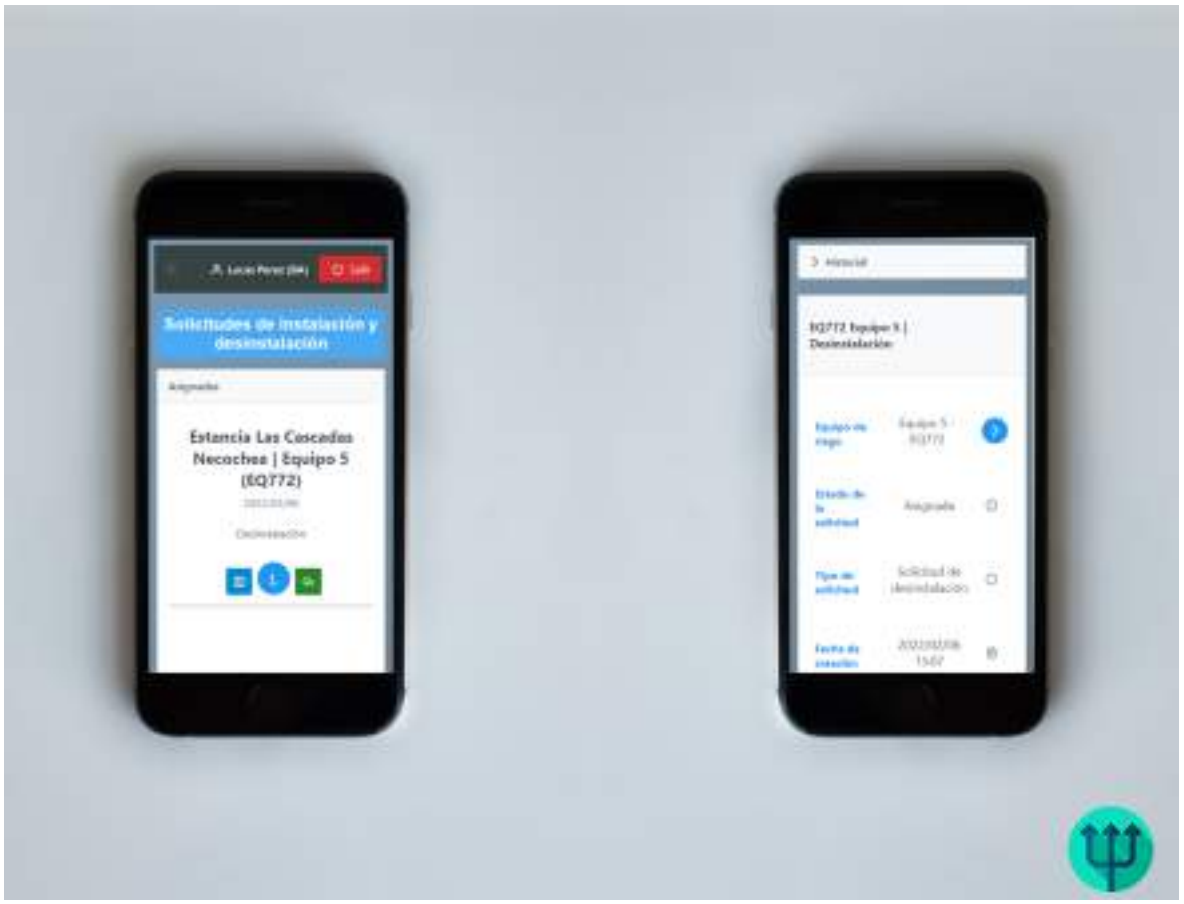


Figura 15. Interfaz web para personal técnico.

La interfaz para técnicos está optimizada para su visualización desde dispositivos móviles. La usabilidad es la máxima prioridad con el objetivo de que el personal técnico pueda informar sobre su trabajo con facilidad.

8.1.5 Mejora en los procesos: análisis de métricas mediante reportes

En el caso particular de la empresa demandante o de alguna empresa que no cuente con un sistema para almacenar y gestionar la información, la utilización del sistema conlleva una mejora en los procesos. La misma radica en que Poseidón CMMS provee reportes de distintas métricas que permiten ver el estado actual de los procesos y dan lugar a su optimización.

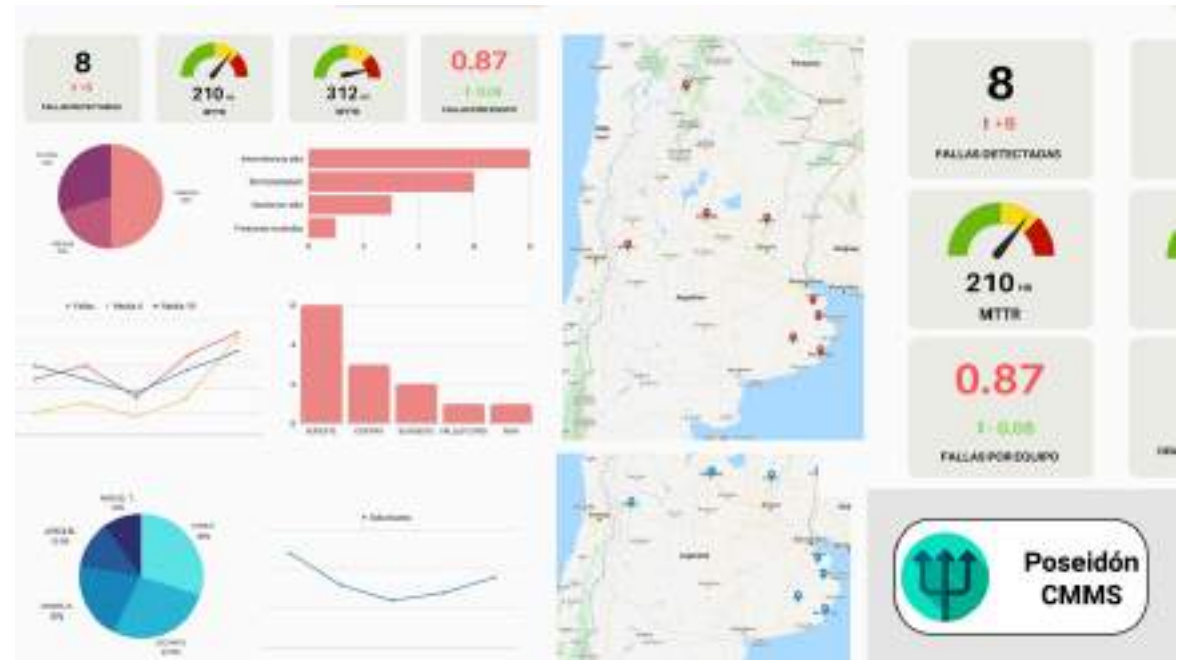


Figura 16. Vista de dashboard con métricas.

8.2. Benchmarking

A la hora de evaluar el producto, es importante realizar una comparación de los aspectos más importantes incluyendo a las alternativas existentes. En esta sección se realizará una comparación con los principales software de tipo CMMS existentes en el mercado. Estos son *Hippo CMMS* [7] y *Limble CMMS* [8].

8.2.1 Comparación de características

8.2.1.1 Documentación detallada de reparaciones

En todo sistema de gestión del mantenimiento, es importante que se detalle las fallas que han existido en los equipos a mantener. Además, la documentación de estos problemas debe ser lo más detallada posible. Con este fin, un CMMS debería permitir la asociación de comentarios e imágenes a cada falla, reparación y reemplazo de equipos.

Esta funcionalidad está presente en ambos productos de la competencia y asimismo en Poseidón CMMS, en el cual se permite la carga de imágenes, logs de los *assets* y comentarios asociados a toda reparación, instalación y reemplazo.

8.2.1.2 Gestión de stock

Otro objetivo de todo CMMS es dar visibilidad a los inventarios de la empresa. Conocer qué activos se posee y sus ubicaciones es prioritario para detectar faltantes. Ambos CMMS de la competencia ofrecen esta funcionalidad del mismo modo que Poseidón CMMS.

8.2.1.3 Distribución geográfica de los *assets* a mantener

Una característica particular del problema analizado en este proyecto es que los *assets* a mantener se encuentran distribuidos geográficamente debido a que deben instalarse en equipos de riego que pueden estar en cualquier parte del mundo.

En este punto, se detectó que las soluciones de gestión del mantenimiento existentes en el mercado se encuentran enfocadas a una cantidad reducida de ubicaciones posibles, o incluso al mantenimiento dentro de una sola locación, como por ejemplo una fábrica. Esto conlleva a que sea difícil definir técnicos por región geográfica, por ejemplo.

En cambio, Poseidón CMMS fue diseñado considerando el hecho de que los *assets* pueden encontrarse en cualquier parte, por lo que gestionar el mantenimiento con distinto personal dependiendo de la zona resulta natural. Esta característica representa una ventaja competitiva sobre la competencia.

8.2.1.4 Interfaz para técnicos y gestión de acceso

Es importante que toda la información que los técnicos pueden proveer sea almacenada dentro del sistema de gestión del mantenimiento. Se observó que *Hippo CMMS* cuenta con esta funcionalidad. Sin embargo, la misma se encuentra pensada en el contexto de un conjunto de *assets* geográficamente próximos (por ejemplo, todos en el mismo edificio). Esta característica se contrasta con el diseño

de Poseidón CMMS, el cual considera que los *assets* están distribuidos geográficamente, como describe el apartado anterior.

Por su parte, *Limble CMMS* también cuenta con una interfaz móvil. Sin embargo, el control de acceso disponible para sus usuarios técnicos es limitado dado que fue pensada considerando que los técnicos son parte de la empresa.
Poseidón CMMS cuenta con una interfaz de usuario para técnicos. El diseño del sistema permite que los técnicos, puedan ingresar al sistema la descripción del trabajo que realizaron. Su control de acceso es estricto, por lo que los técnicos sólo podrán ver los *issues* que les corresponden.

8.2.1.5 Gestión de repuestos

La gestión del mantenimiento no solo involucra el monitoreo y detección de fallas en los *assets* que se encuentren instalados, sino que también se debe tener un control de los repuestos disponibles para repararlas.

Tanto *Limble CMMS* como *Hippo CMMS* cuentan con funciones que permiten cubrir esta necesidad. Asimismo, Poseidón CMMS incorpora el almacenamiento de estos *assets* que no están siendo utilizados activamente. La funcionalidad incluye el almacenamiento de sus ubicaciones y movimientos de stock para maximizar la trazabilidad de los inventarios.

8.2.1.6 Precio

Con respecto al costo que conlleva el uso, los productos ya existentes fijan su precio mensual según la cantidad de usuarios del sistema. Se debe contar como usuario tanto a técnicos como a administradores. El precio de cada uno de ellos es en promedio USD 40 por usuario por mes.

El costo de operación de Poseidón CMMS será el costo de escalar la infraestructura existente en la empresa demandante para acomodar los componentes descritos en la arquitectura. Estos son una base de datos PostgreSQL y un servidor NodeJS para servir tanto la API como el frontend.

8.2.2 Resumen

A partir de la comparación realizada en la sección anterior, surge el siguiente cuadro resumen:

	Hippo CMMS	Limble CMMS	Poseidón CMMS
Documentación detallada de fallas	Sí	Sí	Sí
Gestión de stock	Sí	Sí	Sí
Interfaz para técnicos	Sí, pero para técnicos internos considerados confiables.	Sí, pero para técnicos internos considerados confiables.	Sí, con control de acceso para que cada técnico solo vea las fallas sobre las que debe actuar.
Adaptabilidad a distribución geográfica	No	No	Sí
Gestión de repuestos	Sí	Sí	Sí
Precio	USD 35 / usuario	USD 40 / usuario	Costos de instalación y deployment

Para concluir, la ventaja competitiva del producto consiste en que fue diseñado pensando en el mantenimiento de dispositivos que están distribuidos geográficamente, a diferencia de la competencia que está pensada principalmente para el mantenimiento de maquinaria ubicada en unas pocas locaciones geográficas. Este factor sumado al estricto control de acceso para técnicos y su bajo costo de operación son los factores a destacar por sobre los productos existentes en el mercado.

Capítulo 9: Memoria del proyecto

9.1 Cumplimiento de objetivos

En esta sección se discutirá el grado de cumplimento logrado en cada objetivo. Cada subtítulo corresponde a un objetivo planteado.

9.1.1 Obtener información y entender el dominio del problema, mediante el análisis de los flujos de trabajo pertinentes en la empresa demandante.

Gracias a la metodología de trabajo utilizada, se mantuvieron reuniones constantes con la empresa demandante del sistema. Durante ellas se pudo obtener información sobre el dominio del problema y conocer la operatoria existente en la empresa con el fin de entender la forma de trabajo y diseñar una solución que se adapte a sus necesidades. Finalmente, este objetivo fue cumplido en su totalidad.

9.1.2 Trabajar de manera conjunta con un alumno de Ingeniería Industrial mediante proyectos finales relacionados.

Durante la ejecución de todo el proyecto se mantuvo un trato constante con el proyecto final paralelo. De esta colaboración surgió mucha información del dominio que permitió avanzar más rápido en el proceso de análisis. Además, otro de los productos del trabajo en conjunto fue el reporte de métricas relacionadas al mantenimiento. La selección de métricas es una parte del análisis que se obtuvo colaborando con el proyecto paralelo. Posteriormente, el diseño del sistema fue pensado para almacenar datos de forma que el producto pueda mostrar estas métricas.

9.1.3 Diseñar una solución de gestión del mantenimiento que facilite el trabajo del personal operativo y técnico de la empresa que utilice el sistema.

Se consideraron las características solicitadas por el personal de la empresa y se obtuvo un diseño que permite implementar las funcionalidades solicitadas por el demandante. En conclusión, el objetivo fue cumplido.

9.1.4 Construir una solución que permita una eficiente gestión de los problemas que se presenten en los equipos.

El proyecto incluyó la implementación de un producto a partir del análisis y diseño realizados.

Durante el proceso de implementación se utilizó un modelo de integración continua/despliegue continuo. Gracias a este proceso, se mantuvo disponible una URL de prueba del producto para que los referentes funcionales pudieran validarlo. Además, el entorno sirvió para realizar pruebas en un ambiente remoto por parte del equipo.

Al finalizar, se obtuvo una solución que permite la gestión de los problemas en los equipos, además de control de stock y la funcionalidad descrita por los capítulos anteriores. En conclusión, este objetivo fue cumplido.

9.1.5 Facilitar el almacenamiento de información sobre la ubicación de los dispositivos de la empresa.

La solución desarrollada permite almacenar la ubicación actual de todos los *assets* de la empresa que la utilice. La ubicación de un *asset* puede ser: instalado en algún equipo; en posesión de un técnico; almacenado en un depósito, pero siempre es conocida. En consecuencia, se puede afirmar que este objetivo fue cumplido.

9.1.6 Integrar la detección automática de fallas en los dispositivos para minimizar su tiempo no operativo.

El diseño planteado en el proyecto contempla la integración con cualquier servicio de detección de fallas mediante una interfaz genérica adaptable a cualquier sistema externo. El diseño implica que un sistema externo tenga la posibilidad de notificar acerca de una falla, de manera que la creación de *hardware issues* sea automática. Por lo tanto, se cumplió con este objetivo.

9.2 Trabajo interdisciplinario

La idea inicial de crear un trabajo interdisciplinario surgió debido a que el equipo de trabajo se dió cuenta de la existencia de un proyecto de Ingeniería Industrial sobre la misma temática que se iba a abordar.

Dada esta situación, se propuso el trabajo con la idea de unir conocimientos de ambas disciplinas y poder lograr mediante esta colaboración un producto mejor. Realizar proyectos de manera interdisciplinaria ayuda a obtener experiencia en actividades más parecidas al mundo laboral. De esta forma, cada uno puede darle distinto enfoque a la solución del problema.

Al proponer la idea en el entorno académico, se nos informó que ambos proyectos debían documentarse en forma independiente. Esto significó que los informes de proyecto serían completamente distintos, pero no impidió que la colaboración entre ambos sea total.

Durante la ejecución del proyecto, se llevaron a cabo reuniones para realizar trabajo de manera coordinada, por ejemplo análisis de métricas. Durante las mismas se logró exponer de manera correcta las ideas de cada uno y obtener resultados.

Desde el proyecto de Ingeniería Informática se pudo aprovechar el análisis de selección de métricas del proyecto paralelo. El proyecto de Ingeniería Industrial, a cambio, pudo ver que el resultado de su selección fue incluido dentro de un producto real con uso en una empresa.

Adicionalmente, el hecho de que el equipo del proyecto de Ingeniería Industrial trabaje en la empresa demandante significó que también cumplieron un rol de segundo referente funcional. En consecuencia, el proceso de análisis y diseño del sistema se pudo llevar adelante con mayor facilidad.
En conclusión, el trabajo interdisciplinario resultó ventajoso para ambos proyectos finales y su implementación también fue un éxito lo cual puede observarse en los resultados.

9.3 Metodología de trabajo

9.3.1 Metodología de análisis

La metodología de análisis planteada para el proyecto consistía en analizar el problema por etapas en las que cada etapa se correspondía con un módulo de funcionalidad.

En los comienzos de la realización del proyecto ya se había tenido en cuenta una etapa extendida de análisis inicial del dominio (Etapa 1). Lo que terminó ocurriendo durante esta primera etapa fue un análisis general de todo el dominio del problema. No se profundizó en mucho detalle en cada módulo, pero la complejidad del dominio sumada a la poca experiencia del equipo de trabajo dentro del mismo hizo necesario un breve aprendizaje.

Además, durante la elaboración del protocolo de trabajo final se obtuvo mucha información del dominio que resultó útil para la realización de todo el proyecto. Este lapso de análisis previo a la entrega del protocolo de trabajo final no se encuentra dimensionado dentro de las estimaciones del proyecto. Sin embargo, fue imprescindible para ampliar los conocimientos del equipo en el dominio del problema.

9.3.2 Metodología de diseño

Al igual que el análisis, el proceso de diseño durante el proyecto originalmente fue planteado en etapas, con una etapa inicial extendida para diseñar la arquitectura del sistema. Este cronograma se cumplió acorde a lo esperado.

Durante el proceso de diseño también surgieron dudas sobre entidades del dominio que pudieron ser resueltas mediante la comunicación con el referente funcional. En conclusión, se pudo cumplir con la metodología de diseño modular propuesta inicialmente, por lo que se considera que fue una buena elección.

9.3.3 Metodología de desarrollo / testing

En los comienzos del proyecto, se propuso la utilización de la metodología Kanban además de la separación de la implementación y pruebas en etapas agrupadas con su análisis y diseño correspondientes. Se puede afirmar que esta separación fue cumplida. Sin embargo, no se cumplió el orden esperado de las etapas. Esto se debe a que las etapas 3 y 4 dependían de la funcionalidad a implementar en las etapas 5 y 6, por lo que su orden fue alterado.

Adicionalmente, se propuso la implementación de un proceso de integración continua para el producto en desarrollo. Este proceso fue exitoso y permitió la evaluación constante del producto por parte de la empresa demandante.

9.4 Métricas

9.4.1 Estimaciones iniciales del proyecto

Para detallar las estimaciones de tiempo iniciales planificadas para el proyecto, se incluye el diagrama de Gantt entregado en el protocolo de proyecto final.

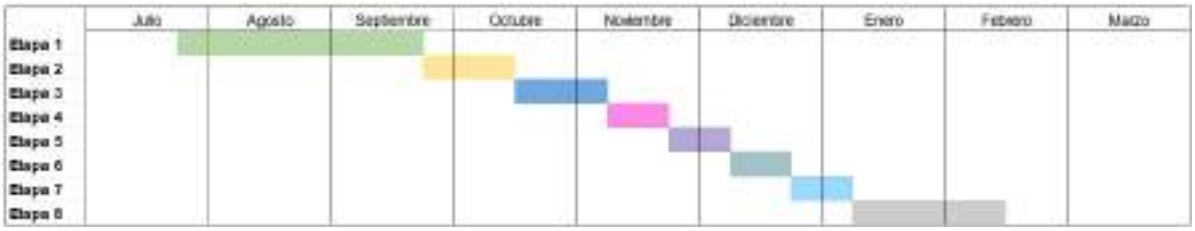


Figura 17. Estimaciones iniciales.

- *Etapa 1: Relevamiento inicial del dominio, requerimientos y elaboración del protocolo.*
  - *Duración: 8 semanas*
- *Etapa 2: Análisis, diseño, implementación y testeo del MVP (minimum viable product) que incluya seguimiento del estado de los assets en tiempo real.*
  - *Duración: 3 semanas*
- *Etapa 3: Análisis, diseño, implementación y testeo del stock de dispositivos.*

- *Duración: 3 semanas*

- *Etapa 4: Análisis, diseño, implementación y testeo de funcionalidad de archivos de registro/ carga de documentos.*
  - *Duración: 2 semanas*

- *Etapa 5: Análisis, diseño, implementación y testeo del módulo de gestión de problemas de hardware. Incluyendo el estado en el que se encuentra (Diagnóstico, Pericia, Análisis forense)*
  - *Duración: 2 semanas*

- *Etapa 6: Análisis, diseño, implementación y testeo del seguimiento de órdenes de trabajo/ service management.*
  - *Duración: 2 semanas*

- *Etapa 7: Análisis, diseño, implementación y testeo del módulo de gestión de contratos de servicio con los clientes. Incluye nivel de servicio contratado y relación con equipos.*
  - *Duración: 2 semanas*

- *Etapa 8: Integración final y pruebas de sistema*
  - *Duración: 5 semanas*

Figura 18. Detalle de las etapas iniciales planteadas en el protocolo.

A la hora de entregar este diagrama, no se detalló la cantidad de horas semanales que se planificaba dedicar al proyecto. Este dato resulta de suma importancia y sin él no se pueden hacer estimaciones de horas. Se puede afirmar que se esperaba emplear una dedicación semanal de 14 horas. Este resultado se obtiene al dividirlas en 3 jornadas de 3 horas en días de semana y una sesión de 5 horas durante el fin de semana.

Considerando los datos expuestos previamente, se obtiene una dedicación total de 266 horas por integrante, dado que se estimaron 14 horas de proyecto semanales por miembro del equipo durante 19 semanas.

A la hora de planificar, se estimó los tiempos a emplear en el análisis, diseño, implementación y pruebas de cada etapa como una unidad a resolver. Por ello, las estimaciones fueron a nivel funcionalidad del sistema.

La planificación implicaba mantener un ritmo de avance constante en el proyecto. Asimismo, no se pensó en ningún tiempo de contingencias por imprevistos que pudieran surgir como festividades o tiempos de reposo por enfermedad.

9.4.2 Tiempos reales

Durante el transcurso del proyecto, se realizó un seguimiento de los tiempos utilizados mediante una planilla. En ella se puede observar los tiempos de ejecución reales de cada etapa del proyecto. A partir de estos datos, en la siguiente sección serán comparados los tiempos esperados contra el tiempo empleado en cada etapa.

Figura 19. Tiempos reales de cada etapa.

La fecha oficial de finalización del proyecto fue el 15 de marzo. Esta fecha se corresponde con el día en el que fue firmada la nota de finalización.

9.4.3 Comparación y verdadero orden de las etapas

A continuación se presenta una comparación entre los tiempos esperados para cada etapa y su verdadera duración. Las barras huecas simbolizan el tiempo previsto que no fue utilizado en esa etapa.

Adicionalmente, es importante hablar de que el orden de las etapas decidido durante la planificación del proyecto fue considerado erróneo y no fue respetado durante su ejecución. El motivo por el cual se tomó esta decisión consiste en que las entidades a analizar y representar en las etapas 3 y 4 resultaban muy dependientes de lo analizado en las etapas 5 y 6, por lo que resultó conveniente cambiar el orden de ejecución para evitar el costo de tiempo resultante de lidiar con esta dependencia. El orden final de las etapas fue 1, 2, 5, 6, 3, 4, 7 y 8.

La fecha de entrega esperada originalmente era el 9 de febrero, pero como se puede ver la verdadera fecha de finalización del proyecto fue el 15 de marzo. Esta diferencia puede explicarse debido a las demoras que se detallan en cada etapa.

Figura 20. Diagrama que compara tiempos estimados con tiempos reales.

9.4.3.1 Resumen

A continuación se detalla lo mencionado en el subtítulo anterior en forma de un gráfico de tiempo total planificado (en horas totales) contra tiempo utilizado realmente.

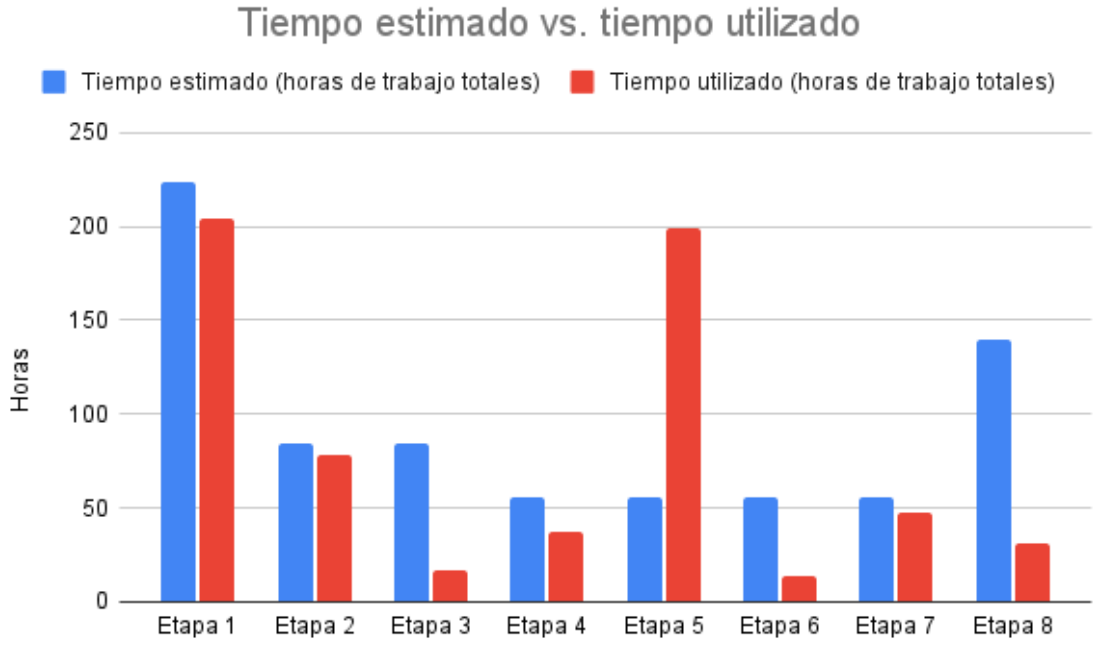


Figura 21. Diagrama que compara tiempos estimados con tiempos reales.

9.4.3.2 Etapa 1

En la etapa 1 se previó realizar un análisis inicial del dominio y desarrollar el protocolo. Dado el nulo conocimiento del equipo sobre el contexto agrícola en el que se encuentra el proyecto, esta etapa demoró 6 semanas más de lo planificado. Durante estas 6 semanas se trabajó por un total de 90 horas

También se puede citar las revisiones del protocolo de trabajo final que debió ser corregido varias veces antes de ser entregado como otra causa de las demoras.

9.4.3.3 Etapa 2

Habiendo solventado esta etapa, se pasó a la etapa 2, donde el objetivo fue el análisis, diseño, implementación y pruebas del módulo de seguimiento de *assets*.

Para esta etapa se cumplió con la estimación propuesta en semanas. Según la información registrada durante el proyecto, en estas semanas se dedicaron 39.3 horas por miembro del equipo. En conclusión, el uso de horas semanales real fue de 13.1 horas por miembro del equipo. Este valor resulta levemente menor al planificado de 14 horas pero a pesar de esto la duración en semanas de la etapa fue correcta.

9.4.3.4 Etapa 5

A continuación, según lo descrito previamente sobre el orden de las etapas, se procedió a la etapa 5. Esta etapa resultó clave ya que se analizaron en profundidad los detalles de una de las entidades más importantes dentro del proyecto, los *hardware issues*. Lo que resultó al concluir esta etapa, fue el análisis, diseño implementación y pruebas del módulo principal del sistema. Esto incluye entidades como el diagnóstico, la pericia, la reparación y la autopsia sobre un *hardware issue*.

Según lo registrado en los archivos del proyecto, se dedicaron un total de 99.7 horas por integrante a esta etapa a lo largo de las 5 semanas. Es decir, un total de 20 horas por semana por integrante, comparadas con las 14 horas semanales

planificadas. Esto excede la estimación realizada al comienzo del proyecto pero resulta razonable dada la complejidad del módulo en cuestión.

9.4.3.5 Etapa 6

Luego de la etapa 5 se procedió a la etapa 6, órdenes de trabajo y service management. Este tema resulta muy importante para el proyecto dado que son las segundas entidades más importantes, después de los *hardware issues*. Durante esta etapa también se trabajó con solicitudes de instalación o desinstalación de *assets* sobre un equipo de riego.

Su ejecución se emplearon 7 horas por integrante durante una semana. Comparando con lo previsto el tiempo resulta mucho menor a lo esperado. Se puede razonar que esto se debe a que las entidades en cuestión son similares a los *hardware issues* por lo que se pudo reutilizar parte del trabajo realizado en etapas previas.

9.4.3.6 Etapa 3

A continuación, se procedió con la realización de la etapa 3: gestión de stock. Para la misma se tuvo que revisar el análisis y diseño previo para modelar la ubicación de cada *asset*. Además, se agregó al producto una vista de movimientos de stock destinada a poder visualizar rápidamente los movimientos tanto automáticos como manuales y crear movimientos nuevos.

Para esta etapa se contempló originalmente una duración de 3 semanas con una dedicación de 14 horas semanales por miembro del equipo. En la práctica, se utilizaron 8.35 horas por miembro del equipo a lo largo de una sola semana. Este error de estimación resulta muy grande. Se puede fundamentar este error en la sobreestimación de la complejidad del almacenamiento de ubicaciones por parte del equipo. En la práctica este módulo no resultó complejo por lo que el tiempo utilizado fue mucho menor.

9.4.3.7 Etapa 4

Posteriormente, se abordó el tema de la carga y almacenamiento de archivos para todas las entidades del dominio (etapa 4). Durante esta etapa se analizó cuáles archivos podían incorporarse en cuáles entidades (por ejemplo, imágenes en una reparación) y se procedió al diseño de los modelos y su implementación.

La duración planificada de esta etapa fue de 2 semanas con 14 horas semanales por miembro. En la práctica, se utilizó solo una semana pero con una cantidad semanal de horas mayor a la estimada. Se utilizaron 18.5 horas semanales por miembro del equipo. La estimación general de todas formas resulta menor dado que la cantidad de horas planificadas eran 28. Se puede considerar que esto se debe a que la funcionalidad en cuestión no resultó compleja y en consecuencia, demoró menos tiempo de lo esperado. En lo que respecta al producto, solo hubo que agregar los campos correspondientes a los formularios, almacenar lo enviado en el servidor y agregar las validaciones necesarias.

9.4.3.8 Etapa 7

Posteriormente, comenzó la etapa 7 del proyecto con la funcionalidad relacionada a la gestión de contratos.

Para la misma se utilizaron 23.9 horas de cada miembro del equipo en total, a lo largo de una sola semana. Esto contrasta con la estimación de 2 semanas a 14 horas por miembro por semana, para un total de 28 horas planificadas. Nuevamente, nuestra estimación resultó menor al tiempo planeado. Se puede considerar como una posible causa que el conocimiento del dominio por parte del equipo ya era extenso por lo que solo hizo falta hacer averiguaciones menores y pasar directamente a la implementación, con casi todo el análisis y diseño resuelto.

9.4.3.9 Etapa 8

Finalmente, en la etapa 8 se realizaron las pruebas finales y corrección de errores. Durante esta etapa se realizaron pruebas generales del producto con el objetivo de encontrar defectos y mejoras.

El tiempo empleado en esta etapa fueron 15.5 horas totales por miembro a lo largo de las 5 semanas. La estimación previa fue de 5 semanas pero empleando 70 horas en total por cada integrante. Se considera que esta diferencia se debe a que la validación constante realizada a lo largo de todo el proyecto sirvió como una base para poder realizar las validaciones finales en un lapso de tiempo menor al planificado.

Capítulo 10: Conclusión

Al comienzo del proyecto se plantearon una serie de objetivos con la finalidad de resolver el problema de la gestión del mantenimiento para empresas que se encargan de monitorear maquinaria agrícola mediante dispositivos IoT. Ahora que el proyecto ha finalizado, podemos afirmar que se logró diseñar un producto que satisface las necesidades de la empresa demandante y puede ser adaptado a otras empresas del mismo rubro. Durante el proyecto se obtuvo mucho conocimiento acerca del dominio del problema, el cual fue necesario para entender los requerimientos del demandante.

Además, el proyecto tuvo como uno de sus principales focos la colaboración interdisciplinaria con un proyecto paralelo de Ingeniería Industrial. En retrospectiva, se puede afirmar que esta colaboración resultó fructífera dado que permitió un relevamiento más eficiente de los requerimientos y una validación constante de los resultados del proyecto. El trabajo en conjunto resultó especialmente efectivo debido a que el equipo del proyecto paralelo de Ingeniería Industrial estaba compuesto por un empleado de la empresa demandante.

En cuanto a los tiempos de entrega, hubo una demora que se puede atribuir a errores en las estimaciones propios de la falta de experiencia. Sin embargo, se considera que esta experiencia resulta didáctica para el refinamiento de estimaciones en proyectos futuros.

Durante el proyecto, se pudo experimentar aspectos de la Ingeniería en Informática que son difíciles de replicar en el entorno académico. En particular, estos aspectos que se pudo reforzar fueron: la interacción con el cliente, el análisis de requerimientos en el contexto de un dominio desconocido, la estimación temporal de ejecución de actividades en un proyecto y su posterior revisión.

Por todo lo descrito anteriormente, se considera que el proyecto fue un éxito desde el punto de vista del producto, pero por sobre todo una gran experiencia de aprendizaje para los alumnos.

10.1 Trabajos futuros

En esta sección se detallarán varias funcionalidades que podrían agregarse al producto y no fueron agregadas por no estar incluidas dentro de la planificación del proyecto.

10.1.1 Sistema de sueldo para técnicos por campaña

El producto permite la gestión de técnicos como un único grupo, es decir no existen subtipos de técnico. Sin embargo, en la empresa demandante existen técnicos con dos modalidades diferentes:

- Por campaña: son contratados una sola vez por un monto estipulado en el contrato y realizan una cantidad indefinida de trabajos. El monto depende de la cantidad de equipos en su zona geográfica.
- Por trabajo: no están bajo un régimen estricto y se les paga un monto fijo por cada trabajo que hacen.

En el producto se llevará un seguimiento de los trabajos realizados por todos los técnicos. Por lo tanto, se cuenta con suficiente información para agregar al producto un módulo de cálculo de pagos a técnicos. Asimismo, en el caso de los técnicos de campaña se podría calcular automáticamente el monto a pagar según cuantos equipos de riego residen en su zona geográfica.

10.1.2 Sistema de control para los técnicos que trabajan por campaña

En el caso de los técnicos contratados por campaña descritos en el subtítulo anterior, resulta útil calcular métricas de los técnicos para poder comparar sus rendimientos. Esta funcionalidad se puede implementar dado que el producto almacena información sobre todas las reparaciones de todos los técnicos.

10.1.3 Integración con CRMs para los contratos

En la actualidad, el sistema cuenta con una gestión de contratos realizada por los usuarios administradores. Una integración con softwares del tipo CRM (del inglés *Customer Relationship Management*, Gestión de relaciones con los clientes) permitiría la creación automática de contratos en Poseidón CMMS cuando estos sean cargados. De los contratos también se podría inferir parámetros importantes como el nivel de servicio contratado para cada equipo de riego.

Apéndices

Apéndice A: Glosario

A.1 CMMS

Un sistema computarizado de gestión de mantenimiento o CMMS (del inglés *Computerized maintenance management system,* sistema de gestión del mantenimiento asistido por computadora) es un software que centraliza la información de mantenimiento y facilita los procesos de las operaciones de mantenimiento. La información ayuda a que los técnicos de mantenimiento puedan realizar efectivamente su trabajo (por ejemplo, determinando qué máquinas requieren mantenimiento y qué dispositivos deben ser reemplazados). Otra característica de este tipo de sistemas es que los datos que almacenan pueden ayudar al área de administración a tomar decisiones (por ejemplo, a través de métricas).

Para poder llevar a cabo un correcto mantenimiento de los activos de una empresa, se requiere tener conocimiento sobre su estado actual. La información es requerida para analizar qué está ocurriendo. Sin la utilización de un sistema de gestión, esta tarea requeriría emplear mucha cantidad de esfuerzo y tiempo en tareas manuales para organizar la información.

Un CMMS puede generar reportes y documentos dando detalles de actividades de mantenimiento. Mientras más sofisticado sea el sistema, los análisis serán más extensos [9].

A.2 API

Una API (del inglés *Application Programming Interface*, interfaz para programación de aplicaciones) es una abstracción de un problema que especifica cómo los clientes deberían interactuar con componentes de software que implementan una solución a este problema. En esencia, las APIs definen bloques reutilizables que

permiten la incorporación de piezas modulares de funcionalidad a aplicaciones para usuarios finales. [10]

En el contexto de Poseidón CMMS, esta definición puede aplicarse considerando que el problema a resolver es la gestión del mantenimiento de activos. Asimismo, el producto presentará una API orientada a resolver este problema. Para lograr este objetivo se debe decidir su tipo, por lo que a continuación se detallarán los dos tipos de API entre los que se tomó la decisión.

A.2.1 REST

Uno puede ver al sistema distribuido como una gran colección de recursos que son gestionados individualmente por componentes. Los recursos pueden crearse o eliminarse por aplicaciones remotas, y asimismo pueden ser consultados y modificados. Este enfoque ha sido ampliamente adoptado para la Web y es conocido como REST (del inglés *Representational State Transfer,* transferencia de estado representacional). Hay cuatro características claves en lo que se conoce como arquitecturas RESTful:

1. Los recursos son identificados a través de un único esquema de nombrado.
2. Todos los servicios proveen la misma interfaz, la cual debe consistir de cuatro operaciones como máximo.
3. Los mensajes enviados o recibidos por un servicio son completamente autodescriptivos.
4. Luego de ejecutar una operación en un servicio, ese componente no guarda información sobre el cliente. [10]

En el contexto del producto, se utiliza un solo endpoint tipo REST como una vía de integración para sistemas externos desde la cual se permite informar sobre fallas detectadas en los equipos de riego. Todo el resto de la API sigue el paradigma GraphQL descrito en la siguiente sección.

A.2.2 GraphQL

GraphQL es un lenguaje de consulta para APIs y un entorno de ejecución para satisfacerlas con los datos existentes. GraphQL permite a los clientes solicitar

exactamente lo que necesitan y de este modo evitar traer datos no utilizados. Además, el lenguaje de consulta es fuertemente tipado [11].

GraphQL es un proyecto creado originalmente por Meta (ex Facebook) para uso interno. En el año 2015 fue liberado como código abierto para uso público.

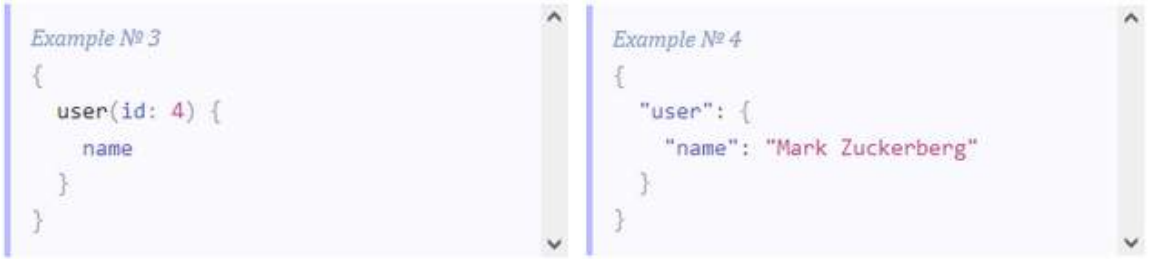


Figura 22. Consulta GraphQL solicitando el nombre del usuario con id 4. A la derecha se muestra la respuesta. [12]

En el producto se optó por la utilización de GraphQL debido a su presencia como nueva tecnología para el desarrollo de APIs, además de la facilidad que otorga para consultar datos. Adicionalmente, varias de las herramientas cuya utilización estaba planeada tenían soporte nativo para GraphQL lo cual también ayudó a tomar la decisión.

A.3 Framework

Un *framework* (del inglés marco de trabajo) es una abstracción en la que se agrega código escrito por el usuario sobre una base de software con funcionalidad genérica. De este modo se logra un software específico. Un framework facilita el desarrollo de software mediante la provisión de módulos reutilizables y un flujo de control manejado por el framework.

A.3.1 Frameworks para clientes web

En el contexto de los clientes web, es común el uso de frameworks para facilitar el desarrollo y permitir la entrega de valor en lapsos de tiempo menores. Se describirán dos de los principales frameworks web discutidos durante este trabajo final.

A.3.1.1 React

React es una biblioteca JavaScript para crear interfaces de usuario [13]. Fue creada por Meta (ex Facebook) en 2013 y es actualmente mantenida por ellos y la comunidad open source.

ReactJS cuenta con su propia extensión al lenguaje JavaScript llamada JSX. JSX permite describir componentes de una página web en un formato similar al XML, lo cual lo hace fácil de aprender para desarrolladores con experiencia previa.

A.3.1.2 VueJS

VueJS es un framework versátil y performante para la construcción de interfaces web [14]. A diferencia de algunos de sus competidores como React, VueJS no cuenta con un dialecto propio de JavaScript sino que utiliza JavaScript clásico. Esto lo hace más fácil de aprender para desarrolladores nuevos.

Además, VueJS cuenta con un sistema de variables reactivas que permiten la actualización de las interfaces en forma automática cuando alguna de ellas cambia.

A.3.2 Frameworks para servidores

En el contexto de los servidores también existen frameworks que permiten simplificar procesos recurrentes y genéricos para lograr minimizar los tiempos de desarrollo. En este caso se describe un framework que cobra relevancia en el contexto de este trabajo final.

A.3.2.1 KeystoneJS

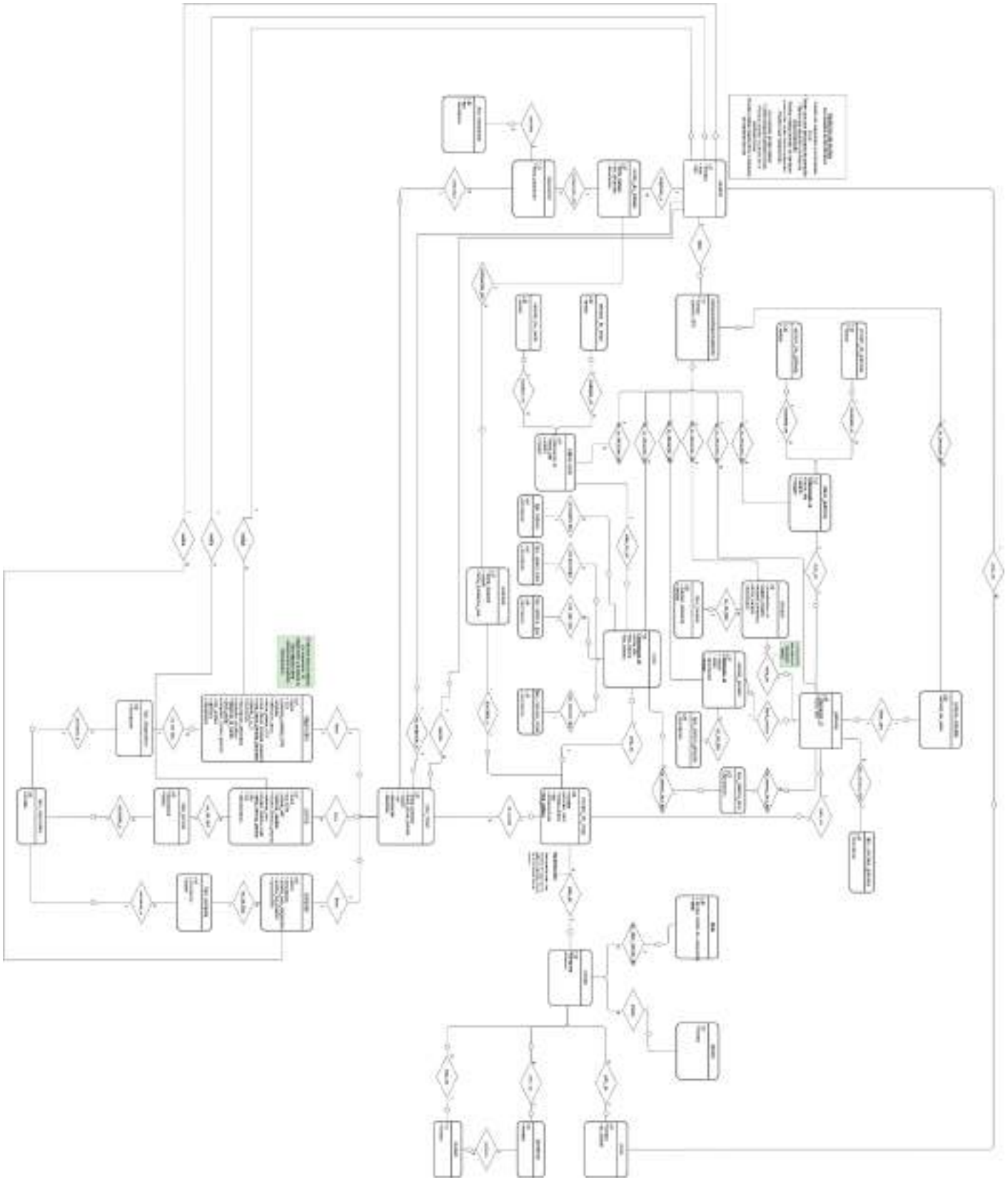
KeystoneJS es un framework de desarrollo de servidor que utiliza el lenguaje JavaScript y permite la creación de backends de APIs GraphQL a partir de la descripción de las entidades del dominio y sus relaciones.

A.4 Google Data Studio

Google Data Studio es una herramienta gratuita que permite crear reportes informativos creados por el usuario a partir de datos obtenidos de fuentes variadas. Un ejemplo de una fuente de datos aceptada por Google Data Studio son las tablas

de una base de datos relacional [15]. Además, permite integrar un reporte dentro de una página web existente.

Apéndice B: DER completo



Apéndice C: Requerimientos funcionales completos

Identificación del requerimiento:	RF01
Nombre del Requerimiento:	Monitoreo en tiempo real de sensores de cada equipo de riego.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario debe contar con permisos de administrador.
Características:	El sistema debe permitir visualizar el estado de transmisión de los sensores de un equipo de riego, detallando qué equipos están transmitiendo normalmente y cuáles están teniendo fallas.

Identificación del requerimiento:	RF02
Nombre del Requerimiento:	Seguimiento de los equipos de riego.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario debe contar con permisos de administrador.
Características:	El sistema debe permitir visualizar todos los sensores y clientes asociados a un determinado equipo de riego.

Identificación del requerimiento:	RF03
Nombre del Requerimiento:	Cálculo de métricas de desempeño.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario debe contar con permisos de administrador.
Características:	El sistema debe calcular métricas de desempeño, propias del contexto del funcionamiento y mostrarlas al usuario.

Identificación del requerimiento:	RF04
Nombre del Requerimiento:	Control de stock de *assets*.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario debe contar con permisos de administrador.
Características:	El sistema debe permitir visualizar la ubicación de los *assets* de la empresa. Además, el sistema debe permitir conocer la cantidad de *assets* de cada ubicación, discriminando por tipo.

Identificación del requerimiento:	RF05

Nombre del Requerimiento:	Seguimiento de fallas y órdenes de trabajo.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario debe contar con permisos de administrador.
Características:	El sistema debe permitir la detección de fallas de transmisión de los *assets* y proveer los medios para realizar un seguimiento del problema, que incluye su asignación a un técnico y su resolución.

Identificación del requerimiento:	RF06
Nombre del Requerimiento:	Control de nivel de servicio para cada cliente.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe dar la posibilidad de asociar a cada equipo un nivel de servicio contratado.

Identificación del requerimiento:	RF07
Nombre del Requerimiento:	Alta, baja, modificación y consulta de equipos de riego.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de equipos de riego, alteraciones a equipos existentes, relaciones con las entidades correspondientes y consulta de equipos ya almacenados.

Identificación del requerimiento:	RF08
Nombre del Requerimiento:	Alta, baja, modificación y consulta de *gateways* (GTW).
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de *gateways*, alteraciones a *gateways* existentes, relaciones con las entidades correspondientes y consulta de *gateways* ya almacenados.

Identificación del requerimiento:	RF09

Nombre del Requerimiento:	Alta, baja, modificación y consulta de nodos.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de nodos GPS, alteraciones a nodos existentes, relaciones con las entidades correspondientes y consulta de nodos ya almacenados.

Identificación del requerimiento:	RF10
Nombre del Requerimiento:	Alta, baja, modificación y consulta de campos.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de campos, alteraciones a campos existentes, relaciones con las entidades correspondientes y consulta de campos ya almacenados.

Identificación del requerimiento:	RF11
Nombre del Requerimiento:	Alta, baja, modificación y consulta de solicitudes de instalación / desinstalación.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de solicitudes, alteraciones a solicitudes existentes, relaciones con las entidades correspondientes y consulta de solicitudes ya almacenadas.

Identificación del requerimiento:	RF12
Nombre del Requerimiento:	Alta, baja, modificación y consulta de técnicos de campo.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de técnicos de campo, alteraciones a técnicos existentes, relaciones con las entidades correspondientes y consulta de técnicos ya almacenados.

Identificación del requerimiento:	RF13
Nombre del Requerimiento:	Alta, baja, modificación y consulta de *hardware issues.*

Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de *hardware issues*, alteraciones a *hardware issues* existentes, relaciones con las entidades correspondientes y consulta de *hardware issues* ya almacenados.

Identificación del requerimiento:	RF14
Nombre del Requerimiento:	Alta, baja, modificación y consulta de clientes.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de clientes, alteraciones a clientes existentes, relaciones con las entidades correspondientes y consulta de clientes ya almacenados.

Identificación del requerimiento:	RF15
Nombre del Requerimiento:	Alta, baja y consulta de tipos de carcasa para *gateway*.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación, eliminación y consulta de tipos de carcasa para *gateway*.

Identificación del requerimiento:	RF16
Nombre del Requerimiento:	Alta, baja, modificación y consulta de módems satelitales.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de modems satelitales, alteraciones a módems satelitales existentes, relaciones con las entidades correspondientes y consulta de módems satelitales ya almacenados.

Identificación del requerimiento:	RF17
Nombre del	Alta, baja y consulta de tipos de módem satelital.

Requerimiento:	
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación, eliminación y consulta de tipos de módem satelital.

Identificación del requerimiento:	RF18
Nombre del Requerimiento:	Alta, baja y consulta de tipos de antena satelital.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación, eliminación y consulta de tipos de antena satelital.

Identificación del requerimiento:	RF19
Nombre del Requerimiento:	Alta, baja y consulta de tipos de baterías.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación, eliminación y consulta de tipos de baterías utilizadas en los dispositivos.

Identificación del requerimiento:	RF20
Nombre del Requerimiento:	Alta, baja y consulta de tipos de antena GPS.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación, eliminación y consulta de tipos de antenas GPS.

Identificación del requerimiento:	RF21
Nombre del Requerimiento:	Alta, baja y consulta de tipos de carcasa para nodo.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema.

	- El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación, eliminación y consulta de tipos de carcasa para nodo.

Identificación del requerimiento:	RF22
Nombre del Requerimiento:	Alta, baja y consulta de tipos de antena LoRa.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación, eliminación y consulta de tipos de antena LoRa.

Identificación del requerimiento:	RF23
Nombre del Requerimiento:	Alta, baja y consulta de tipos de panel solar.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación, eliminación y consulta de tipos de panel solar.

Identificación del requerimiento:	RF24
Nombre del Requerimiento:	Alta, baja, modificación y consulta de placas de nodo.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de placas de nodo, alteraciones a placas de nodo existentes, relaciones con las entidades correspondientes y consulta de placas de nodo ya almacenadas.

Identificación del requerimiento:	RF25
Nombre del Requerimiento:	Alta, baja, modificación y consulta de versiones de hardware de nodo
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema.

	- El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de tipos de versiones de hardware de nodo y consulta de tipos de versiones ya almacenadas.

Identificación del requerimiento:	RF26
Nombre del Requerimiento:	Alta, baja, modificación y consulta de versiones de firmware de nodo
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de tipos de versiones de firmware de nodo y consulta de tipos de versiones ya almacenadas.

Identificación del requerimiento:	RF27
Nombre del Requerimiento:	Alta, Baja, Modificación y Consulta de Equipos de riego
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación de equipos de riego, eliminación de los mismos y alteraciones a equipos existentes.

Identificación del requerimiento:	RF28
Nombre del Requerimiento:	Alta, baja, modificación y consulta de placas de GTW
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de placas de GTW, alteraciones a placas de GTW existentes, relaciones con las entidades correspondientes y consulta de placas ya almacenadas.

Identificación del requerimiento:	RF29
Nombre del Requerimiento:	Alta, baja, modificación y consulta de versiones de firmware de GTW
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.

Características:	El sistema debe permitir la creación y eliminación de tipos de versiones de firmware de GTW y consulta de tipos de versiones ya almacenadas.

Identificación del requerimiento:	RF30
Nombre del Requerimiento:	Alta, baja, modificación y consulta de versiones de hardware de GTW
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de tipos de versiones de hardware de GTW y consulta de tipos de versiones ya almacenadas.

Identificación del requerimiento:	RF25
Nombre del Requerimiento:	Alta, baja, modificación y consulta de tipos de *assets*
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de tipos de *assets*, y consulta de tipos de *assets* ya almacenados.

Identificación del requerimiento:	RF31
Nombre del Requerimiento:	Alta, baja, modificación y consulta de órdenes de trabajo
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de órdenes de trabajo, alteraciones a órdenes de trabajo existentes, relaciones con las entidades correspondientes y consulta de órdenes de trabajo ya almacenadas.

Identificación del requerimiento:	RF32
Nombre del Requerimiento:	Alta, baja y consulta de tipos de sensores de presión
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de tipos sensores de

	presión y consulta de tipos de sensores de presión ya almacenados.

Identificación del requerimiento:	RF33
Nombre del Requerimiento:	Alta, baja, modificación y consulta de sensores de presión
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de sensores de presión, alteraciones a sensores existentes, relaciones con las entidades correspondientes y consulta de sensores ya almacenados.

Identificación del requerimiento:	RF34
Nombre del Requerimiento:	Alta, baja, modificación y consulta de reparaciones de *assets*
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema.
Características:	El sistema debe permitir la creación y eliminación de reparaciones, alteraciones a reparaciones existentes, relaciones con las entidades correspondientes y consulta de relaciones ya almacenadas. En el caso de los técnicos, se debe permitir solo en caso de que el *hardware issue* correspondiente se le haya asignado.

Identificación del requerimiento:	RF35
Nombre del Requerimiento:	Alta, baja, modificación y consulta de ubicaciones de almacenamiento de *assets* de la empresa.
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.
Características:	El sistema debe permitir la creación y eliminación de ubicaciones de almacenamiento de *assets*, alteraciones a ubicaciones existentes, relaciones con las entidades correspondientes y consulta de ubicaciones ya almacenadas.

Identificación del requerimiento:	RF36
Nombre del Requerimiento:	Alta, baja, modificación y consulta de usuarios administradores
Precondiciones:	- El usuario debe haber iniciado sesión con sus credenciales en el sistema. - El usuario que realiza la modificación, alta o baja, debe contar con permisos de administrador.

Características:	El sistema debe permitir la creación y eliminación de usuarios administradores, alteraciones a usuarios existentes, relaciones con las entidades correspondientes y consulta de usuarios ya almacenados.

Apéndice D: Casos de uso completos

CU01: Login al sistema

Actores: Usuario (tanto técnico como usuario administrador).

Precondición:

- El usuario debe tener una cuenta registrada en el sistema.

Postcondición: El usuario se ha logueado al sistema.

Propósito: Ingresar al sistema para poder interactuar con el mismo.

Resumen: El usuario ingresa al sistema, el cual le solicita las credenciales. En caso de que los datos sean correctos, el usuario se habrá logueado correctamente en el sistema.

Tipo: Primario y esencial.

Curso normal de eventos:

Acción del usuario	Respuesta del sistema
1. El usuario ingresa al sistema sin haberse logueado	2. El sistema solicita las credenciales correspondientes
3. El usuario se ha logueado al sistema correctamente	

Cursos alternos:

3.1 Las credenciales son incorrectas, el usuario no ha podido ingresar al sistema.

CU02: Monitoreo general de equipos de riego y el estado de transmisión de sus sensores

Actores: Usuario administrador.

Precondición:

- El usuario debe haber ingresado al sistema con sus respectivas credenciales.

Postcondición: El usuario conoce fehacientemente el estado general de los *assets*.

Propósito: Permitir al usuario conocer el estado de los equipos, que *assets* tiene cada uno y si están transmitiendo o no en la actualidad.

Resumen: El usuario obtiene un listado de los equipos que describe sus *assets* asociados y el estado actual de transmisión de cada uno.

Tipo: Primario y esencial.

Curso normal de eventos:

Acción del usuario	Respuesta del sistema
1. El usuario ingresa a la vista principal.	2. El sistema muestra un listado de los equipos de riego presentes y sus sensores asociados.

Cursos alternos:

2.1 No existen equipos registrados en el sistema. En este caso la lista estará vacía y no se podrá proceder hasta que sean ingresados nuevos equipos.

CU03: Monitoreo de sensores de un equipo de riego

Actores: Usuario administrador.

Precondición:

- El usuario debe haber ingresado al sistema con sus respectivas credenciales.
- El equipo de riego a ser monitoreado debe tener al menos un sensor instalado.

Postcondición: El usuario administrador conoce fehacientemente el estado de los sensores de un equipo de riego.

Propósito: Obtener información sobre el funcionamiento de los sensores en un equipo de riego en particular.

Resumen: El usuario selecciona un equipo de riego para observar, a lo que el sistema responde con un detalle de los sensores y *assets* instalados. Esta visualización incluye el estado de transmisión actual de los mismos.

Tipo: Primario y esencial.

Curso normal de eventos:

Acción del usuario	Respuesta del sistema
CU02	
1. El usuario selecciona un equipo de riego de la lista.	2. El sistema presenta información detallada sobre el mismo.

Cursos alternos:

1.1 No existen equipos registrados en el sistema. En este caso la lista estará vacía y no habrá equipos por monitorear.

CU04: Detección automática de *hardware issues*.

Actores: Usuario administrador y sistema proveedor de información sobre los *assets*.

Precondición:

- El usuario administrador debe haber iniciado sesión en el sistema.

Postcondición: El usuario administrador concreta la creación de un *hardware issue* vinculado a un equipo de riego.

Propósito: Detectar fallas en los sensores y guardar información sobre ellas en el sistema para su posterior inspección y corrección.

Resumen: El sistema, mediante la consulta al sistema externo, detecta posibles *hardware issues,* informando al usuario administrador. De esta forma, el usuario es quién puede detectar si es un problema real o no.

Tipo: Primario y esencial.

Curso normal de eventos:

Acción del usuario	Respuesta del sistema	Sistema externo
		1. El sistema externo informa de una posible falla en un equipo de riego. Se incluye el identificador del equipo y su posible falla.
	2. El sistema genera un *hardware issue*	

	correspondiente, cargado sin diagnóstico para su posterior revisión.	
3. El usuario administrador observa el *hardware issue* generado automáticamente y tiene la opción de crearle un diagnóstico.		

Cursos alternos:

3.1 El usuario desestima el posible *hardware issue.* No se requiere acción alguna del sistema.

CU05: Consulta del estado de un *hardware issue*

Actores: Usuario administrador o técnico.

Precondición:

- El usuario administrador debe haber iniciado sesión en el sistema.
- El usuario técnico debe haber iniciado sesión en el sistema y debe tener asignado el *hardware issue* a consultar.

Postcondición: El usuario obtiene información sobre el *hardware issue* solicitado.

Propósito: Obtener información sobre el *hardware issue* para poder realizar distintas acciones.

Resumen: El usuario accede al módulo de *hardware issues* y selecciona uno para ver su estado.

Tipo: Primario y esencial.

Curso normal de eventos:

Acción del usuario	Respuesta del sistema
1. El usuario ingresa al módulo de *hardware issues.*	2. El sistema presenta todos los *hardware issues* válidos que el usuario tiene acceso con sus estados actuales.
3. El usuario selecciona un *hardware issue* para obtener más detalles.	4. El sistema presenta los datos pertinentes a este *hardware issue*.

Cursos alternos:

3.1 El usuario no tiene acceso a ningún *hardware issue* o no hay ninguno para visualizar.

CU06: Diagnóstico de *hardware issue* válido.

Actores: Usuario administrador.

Precondición:

- El usuario administrador debe haber iniciado sesión en el sistema.
- Debe existir un *hardware issue* válido que no haya sido diagnosticado.

Postcondición:

- Se agrega al *hardware issue* una sección diagnóstico en la que se detalla el problema que ese *asset* está sufriendo.

Propósito: registrar en el sistema la falla en el *asset*. De esta forma se puede proceder a su reparación considerando las características del problema.

Resumen: el usuario administrador observa los datos del *hardware issue* y describe una aproximación del problema con un diagnóstico.

Tipo: primario y esencial.

Curso normal de eventos:

Acción del usuario	Respuesta del sistema
CU05	
1. El usuario analiza los datos y saca una conclusión. Posteriormente, le asigna un diagnóstico al *hardware issue.*	2. El sistema persiste los cambios.

Cursos alternos:

1.1 Luego de analizar los datos, el usuario concluye que el *hardware issue* no era válido y procede a desestimarlo.

CU07: Asignación de un *hardware issue* un técnico.

Actores: Usuario administrador y técnico.

Precondición:

- El usuario administrador debe haber iniciado sesión en el sistema.
- El usuario técnico debe haber iniciado sesión en el sistema.

Postcondición:

- Un *hardware issue* se asigna a un técnico para que lo resuelva.

Propósito: Registrar en sistema a quién está asignado un *hardware issue.* Establece la responsabilidad del técnico sobre este problema en particular.

Resumen: El usuario administrador elige un *hardware issue* que no está actualmente asignado a ningún técnico y lo asigna a alguno que opere en la zona donde se encuentra el *asset* con problemas.

Tipo: Primario y esencial.

Curso normal de eventos:

Acción del usuario	Respuesta del sistema	Acción del Técnico
CU05		
1. El usuario selecciona un *hardware issue* diagnosticado que aún no fue asignado a ningún técnico.	2. El sistema presenta los posibles técnicos que operan en la zona donde está el *asset* con problemas.	
3. El usuario toma una decisión y selecciona a uno de los técnicos	4. El sistema persiste los cambios y cambia el estado del *hardware*	5. Al ingresar al sistema, el técnico visualiza el *hardware issue*

como el encargado de este *hardware issue*.	*issue* a “asignado”.	recientemente asignado.

Cursos alternos:

1.1 No hay ningún *hardware issue* para seleccionar. No hace falta la asignación de ningún *hardware issue*.

3.1 No hay técnicos posibles para seleccionar en esa zona. Se deberá asignar algún técnico a esa zona en particular.

5.1 El técnico no tiene ningún *hardware issue* asignado para visualizar.

CU08: Carga de peritaje y reparación

Actores: Técnico.

Precondición:

- El usuario técnico debe haber iniciado sesión en el sistema.
- El usuario técnico solucionó el *hardware issue* físicamente.
- Si el usuario técnico repara el *hardware issue* mediante un recambio de algún *asset*, el *asset* a colocar debe estar en su inventario.

Postcondición:

- Se registra una reparación asociada a *hardware issue* con los cambios que se realizaron sobre los equipos. El nuevo estado del *hardware issue* es “reparado” o “cerrado” dependiendo de los cambios realizados.

Propósito: Continuar el seguimiento de un *hardware issue* mediante el almacenamiento de los cambios físicos hechos por el técnico.

Resumen: El técnico asignado al *hardware issue* asiste al campo donde se encuentra el *asset* que presenta problemas. Realiza una pericia sobre el *asset* y documenta los logs obtenidos y los cambios hechos en el sistema. En caso de que se permita, el técnico realiza la reparación de la falla.

Tipo: Primario y esencial.

Curso normal de eventos:

Acción del técnico	Respuesta del sistema
CU05	
1. El técnico selecciona el *hardware issue* que resolvió en campo.	2. El sistema presenta un formulario.
3. El técnico almacena los logs que	4. El sistema persiste los cambios y

obtuvo y las acciones que realizó sobre los dispositivos con problemas.	cambia el estado del *hardware issue* a “reparado” si hizo falta cambiar algún *asset*, o a “cerrado” si solo hizo falta un ajuste menor sobre el *asset* existente.

Cursos alternos:

1.1 No hay ningún *hardware issue* para seleccionar. No tiene ningún *asset* para peritar ni reparar.

CU09: Carga de autopsia

Actores: Usuario administrador.

Precondición:

- El usuario administrador debe haber iniciado sesión en el sistema.

Postcondición:

- Se registra una autopsia asociada a *hardware issue* con información detallada sobre la falla en el *asset*. El nuevo estado del *hardware issue* es “cerrado”.

Propósito: Proseguir con el seguimiento de un *hardware issue* mediante el almacenamiento de las fallas que tuvo un *asset*.

Resumen: El usuario administrador realiza pruebas sobre el *asset* con fallas. De esta forma, se puede determinar la raíz del problema y almacenarla en el sistema.

Tipo: Primario y esencial.

Curso normal de eventos:

Acción del técnico	Respuesta del sistema
CU05	
1. El usuario selecciona el *hardware issue* correspondiente al *asset* sobre el que realizó la autopsia.	2. El sistema presenta un formulario para la carga de los datos necesarios.
3. El usuario completa el formulario: almacena los logs pertinentes y la causa de la falla.	4. El sistema persiste los cambios y cambia el estado del *hardware issue* a “cerrado”.

Cursos alternos:

1.1 No hay ningún *hardware issue* para seleccionar. En este caso no se podrá cargar una autopsia.

CU10: Seguimiento de stock de *assets*

Actores: Usuario administrador.

Precondición:

- El usuario administrador debe haber iniciado sesión en el sistema.

Postcondición:

- El usuario administrador se informa acerca de la cantidad y tipo de *assets* en cada ubicación del sistema.

Propósito: Poder realizar un seguimiento de los *assets* de la empresa y conocer su ubicación.

Resumen: El usuario administrador ingresa al módulo de stock. Luego, revisa el inventario de los diferentes dispositivos en las ubicaciones de la empresa.

Tipo: Secundario.

Curso normal de eventos:

Acción del técnico	Respuesta del sistema
1. El usuario ingresa al módulo de stock de *assets*.	2. El sistema presenta información sobre los inventarios actuales de los dispositivos.

Cursos alternos:

N/A

CU11: Seguimiento del nivel de servicio prestado

Actores: Usuario administrador.

Precondición:

- El usuario administrador debe haber iniciado sesión en el sistema.

Postcondición:

- El usuario administrador se informa acerca del grado de cumplimiento de la empresa respecto a los contratos de los clientes.

Propósito: Dar información a los usuarios administradores sobre el cumplimiento de la empresa con los niveles de servicio prometidos contractualmente.

Resumen: El usuario administrador ingresa a la vista del dashboard y puede observar el nivel de servicio prestado para cada cliente. Puede utilizar esta información para tomar decisiones futuras.

Tipo: Secundario.

Curso normal de eventos:

Acción del técnico	Respuesta del sistema
1. El usuario ingresa a la vista de niveles de servicio.	2. El sistema presenta información sobre los niveles de servicio prestados para cada cliente.

Cursos alternos:

N/A

CU12: Vista del panel de control del sistema

Actores: Usuario administrador.

Precondición:

- El usuario administrador debe haber iniciado sesión en el sistema.

Postcondición:

- El usuario administrador obtiene información sobre el funcionamiento actual de los dispositivos y métricas importantes de su funcionamiento.

Propósito: Dar información a los usuarios administradores sobre los dispositivos actualmente desplegados.

Resumen: El usuario administrador ingresa al panel de control donde puede observar métricas importantes sobre el sistema, como por ejemplo: cantidad de equipos con *assets* instalados comparado con cantidad total de equipos, promedio de tiempo transcurrido desde que se diagnosticó un *hardware issue* hasta que fue resuelto.

Tipo: Secundario.

Curso normal de eventos:

Acción del técnico	Respuesta del sistema
1. El usuario ingresa a la vista de niveles de servicio.	2. El sistema presenta información sobre las distintas métricas.

Cursos alternos:

N/A

Bibliografía

[1] Uribe, C; Lagos, L; y Holzaphel, E, *Pivote central*. 2001. https://biblioteca.inia.cl/bitstream/handle/123456789/33791/NR25837.pdf (accedido el 22 de noviembre de 2021).

[2] "the mistery of the round fields". Elo Vazquez. Licencia CC BY-NC-ND 2.0. https://www.flickr.com/photos/elo/2561962498/ (accedido el 9 de marzo de 2022).

[3] "Irrigated field in west central Texas ". Joel Dunn. Licencia Unsplash. https://unsplash.com/photos/zm3b7LeusHI (accedido el 9 de marzo de 2022).

[4] "GatewayPONCE". Bichos de Campo. https://bichosdecampo.com/en-homenaje-a-ponce-combinando-informatica-y-electronica-tres-jovenes-marplatenses-crearon-una-startup-para-monitorear-y-eficientizar-los-equipos-de-riego-en-argentina/gatewayponce/ (accedido el 8 de febrero de 2022).

[5] "Types of Pressure Sensors - gauge, absolute, sealed & differential". ES Systems. https://esenssys.com/differences-between-pressure-sensors/ (accedido el 22 de noviembre de 2021).

[6] "What is lora". Semtech. https://www.semtech.com/lora/what-is-lora (accedido el 4 de marzo de 2022).

[7] "Pricing". Limble. https://limblecmms.com/pricing/ (accedido el 22 de noviembre de 2021).

[8] "Leading CMMS Software | Hippo CMMS". Hippo CMMS. https://hippocmms.iofficecorp.com/solutions/cmms-software (accedido el 22 de noviembre de 2021).

[9] K. Bagadia, *Computerized Maintenance Management Systems Made Easy: How to Evaluate, Select, and Manage CMMS*. McGraw Hill Professional, 2010.

[10] M. Reddy, *API design for C++*. Burlington, MA: Morgan Kaufmann, 2011.

[11] "GraphQL | A query language for your API". https://graphql.org/ (accedido el 3 de marzo de 2022).

[12] "GraphQL". GraphQL Specification Versions. https://spec.graphql.org/October2021/ (accedido el 3 de marzo de 2022).

[13] "React – Una biblioteca de JavaScript para construir interfaces de usuario". https://es.reactjs.org/ (accedido el 3 de marzo de 2022).

[14] "Vue.js - The Progressive JavaScript Framework | Vue.js". https://vuejs.org/ (accedido el 3 de marzo de 2022).

[15] "Welcome to Data Studio!" Google Help. https://support.google.com/datastudio/answer/6283323?hl=en (accedido el 3 de marzo de 2022).