UNIVERSIDAD DE CONCEPCIÓN

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INFORMATICA Y CIENCIAS DE LA COMPUTACIÓN

Profesor Patrocinante:

Geoffrey Hecht

Ingenieros supervisores:
Rodrigo Vergara Arias
Alejandro Sánchez Fuenzalida

Informe de Memoria de Título
para optar al título de:

Ingeniero Civil Informático

LogBook: Sistema de gestión de eventos de mantenimiento enfocada a la industria de la gran minería.

Concepción, Julio de 2024

Matías Guido Eduardo Cáceres Figueroa

UNIVERSIDAD DE CONCEPCIÓN
Facultad de Ingeniería
Departamento de Informática y
Ciencias de la computación

Profesor Patrocinante:
Geoffrey Hecht

LogBook: Sistema de gestión de eventos de mantenimiento enfocada a la industria de la gran minería

Matías Guido Eduardo Cáceres Figueroa

Informe de Memoria de Título
para optar al Título de

Ingeniero Civil Informático

Julio 2024

Resumen

Uno de los grandes problemas de las industrias es que la productividad se puede ver mermada debido a problemas como interrupciones imprevistas causadas por fallas mecánicas. Con el desarrollo de la tecnología a lo largo de los años, poco a poco esta se ha ido introduciendo en los procesos industriales para optimizar y revolucionar varios procesos, lo que se conoce como Industria 4.0. En la industria minera, la transición hacia la Industria 4.0 se dificulta debido a, entre otras cosas, la inconsistencia en la aplicación de estándares sobre cómo organizar y gestionar la información referente a datos técnicos o de mantención para la variedad de equipos y componentes existentes. Bajo este contexto, Simpro está desarrollando una suite de aplicaciones con la cual facilitar esta transición. Esta memoria aborda el desarrollo de una de las aplicaciones de esta suite, LogBook, cuyo fin es mejorar la gestión de activos y el registro de eventos de mantenimiento. Para esto se siguió una metodología ágil e iterativa con tres fases de desarrollo. El prototipo obtenido facilitó las tareas de identificar la ubicación física de un equipo, así como los distintos sistemas y partes que lo componen. A su vez, permite observar las fallas y mantenciones asociadas. Se destacaron potenciales mejoras futuras como la incorporación de una interfaz gráfica y la capacidad de agregar información específica de equipos de los cuales no exista registro. Esta memoria representa un avance hacia la digitalización y optimización de la gestión de activos en la industria minera, ofreciendo una base sólida para futuras mejoras y ampliaciones.

Tabla de contenidos

1 INTRODUCCIÓN 1

1.1 CONTEXTO DEL PROYECTO 1
1.2 DESCRIPCIÓN DEL PROBLEMA 2
1.3 SOLUCIÓN PROPUESTA 2
1.4 OBJETIVO GENERAL 3
1.5 OBJETIVOS ESPECÍFICOS 3
1.6 ALCANCES Y LIMITACIONES 3
1.7 METODOLOGÍA 4

2 MARCO TEÓRICO 5

2.1 INDUSTRIA 4.0 5
2.2 NORMA ISO 14224 6

3 DISEÑO DEL SOFTWARE 8

3.1 USUARIOS 8
3.2 HISTORIAS DE USUARIO 9
3.3 ARQUITECTURA 11
3.3.1 Diagrama de Contexto 11
3.3.2 Diagrama de Contenedores 12
3.3.3 Diagrama de Componentes 13
3.3.4 Base de datos 14
3.3.5 Modelo entidad relación 14
3.3.6 Modelo relacional 17

4 IMPLEMENTACIÓN 18

4.1 TECNOLOGÍAS UTILIZADAS 18
4.1.1 FastAPI 19
4.1.2 Docker 19
4.1.3 PostgreSQL 19
4.1.4 AWS 20
4.1.5 Postman 20

5 RESULTADOS 21

5.1 FUNCIONALIDADES PRINCIPALES 21
5.1.1 Jerarquía de equipos 21
5.1.2 Eventos de mantenimiento 26
5.1.3 Información específica de cada equipo 30
5.2 RETROALIMENTACIÓN DE LOS USUARIOS 34

6 CONCLUSIONES 43

7 TRABAJO FUTURO 44

8 BIBLIOGRAFÍA 45

9 ANEXOS… 46

9.1 DETALLE DE LAS ENTIDADES DEL MODELO ENTIDAD RELACIÓN 46
9.2 LISTA COMPLETA DE ENDPOINTS DE LA API 53

Lista de tablas

Tabla 1 Respuestas encuesta de satisfacción..... 42
Tabla 2 Detalle entidad Plant 47
Tabla 3 Detalle entidad Section 47
Tabla 4 Detalle entidad Equipment..... 47
Tabla 5 Detalle entidad Subunit 48
Tabla 6 Detalle entidad Component..... 48
Tabla 7 Detalle entidad Part 49
Tabla 8 Detalle entidad Screen..... 49
Tabla 9 Detalle entidad Crusher..... 49
Tabla 10 Detalle entidad Sensor..... 50
Tabla 11 Detalle entidad Failure 50
Tabla 12 Detalle entidad Report..... 50
Tabla 13 Detalle entidad Maintenance..... 51
Tabla 14 Detalle entidad Collection..... 51
Tabla 15 Detalle entidad Failure mechanism 51
Tabla 16 Detalle entidad Failure cause 51
Tabla 17 Detalle entidad Detection method..... 52
Tabla 18 Detalle entidad Failure mode 52
Tabla 19 Detalle entidad Maintenance activity..... 52
Tabla 20 Lista completa de endpoints de la API de LogBook..... 53

Lista de figuras

Figura 1 Clasificación taxonómica de LogBook, hecha según lo establecido en la sección 8.2 de la ISO 14224:2016. 7
Figura 2 Diagrama de contexto de la plataforma. 11
Figura 3 Diagrama de contenedores de la plataforma. 12
Figura 4 Diagrama de componentes de la plataforma. 13
Figura 5 Vista general modelo entidad-relación de la base de datos de LogBook. 14
Figura 6 Modelo relacional de la base de datos de LogBook. 17
Figura 7 Diagrama de tecnologías de LogBook. 18
Figura 8 Estructura jerárquica devuelta a partir del nivel de plantas. 22
Figura 9 Estructura jerárquica devuelta a partir del nivel de equipos. 23
Figura 10 Ejemplo de obtención de la jerarquía de todas las plantas. 24
Figura 11 Ejemplo de obtención de la jerarquía de todos los equipos. 25
Figura 12 Ejemplo de obtención de fallas. 27
Figura 13 Ejemplo de obtención de mantenciones. 28
Figura 14 Ejemplo de obtención de reportes. 29
Figura 15 Ejemplo de obtención de información específica sobre chancadores. 31
Figura 16 Ejemplo de obtención de información específica sobre harneros. 32
Figura 17 Ejemplo de obtención de los sensores asociados a un chancador. 33
Figura 18 Drawer de la interfaz gráfica de Sentinel (TCMS) con información proveniente de LogBook. 34
Figura 19 Drawer de la interfaz gráfica de AI Detector (TCMS) con información proveniente de LogBook. 35
Figura 20 Lista de chancadores en AI Detector, cargada desde LogBook. 36
Figura 21 Ejemplo de predicción modelo Autoencoder en AI Detector. 37
Figura 22 Documentación del endpoint GET /hierarchy/component. 38
Figura 23 Documentación del endpoint GET /hierarchy/equipment/crusher. 39
Figura 24 Documentación del método POST /maintenances. 40
Figura 25 Descripción de los parámetros del body del método POST /maintenances. 41
Figura 26 Ejemplo de respuesta del método POST /maintenances. 41

1 Introducción

1.1 Contexto del proyecto

En el actual contexto de los sectores industriales caracterizado por una creciente incorporación de tecnologías avanzadas en sus procesos, Simpro se destaca como una empresa de consultoría y servicios de ingeniería enfocada al mantenimiento y monitoreo de equipos industriales, particularmente de la gran minería.

Uno de los problemas críticos de estos sectores es que la productividad se puede ver mermada por diversos motivos, como interrupciones imprevistas a causa de fallas mecánicas [1].

Las máquinas rotatorias impulsadas por motores de inducción son un activo presente en casi todos los entornos industriales y, por lo tanto, típico objeto de mantención. De sus componentes, los rodamientos cuentan por cerca del 40% del total de fallas registradas por estos motores [2]. Por lo tanto, es importante para las industrias el incorporar soluciones tecnológicas para minimizar este tipo de eventos.

Por otro lado, la información referente a los datos técnicos y la condición de los diferentes equipos está dispersa a lo largo de diferentes fuentes como lo pueden ser un informe de tribología, un informe de monitoreo y vibraciones (MONCON) o un reporte con las tendencias de corriente de un motor, lo cual ralentiza y dificulta el monitoreo, análisis y toma de decisiones, para, por ejemplo, decidir si es necesario realizar una mantención o reemplazar un elemento.

Para remediar esta situación, uno de los focos de desarrollo de Simpro es la transición e integración digital, dentro del marco de la *Industria 4.0*, mediante el desarrollo de la suite TCMS y sus aplicaciones, asociadas a diversas facetas del monitoreo basado en condición de activos y plantas industriales.

1.2 Descripción del problema

El problema que se quiere abordar con este proyecto es la gestión de activos y el seguimiento histórico de eventos de mantenimiento asociados a equipos de la industria minera. En la actualidad, la industria minera enfrenta desafíos significativos debido a la falta de estandarización en la recopilación y almacenamiento de este tipo de datos, lo que dificulta la implementación efectiva de soluciones tecnológicas y la transición hacia la industria 4.0.

1.3 Solución propuesta

Ante el presente desafío, se propone la creación de un software que permita almacenar y gestionar información relacionada a activos y eventos de mantenimiento en el área de la minería. Esta será uno de los pilares centrales de la suite de Simpro, *Total Condition Monitoring System* (TCMS), actualmente en desarrollo.

La solución planteada se denomina LogBook. Con esta se espera lograr jerarquizar la información, permitiendo identificar de manera clara y sencilla la ubicación física de un equipo, su información técnica y la relación entre los distintos elementos que lo conforman. A su vez, esta debe de actuar como un registro histórico de los distintos eventos de mantenimiento relacionados a un equipo, en donde como mínimo, se especifiquen la fecha, ubicación física y causa del evento en particular. Junto con esto, el sistema debe permitir agrupar distintos eventos de mantenimiento para generar una traza y dar con la causa del mal funcionamiento o detención de un equipo.

La complejidad del problema radica en que la plataforma debe de ser lo suficientemente robusta para gestionar la información que provendrá del resto de las aplicaciones de TCMS, además de tener que evaluar la necesidad de una comunicación unidireccional o bidireccional con cada uno de los demás módulos.

Se espera que la solución propuesta sirva como un sistema centralizado de información el cual procese las consultas más comunes relacionadas a los equipos, apoyando así a las demás aplicaciones de la suite, permitiendo aumentar la productividad y tiempos de respuesta de estas. La eficiencia y eficacia de la misma se validarán mediante pruebas y evaluaciones. De este modo, se espera que la plataforma responda adecuadamente a los retos planteados.

1.4 Objetivo general

Desarrollar una solución tecnológica integral mediante una plataforma integradora de información proveniente de diversas fuentes, que permita almacenar y gestionar información de mantenimiento asociada a distintos equipos, subunidades y componentes de una planta de la gran minería, mejorando la eficiencia operativa y reduciendo los tiempos de inactividad no planificados gracias al seguimiento efectivo de la condición de maquinaria industrial.

1.5 Objetivos específicos

- **Diseñar e implementar una base de datos que de soporte al sistema:** que sea capaz de registrar y gestionar la jerarquía de equipos, subunidades, componentes y las relaciones entre estos.
- **Almacenar y gestionar eventos de mantenimiento:** la base de datos también debe de ser capaz de almacenar los eventos de mantenimiento asociados a elementos de la jerarquía, permitiendo agregar, modificar y eliminar eventos de forma segura y eficaz.
- **Implementar e implementar API:** implementar una API para almacenar y gestionar eventos de mantenimiento asociados a elementos de la jerarquía provenientes de diversas fuentes.
- **Integración con suite TCMS:** se debe considerar la integración de la aplicación con el resto de las aplicaciones de la suite.

1.6 Alcances y limitaciones

- El sistema debe de quedar preparado para acoplar una interfaz gráfica en el futuro (fuera del alcance de esta memoria)
- El software producido será propiedad exclusiva de Simpro.

1.7 Metodología

El proyecto se rigió por una metodología ágil e iterativa, con sprints adaptados a las condiciones de trabajo de Simpro. En esta, se distinguen tres fases principales definidas a continuación.

Fase 1: Planificación inicial

Durante esta etapa, se realizó una búsqueda sobre normativa asociada al contexto del proyecto, se definieron los objetivos para cada sprint, se diseñó un prototipo inicial de la aplicación y se obtuvieron los recursos tecnológicos necesarios para el proyecto.

Fase 2: Desarrollo e implementación

En esta fase se desarrolló e implementó la plataforma según los diseños realizados en la etapa anterior junto con su puesta en marcha en el servidor correspondiente.

Fase 3: Evaluación de la Plataforma

Durante esta etapa, la plataforma se sometió a pruebas entregándola a los usuarios finales para obtener retroalimentación en un entorno de uso real. Se proporcionaron nuevas versiones semanalmente y se recopilaron comentarios para realizar ajustes.

2 Marco teórico

En esta sección se presentará una vista inicial a términos claves que proporcionaran al lector una base sólida para comprender el contexto del proyecto.

2.1 Industria 4.0

El contexto del proyecto está relacionado con la Industria 4.0, la cual representa una transformación en los procesos productivos mediante la adopción de tecnologías digitales avanzadas. Su objetivo es crear un sistema de manufactura más digitalizado, interconectado, inteligente y autónomo, que mejore la eficiencia, reduzca costos y aumente la calidad. [3] plantea 9 pilares fundamentales para la comprensión y adopción de la Industria 4.0: Big data, Robots Autónomos, Simulaciones, Integración horizontal y vertical, Internet de las cosas (IoT), Cloud Computing, Manufactura aditiva, Realidad Aumentada (AR) y Ciberseguridad.

La integración de estas tecnologías en el sector minero se denomina Minería 4.0, en donde, por ejemplo, los dispositivos IoT pueden utilizarse para supervisar el estado de los equipos de minería en tiempo real, lo que permite realizar un mantenimiento predictivo y reducir el tiempo de inactividad debido a fallos inesperados.

Estas innovaciones contribuyen al ahorro de costes y a la mejora de los procesos de toma de decisiones, ejemplificando así los beneficios de la industria 4.0 en el área de la minería.

Una de las bases para realizar esta transición es contar con un sistema en donde se centralice la información como por ejemplo los diferentes equipos con los que se cuenta, su ubicación, los datos que se están monitoreando de cada uno de estos o cada cuanto se les realiza cambios de piezas, y con el desarrollo de LogBook se espera suplir esta necesidad. Junto con esto. a futuro, Simpro planea desarrollar simulaciones con realidad virtual de equipos reales para la capacitación de personal, tomando como base la información que se recopile en LogBook para esto.

2.2 Norma ISO 14224

Uno de los problemas en la industria minera para adoptar las prácticas de la Industria 4.0 es la inconsistencia en la aplicación de referencias o estándares asociados a este sector que permitan respaldar la toma de decisiones y ofrezcan seguridad a las empresas. Para hacer frente a esta carencia, este proyecto utiliza como base la estructura y metodologías descritas en la norma ISO 14224:2016 [4] la cual, aunque se diseñó originalmente para la industria del petróleo y el gas, proporciona un marco de referencia para la recopilación y el análisis de datos de mantenimiento, que puede adaptarse para satisfacer las necesidades de las operaciones mineras.

El principal beneficio que se obtiene al adoptar esta norma es que se resuelve el problema de definir una estructura para jerarquizar y organizar la información referente a los equipos que se utilizan en la industria, esta se puede apreciar en la sección 8, específicamente 8.2, ilustrada por la figura 3 de la norma, titulada ***Taxonomy classification with taxonomic levels*.**

En esta, los niveles 1 al 5 son usados para describir la ubicación física del equipo, distinguiendo el tipo de industria y aplicación, ya que una misma máquina puede ser usada en diferentes industrias y tipos de plantas.

Los niveles 6 al 9 se utilizan para indicar los diferentes elementos que componen a un equipo y el cómo estas se relacionan entre sí, estableciendo una relación padre – hijo entre un nivel superior y un nivel inferior, i.e. una subunidad pertenece a un equipo, un componente pertenece a una subunidad y así sucesivamente.

Para el caso particular de esta memoria, se llegó a un acuerdo con la empresa y se estableció que para el desarrollo de LogBook se utilizarían los niveles 4 al 9 específicamente, ya que esto se ajusta a su caso de uso. Haciendo uso de esta jerarquía, podemos identificar de forma clara en donde se encuentra un chancador o a qué equipo pertenece el motor 7.

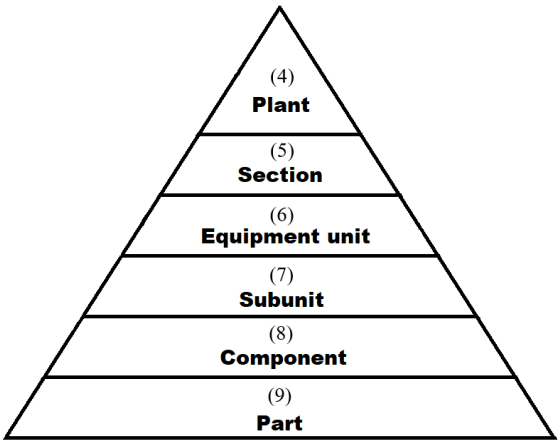

Figura 1 Clasificación taxonómica de LogBook, hecha según lo establecido en la sección 8.2 de la ISO 14224:2016.

Ya resuelto el tema de la jerarquía, el siguiente paso es identificar qué información es relevante almacenar respecto a cada equipo, lo cual es abarcado en la sección 9.4 – *equipment data* de la norma ISO 14224. En esta se menciona que la información de un equipo se compone de dos tipos de datos: datos comunes a toda clase de equipos, y datos específicos de cada clase, esto se explica más en detalle en la tabla 5 de la misma, en donde se remarca el mínimo de información que debería ser recolectada para todos los equipos como por ejemplo su código, tipo, clase, fabricante, entre otros.

Análogamente, es necesario conocer que información es relevante almacenar para tener un registro de las fallas y mantenciones para un equipo a lo largo del tiempo. Esto es visto en detalle en las secciones 9.5 – *Failure data* y 9.6 – *Maintenance data* de la norma respectivamente.

Mediante el uso de esta estructura y metodología, se facilita una transición más fluida hacia la Industria 4.0 enfocada al sector de la minería, estableciendo una base robusta para la implementación de prácticas de mantenimiento predictivo y gestión de activos.

3 Diseño del software

A lo largo del diseño y planificación del software, se utilizaron diversos artefactos, cada uno de estos representando un aspecto distinto de la API desarrollada. A continuación, se presentan las distintas documentaciones según el orden en que se implementaron.

3.1 Usuarios

En esta instancia en particular, no habrá usuarios comunes que hagan uso directo de LogBook, pero se proyecta a futuro generar una interfaz para la visualización y gestión de los datos de LogBook de una forma más interactiva. De momento, serán las demás aplicaciones de la suite TCMS y empleados con el rol de administrador los que interactuarán con LogBook, teniendo identificadas las siguientes interacciones:

- **AI Detector**: Aplicación dedicada al monitoreo de anomalías analizando información proveniente de sensores SCADA (Supervisory Control And Data Acquisition). Dentro de sus características se encuentra el uso métodos de inteligencia artificial para calcular las probabilidades de que un equipo o una parte del mismo genere una lectura anómala dentro de un rango de tiempo definido. Respecto a su relación con LogBook, este accederá a información de la jerarquía para un elemento específico, como equipos y sensores, la relación entre estos, y enviará resúmenes de reportes de episodios SCADA.
- **Sentinel:** Software enfocado al análisis de vibraciones en equipos industriales. Este accederá a información de la jerarquía para un elemento en específico, como equipos y sensores, la relación entre estos y enviará resúmenes de reportes de vibración hacia LogBook.
- **RCM Tool:** Software dedicado a realizar reportes en terreno de fallas o daños a equipos industriales. Este accederá a información de la jerarquía para un elemento en específico y enviará resúmenes de reportes de condición hacia LogBook, a que equipo.
- **Administrador:** Responsable de garantizar el buen funcionamiento de la aplicación. Se encarga de poblar y mantener la base de datos, puede agregar, modificar, eliminar o consultar todo tipo de información relacionada a un elemento o evento de mantenimiento.

3.2 Historias de usuario

Con el propósito de especificar con mayor precisión las metas de diseño y desarrollo de la plataforma, se crearon las siguientes historias de usuario:

- **HU1:** Como LogBook quiero gestionar elementos de una jerarquía para actualizar la información de la misma.
- **HU2:** Como sistema externo quiero recibir una lista que contenga la jerarquía completa de una planta para hacer uso de esta.
- **HU3:** Como sistema externo quiero ver el detalle de una planta en específico para hacer uso de esta información.
- **HU4:** Como sistema externo quiero ver el detalle de un área en específico para hacer uso de esta información.
- **HU5:** Como sistema externo quiero ver el detalle de un equipo en específico para hacer uso de esta información.
- **HU6:** Como sistema externo quiero ver el detalle de una subunidad en específico para hacer uso de esta información.
- **HU7:** Como sistema externo quiero ver el detalle de un componente en específico para hacer uso de esta información.
- **HU8:** Como Logbook quiero gestionar eventos de mantenimiento para actualizar la información de estos.
- **HU9:** Como LogBook quiero recibir elementos que me permitan crear eventos de mantenimiento.
- **HU10:** Como sistema externo quiero una lista que contenga todos los eventos asociados a un elemento de la jerarquía para facilitar el análisis y diagnóstico de fallas para dicho elemento.
- **HU11:** Como LogBook quiero gestionar colecciones de eventos de mantenimiento para actualizar la información de estos.
- **HU12:** Como LogBook quiero recibir una lista de eventos para crear una colección.
- **HU13:** Como sistema externo quiero una lista que contenga la información de todas las colecciones de eventos asociadas a un elemento de la jerarquía para facilitar el análisis y diagnóstico de fallas para dicho elemento.

- **HU14:** Como sistema externo quiero poder obtener la información asociada a los chancadores para hacer uso de esta, ya sea una lista con todos, o un chancador en específico.
- **HU15:** Como sistema externo quiero poder obtener la información asociada a los sensores para hacer uso de esta, ya sea una lista con todos, o un sensor en específico.
- **HU16:** Como sistema externo quiero poder obtener la información asociada a los harneros para hacer uso de esta, ya sea una lista con todos, o un harnero en específico.
- **HU17:** Como sistema externo quiero obtener la ficha médica de un elemento de la jerarquía para hacer uso de esta.
- **HU18:** Como Logbook quiero almacenar un resumen de los reportes generados por RCM Tool para su posterior uso.

3.3 Arquitectura

En esta sección, se describe el diseño de la arquitectura del software LogBook haciendo uso del modelo C4. Este modelo incluye un conjunto de diagramas que representan la plataforma en diferentes niveles de abstracción: el diagrama de contexto, que ilustra la relación de la plataforma con otros sistemas, el diagrama de contenedores, que detalla la interacción entre las distintas partes del sistema, y el diagrama de componentes, que describe la estructura interna de cada contenedor. Cada uno de estos será explicado en detalle a continuación.

3.3.1 Diagrama de Contexto

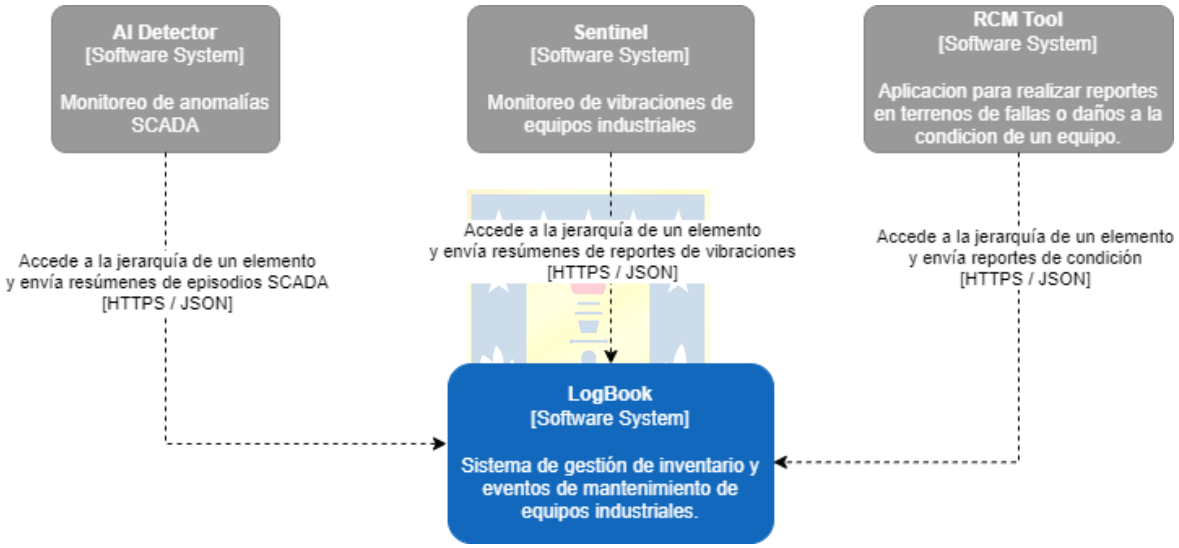

Figura 2 Diagrama de contexto de la plataforma.

En este nivel, se observan el cómo las demás aplicaciones interactúan con LogBook. Como interacción común, todas las aplicaciones acceden a la jerarquía, sin embargo, cada una accede a un elemento o nivel distinto y cada una hace uso de información diferente, ya sea técnica o referente a la ubicación física del elemento. Por el otro lado, se espera que cada aplicación haga envío de un resumen de reportes de diferente índole hacia LogBook.

3.3.2 Diagrama de Contenedores

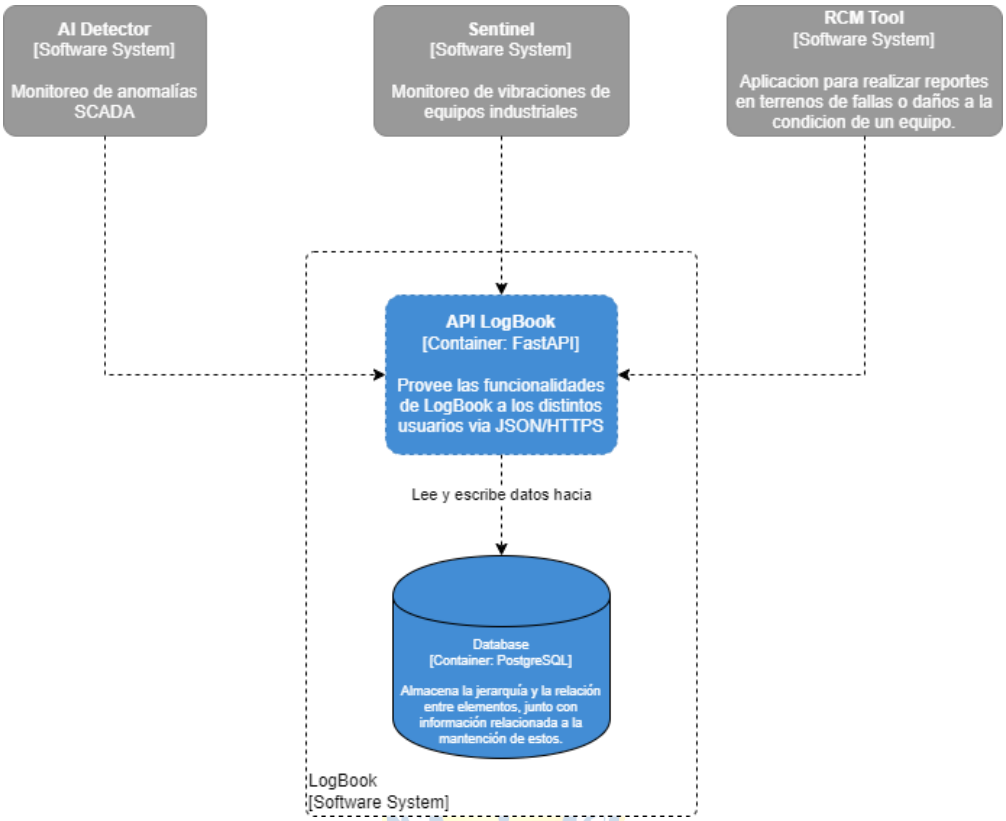

Figura 3 Diagrama de contenedores de la plataforma.

Como se muestra en la figura 3, el sistema cuenta con dos partes claramente definidas, una API REST que actúa como backend del sistema, y un servidor de base de datos. La API es el componente central del sistema y es responsable de establecer conexiones seguras tanto con las demás aplicaciones como con la base de datos, mientras que esta última, es la encargada de almacenar la información proveniente de las distintas fuentes de datos.

3.3.3 Diagrama de Componentes

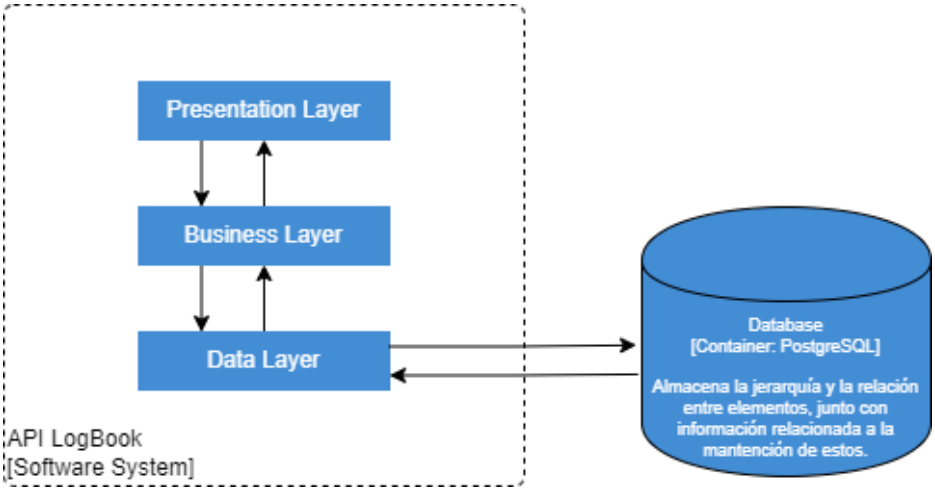

Figura 4 Diagrama de componentes de la plataforma.

Para el desarrollo de la API, se optó por una arquitectura de tres capas: capa de presentación, capa de negocio y capa de datos, cada una con una responsabilidad en específico. Esta arquitectura permite que cada capa de la aplicación escale de manera independiente según las necesidades de la aplicación. A continuación, se explicará en detalle el propósito de cada capa.

Capa de presentación: Responsable de manejar las solicitudes provenientes de las demás aplicaciones de la suite TCMS. Pone a disposición las rutas de la API, gestiona las solicitudes HTTP y las redirige a la capa de negocio. A su vez, establece el formato en que se deben recibir y entregar los datos.

Capa de negocio: Contiene todo lo referente a la lógica con la que opera la aplicación. Es la encargada de procesar las peticiones, realizar validación de datos y aplicar las reglas de negocio. Esta capa se encarga de la comunicación entre la capa de presentación y la capa de datos.

Capa de datos: Se encarga de gestionar el acceso y la manipulación de los datos almacenados en la base de datos interna del sistema.

3.3.4 Base de datos

La base de datos es el pilar central de la aplicación, se puso énfasis en establecer una jerarquía definida para cada grupo de elementos, que fuera intuitiva y de rápido acceso. En las siguientes secciones se presentan tanto el modelo entidad-relación como el modelo relacional de esta. Estos diagramas proporcionan un detalle exhaustivo de cada uno de los elementos que conforman la base de datos.

3.3.5 Modelo entidad relación

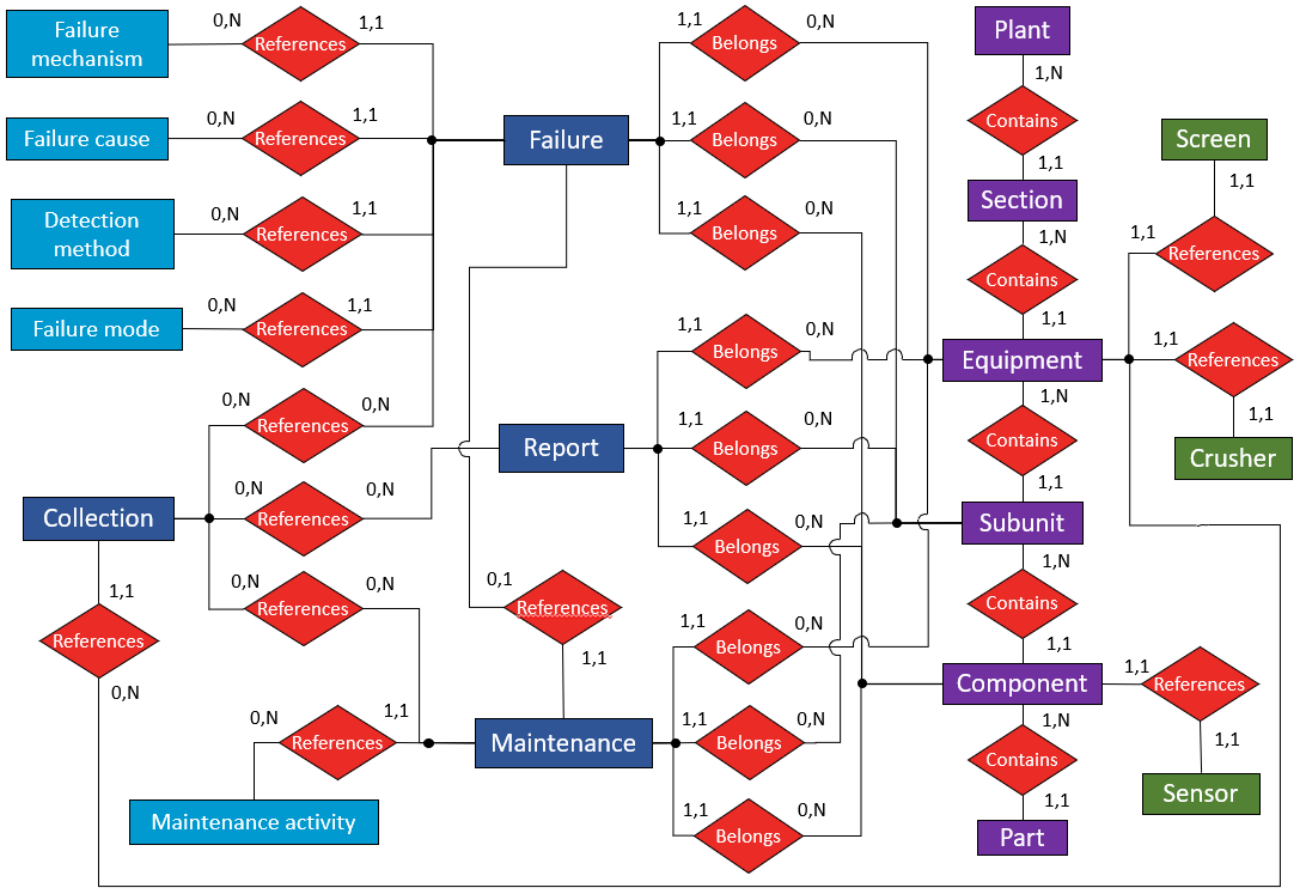

Figura 5 Vista general modelo entidad-relación de la base de datos de LogBook.

Dentro de este modelo se distinguen cuatro partes principales:

Jerarquía: Destacado en morado, almacena información de ubicación, detalles generales y la relación entre los distintos elementos de un equipo.

Tablas complementarias a la jerarquía: Destacadas en verde, en ellas, se almacena información que es única y específica de cada tipo de elemento.

Tablas relacionadas a eventos de mantenimiento: Destacadas en azul, almacenan todas las fallas, mantenciones y reportes en terreno asociados a los distintos equipos.

Tablas complementarias a eventos de mantenimiento: Destacadas en celeste, contienen códigos que ayudan a explicar con detalle que fue lo que paso en un evento de mantenimiento en particular.

Debido a la cantidad de entidades y de argumentos, en la figura 5 se presenta una vista general del modelo, dando una breve explicación sobre cada entidad a continuación. La explicación en detalle de los parámetros de cada entidad se encuentra en los anexos correspondientes a cada una de estas.

- **Plant:** El primer nivel de la jerarquía, representa la información básica de una planta minera, por ejemplo, su nombre. Ver anexos, tabla 2: Detalle entidad Plant.
- **Section:** Segundo nivel de la jerarquía, representa las distintas secciones de una planta, por ejemplo, chancado primario, secundario o terciario. Ver anexos, tabla 3: Detalle entidad Section.
- **Equipment:** Tercer nivel de la jerarquía, se basa en la tabla 5: *Equipment data common to all equipment classes* de la norma ISO 14224:2016. En esta tabla se encuentra toda la información que es común a los distintos equipos de la gran minería, como su código, numero serial y manufacturador. Ver anexos, tabla 4: Detalle entidad Equipment.
- **Subunit:** Cuarto nivel de la jerarquía, se basa parcialmente en la tabla 5: *Equipment data common to all equipment classes* de la norma ISO 14224:2016. Representa las subunidades que componen un equipo, por ejemplo, el sistema motriz de un harnero. Ver anexos, tabla 5: Detalle entidad Subunit.
- **Component:** Quinto nivel de la jerarquía, se basa parcialmente en la tabla 5: *Equipment data common to all equipment classes* de la norma ISO 14224:2016. Representa los componentes que conforman una subunidad, por ejemplo: el motor 1 del sistema motriz. Ver anexos, tabla 6: Detalle entidad Component.
- **Part:** Sexto nivel de la jerarquía, se basa parcialmente en la tabla 5: *Equipment data common to all equipment classes* de la norma ISO 14224:2016. Representa las distintas partes que conforman un componente como mangueras, rotores o tuercas. Ver anexos, tabla 7: Detalle de entidad Part.
- **Screen:** Entidad complementaria que representa información específica sobre harneros, como su tamaño y capacidad. Ver anexos, tabla 8: Detalle entidad Screen.
- **Crusher:** Entidad complementaria que representa información específica sobre chancadores, como su capacidad. Ver anexos, tabla 9: Detalle entidad Crusher.

- **Sensor:** Entidad complementaria que representa información específica sobre sensores, como la magnitud y frecuencia de medición. Ver anexos, tabla 10: Detalle entidad Sensor.
- **Failure:** Almacena información relacionada a las fallas de un equipo, subunidad o componente, e indica si la falla esta activa o es histórica, se basa en la tabla 6: *Failure data* de la norma ISO 14224:2016. Ver anexos, tabla 11: Detalle entidad Failure.
- **Report:** Contiene un resumen de los reportes generados por el software RCM Tool V2, este puede estar asociado a un equipo o a una parte de este. Ver anexos, tabla 12: Detalle entidad Report.
- **Maintenance:** Almacena información relacionada a las mantenciones realizadas o por realizar a un equipo, subunidad o componente, se basa en la tabla 8: *Maintenance data* de la norma ISO 14224:2016. Ver anexos, tabla 13: Detalle entidad Maintenance.
- **Collection:** Representa un grupo de eventos de mantenimiento asociados a un equipo, que están relacionados entre sí, estos pueden ser fallas, mantenimientos y/o reportes. Ver anexos, tabla 14: Detalle entidad Collection.
- **Failure mechanism:** Contiene códigos que representan y especifican el mecanismo de falla, por ejemplo, falla mecánica o falla eléctrica. Se basa en la tabla *B.2 – Failure mechanism* de la norma ISO 14224:2016. Ver anexos, tabla 15: Detalle entidad Failure mechanism.
- **Failure cause:** Contiene códigos que representan la causa de falla. Se basa en la tabla *B.3 – Failure causes* de la norma ISO 14224:2016. Ver anexos, tabla 16: Detalle entidad Failure cause.
- **Detection method:** Contiene códigos que representan como se detectó una falla, por ejemplo, observación casual o mantenimiento periódico. Se basa en la tabla *B.4 – Detection method* de la norma ISO 14224:2016. Ver anexos, tabla 17: Detalle entidad Detection method.
- **Failure mode:** Contiene códigos que representan el modo de falla de un elemento. Se basa en las tablas *B.6* a *B.15* de la norma ISO 14224:2016. Ver anexos, tabla 18: Detalle entidad Failure mode.
- **Maintenance activity:** Contiene códigos que representan la actividad de mantención que se le realizó o realizara a un elemento de la jerarquía. Se basa en la tabla *B.5 – Maintenance activity* de la norma ISO 14224:2016. Ver anexos, tabla 19: Detalle entidad Maintenance activity.

3.3.6 Modelo relacional

En base al diagrama de la sección anterior, se realizó el siguiente esquema relacional, en el cual se han agregado algunos atributos y restricciones para obtener la estructura final de la base de datos, entre ellas se destacan las tablas intermedias *Collection_failures*, *Collection_reports* y *Collection_maintenances*, tal como se muestra a continuación:

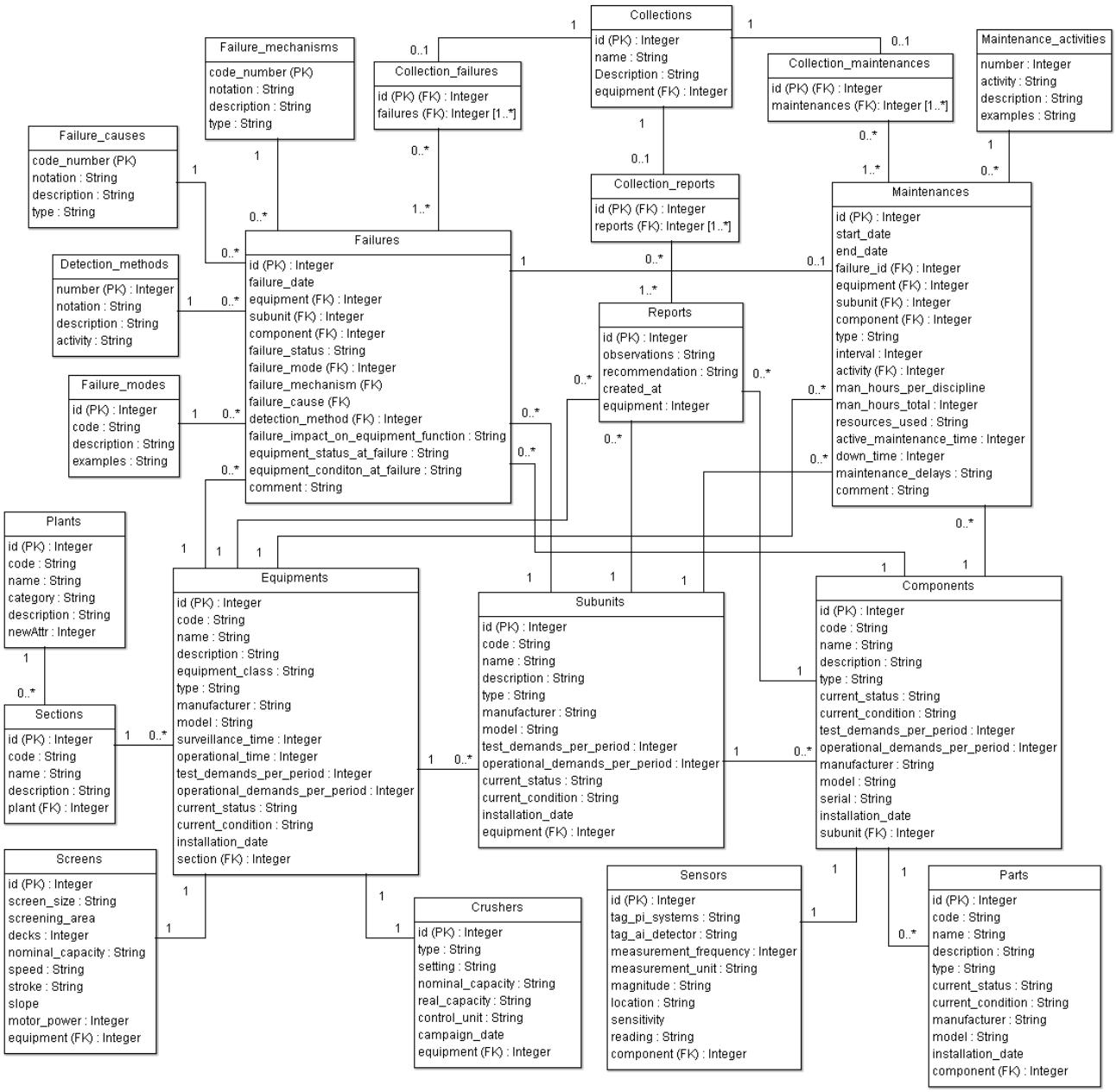

Figura 6 Modelo relacional de la base de datos de LogBook.

4 Implementación

En este capítulo se detallan las distintas herramientas utilizadas para el desarrollo del proyecto, junto con las distintas razones de porque estas fueron escogidas en primer lugar.

4.1 Tecnologías utilizadas

Para llevar a cabo la implementación de la aplicación, se eligieron tecnologías que cumplieran con los requerimientos establecidos. Entre estas, se encuentran FastAPI para el desarrollo del backend, y PostgreSQL como el servidor de bases de datos. A continuación, se puede observar el cómo estas se relacionan:

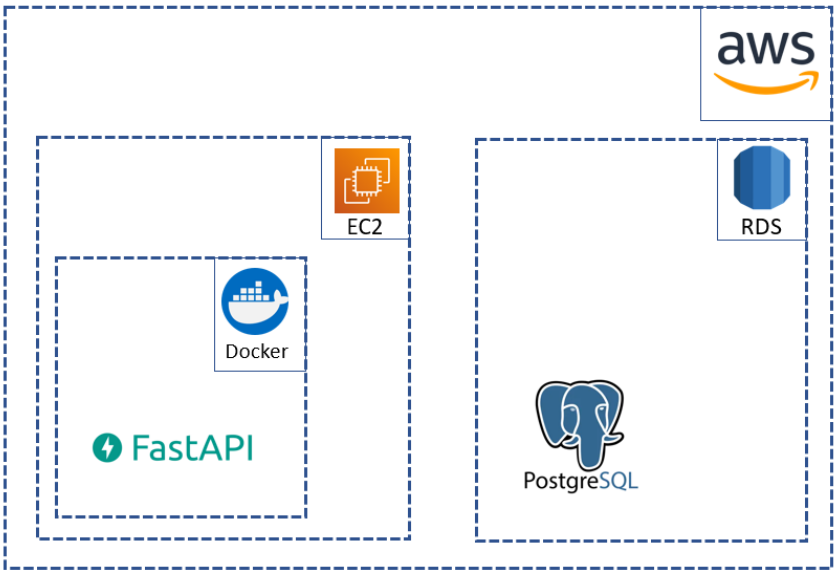

Figura 7 Diagrama de tecnologías de LogBook.

4.1.1 FastAPI

FastAPI [5] es un framework web de última generación, de código abierto, enfocado en la construcción rápida y eficiente de APIs web utilizando Python 3.8 o superior. Se basa en estándares como JSON y OpenAPI, entre otros, e incluye características que facilitan el despliegue como validación automática de errores, manejo de errores, y la generación automática de documentación siguiendo los estándares de OpenAPI. Otro de sus puntos fuertes es el uso de la programación asíncrona, lo que permite que múltiples tareas se ejecuten al mismo tiempo sin interferir entre sí.

Se optó por FastAPI por ser fácil de comprender, poseer buena documentación, y familiaridad con Python.

4.1.2 Docker

Docker [6] es una plataforma de código abierto que permite crear, testear y desplegar aplicaciones de manera rápida, junto con esto, brinda la capacidad de separar las aplicaciones de la infraestructura en donde se están desarrollando, lo que permite que puedan ser ejecutadas independiente de la máquina en la que se encuentren. Principalmente, Docker facilita la creación de contenedores que encapsulan las aplicaciones y todo lo que esta necesita para ser ejecutada, como librerías y herramientas de sistema.

Debido a la facilidad de manejo, flexibilidad brindada y a la baja curva de aprendizaje es que se optó por Docker para desplegar la aplicación.

4.1.3 PostgreSQL

PostgreSQL [7], o simplemente Postgres, es un sistema de gestión de bases de datos relacional orientado a objetos de código abierto. Su desarrollo está gestionado por una comunidad de desarrolladores voluntarios, independientes o respaldados por empresas, conocida como el PGDG (PostgreSQL Global Development Group).

Se optó por usarlo en el proyecto debido a su solidez, sencilla configuración y confiabilidad para crear un servidor SQL.

4.1.4 AWS

Amazon Web Services [8] es una plataforma que ofrece soluciones basadas en computación en la nube, flexibles, fiables, escalables, intuitivas y de bajo costo. Esta plataforma combina infraestructura como servicio (IaaS), plataforma como servicio (PaaS) y paquetes de software como servicio (SaaS).

En la arquitectura del proyecto, AWS se refiere al ecosistema que integra y pone a disposición del público los servicios implementados en la aplicación, específicamente en la región us-east-1.

4.1.5 Postman

Postman [9] es una herramienta para el desarrollo y prueba de APIs. Permite a los desarrolladores enviar solicitudes HTTP a un servidor y analizar las respuestas sin necesidad de escribir código. Además, provee herramientas para la documentación y colaboración en equipo, lo que hace que sea una herramienta esencial para asegurar la calidad y funcionalidad de las APIs en proyectos de software.

5 Resultados

Este capítulo se centrará en presentar las pruebas realizadas para validar y asegurar que la aplicación desarrollada cumple con los requisitos establecidos. Para esto, se dará un vistazo a las funcionalidades principales que se implementaron, las cuales fueron testeadas con Postman. Como se mencionó anteriormente, este es un software que permite enviar peticiones http. Cada prueba consistió en mandar peticiones, variando los parámetros, a cada endpoint de la API, para verificar el resultado esperado de cada petición. Debido a la gran cantidad de endpoints, en este informe solo se incluyen las pruebas realizadas a las partes más utilizadas de la API. Luego de esto, se analizarán los comentarios y resultados obtenidos de encuestas de utilidad realizadas a los empleados de Simpro. El listado completo de endpoints, y su propósito se puede encontrar en los anexos, tabla 20.

5.1 Funcionalidades principales

Para las pruebas realizadas, se puso como objetivo que los tiempos de respuesta promedio fueran menores a un segundo. Esto debido a que el tiempo de una consulta a LogBook se suma al tiempo de procesamiento de las demás aplicaciones, por tanto, es crucial minimizar esto tanto como sea posible. Sin embargo, a veces pueden surgir tiempos de respuesta mayores a un segundo debido a la lejanía con los servidores de Amazon en Virginia, Estados Unidos, lugar en donde se alojan las aplicaciones de Simpro.

5.1.1 Jerarquía de equipos

- **GET** /hierarchy/plant
- **GET** /hierarchy/section
- **GET** /hierarchy/equipment
- **GET** /hierarchy/subunit
- **GET** /hierarchy/component

Estos endpoints son fundamentales, ya que permiten obtener la jerarquía completa de uno o más elementos en particular. La estructura devuelta por cada uno de estos, es una lista anidada que comienza en el nivel de la jerarquía con el mismo nombre, y desciende hacia los niveles inferiores. Por ejemplo, una petición a GET /hierarchy/plant, devolvería la siguiente estructura:

```
[ # Nivel de plantas
    {
        "code": "PSEC",
        "name": "Planta secundaria",
        # más campos
        "sections": [
            {
                "code": "PSEC-CHT",
                "name": "Chancado primario",
                # más campos
                "equipments": [
                    {
                        "code": "PSEC-CHT-CR006",
                        "name": "Chancador 006",
                        # más campos
                        "subunits": [
                            {
                                "code": "PSEC-CHT-CR006-SMT",
                                "name": "Sistema motriz chancador 6",
                                # más campos
                                "components": [
                                    {
                                        "code": "PSEC-CHT-CR006-SMT-M001",
                                        "name": "Motor 1 sistema motriz",
                                        # más campos
                                    }
                                ]
                            }
                        ]
                    }
                ]
            }
        ]
    }
]
```

Figura 8 Estructura jerárquica devuelta a partir del nivel de plantas.

Mientras que un llamado a GET /hierarchy/equipment devolvería:

```
[ # Nivel de equipos.
    {
        "code": "PSEC-CHT-SN009",
        "name": “Harnero 9",
        # más campos
        "subunits": [
            {
                "code": "PSEC-CHT-SN009-SMT",
                "name": "Sistema motriz harnero 9",
                # más campos
                "components": [
                    {
                        "code": "PSEC-CHT-SN009-SMT-M001",
                        "name": "Motor 1 sistema motriz",
                        # más campos
                    }
                ]
            }
        ]
    }
]
```

Figura 9 Estructura jerárquica devuelta a partir del nivel de equipos.

De esta forma, se puede observar de manera fácil el cómo se relacionan los distintos elementos, pudiendo obtener la ubicación física de un equipo, su información general, y las partes que lo componen. Ejemplos reales de usos de estos endpoints se pueden encontrar en las figuras 10 y 11 a continuación:

GET localhost:8006/logbook-container/hierarchy/plant Send

Params Authorization Headers (7) Body Scripts Tests Settings Cookies

Body Cookies Headers (4) Test Results Status: 200 OK Time: 2.17 s Size: 42.26 KB Save as example

Pretty Raw Preview Visualize JSON


```
[
    {
        "id": 1,
        "code": "PSEC",
        "name": "Planta secundaria",
        "category": "test category",
        "description": "Planta secundaria división Gabriela Mistral",
        "sections": [
            {
                "id": 3,
                "code": "PSEC-CHT",
                "name": "Chancado terciario",
                "description": "Chancado terciario planta secundaria División Gabriela Mistral",
                "plant": 1,
                "equipments": [
                    {
                        "id": 6,
                        "code": "PSEC-CHT-CR011",
                        "name": "Chancador CR011",
                        "description": "Chancador 11",
                        "equipment_class": "crusher",
                        "type": "cone",
                        "manufacturer": "Metso",
                        "model": "MP1000",
                        "surveillance_time": 1000,
                        "operational_time": 214,
                        "test_demands_per_period": 4,
                        "operational_demands_per_period": 4,
                        "current_status": "operating",
                        "current_condition": "excellent",
                        "installation_date": "2024-05-28T16:45:25.965496Z",
                        "section": 3,
                        "subunits": [
                            {
                                "id": 12,
                                "code": "PSEC-CHT-CR011-STRUCT",
                                "name": "STRUCT CR011",
                                "description": "Subunidad estructural del chancador 11",
                                "type": "structural",
                                "manufacturer": "test manufacturer 6",
                                "model": "test model 6",
                                "test_demands_per_period": 0,
                                "operational_demands_per_period": 0,
                                "current_status": "operating",
                                "current_condition": "excellent",
                                "installation_date": "2024-05-28T16:45:32.688416Z",
                                "equipment": 6,
                                "components": [
                                    {
                                        "id": 90,
                                        "code": "PSEC-CHT-CR011-STRUCT-CR",
                                        "name": "estructura chancador",
                                        "description": "Estructura chancador CR0011",
                                        "type": "structural",
                                        "current_status": "operating",
                                        "current_condition": "excellent",
                                        "test_demands_per_period": 1,
                                        "operational_demands_per_period": 1,
                                        "manufacturer": "-",
                                        "model": "-",
                                        "serial": "-",
                                        "installation_date": "2024-05-28T16:46:44.404662Z",
                                        "subunit": 12
                                    }
                                ]
                            }
                        ]
                    }
                ]
            }
        ]
    }
]
```


Figura 10 Ejemplo de obtención de la jerarquía de todas las plantas.

```
[
    {
        "id": 1,
        "code": "PSEC-CHT-CR006",
        "name": "Chancador CR006",
        "description": "Chancador 6",
        "equipment_class": "crusher",
        "type": "cone",
        "manufacturer": "Metso",
        "model": "MP1000",
        "surveillance_time": 1200,
        "operational_time": 750,
        "test_demands_per_period": 4,
        "operational_demands_per_period": 4,
        "current_status": "operating",
        "current_condition": "excellent",
        "installation_date": "2024-05-28T16:45:25.965496Z",
        "section": 3,
        "subunits": [
            {
                "id": 1,
                "code": "PSEC-CHT-CR006-CTRL",
                "name": "CTRL CR006",
                "description": "Grupo de sensores de control y monitoreo del chancador 6",
                "type": "control_monitor",
                "manufacturer": "test manufacturer 1",
                "model": "test model 1",
                "test_demands_per_period": 2,
                "operational_demands_per_period": 4,
                "current_status": "operating",
                "current_condition": "excellent",
                "installation_date": "2024-05-28T16:45:32.688416Z",
                "equipment": 1,
                "components": [
                    {
                        "id": 1,
                        "code": "PSEC-CHT-CR006-CTRL-AI",
                        "name": "Corriente Motor",
                        "description": "Corriente motor CR006",
                        "type": "SCADA",
                        "current_status": "operating",
                        "current_condition": "excellent",
                        "test_demands_per_period": 1,
                        "operational_demands_per_period": 1,
                        "manufacturer": "-",
                        "model": "-",
                        "serial": "-",
                        "installation_date": "2024-05-28T16:45:48.700533Z",
                        "subunit": 1
                    },
                    {
                        "id": 2,
                        "code": "PSEC-CHT-CR006-CTRL-TIT157",
                        "name": "Temp. aceite buje ex. CR006",
                        "description": "Temperatura aceite buje excentrica CR006",
                        "type": "SCADA",
                        "current_status": "operating",
                        "current_condition": "excellent",
                        "test_demands_per_period": 1,
                        "operational_demands_per_period": 1,
                        "manufacturer": "-",
                        "model": "-",
                        "serial": "-",
                        "installation_date": "2024-05-28T16:45:48.700533Z",
                        "subunit": 1
                    }
                ]
            }
        ]
    },
    {
        "id": 7,
        "code": "PSEC-CHT-SCR006",
        "name": "Harnero SCR006",
        "description": "Harnero 6",
        "equipment_class": "screen",
        "type": "linear",
        "manufacturer": "Lokotrack",
        "model": "ST2.3",
        "surveillance_time": 800,
        "operational_time": 600,
        "test_demands_per_period": 4,
        "operational_demands_per_period": 4,
        "current_status": "operating",
        "current_condition": "excellent",
        "installation_date": "2024-05-29T15:03:29.059713Z",
        "section": 3,
        "subunits": []
    }
]
```


Figura 11 Ejemplo de obtención de la jerarquía de todos los equipos.

5.1.2 Eventos de mantenimiento

- **GET** /failures
- **GET** /maintenances
- **GET** /reports

Estos endpoints son esenciales para obtener un historial de lo que ha pasado con un equipo y así una visión integral del estado de los distintos equipos.

Las fallas indican la ubicación del daño físico o mal funcionamiento de un equipo, especificando que parte del mismo está siendo afectada, contiene códigos para indicar el mecanismo, modo y causa de falla junto con el método de detección de esta. Por último, se incluye un campo para indicar si la falla esta activa o es histórica.

Por otro lado, por cada falla activa surge la necesidad de registrar una mantención, dentro de los detalles de una mantención se indican la fecha de inicio y termino, el tipo de mantención, horas invertidas por disciplina, recursos utilizados, entre otros.

Respecto a los reportes, estos son realizados por personal en terreno, contienen observaciones y recomendaciones sobre el estado o condición de un equipo, ya sea una estructura dañada, componente desgastado u otro. Ejemplos de uso de cada endpoint se pueden encontrar a continuación:

GET localhost:8006/logbook-container/failures Send

Params Authorization Headers (7) Body Scripts Tests Settings Cookies

Body 200 OK 455 ms 1.71 KB Save as example

Pretty Raw Preview Visualize JSON


```
[
    {
        "id": 1,
        "failure_date": "2024-05-28T16:48:42.231002Z",
        "equipment": 1,
        "subunit": 7,
        "component": null,
        "failure_status": "active",
        "failure_mode": null,
        "failure_mechanism": null,
        "failure_cause": null,
        "detection_method": "Casual observation",
        "failure_impact_on_equipment_function": "degraded",
        "equipment_status_at_failure": "operating",
        "equipment_condition_at_failure": "good",
        "comment": "Quema bujes desarme chancador para cambio corazas"
    },
    {
        "id": 2,
        "failure_date": "2024-05-28T16:48:42.231002Z",
        "equipment": 1,
        "subunit": 7,
        "component": null,
        "failure_status": "active",
        "failure_mode": null,
        "failure_mechanism": null,
        "failure_cause": null,
        "detection_method": "Casual observation",
        "failure_impact_on_equipment_function": "degraded",
        "equipment_status_at_failure": "operating",
        "equipment_condition_at_failure": "fair",
        "comment": "Paso inchancable, cambio coraza"
    },
    {
        "id": 3,
        "failure_date": "2024-05-28T16:48:42.231002Z",
        "equipment": 2,
        "subunit": 8,
        "component": null,
        "failure_status": "active",
        "failure_mode": null,
        "failure_mechanism": null,
        "failure_cause": null,
        "detection_method": "Casual observation",
        "failure_impact_on_equipment_function": "degraded",
        "equipment_status_at_failure": "operating",
        "equipment_condition_at_failure": "excellent",
        "comment": "Paso inchancable"
    }
]
```


Figura 12 Ejemplo de obtención de fallas.

GET localhost:8006/logbook-container/maintenances Send

Params Authorization Headers (7) Body Scripts Tests Settings Cookies

Body 200 OK 440 ms 1.63 KB Save as example

Pretty Raw Preview Visualize JSON


```
[
    {
        "id": 1,
        "start_date": "2024-05-28T16:48:47.970928Z",
        "end_date": "2024-05-28T16:48:47.970928Z",
        "failure_id": 2,
        "equipment": 1,
        "subunit": 7,
        "component": null,
        "type": "corrective",
        "interval": null,
        "activity": "Replace",
        "man_hours_per_discipline": {
            "mechanic": 3
        },
        "man_hours_total": 3,
        "resources_used": "-",
        "active_maintenance_time": 0,
        "down_time": 3,
        "maintenance_delays": "-",
        "comment": "Cambio de coraza"
    },
    {
        "id": 2,
        "start_date": "2024-05-28T16:48:47.970928Z",
        "end_date": "2024-05-28T16:48:47.970928Z",
        "failure_id": 1,
        "equipment": 1,
        "subunit": 7,
        "component": null,
        "type": "corrective",
        "interval": null,
        "activity": "Replace",
        "man_hours_per_discipline": {
            "mechanic": 5
        },
        "man_hours_total": 5,
        "resources_used": "-",
        "active_maintenance_time": 2,
        "down_time": 3,
        "maintenance_delays": "-",
        "comment": "Cambio de bujes"
    },
    {
        "id": 3,
        "start_date": "2024-05-28T16:48:47.970928Z",
        "end_date": "2024-05-28T16:48:47.970928Z",
        "failure_id": 3,
        "equipment": 2,
        "subunit": 8,
        "component": null,
        "type": "preventive",
        "interval": 90,
        "activity": "Replace",
        "man_hours_per_discipline": {
            "mechanic": 3
        },
        "man_hours_total": 3,
        "resources_used": "-",
        "active_maintenance_time": 0,
        "down_time": 3,
        "maintenance_delays": "-",
        "comment": "Cambio de coraza"
    },
]
```


Figura 13 Ejemplo de obtención de mantenciones.

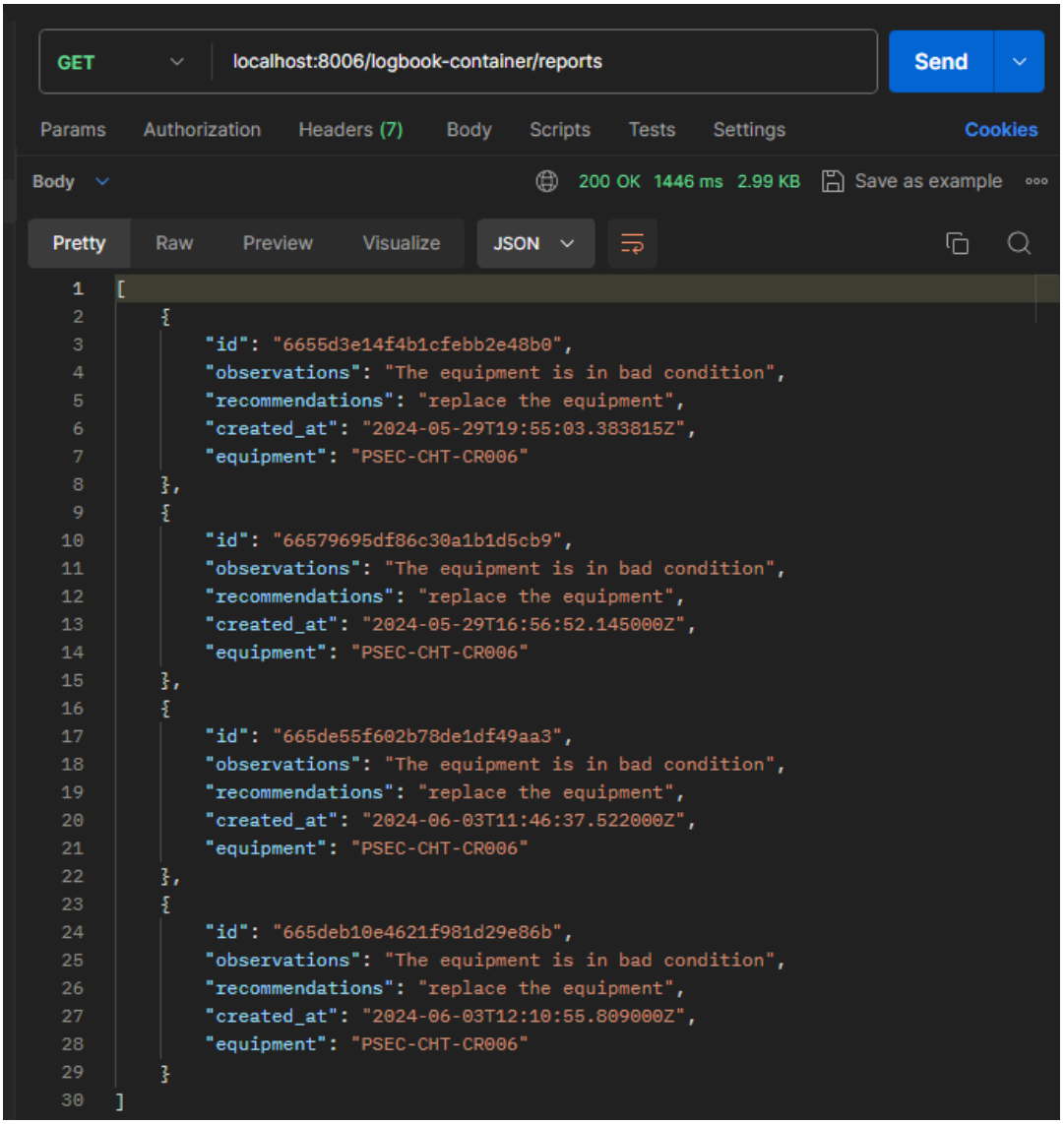

Figura 14 Ejemplo de obtención de reportes.

5.1.3 Información específica de cada equipo

- **GET** /hierarchy/equipment/crusher
- **GET** /hierarchy/equipment/screens
- **GET** /hierarchy/equipment/crusher/{crusher_id}/sensor-data
- **GET** /hierarchy/component/sensor

Permiten acceder a la información técnica que es única para cada tipo de equipo, ya sea capacidad nominal, capacidad real, velocidad de operación, inclinación u otra característica. Además, permite obtener datos de sensores asociados a los equipos, lo que es particularmente útil para realizar mantenimiento predictivo y análisis de rendimiento. Ejemplos de uso se pueden encontrar en las siguientes figuras:

GET localhost:8006/logbook-container/hierarchy/equipment/crusher/ Send

Params Authorization Headers (7) Body Scripts Tests Settings Cookies

Body Cookies Headers (4) Test Results 200 OK 422 ms 1.4 KB Save as example

Pretty Raw Preview Visualize JSON


```
[
    {
        "id": 1,
        "type": "tertiary",
        "setting": "38 mm",
        "nominal_capacity": "1750 mtph",
        "real_capacity": "1650 mtph",
        "control_unit": "PLC-SAP-006",
        "campaign_date": "2024-05-28T16:47:01.534992Z",
        "equipment": 1,
        "code": "PSEC-CHT-CR006"
    },
    {
        "id": 2,
        "type": "tertiary",
        "setting": "38 mm",
        "nominal_capacity": "1750 mtph",
        "real_capacity": "1650 mtph",
        "control_unit": "PLC-SAP-007",
        "campaign_date": "2024-05-28T16:47:01.534992Z",
        "equipment": 2,
        "code": "PSEC-CHT-CR007"
    },
    {
        "id": 3,
        "type": "tertiary",
        "setting": "25 mm",
        "nominal_capacity": "1210 mtph",
        "real_capacity": "1100 mtph",
        "control_unit": "PLC-SAP-008",
        "campaign_date": "2024-05-28T16:47:01.534992Z",
        "equipment": 3,
        "code": "PSEC-CHT-CR008"
    },
    {
        "id": 4,
        "type": "tertiary",
        "setting": "25 mm",
        "nominal_capacity": "1210 mtph",
        "real_capacity": "1100 mtph",
        "control_unit": "PLC-SAP-009",
        "campaign_date": "2024-05-28T16:47:01.534992Z",
        "equipment": 4,
        "code": "PSEC-CHT-CR009"
    },
]
```


Figura 15 Ejemplo de obtención de información específica sobre chancadores.

GET localhost:8006/logbook-container/hierarchy/equipment/screens/ Send

Params Authorization Headers (7) Body Scripts Tests Settings Cookies

Body Cookies Headers (4) Test Results 200 OK 425 ms 682 B Save as example

Pretty Raw Preview Visualize JSON


```
[
    {
        "id": 1,
        "screen_size": "10' x 5'",
        "screening_area": 4.6,
        "decks": 2,
        "nominal_capacity": "350 mtph",
        "speed": "1800 to 3600",
        "stroke": "up to 13 mm",
        "slope": 25.5,
        "motor_power": 55,
        "equipment": 7
    },
    {
        "id": 2,
        "screen_size": "10' x 5'",
        "screening_area": 4.6,
        "decks": 2,
        "nominal_capacity": "350 mtph",
        "speed": "1800 to 3600",
        "stroke": "up to 13 mm",
        "slope": 25.5,
        "motor_power": 55,
        "equipment": 8
    },
    {
        "id": 3,
        "screen_size": "10' x 5'",
        "screening_area": 4.6,
        "decks": 2,
        "nominal_capacity": "350 mtph",
        "speed": "1800 to 3600",
        "stroke": "up to 13 mm",
        "slope": 25.5,
        "motor_power": 55,
        "equipment": 9
    }
]
```


Figura 16 Ejemplo de obtención de información específica sobre harneros.

GET localhost:8006/logbook-container/hierarchy/equipment/crusher/1/sensor-data Send

Params Authorization Headers (7) Body Scripts Tests Settings Cookies

Body Cookies Headers (4) Test Results 200 OK 426 ms 1.21 KB Save as example

Pretty Raw Preview Visualize JSON


```
[
    {
        "tag_ai_detector": "CR006_AI",
        "magnitude": "current",
        "crusher_code": null
    },
    {
        "tag_ai_detector": "CR006_TIT157",
        "magnitude": "temperature",
        "crusher_code": null
    },
    {
        "tag_ai_detector": "CR006_TIT158",
        "magnitude": "temperature",
        "crusher_code": null
    },
    {
        "tag_ai_detector": "CR006_TIT159",
        "magnitude": "temperature",
        "crusher_code": null
    },
    {
        "tag_ai_detector": "CR006_TIT161",
        "magnitude": "temperature",
        "crusher_code": null
    },
    {
        "tag_ai_detector": "CR006_TIT162",
        "magnitude": "temperature",
        "crusher_code": null
    },
    {
        "tag_ai_detector": "CR006_TIT165",
        "magnitude": "temperature",
        "crusher_code": null
    },
    {
        "tag_ai_detector": "CR006_PDIT163",
        "magnitude": "pressure",
        "crusher_code": null
    },
    {
        "tag_ai_detector": "CR006_PIT160",
        "magnitude": "pressure",
        "crusher_code": null
    },
    {
        "tag_ai_detector": "CR006_VIT155A",
        "magnitude": "vibration",
        "crusher_code": null
    },
    {
        "tag_ai_detector": "CR006_VIT155B",
        "magnitude": "vibration",
        "crusher_code": null
    },
    {
        "tag_ai_detector": "CR006_VIT155C",
        "magnitude": "vibration",
        "crusher_code": null
    },
    {
        "tag_ai_detector": "CR006_VIT155D",
        "magnitude": "vibration",
        "crusher_code": null
    },
    {
        "tag_ai_detector": "CR006_ZI166",
        "magnitude": "setting",
        "crusher_code": null
    }
]
```


Figura 17 Ejemplo de obtención de los sensores asociados a un chancador.

5.2 Retroalimentación de los usuarios

Durante el periodo la memoria, el equipo que estaba a cargo del desarrollo de la suite TCMS consto de cuatro personas, las cuales interactuaron con LogBook a través de peticiones http realizadas en los lenguajes de programación Python y Dart. Una de las peticiones más realizadas fue la obtención de la jerarquía de una planta para desplegarla en la interfaz gráfica de las distintas aplicaciones, con tal de que los usuarios puedan navegar fácilmente por los distintos elementos de la misma. Ejemplo de esto se puede ver en las figuras 18, 19 y 20 en donde se observa la jerarquía de la planta *"División Gabriela Mistral"* (DGM):

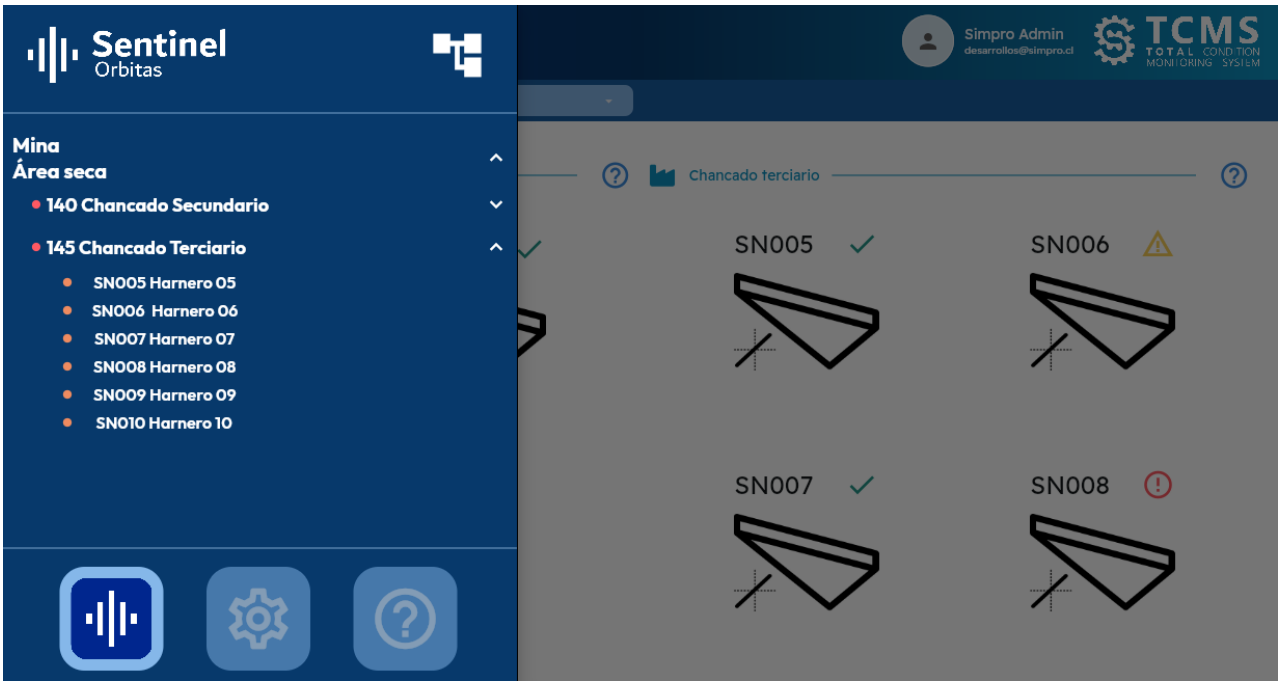

Figura 18 Drawer de la interfaz gráfica de Sentinel (TCMS) con información proveniente de LogBook.

Figura 19 Drawer de la interfaz gráfica de AI Detector (TCMS) con información proveniente de LogBook.

Figura 20 Lista de chancadores en AI Detector, cargada desde LogBook.

Cabe aclarar que en la figura 20 que solo los nombres de los chancadores y la información de la barra superior celeste son extraídos de LogBook, mientras que para los apartados *Telemetría* e *IA* la información se obtiene de otros modulos de AI Detector.

Junto con esto, en AI Detector, se realizaban peticiones para obtener el código y la magnitud de los sensores asociados a cada chancador, con tal de poder analizar las mediciones provenientes de estos y aplicar un nivel de alerta en caso de que existan valores anómalos, según las predicciones realizadas por modelos de inteligencia artificial, ejemplo de esto se puede ver en la figura 21 a continuación:

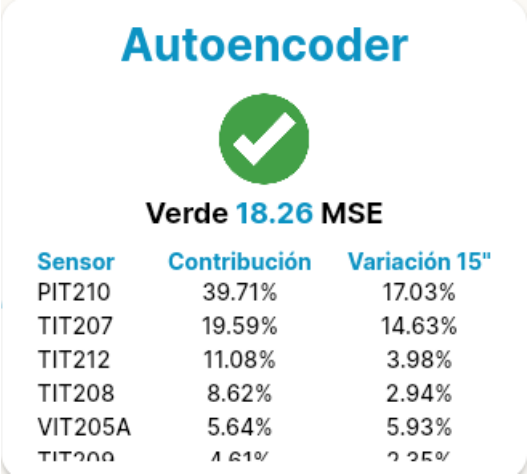

Figura 21 Ejemplo de predicción modelo Autoencoder en AI Detector.

Por otro lado, en caso de que a los miembros del equipo les surgiera duda respecto a cómo solicitar información de LogBook, se les proveyó de documentación en el repositorio de Github del proyecto en donde se detalla de manera exhaustiva el uso de cada endpoint de la API, lo que incluye ejemplos de lo que se debiera devolver como respuesta para cada uno de estos, y la descripción de cada parámetro del body en caso de peticiones POST y PUT. Esto se ve reflejado en las siguientes figuras:

GET: All components

Returns the hierarchy of all the components.

```
GET /hierarchy/component
```

Response Example


```
[
    {
        "id": 1,
        "code": "PSEC-CHT-CR006-CTRL-AI",
        "name": "Corriente Motor",
        "description": "Corriente motor CR006",
        "type": "SCADA",
        "current_status": "operating",
        "current_condition": "excellent",
        "test_demands_per_period": 4,
        "operational_demands_per_period": 4,
        "manufacturer": "Example Manufacturer Co.",
        "model": "lorem ipsum",
        "serial": "lorem ipsum",
        "installation_date": "2024-04-25T14:44:18",
        "subunit": 1
    }
]
```


Figura 22 Documentación del endpoint GET /hierarchy/component.

GET: All crushers

Returns technical specific data of all crushers.

```
GET /hierarchy/equipment/crusher/
```

Response Example


```
[
    {
        "id": 1,
        "type": "tertiary",
        "setting": "38 mm",
        "nominal_capacity": "1750 mtph",
        "real_capacity": "1650 mtph",
        "control_unit": "PLC-SAP-006",
        "campaign_date": "2024-05-27T16:55:52.351776+00:00",
        "equipment": 1,
        "code": "PSEC-CHT-CR006"
    }
]
```


Figura 23 Documentación del endpoint GET /hierarchy/equipment/crusher.

POST: Add a new maintenance

Add a new maintenance to the database.

```
POST /maintenances
```

Request body


```
{
    "start_date": "2024-05-27T15:05:34",
    "end_date": "2024-05-27T15:05:34",
    "failure_id": 1,
    "equipment": 1,
    "subunit": 1,
    "component": 1,
    "type": "str",
    "interval": 5,
    "activity": "int",
    "man_hours_per_discipline": {
        "discipline_1": 5,
        "discipline_2": 2
    },
    "man_hours_total": 7,
    "resources_used": "str",
    "active_maintenance_time": 1,
    "down_time": 1,
    "maintenance_delays": "str",
    "comment": "str"
}
```


Figura 24 Documentación del método POST /maintenances.

Parameter	Type	Description
`start_date`	`datetime`	**Required**. Date when the maintenance started
`end_date`	`datetime`	**Required**. Date when the maintenance ended
`failure_id`	`int`	**Required**. Id of the failure that was treated
`equipment`	`int`	**Required**. Equipment id
`subunit`	`int`	**Optional**. Subunit id
`component`	`int`	**Optional**. Component id
`type`	`str`	**Required**. if the maintenances was predictive or corrective
`interval`	`int`	**Required**. If the maintenance is predictive, it indicates the period of time between maintenances
`activity`	`int`	**Required**. Number code of the activity that was performed
`man_hours_per_discipline`	`dict[str, int]`	**Required**. Indicates the distribution of time by discipline
`man_hours_total`	`int`	**Required**. Indicates total maintenance time
`resources_used`	`str`	**Required**. e.g. intervention vessel, crane
`active_maintenance_time`	`int`	**Required**. Time duration for active maintenance work being done on the equipment
`down_time`	`int`	**Required**. Time duration during which an item is in a down state
`maintenance_delays`	`str`	**Required**. Prolonged down time causes, e.g. logistics, weather, scaffolding.
`comment`	`str`	**Optional**. Field to add extra observations

Figura 25 Descripción de los parámetros del body del método POST /maintenances.

Response Example

```
{
    "message": "TABLE maintenances: object was added successfully"
}
```

Figura 26 Ejemplo de respuesta del método POST /maintenances.

Además, a los integrantes del equipo se les pidió que contestaran una encuesta por medio de Google Forms. Debido a la naturaleza de la aplicación, existe poca o nula interacción humana con esta, por lo que la encuesta se centra en evaluar que tan bien se integró LogBook con las demás aplicaciones de la suite TCMS. Esta consta de 5 preguntas descritas a continuación, con una escala del 1 al 5, en donde 1 es muy en desacuerdo y 5 muy de acuerdo:

1. ¿La documentación de la API es clara y fácil de entender?
2. ¿La API cubre todas las funcionalidades necesarias para la gestión de activos y el registro de eventos de mantenimiento?
3. ¿Fue fácil integrar la API con otras aplicaciones de la empresa?
4. ¿Las respuestas de la API son rápidas y eficientes?
5. ¿La API maneja adecuadamente los errores y proporciona mensajes útiles?

Las respuestas obtenidas fueron las siguientes:

Pregunta	**Integrante 1**	**Integrante 2**	**Integrante 3**	**Integrante 4**	**Promedio**
Pregunta 1	5	5	4	4	4.5
Pregunta 2	5	5	5	5	5
Pregunta 3	4	5	4	5	4.5
Pregunta 4	5	5	5	5	5
Pregunta 5	4	5	4	3	4

Tabla 1 Respuestas encuesta de satisfacción.

En general se puede observar que la API fue bien evaluada, cumpliendo con los objetivos establecidos y permitiendo agilizar el proceso de desarrollo del equipo.

Junto con esto, la mayoría de los integrantes del equipo recalco la importancia de contar con mensajes de error más descriptivos a futuro, ya que, debido a la restricción de tiempo, gran parte de estos son genéricos y podrían proporcionar información de mayor utilidad.

6 Conclusiones

Durante el desarrollo de este proyecto, se implementó la API LogBook, diseñada para centralizar la información técnica y eventos de mantenimiento de equipos en la industria minera. LogBook facilita la gestión de inventarios y el seguimiento histórico de eventos, permitiendo mejorar la eficiencia del resto de aplicaciones de la suite TCMS al delegar estas responsabilidades en sí.

Se evaluaron diversas herramientas para su realización, tomando en cuenta sus fortalezas y debilidades.

En general, la experiencia adquirida a lo largo del desarrollo de LogBook fue enriquecedora, me permitió aplicar conocimientos teóricos en un contexto práctico y mejorar las habilidades relacionadas al desarrollo de software y gestión de proyectos.

Se destaca que la evaluación inicial de la API LogBook es bastante positiva, ya que el sistema responde adecuadamente a las necesidades identificadas al inicio del proyecto

Sin embargo, debido al cambio de prioridades de la empresa hacia el final del proyecto, el desarrollo de la interfaz visual de LogBook, la cual haría uso de todas las funcionalidades implementadas en la API, se ha postergado hasta plazo indefinido, por lo que, en la actualidad, no se hace uso de la API más allá de lo enseñado en este informe.

7 Trabajo futuro

A pesar de que la realización del presente proyecto ha permitido la creación de un prototipo funcional, es necesario continuar con su desarrollo con tal de aumentar sus funcionalidades y optimizar su rendimiento. Entre los posibles trabajos futuros se encuentra:

- **Poder filtrar la información:** con el fin de mejorar la precisión de la búsqueda al momento de obtener datos de interés, algunos ejemplos de esto serian poder obtener todos los equipos que tengan más de tres fallas activas u obtener los equipos con una antigüedad mayor a 5 años.
- **Mensajes de error:** Como se recalcó por parte del equipo de la suite TCMS, es necesario mejorar los mensajes de error de la plataforma, para que den información de mayor utilidad.
- **Desarrollar una interfaz gráfica:** Como se mencionó en la sección 1.7, a futuro será necesario incorporar una interfaz que permita a la persona a cargo revisar de forma rápida y eficiente toda la información contenida en LogBook, de momento, una de las funcionalidades clave que debe de tener esta interfaz es el permitir declarar los campos de una nueva tabla dentro de la base de datos, en caso de que se deba de almacenar información sobre un tipo de elemento que no esté registrado en esta.
- **Aplicación de realidad virtual para móviles:** con los datos proporcionados por LogBook se considera realizar una aplicación de realidad virtual enfocada a simular e interactuar con las diversas partes de una máquina en un entorno seguro, con el fin de preparar al personal de una planta para operaciones de alto riesgo. Para esto, será necesario investigar sobre los datos que se requieren para crear simulaciones precisas y cómo integrar la API de LogBook con tecnologías de realidad virtual.

8 Bibliografía

[1] R. Pascual, V. Meruane y P. Rey, «On the effect of downtime costs and budget constraint on preventive and replacement policies,» 2008.

[2] K. S. Gaeld y H. W. Ping, «Wavelet fault diagnosis and tolerant of induction motor: A review,» 2011.

[3] G. Erboz, «How to Define Industry 4.0: The Main Pillars Of Industry 4.0.,» 2017.

[4] I. O. f. S. ISO, «ISO 14224:2016 - Petroleum, petrochemical and natural gas industries — Collection and exchange of reliability and maintenance data for equipment».

[5] FastAPI, «FastAPI Home page,» [En línea]. Available: https://fastapi.tiangolo.com. [Último acceso: 10 abril 2024].

[6] Docker, «Docke Home Page,» [En línea]. Available: https://www.docker.com. [Último acceso: 10 abril 2024].

[7] PostgreSQL, «PostgreSQL Home Page,» [En línea]. Available: https://www.postgresql.org. [Último acceso: 10 abril 2024].

[8] A. W. S. AWS, «AWS Home Page,» [En línea]. Available: https://aws.amazon.com. [Último acceso: 10 abril 2024].

[9] Postman, «Postman Home Page,» [En línea]. Available: https://www.postman.com. [Último acceso: 10 abril 2024].

9 Anexos

9.1 Detalle de las entidades del modelo entidad relación

Como aclaración, se asume que todas las entidades tienen un campo extra implícito llamado id, con excepción de las tablas 15 a 19. A continuación se explicara la razón de ser de esto:

Tabla 15: es una copia de la tabla B.2 de la norma ISO 14224:2016.

Tabla 16: es una copia de la tabla B.3 de la norma ISO 14224:2016.

Tabla 17: es una copia de la tabla B.4 de la norma ISO 14224:2016.

Tabla 18: es una compilación de la información presente en las tablas B.6 a B.15 de la norma ISO 14224:2016, la cual también puede incluir códigos o modos de falla personalizados en caso de ser requerido.

Tabla 19: es una copia de la tabla B.5 de la norma ISO 14224:2016.

Tabla 2 Detalle entidad Plant

Atributo	Descripción
code	Código único que utilizara LogBook para identificar la planta dentro del sistema.
name	Nombre de la planta.
category	Categoría de la planta.
description	Breve descripción de la planta.

Tabla 3 Detalle entidad Section

Atributo	Descripción
code	Código único que utilizara LogBook para identificar la sección.
name	Nombre de la sección, e.g. chancado primario.
description	Breve descripción de la sección.

Tabla 4 Detalle entidad Equipment

Atributo	Descripción
code	Código único del equipo
name	Nombre del equipo
description	Breve descripción del equipo
equipment class	Clase del equipo, e.g. bomba, motor o intercambiador de calor.
type	Tipo del equipo, e.g. eléctrico o hidráulico.
manufacturer	Nombre de la empresa que fabrica el equipo.
model	Modelo que el fabricante designa al equipo.
surveillance time	Cantidad de tiempo durante el cual se monitorea el estado, condición y funcionamiento del equipo. Se calcula de forma teórica tomando como punto de inicio la fecha de instalación del mismo. Incluye los momentos de operación, periodos de inactividad, periodos de mantención y down-time, entre otros. Se mide en horas.
operational time	Cantidad de tiempo durante el cual el equipo se encuentra operando normalmente. Se mide en horas
test demands per period	Cantidad de test realizados durante un periodo, la frecuencia de estos queda a cargo del planificador.
operational demands per period	Cantidad de demandas operacionales por periodo, la frecuencia de estos queda a cargo del planificador.
current status	Estado operativo actual del equipo, i.e. operando, en mantención o inactivo.
current condition	Condición actual del equipo, i.e. excelente, aceptable, pobre o crítica. Referente a daños físicos en la máquina.
installation date	Fecha y hora, incluyendo los segundos, en que el equipo empezó a operar.

Tabla 5 Detalle entidad Subunit

Atributo	Descripción
code	Código único de la subunidad.
name	Nombre de la subunidad.
description	Breve descripción de la subunidad.
type	Tipo de subunidad, e.g. sistema motriz, sistema de lubricación.
manufacturer	Nombre de la empresa que fabrica la subunidad.
model	Modelo que el fabricante asigno a la subunidad.
test demands per period	Cantidad de test realizados durante un periodo, la frecuencia de estos queda a cargo del planificador.
operational demands per period	Cantidad de demandas operacionales por periodo, la frecuencia de estos queda a cargo del planificador.
current status	Estado operativo actual de la subunidad, i.e. operando, en mantención o inactivo.
current condition	Condición actual de la subunidad, i.e. excelente, aceptable, pobre o crítica. Referente a daños físicos en la máquina.
installation date	Fecha y hora, incluyendo los segundos, en que la subunidad empezó a operar.

Tabla 6 Detalle entidad Component

Atributo	Descripción
code	Código único del componente
name	Nombre del componente
description	Breve descripción del componente
type	Tipo de componente, e.g. motor, condensador.
current status	Estado operativo actual del componente, i.e. operando, en mantención o inactivo.
current condition	Condición actual del componente, i.e. excelente, aceptable, pobre o crítica. Referente a daños físicos en la máquina.
test demands per period	Cantidad de test realizados durante un periodo, la frecuencia de estos queda a cargo del planificador.
operational demands per period	Cantidad de demandas operacionales por periodo, la frecuencia de estos queda a cargo del planificador.
manufacturer	Nombre de la empresa que fabrica el componente.
model	Modelo que el fabricante asigno al componente
serial	Código alfanumérico único asignado por el fabricante.
installation date	Fecha y hora, incluyendo los segundos, en que el componente empezó a operar.

Tabla 7 Detalle entidad Part

Atributo	Descripción
code	Código único de la parte.
name	Nombre de la parte.
description	Descripción breve de la parte.
type	Tipo de parte, e.g., manguera, rotor.
current status	Estado operativo actual de la parte, i.e. operando, en mantención o inactivo.
current condition	Condición actual de la parte, i.e. excelente, aceptable, pobre o crítica. Referente a daños físicos en la máquina.
manufacturer	Nombre de la empresa que fabrica la parte.
model	Modelo que el fabricante asigno a la parte.
installation date	Fecha y hora, incluyendo los segundos, en que la parte empezó a operar.

Tabla 8 Detalle entidad Screen

Atributo	Descripción
screen size	Largo y ancho de la superficie de cribado, en pulgadas.
screening area	Área de la superficie de cribado, en metros cuadrados.
decks	Número de decks del harnero.
nominal capacity	Capacidad nominal del harnero, en toneladas métricas por hora.
speed	Frecuencia a la que opera el harnero, medido en RPM.
stroke	La distancia entre los extremos de un movimiento lineal, o bien el diámetro de un movimiento circular, medido en milímetros.
slope	El ángulo con la horizontal formado por la primera sección de cribado, medido en grados.
motor power	Potencia del motor, medida en kilowatts.

Tabla 9 Detalle entidad Crusher

Atributo	Descripción
type	Tipo de chancador, e.g. primario, secundario o terciario.
setting	Closed side setting del crusher, se mide en milímetros.
nominal capacity	Capacidad teórica de material que puede procesar el chancador, medido en toneladas métricas por hora.
real capacity	Capacidad medida de material que puede procesar el chancador, medido en toneladas métricas por hora.
control unit	Código del PLC asociado al chancador.
campaign date	Fecha de mantenimiento del buje de un chancador, marca una campaña pues se debe cambiar la coraza entera del chancador.

Tabla 10 Detalle entidad Sensor

Atributo	Descripción
tag pi systems	Identificador único del sensor en pi systems, no pertenece a TCMS.
tag ai detector	Identificador único del sensor dentro de ai detector, pertenece a TCMS.
measurement frequency	Intervalo de tiempo entre medición y medición, en segundos.
measurement unit	La unidad en la que se mide la magnitud del sensor.
magnitude	La magnitud que mide el sensor, e.g. temperatura.
location	Ubicación física del sensor en un equipo.
sensitivity	Relación entre la variación de la magnitud de salida y la variación de la magnitud de entrada.
reading	Unidad de medida de la salida del sensor, se puede obtener la medición de un sensor con la fórmula: Medición = lectura * sensibilidad.

Tabla 11 Detalle entidad Failure

Atributo	Descripción
failure date	Fecha y hora en que se dio aviso de la falla, incluye segundos.
failure status	Indica si la falla esta activa o es histórica.
failure impact on equipment function	Efecto de la falla en el funcionamiento del equipo: fallo crítico, degradado o incipiente.
equipment status at failure	Estado del equipo al momento de la falla.
equipment condition at failure	Condición del equipo al momento de la falla.
comment	Campo para indicar información adicional sobre la falla.

Tabla 12 Detalle entidad Report

Atributo	Descripción
observations	Observaciones incluidas en el reporte.
recommendations	Recomendaciones incluidas en el reporte.
created at	Fecha y hora, con segundos, en que se creó el reporte.

Tabla 13 Detalle entidad Maintenance

Atributo	Descripción
start date	Fecha y hora, con segundos, en la que empezó la mantención.
end date	Fecha y hora con segundos en que termino la mantención.
type	Tipo de mantención, i.e. correctiva o preventiva.
ynterval	Aplica solo para mantención preventiva, intervalo de tiempo en días entre mantención y mantención.
man hours per discipline	Horas hombre de la mantención, por disciplina, e.g. mecánica, electricidad.
man hours total	Horas hombre totales.
resources used	Recursos utilizados, e.g. herramientas de diagnóstico, equipo de levantamiento.
active maintenance time	Tiempo de la mantención en el que el equipo se encuentra activo, i.e. operando, en espera, inactivo, entre otros.
down time	Tiempo de la mantención en el que el equipo no se encuentra activo.
maintenance delays	Tiempo de inactividad prolongado por causas logísticas, meteorológicas, retraso del personal de reparación, etc.
comment	Campo para indicar información adicional sobre la mantención.

Tabla 14 Detalle entidad Collection

Atributo	Descripción
Name	Nombre de la colección.
description	Descripción de la colección.

Tabla 15 Detalle entidad Failure mechanism

Atributo	Descripción
code number	Equivale a la columna ***Subdivision code number,*** identificador primario, flotante.
notation	Equivale a la columna ***Subdivision notation***.
description	Equivale a la columna ***Description of the faiure mechanism***.
type	Equivale a la columna ***Failure mechanism notation***.

Tabla 16 Detalle entidad Failure cause

Atributo	Descripción
code number	Equivale a la columna ***Subdivision code number,*** identificador primario, flotante.
notation	Equivale a la columna ***Subdivision of the failure cause***.
description	Equivale a la columna ***Description of the faiure cause***.
type	Equivale a la columna ***notation***.

Tabla 17 Detalle entidad Detection method

Atributo	Descripción
number	Equivale a la columna ***Number,*** identificador primario, entero.
notation	Equivale a la columna ***Notation***.
description	Equivale a la columna ***Description***.
activity	Equivale a la columna ***Activity***.

Tabla 18 Detalle entidad Failure mode

Atributo	Descripción
id	Identificador único, índice entero.
code	Código del modo de falla, compuesto de tres letras, e.g. AIR, BRD.
description	Descripción del código del modo de falla.
examples	Ejemplos de sucesos que abarca un modo de falla especifico.

Tabla 19 Detalle entidad Maintenance activity

Atributo	Descripción
number	Equivale a la columna ***Number,*** identificador primario, entero.
activity	Equivale a la columna ***Activity***.
description	Equivale a la columna ***Description***.
examples	Equivale a la columna ***Example***.

9.2 Lista completa de endpoints de la API

Tabla 20 Lista completa de endpoints de la API de LogBook

Método	Ruta	Descripción
GET	/hierarchy/plant	Devuelve una lista anidada con toda la información de cada planta y sus elementos.
GET	/hierarchy/plant/{id}	Devuelve una lista anidada con toda la información de una planta específica y sus elementos.
GET	/hierarchy/section	Devuelve una lista anidada con toda la información de cada sección y sus elementos.
GET	/hierarchy/section/{id}	Devuelve una lista anidada con toda la información de una sección específica y sus elementos.
GET	/hierarchy/equipment	Devuelve una lista anidada con toda la información de cada equipo y sus elementos.
GET	/hierarchy/equipment/{id}	Devuelve una lista anidada con toda la información de un equipo específico y sus elementos.
GET	/hierarchy/subunit	Devuelve una lista anidada con toda la información de cada subunidad y sus elementos.
GET	/hierarchy/subunit/{id}	Devuelve una lista anidada con toda la información de una subunidad específica y sus elementos.
GET	/hierarchy/component/	Devuelve una lista con la información de cada componente.
GET	/hierarchy/component/{id}	Devuelve la información de un componente específico.
GET	/hierarchy/equipment/crusher	Devuelve una lista con la información específica de todos los chancadores.
GET	/hierarchy/equipment/screens	Devuelve una lista con la información específica de todos los harneros.

GET	/hierarchy/equipment/crusher/ {crusher-id}/sensor-data	Devuelve una lista con el tag y la magnitud de todos los sensores asociados a un chancador.
GET	/hierarchy/equipment/crusher/ {crusher-id}/campaign-date	Devuelve la fecha de campaña de un chancador.
GET	/hierarchy/component/sensor	Devuelve la información de todos los sensores.
GET	/hierarchy/component/sensor/{id}	Devuelve la información de un sensor filtrado por id.
GET	/hierarchy/component/sensor/code/{code}	Devuelve la información de un sensor filtrado por código.
GET	/failures	Devuelve la información de todas las fallas, tanto activas como históricas. Se puede filtrar para un equipo, subunidad o componente en específico.
GET	/maintenances	Devuelve la información de todas las mantenciones. Se puede filtrar para un equipo, subunidad o componente en específico.
GET	/reports	Devuelve la información de todos los reportes hechos en terreno.
GET	/collections	Devuelve una lista con todas las colecciones asociadas a un equipo.
GET	/medical-records	Devuelve la ficha médica para un equipo, esta se compone de su información general, información específica, lista de fallas y lista de mantenciones.
POST	/hierarchy/plant	Añade una nueva planta a la base de datos.
POST	/hierarchy/section	Añade una nueva sección a la base de datos.
POST	/hierarchy/equipment	Añade un nuevo equipo a la base de datos.
POST	/hierarchy/subunit	Añade una nueva subunidad a la base de datos.
POST	/hierarchy/component	Añade un nuevo componente a la base de datos.
POST	/hierarchy/component/sensor	Añade un nuevo sensor a la base de datos, se debe especificar tanto su información general como la especifica.
POST	/failures	Añade una nueva falla a la base de datos.
POST	/maintenances	Añade una nueva mantención a la base de datos.

POST	/reports	Añade un nuevo resumen de reporte a la base de datos.
POST	/collections	Añade una nueva colección a la base de datos.
PUT	/hierarchy/plant/{id}	Modifica la información de una planta.
PUT	/hierarchy/section/{id}	Modifica la información de una sección.
PUT	/hierarchy/equipment /{id}	Modifica la información de un equipo.
PUT	/hierarchy/subunit /{id}	Modifica la información de una subunidad.
PUT	/hierarchy/component /{id}	Modifica la información de un componente.
PUT	/failures/{id}	Modifica la información de una falla.
PUT	/maintenances/{id}	Modifica la información de una mantención.
PUT	/collections/{id}	Modifica la información de una colección.
DELETE	/hierarchy/plant/{id}	Elimina la información de una planta.
DELETE	/hierarchy/section/{id}	Elimina la información de una sección.
DELETE	/hierarchy/equipment /{id}	Elimina la información de un equipo.
DELETE	/hierarchy/subunit /{id}	Elimina la información de una subunidad.
DELETE	/hierarchy/component /{id}	Elimina la información de un componente.
DELETE	/failures/{id}	Elimina la información de una falla.
DELETE	/maintenances/{id}	Elimina la información de una mantención.
DELETE	/collections/{id}	Elimina la información de una colección.