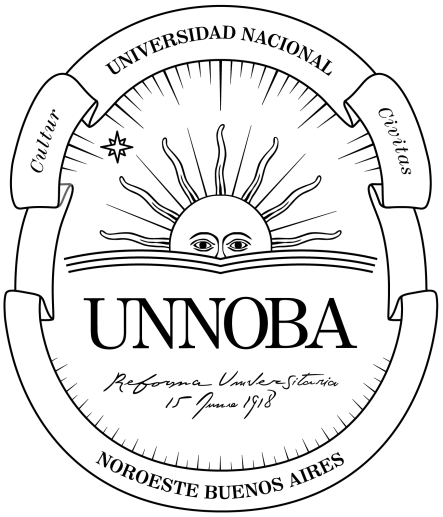
Anexo V



Universidad Nacional del Noroeste de la Provincia de Buenos Aires

Título: Quality Engineering en Hydrolix Innovations Lab

Carrera: Ingeniería en Informática

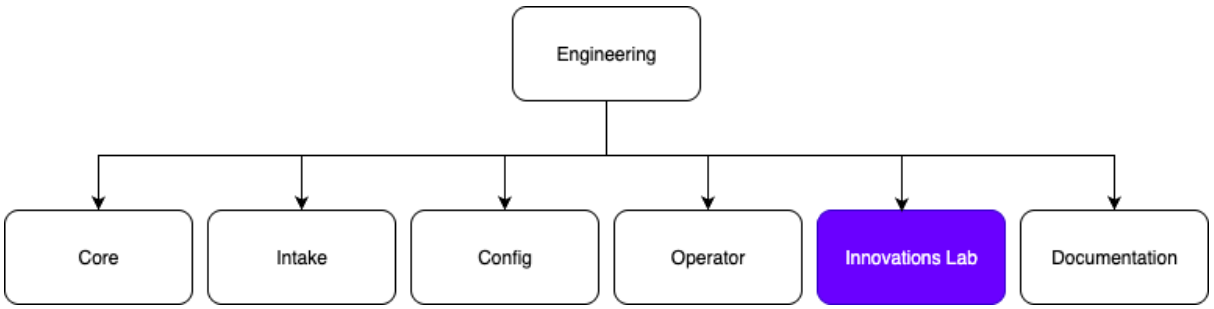
Práctica Profesional Supervisada

Introducción

Contexto en el que se realizó la práctica profesional supervisada

El presente informe surge a partir de la oportunidad de trabajo ofrecida por la empresa de consultoría de software **Clarolab**, con sede en Junín, Buenos Aires. Clarolab se dedica a brindar servicios de consultoría a empresas y organizaciones desde su fundación en 2008. Una de las compañías que se beneficia de sus servicios es **Hydrolix**, una empresa dedicada al desarrollo de una plataforma de base de datos en la nube. El principal objetivo de Hydrolix es actuar como un *streaming data lake*, simplificando el uso y transformando la gestión de grandes volúmenes de datos a altas velocidades, reduciendo significativamente los costos asociados a su manejo.

Dentro de Hydrolix, el área de ingeniería está compuesta por varios equipos: **Intake**, encargado de la ingesta de datos; **Core**, que procesa y retorna los resultados de consultas; **Config**, responsable de la API y la actualización de la interfaz de usuario; **Control Plane**, encargado de la infraestructura; y **Innovation Lab**, el equipo más reciente, creado en 2024, cuyo principal objetivo es incrementar el ecosistema de Hydrolix, especialmente en áreas relacionadas con la inteligencia artificial y mejorar la compatibilidad del producto con espacios de *data science*.



En el equipo de **Innovation Lab**, se desarrollaron las tareas que conforman la práctica profesional detallada en este informe. El rol desempeñado dentro de este equipo fue en el área de **Quality Engineering (QE)**, donde se tuvo como responsabilidad principal asegurar que los procesos de desarrollo de software mantuvieran un estándar de calidad alto, facilitando iteraciones rápidas y sin errores. Además, como **Team Leader de QE** en Innovation Lab, también se asumieron tareas de gestión y organización del equipo, asignación de tareas y creación de soluciones de automatización.

Esta práctica profesional supervisada se enmarca dentro del trabajo **full-time** que, desde junio de 2024, se realiza en la empresa **Hydrolix**, particularmente en el equipo de Innovations Lab.

Durante mayo de ese mismo año, se llevó a cabo un período de entrenamiento y colaboración en tareas iniciales junto a distintos equipos de la organización

A grandes rasgos, las actividades desarrolladas durante la práctica se centraron en:

- O**rganización del ecosistema Jira + GitLab**.
- G**estión de tareas del equipo y onboarding de nuevos miembros**
- **Administración de clusters e infraestructura.**
- M**igración y recuperación de datos**.
- D**esarrollo de un framework de testing automatizado** para los conectores de Splunk y Spark.

A lo largo del tiempo en el equipo de Innovation Lab, una parte crucial del trabajo consistió en la **organización del ecosistema Jira + GitLab**, lo cual resultó fundamental para mantener un control preciso y prolijo de todas las tareas del equipo, asegurando que el flujo de trabajo se desarrollara de manera eficiente y sin contratiempos. También se asumieron tareas de gestión, organizando y asignando las tareas del equipo, mientras se entrenaba y capacitaba a los nuevos miembros de QE y desarrollo, explicando el funcionamiento de los productos y el flujo de trabajo de desarrollo.

Además, debido a la diferencia horaria con el jefe y los equipos internacionales (como los de Ucrania trabajando con Spark, los equipos de Customer Success de Europa y los de Documentación), se estuvo involucrado en la resolución diaria de cuestiones urgentes y en la coordinación de actividades a través de distintos husos horarios.

En cuanto a las tareas de infraestructura, se asumió la responsabilidad del **mantenimiento de clusters** y también se participó en tareas de **migración y restauración de datos**, asegurando que los datos se gestionaran adecuadamente en todos los entornos de trabajo. Asimismo, en el área de **testing manual**, se supervisaron los procesos y herramientas de prueba para garantizar su correcta ejecución antes de cualquier despliegue.

Finalmente, la contribución más destacada fue el **desarrollo de un framework de testing automatizado** para los conectores de **Splunk y Spark**. **Splunk** es una plataforma utilizada para el análisis y monitoreo de datos en tiempo real, mientras que **Apache Spark** es un motor de procesamiento distribuido ampliamente empleado en ciencia de datos y análisis de grandes volúmenes de información. Los conectores de Hydrolix para estas herramientas tienen como objetivo **permitir que los usuarios puedan consultar y procesar datos directamente desde Hydrolix**, integrando sus flujos de trabajo habituales sin necesidad de modificar sus entornos.

Ésta framework de testing automatizado permitió mejorar significativamente la eficiencia, detectar errores en etapas tempranas y elevar la calidad general del ciclo de desarrollo y del producto en sí.

Objetivos

Objetivo general

El objetivo principal del trabajo desarrollado durante esta práctica profesional es **asegurar que los productos creados por el equipo de Innovation Lab de Hydrolix sean confiables, funcionales y útiles para sus usuarios**. Para lograrlo, se trabaja en la construcción de un ecosistema de calidad que permite validar cada nuevo desarrollo desde sus primeras etapas hasta su puesta en producción, a través de pruebas manuales y automatizadas, procesos de revisión y seguimiento de errores, e integración con herramientas de control como GitLab y Jira.

A su vez, se busca **facilitar el trabajo del equipo de desarrollo**, proveyendo los entornos, configuraciones y estructuras de datos necesarias para ejecutar pruebas de manera efectiva, permitiendo que las iteraciones técnicas se realicen con agilidad y precisión. En este rol también se asume la responsabilidad de liderar y organizar al equipo de **Quality Engineering**, acompañando el crecimiento del área mediante la incorporación de nuevas herramientas, metodologías de trabajo y espacios de formación para los nuevos integrantes.

Esta práctica profesional no se enmarca en un ejercicio académico simulado, sino que corresponde al **trabajo diario desempeñado de manera *full-time* en un entorno real de producción**, con autonomía en la toma de decisiones y un enfoque constante en la mejora continua de los procesos y productos de la empresa.

Objetivos específicos

1. Garantizar la trazabilidad y documentación del proceso de desarrollo y testing

Implementar y mantener buenas prácticas en la gestión de tickets, descripciones de tareas, merge requests y resultados de pruebas, asegurando que cada cambio esté debidamente documentado, justificado y probado, facilitando el seguimiento y análisis de errores a lo largo del ciclo de vida del software.

2. Ejecutar pruebas manuales sobre los conectores Splunk y Spark

Validar el correcto funcionamiento del conector de Splunk, centrándose en la traducción de consultas SPL a SQL de ClickHouse, y en el comportamiento de su interfaz gráfica. Las

pruebas incluyeron diversidad de datos y queries, así como intentos deliberados de romper la interfaz para detectar fallos en validaciones y configuración. En el caso de Spark, el trabajo consistió en probar manualmente la integración del conector con Hydrolix, utilizando su propia shell de queries, validando desde la correcta lectura de datos hasta el procesamiento de consultas complejas. Estas pruebas exploraron distintos tipos de datos, funciones de agregación, operaciones de machine learning y escenarios esperados y no esperados, para asegurar estabilidad y confiabilidad.

3. Automatizar pruebas de los conectores Splunk y Spark

Desarrollar e implementar frameworks de pruebas automatizadas para los conectores Splunk y Spark. En el caso de Splunk, emplear Python utilizando librerías como Selene y Splunklib para validar distintos escenarios de uso tanto a nivel de queries como de interfaz, facilitando la detección de regresiones y reduciendo significativamente el tiempo de validación manual. Para Spark, reutilizar parte del código común y desarrollar en PySpark la validación de múltiples funcionalidades del conector. Esta solución permitió detectar fallos en etapas tempranas del ciclo de integración continua, incluso revelando errores en otras áreas del sistema al comparar resultados entre el motor de queries de Spark y el de Hydrolix.

4. Integrar las pruebas automatizadas en GitLab CI

Coordinar con el equipo de CI la integración del framework de testing en los pipelines de GitLab, permitiendo la ejecución automática de las pruebas con cada cambio en las ramas. Esta integración agilizó la detección de errores, mejoró la calidad general del producto y permitió enfocar los esfuerzos manuales en funcionalidades nuevas.

5. Replicar las pruebas automatizadas en entorno Databricks

Adaptar el framework de pruebas del conector de Spark para su ejecución en la plataforma Databricks, conservando la capacidad de validación de funcionalidades críticas. Esto permitió evaluar compatibilidad, performance y posibles mejoras del conector en un entorno de nube ampliamente utilizado en la industria.

6. Migrar datos hacia Delta Tables para benchmarking en Databricks

Liderar la migración de aproximadamente 8 mil millones de registros de logs desde Hydrolix hacia Delta Tables en Databricks, con el objetivo de realizar comparativas de rendimiento. Esta tarea implicó la ejecución de múltiples workflows en paralelo utilizando Spark, y permitió generar contenido técnico para difundir las capacidades del conector.

7. Recuperar datos perdidos en el cluster principal

Realizar tareas avanzadas de recuperación de datos tras una purga accidental de información valiosa, accediendo directamente al bucket de almacenamiento y al catálogo de PostgreSQL.

Este proceso implicó el manejo cuidadoso de transacciones y la automatización de scripts en Python para la restauración precisa y completa de los datos.

8. Liderar al equipo de Quality Engineering en Innovation Lab

Organizar y gestionar al equipo de QE mediante asignación de tareas, definición de prioridades, entrenamiento de nuevos integrantes y coordinación con el resto del equipo de desarrollo. El objetivo fue mantener la calidad del software desde etapas tempranas de desarrollo y asegurar ciclos de iteración eficientes.

9. Coordinar actividades con equipos distribuidos internacionalmente

Facilitar la colaboración con equipos de diferentes regiones (Ucrania, Europa, etc.), resolviendo incidencias en distintos husos horarios y asegurando una comunicación efectiva. Esta coordinación fue clave para mantener la continuidad del trabajo, especialmente en contextos de urgencia o durante releases críticos.

Plan de Trabajo y Carga Horaria

A continuación, se detalla el plan de trabajo y una estimación de la carga horaria invertida en cada una de las actividades desarrolladas durante el período de práctica profesional. Cabe aclarar que al tratarse de una actividad laboral full-time, esta distribución no responde a una planificación académica previa, sino a un resumen de los principales bloques de tareas efectivamente realizadas, organizadas por tipo de actividad.

Actividad	Tareas relacionadas	Horas estimadas
Garantizar la trazabilidad y documentación del proceso de desarrollo y testing	Implementación de buenas prácticas en la gestión de tickets y merge requests, documentación en Confluence, creación de dashboards en Jira y automatización de enlaces con GitLab.	40
Testing manual - Splunk & Spark	Ejecución de queries SPL en Splunk y validación de resultados; pruebas de interfaz; ejecución de consultas con Spark Shell sobre distintos tipos de datos y escenarios.	100
Testing automatizado - Splunk	Diseño y desarrollo de frameworks de testing:	120

& Spark	uso de Python, Selene y Splunklib para Splunk, y PySpark para Spark; validación de regresiones, manejo de datasets y resultados comparativos.	
Testing en Databricks	Adaptación de pruebas automatizadas y validación en la plataforma	40
Integración con GitLab CI	Trabajo con el equipo de CI, integración de pipelines de testing	25
Migración de datos a Delta Tables	Ejecución de workflows Spark	30
Recuperación de datos	Ejecución de scripts, restauración manual y automatizada de datos	25
Gestión de equipo de QE	Asignación de tareas, organización del trabajo, soporte a nuevos miembros	40
Colaboración internacional	Coordinación con equipos remotos, resolución de incidencias	30

Observaciones

- Las horas estimadas no son acumulativas por día o semana, sino que reflejan bloques de trabajo concentrados en distintos períodos a lo largo de varios meses.

- Muchas de estas actividades se realizaron en paralelo, como parte del flujo de trabajo normal del equipo.

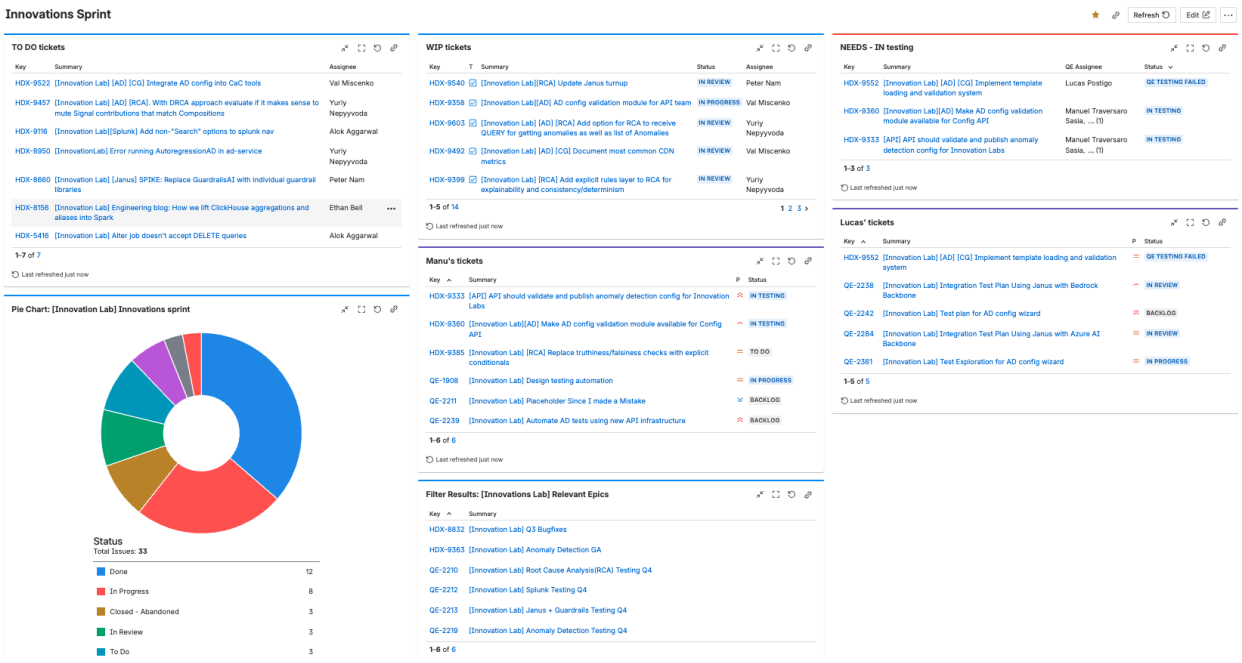
Descripción de la Práctica Profesional Efectuada

1. Garantizar la trazabilidad y documentación del proceso de desarrollo y testing

Desde la conformación inicial del equipo se trabajó en establecer prácticas que aseguraran la trazabilidad y documentación de cada tarea. Todo el trabajo se gestionó mediante **tickets en Jira**, transformando cada necesidad en una tarea registrada y medible. A su vez, se documentaron de manera sistemática en **Confluence** tanto los pasos para utilizar los productos como las tecnologías involucradas, facilitando el acceso al conocimiento compartido.

Se implementaron **dashboards y visualizaciones en Jira**, como la que se puede observar en la figura 1, para dar visibilidad al trabajo en curso y al estado de cada tarea, incluyendo tanto aquellas asociadas a desarrollo de código como las de carácter organizativo. Además, se incorporaron **automatizaciones entre GitLab y Jira** para enlazar tickets y *merge requests*, lo que permitió estandarizar y agilizar el proceso de desarrollo.

Figura 1. Dashboard con información del Sprint actual.

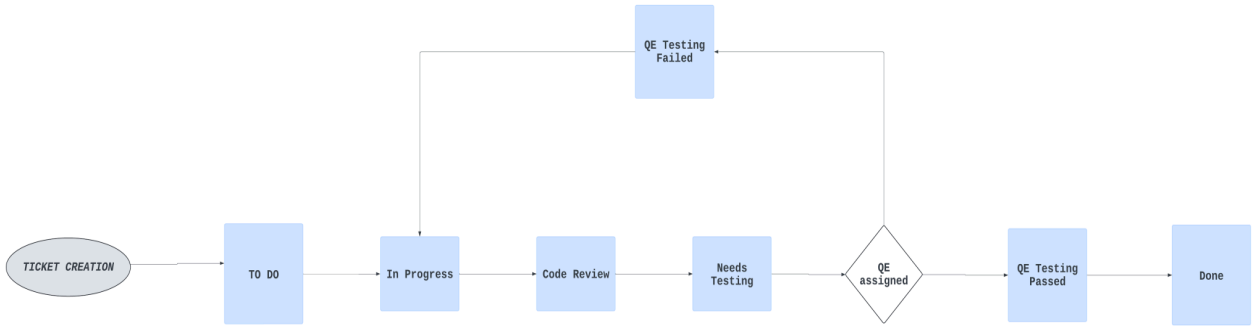


Nota: Dashboards del estilo se usaron para mantener constante control de las tareas del equipo

Este enfoque se apoyó en conceptos y metodologías vistas en la carrera, como **metodologías ágiles, elicitación de requisitos y trabajo colaborativo en equipo**, aplicados en un contexto real. El impacto fue claro: se logró reducir la confusión, mejorar la comunicación, detectar errores en fases tempranas, y mantener una trazabilidad precisa de los cambios y su impacto en el sistema, así como de los pasos a seguir en cada iteración.

Cabe destacar que parte de este proceso ya se encontraba **estandarizado por la organización de QE**, sobre lo cual se trabajó constantemente para ajustar a las necesidades específicas de cada equipo. Como base de dicha estandarización, cada ticket debía recorrer una serie de estados definidos en Jira: **En Progreso → Revisión de Código → Necesita Testing → En Testing → QE Aprueba / QE Fallo → Tarea Finalizada**. Al igual que se ve en la figura 2.

Figura 2. Diagrama de estados de un ticket Jira



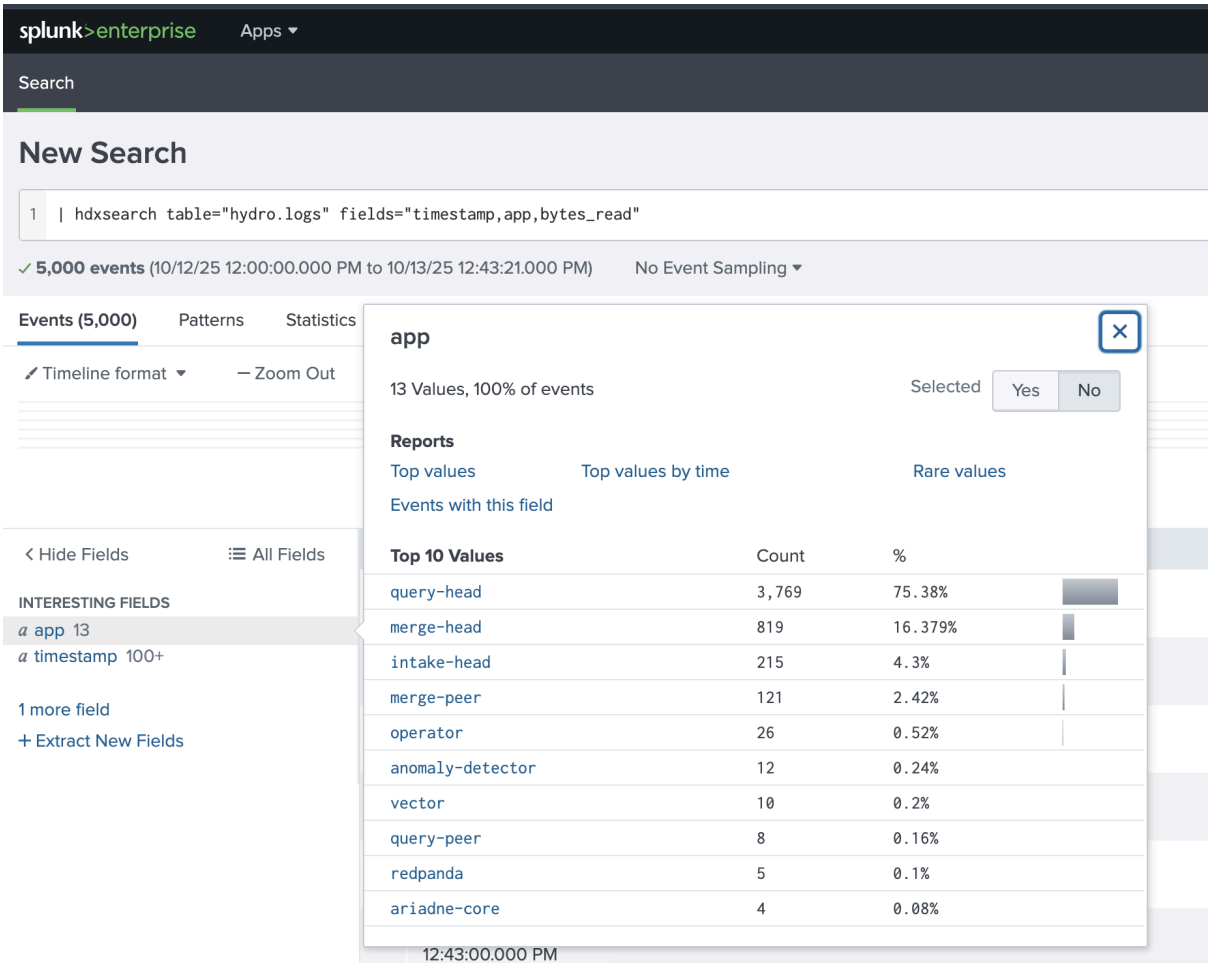
Nota: Mi rol usualmente corresponde desde que el ticket llega a 'needs testing' hasta el final, aunque con mi crecimiento profesional me empecé a involucrar en etapas anteriores, siendo parte del diseño y review del código.

2. Pruebas manuales sobre los conectores Splunk y Spark.

Luego de establecer un proceso ordenado de documentación y trazabilidad, el siguiente paso consistió en realizar pruebas manuales de los conectores Splunk y Spark, con el fin de validar su funcionamiento tanto de manera aislada como en escenarios de integración con el cluster de Hydrolix. Esto permitió observar el comportamiento de los conectores frente a distintos estados de los pods y asegurar su confiabilidad en situaciones de uso real. Al mismo tiempo, estas pruebas representaron una instancia de entrenamiento, ya que implicaron aprender desde cero conceptos fundamentales de ambas tecnologías.

En el caso del conector de Splunk, véase figura 3, se llevaron adelante pruebas sobre su interfaz gráfica de configuración, donde se identificaron múltiples fallos que fueron corregidos en sucesivas iteraciones hasta alcanzar un estado apto para producción. El foco principal estuvo en ejecutar consultas SPL, verificando que fueran traducidas correctamente a SQL, que las funciones de agregación se aplicaran del lado de Hydrolix y que los resultados retornados fueran consistentes. Estas validaciones abarcaron desde consultas simples hasta casos más complejos, encontrándose errores importantes en el manejo de palabras clave debido al uso interno de librerías de sanitización.

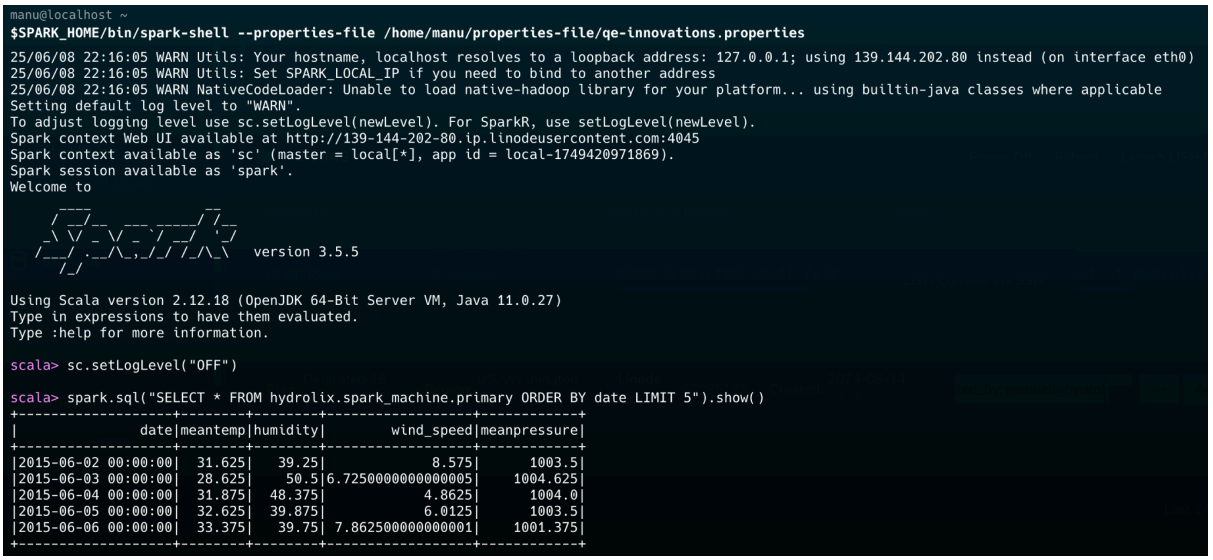
Figura 3. Interfaz de Splunk Enterprise.



Nota: Ejemplo de query básica a una tabla nativa de un cluster Hydrolix, donde se lista el timestamp en el que se registró el log y de qué servicio provino.

Por su parte, las pruebas sobre el conector de Spark fueron más extensivas, debido a la mayor amplitud del lenguaje y la variedad de funciones disponibles. Se exploraron distintos datasets y particiones, consultas con condicionales, filtros, order by, where, funciones de agregación y escenarios de machine learning. También se profundizó en el uso de tablas nativas de Hydrolix, evaluando la correcta deserialización de datos hacia Spark en distintos contextos de ejecución.

Figura 4. Interfaz Spark Shell.



Nota: Se usó tanto en la etapa inicial para analizar el funcionamiento del conector y sobre el final del desarrollo, para analizar con mayor detalle los bugs encontrados por la automatización.

Este proceso permitió aplicar conocimientos de la carrera en bases de datos, SQL, DDL, DML y DCL, así como experiencia previa con PostgreSQL, complementados con aprendizajes nuevos en SPL y PySpark. El mayor impacto estuvo en la detección temprana de errores durante el desarrollo y en el feedback sobre diseño, implementación y experiencia de usuario, lo que incrementó significativamente la calidad de ambos conectores antes de avanzar hacia la etapa de automatización.

3. Automatización de pruebas de los conectores Splunk y Spark.

En línea con la filosofía de la organización de automatizar todo lo posible, se avanzó hacia la automatización cuando el equipo comprendió suficientemente el funcionamiento de los productos. El objetivo fue ahorrar tiempo, ejecutar comprobaciones críticas de manera permanente y enfocar las pruebas manuales en lo exclusivo de cada nueva funcionalidad. La priorización comenzó por las queries más básicas y frecuentes, aquellas que no podían fallar.

Criterios y elecciones tecnológicas. Se eligió Python por rapidez de desarrollo, experiencia del equipo y por el ecosistema disponible. Selene fue recomendado por el equipo de QE especializado en API/UI, mientras que Splunklib resultó la opción natural para la integración con Splunk desde Python. Para Spark se trabajó con PySpark, lo que permitió crear sesiones en un entorno controlado y reproducible.

Estructura común de los tests. Todos los proyectos compartieron una fase inicial de setup, en la que se creaban recursos en Hydrolix mediante llamadas a la API, y a partir de ahí se diversificaban según el conector.

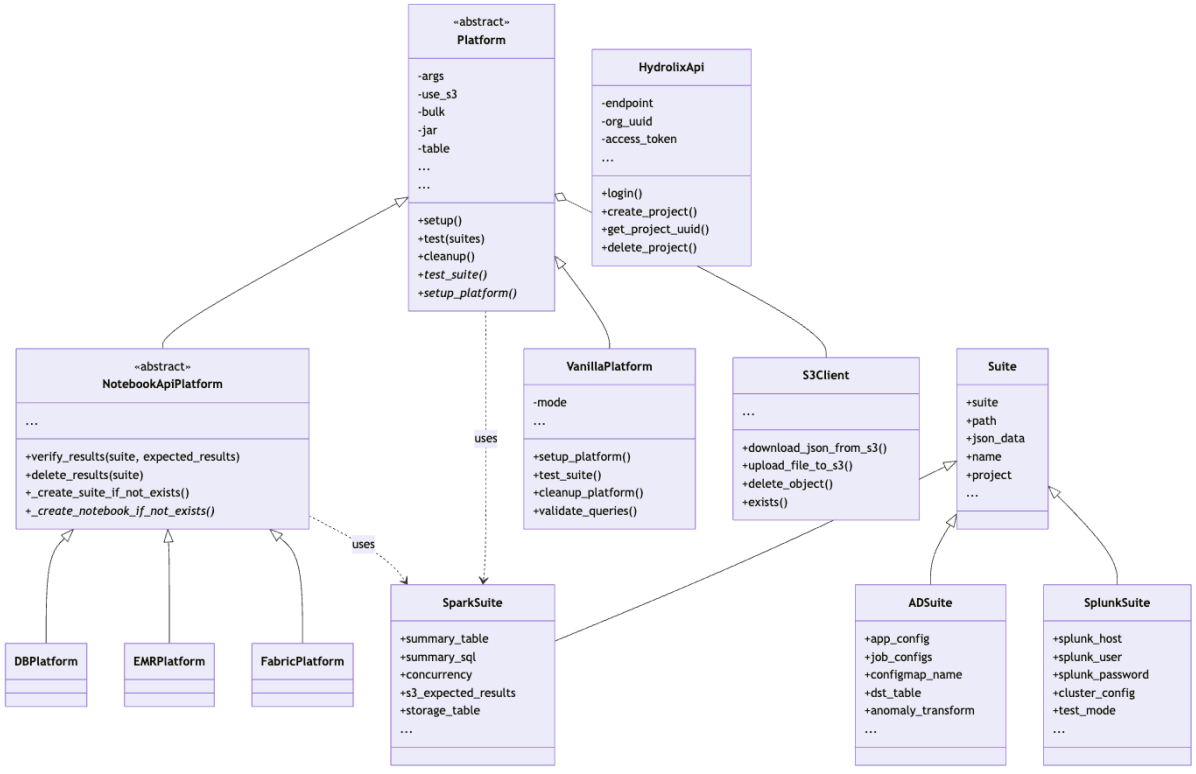
- En Splunk: primero se validaron queries básicas, luego se agregaron tests de palabras clave (donde se detectaron fallos relacionados con la librería de sanitización de SQL) y finalmente pruebas de la interfaz de configuración. En general, fue un producto

sencillo de automatizar y sin mayores complicaciones.

- En Spark: la complejidad fue mayor debido a su carácter distribuido, el uso de multithreading y la variedad de entornos. Se ejercitaron consultas con condicionales, filtros, `WHERE`, `ORDER BY`, funciones de agregación, diversidad de datasets y validaciones sobre tablas nativas de Hydrolix, especialmente en el proceso de deserialización de datos hacia Spark.

Arquitectura y reutilización. Se diseñó una estructura orientada a objetos con una clase base de suite/test y subclases para Splunk y Spark. Gracias al polimorfismo, cada tipo de prueba sabía cómo realizar su propio setup o cómo comparar resultados, reutilizando gran parte del código común. También se aplicó el patrón Singleton para gestionar sesiones y evitar reinicializaciones innecesarias. Finalmente, los tests se adaptaron a diferentes protocolos, pudiendo ejecutarse tanto por HTTP/HTTPS como por TCP.

Figura 5. Diagrama de clases de la testing framework



Nota: La estructura pasó por algunas iteraciones donde se añadieron funcionalidades y otras se corrigieron, pero el diseño orientado a objetos y la reutilización de código son conceptos que estuvieron desde el día 1.

Resultado. El trabajo derivó en un set de pruebas más estable, escalable y reutilizable, que permitió detectar errores en fases tempranas y asegurar que los escenarios críticos se validaran continuamente. Además, se sentaron las bases para la integración en pipelines de CI/CD.

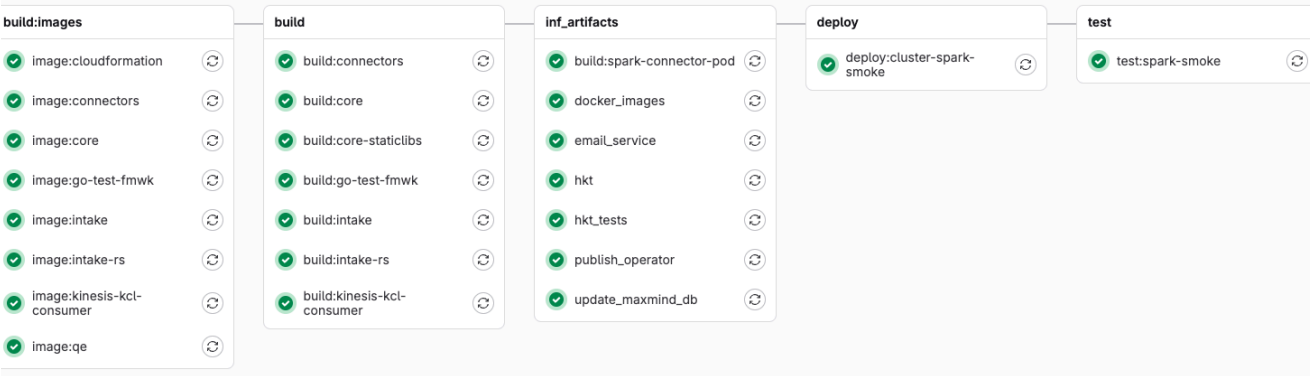
4. Integración de pruebas automatizadas en GitLab CI.

El siguiente paso fue integrar las pruebas automatizadas en los pipelines de GitLab CI, con el objetivo de que se ejecutaran en cada cambio realizado durante la fase de desarrollo. Esto permitió detectar errores de manera inmediata, incluso antes de llegar a la etapa de revisión, evitando retrabajos y reduciendo el tiempo de intercambio entre los miembros del equipo.

El trabajo de coordinación con el equipo de CI implicó un proceso iterativo de ajustes. Fue necesario adaptar parte del framework para que pudiera ejecutarse en los clusters propios de CI, que tenían diferencias técnicas con los entornos de testing habituales. En el caso de Spark, el desafío más significativo fue realizar la integración como pod de Kubernetes, algo que no se había hecho previamente y que requirió investigación y pruebas adicionales. En cuanto a la infraestructura de GitLab, los runners se organizaron siguiendo esquemas ya utilizados en otras áreas de QE, lo que facilitó la adopción.

La integración demandó un esfuerzo de coordinación continua entre tickets, ajustes en el framework y la habilitación del equipo de CI para aplicar correcciones específicas. Una vez consolidado el proceso, los pipelines se transformaron en una herramienta confiable para detectar fallos de manera temprana y sostener la calidad del software en cada iteración.

Figura 6. Visualización de pipelines en Gitlab CI.



Nota: En cada stage se realizan los diferentes pasos necesarios para realizar un test de integración. En build se crea el .jar del spark connector, en deploy se levanta un cluster con una instancia de spark en un pod, y en test el conector corre las queries de validación necesarias.

5. Automatización en entorno Databricks.

En conjunto con el equipo y bajo la guía de mi manager se identificó que muchos de los potenciales clientes trabajaban a diario en **Databricks**, una plataforma ampliamente utilizada en la industria para el análisis de datos y el trabajo colaborativo en ciencia de datos e inteligencia artificial. Databricks ofrece entornos administrados basados en **clusters de Spark**, con notebooks integrados y herramientas para procesamiento distribuido, lo que lo convierte en un ecosistema con varias similitudes respecto a Hydrolix.

Una vez adaptado el conector para funcionar en Databricks, el siguiente paso fue llevar allí las pruebas automatizadas, con el objetivo de validar **rendimiento, compatibilidad con distintas versiones** y la correcta interacción de todo el ciclo **Databricks → Spark → Spark connector → Cluster Hydrolix**, incluyendo la deserialización de datos en ambos sentidos.

Para lograrlo fue necesario construir un **módulo nuevo**, ya que la forma de manejar sesiones e instalar el `.jar` del conector difería respecto a la configuración de Spark tradicional. Además, el proceso de setup debía levantar un **cluster propio de Databricks**, lo que implicó adaptar el flujo de las suites. En esta instancia resultó útil el conocimiento adquirido en la materia **Sistemas Inteligentes**, particularmente sobre notebooks, que facilitó la ejecución de pruebas manuales en la plataforma.

En cuanto al alcance, se mantuvo la misma estructura de suites y cobertura de automatización que ya existía para Spark, realizando únicamente los ajustes necesarios en las partes de interacción con la API, el setup y la validación de resultados. Esta réplica de pruebas no solo confirmó la compatibilidad del conector en un entorno clave para la industria, sino que además habilitó la posibilidad de avanzar con pruebas de **performance** que derivaron en la práctica de *benchmarking*, desarrollada en la siguiente etapa.

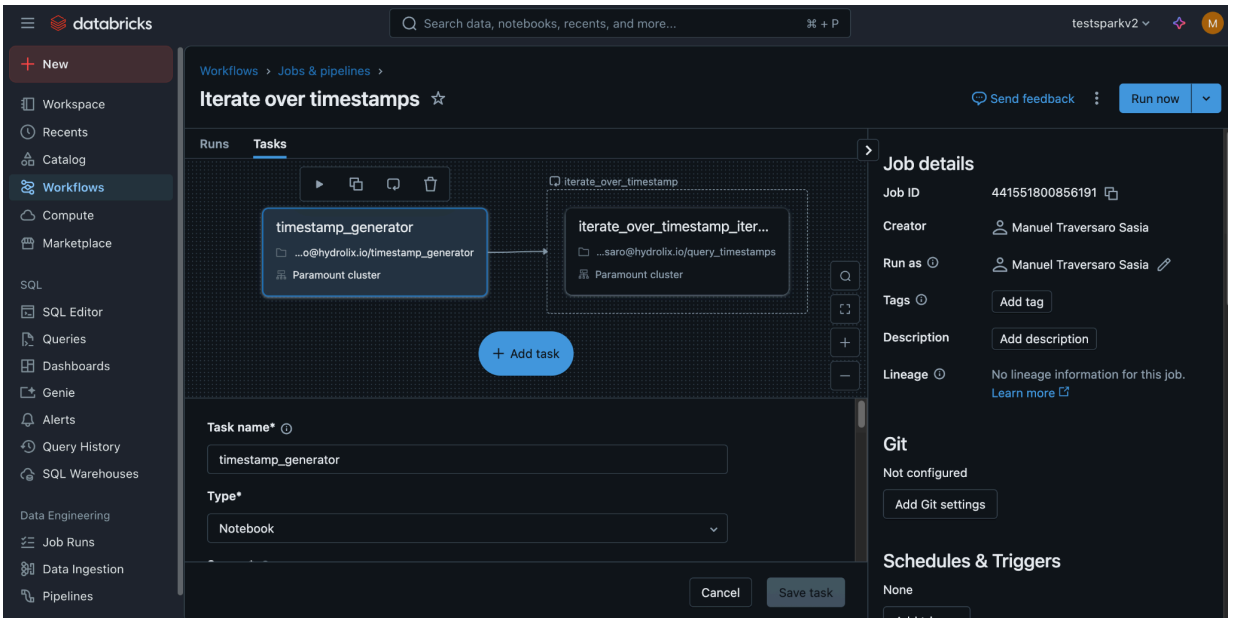
6. Migración de datos hacia Delta Table para benchmarking en Databricks.

Una vez que el **conector de Spark** alcanzó un estado estable y funcional, el siguiente paso fue evaluar su rendimiento frente a las consultas nativas de **Databricks** sobre sus propias **Delta Tables**. Este proceso de *benchmarketing* tenía como propósito medir la **performance** y **capacidad de respuesta** en distintos escenarios de consulta, complementando el discurso técnico que distingue a **Hydrolix**: si ya se sabía que el **costo operativo** del sistema era menor, el desafío ahora era demostrar que su rendimiento podía igualar o superar al de las soluciones que se ofrecen en el **estándar del mercado**, como Databricks.

Los datos utilizados provenían de una **copia ofuscada de información real de clientes**, recolectada originalmente durante **eventos deportivos transmitidos en vivo**, donde el sistema mantenía observabilidad sobre métricas de tráfico y rendimiento. El volumen total alcanzaba aproximadamente **ocho mil millones de registros**, lo que representó un caso de uso realista y exigente para las pruebas. Los datos se migraron desde una **tabla nativa de Hydrolix** hacia una **Delta Table** en Databricks, preservando la estructura y las relaciones necesarias para los experimentos posteriores.

El proceso se orquestó mediante **workflows altamente concurrentes**, con jobs que ejecutaban consultas segmentadas por fecha, ejemplo en la figura 7. Cada consulta abarcaba un rango de cuatro horas, y múltiples consultas simultáneas cubrían el intervalo de un día completo. Cada ejecución del workflow procesaba un mes de datos, con **reintentos manuales** en caso de interrupciones, asegurando que las operaciones fueran **transaccionales y libres de duplicados**. Se mantuvo una **tabla duplicada auxiliar** que permitía monitorear el progreso de la migración y verificar qué fragmentos de datos habían sido completados con éxito.

Figura 7. Workflows de Databricks



Nota: la alta concurrencia no es visible en esta interfaz, se configura dentro de cada task.

La **validación de calidad** se realizó mediante recuentos de filas y consultas de verificación para asegurar la integridad y completitud del proceso. Una vez concluida la migración, el **equipo de ingeniería de datos** tomó la posta para realizar las pruebas de rendimiento y análisis comparativo entre los entornos Hydrolix y Databricks.

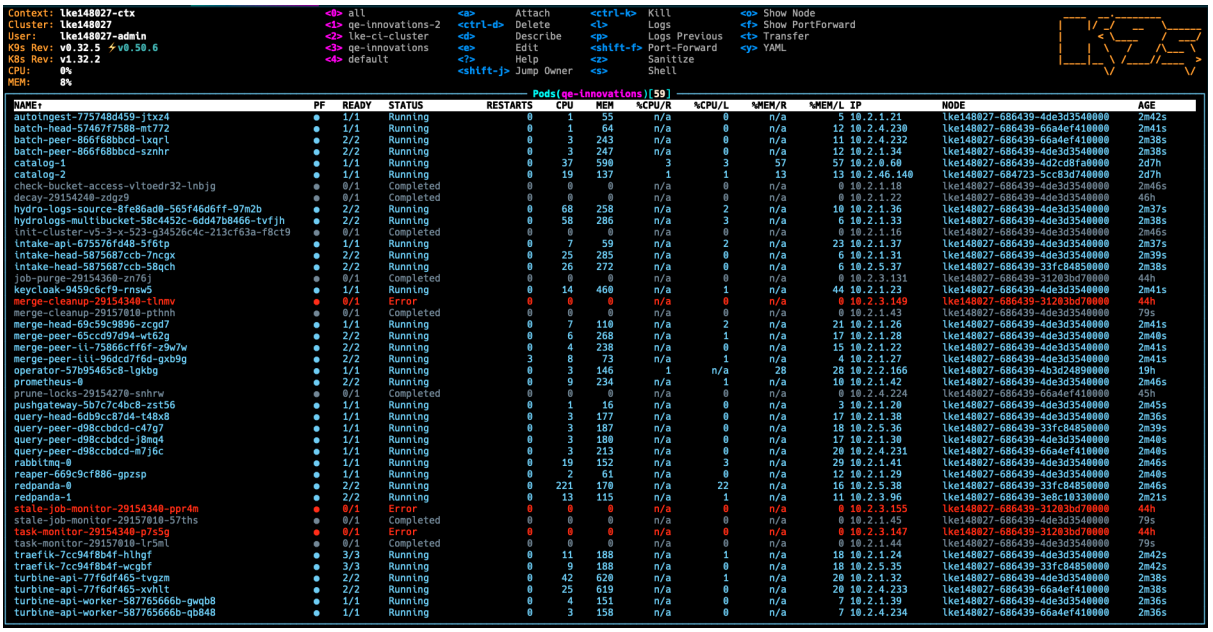
Esta experiencia no solo permitió disponer de un **dataset confiable para benchmarking**, sino que también fortaleció el entendimiento del funcionamiento de Delta Tables y consolidó el aprendizaje sobre el **manejo de grandes volúmenes de datos**, la **planificación de workflows distribuidos** y la **validación de resultados en entornos de alto rendimiento**.

7. Recuperación de datos perdidos en el cluster principal.

Durante esta etapa se presentó una situación imprevista que requirió la **recuperación de información crítica** en uno de los clusters principales de **Hydrolix**. El incidente ocurrió cuando un miembro del equipo configuró erróneamente una opción de **limpieza automática** en la tabla utilizada para las pruebas de **benchmarking**, lo que provocó la eliminación de todos los registros con más de 90 días de antigüedad. La intención era liberar espacio en disco, pero sin conocer que esa tabla contenía **un año completo de logs fundamentales** para las comparaciones históricas de rendimiento del **Spark Connector**.

La pérdida de estos datos impedía continuar con los ejercicios de benchmarking comparativo entre versiones, ya que sin el histórico no era posible medir de forma consistente la evolución del rendimiento —por ejemplo, comparar si una nueva versión era “20% más rápida que el mes pasado”. El problema fue detectado al observar una **disminución abrupta en la cantidad total de filas** de la tabla principal.

Figura 8. Vista general de los servicios del cluster.



```
Context: lke148027-ctx                <0> all              <a>      Attach      <ctrl-k> Kill          <o> Show Node
Cluster: lke148027                    <1> qe-innovations-2  <ctrl-d> Delete      <l>      Logs          <f> Show PortForward
User:    lke148027-admin              <2> lke-ci-cluster    <d>      Describe    <p>      Logs Previous <t> Transfer
K9s Rev: v0.32.5 ⚡v0.50.6             <3> qe-innovations    <e>      Edit        <shift-f> Port-Forward <y> YAML
K8s Rev: v1.32.2                      <4> default           <?>      Help        <z>      Sanitize
CPU:     0%                                                 <shift-j> Jump Owner <s>      Shell
MEM:     0%
                                                  Pods(qe-innovations)[59]
NAME↑                                  PF  READY  STATUS     RESTARTS  CPU  MEM  %CPU/R  %CPU/L  %MEM/R  %MEM/L  IP           NODE                                AGE
autoingest-775748d459-jtxz4            ●   1/1    Running           0    1   55     n/a       0     n/a       5  10.2.1.21    lke148027-686439-4de3d3540000       2m42s
batch-head-57467f7588-mt772            ●   1/1    Running           0    1   64     n/a       0     n/a      12  10.2.4.230   lke148027-686439-66a4ef410000       2m41s
batch-peer-866f68bbcd-lxqrl            ●   2/2    Running           0    3  243     n/a       0     n/a      11  10.2.4.232   lke148027-686439-66a4ef410000       2m38s
batch-peer-866f68bbcd-sznhr            ●   2/2    Running           0    3  247     n/a       0     n/a      12  10.2.1.34    lke148027-686439-4de3d3540000       2m38s
catalog-1                              ●   1/1    Running           0   37  590       3       3      57      57  10.2.0.60    lke148027-686439-4d2cd8fa0000       2d7h
catalog-2                              ●   1/1    Running           0   19  137       1       1      13      13  10.2.46.140  lke148027-684723-5cc83d740000       2d7h
check-bucket-access-vltoedr32-lnbjg    ●   0/1    Completed         0    0    0     n/a       0     n/a       0  10.2.1.18    lke148027-686439-4de3d3540000       2m46s
decay-29154240-zdgz9                   ●   0/1    Completed         0    0    0     n/a       0     n/a       0  10.2.1.22    lke148027-686439-4de3d3540000       46h
hydro-logs-source-8fe86ad0-565f46d6ff-97m2b ●  2/2  Running       0   68  258     n/a       2     n/a      10  10.2.1.36    lke148027-686439-4de3d3540000       2m37s
hydrologs-multibucket-58c4452c-6dd47b8466-tvfjh ●  2/2  Running   0   58  286     n/a       3     n/a       6  10.2.1.33    lke148027-686439-4de3d3540000       2m38s
init-cluster-v5-3-x-523-g34526c4c-213cf63a-f8ct9 ●  0/1  Completed  0    0    0     n/a       0     n/a       0  10.2.1.16    lke148027-686439-4de3d3540000       2m46s
intake-api-675576fd48-5f6tp            ●   1/1    Running           0    7   59     n/a       2     n/a      23  10.2.1.37    lke148027-686439-4de3d3540000       2m37s
intake-head-5875687ccb-7ncgx           ●   2/2    Running           0   25  285     n/a       0     n/a       6  10.2.1.31    lke148027-686439-4de3d3540000       2m39s
intake-head-5875687ccb-58qch           ●   2/2    Running           0   26  272     n/a       0     n/a       6  10.2.5.37    lke148027-686439-33fc84850000       2m38s
job-purge-29154360-zn76j               ●   0/1    Completed         0    0    0     n/a       0     n/a       0  10.2.3.131   lke148027-686439-31203bd70000       44h
keycloak-9459c6cf9-rnsw5               ●   1/1    Running           0   14  460     n/a       1     n/a      44  10.2.1.23    lke148027-686439-4de3d3540000       2m41s
merge-cleanup-29154340-tlnmv           ●   0/1    Error             0    0    0     n/a       0     n/a       0  10.2.3.149   lke148027-686439-31203bd70000       44h
merge-cleanup-29157010-pthnh           ●   0/1    Completed         0    0    0     n/a       0     n/a       0  10.2.1.43    lke148027-686439-4de3d3540000       79s
merge-head-69c59c9896-zcgd7            ●   1/1    Running           0    7  110     n/a       2     n/a      21  10.2.1.26    lke148027-686439-4de3d3540000       2m41s
merge-peer-65ccd97d94-wt62q            ●   2/2    Running           0    6  268     n/a       1     n/a      17  10.2.1.28    lke148027-686439-4de3d3540000       2m40s
merge-peer-ii-75866cff6f-z9w7w         ●   2/2    Running           0    4  238     n/a       0     n/a      15  10.2.1.22    lke148027-686439-4de3d3540000       2m41s
merge-peer-iii-96dcd7f6d-gxb9g         ●   2/2    Running           3    8   73     n/a       1     n/a       4  10.2.1.27    lke148027-686439-4de3d3540000       2m41s
operator-57b95465c8-lgkbg              ●   1/1    Running           0    3  146       1     n/a      28      28  10.2.2.166   lke148027-686439-4b3d24890000       19h
prometheus-0                           ●   2/2    Running           0    9  234     n/a       1     n/a      10  10.2.1.42    lke148027-686439-4de3d3540000       2m46s
prune-locks-29154270-snhrw             ●   0/1    Completed         0    0    0     n/a       0     n/a       0  10.2.4.224   lke148027-686439-66a4ef410000       45h
pushgateway-5b7c7c4bc8-zst56           ●   1/1    Running           0    1   16     n/a       0     n/a       3  10.2.1.20    lke148027-686439-4de3d3540000       2m45s
query-head-6db9cc87d4-t48x8            ●   1/1    Running           0    3  177     n/a       0     n/a      17  10.2.1.38    lke148027-686439-4de3d3540000       2m36s
query-peer-d98ccbdcd-c47g7             ●   1/1    Running           0    3  187     n/a       0     n/a      18  10.2.5.36    lke148027-686439-33fc84850000       2m39s
query-peer-d98ccbdcd-j8mq4             ●   1/1    Running           0    3  180     n/a       0     n/a      17  10.2.1.30    lke148027-686439-4de3d3540000       2m40s
query-peer-d98ccbdcd-m7j6c             ●   1/1    Running           0    3  213     n/a       0     n/a      20  10.2.4.231   lke148027-686439-66a4ef410000       2m40s
rabbitmq-0                             ●   1/1    Running           0   19  152     n/a       3     n/a      29  10.2.1.41    lke148027-686439-4de3d3540000       2m46s
reaper-669c9cf886-gpzsp                ●   1/1    Running           0    2   61     n/a       0     n/a      12  10.2.1.29    lke148027-686439-4de3d3540000       2m40s
redpanda-0                             ●   2/2    Running           0  221  170     n/a      22     n/a      16  10.2.5.38    lke148027-686439-33fc84850000       2m46s
redpanda-1                             ●   2/2    Running           0   13  115     n/a       1     n/a      11  10.2.3.96    lke148027-686439-3e8c10330000       2m21s
stale-job-monitor-29154340-ppr4m       ●   0/1    Error             0    0    0     n/a       0     n/a       0  10.2.3.155   lke148027-686439-31203bd70000       44h
stale-job-monitor-29157010-57ths       ●   0/1    Completed         0    0    0     n/a       0     n/a       0  10.2.1.45    lke148027-686439-4de3d3540000       79s
task-monitor-29154340-p7s9g            ●   0/1    Error             0    0    0     n/a       0     n/a       0  10.2.3.147   lke148027-686439-31203bd70000       44h
task-monitor-29157010-lr5ml            ●   0/1    Completed         0    0    0     n/a       0     n/a       0  10.2.1.44    lke148027-686439-4de3d3540000       79s
traefik-7cc94f8b4f-hlhgf               ●   3/3    Running           0   11  188     n/a       1     n/a      18  10.2.1.24    lke148027-686439-4de3d3540000       2m42s
traefik-7cc94f8b4f-wcgbf               ●   3/3    Running           0    9  188     n/a       0     n/a      18  10.2.5.35    lke148027-686439-33fc84850000       2m42s
turbine-api-77f6df465-tvgzm            ●   2/2    Running           0   42  620     n/a       1     n/a      20  10.2.1.32    lke148027-686439-4de3d3540000       2m38s
turbine-api-77f6df465-xvhlt            ●   2/2    Running           0   25  619     n/a       0     n/a      20  10.2.4.233   lke148027-686439-66a4ef410000       2m38s
turbine-api-worker-587765666b-gwqb8    ●   1/1    Running           0    4  151     n/a       0     n/a       7  10.2.1.39    lke148027-686439-4de3d3540000       2m36s
turbine-api-worker-587765666b-qb848    ●   1/1    Running           0    3  158     n/a       0     n/a       7  10.2.4.234   lke148027-686439-66a4ef410000       2m36s
```

Nota: Algunos de estos servicios realizan las tareas de ciclo de vida necesarias para que, una vez eliminada la tabla, se borren los datos del catálogo y posteriormente del bucket.

A partir de ese hallazgo, se planificó el proceso de recuperación accediendo directamente al **bucket de almacenamiento**. Se desarrollaron dos **scripts en Python**: el primero para leer la información de las particiones existentes y el segundo para formatearla según la estructura esperada por el **catálogo interno de Hydrolix**. Ambos scripts se ejecutaron con **alta concurrencia** para acelerar la reconstrucción, mientras que una tercera fase consistió en realizar **inserciones transaccionales en el catálogo PostgreSQL**, garantizando consistencia e integridad. Una vez rehabilitadas las particiones en el catálogo, los propios componentes del sistema reconstruyeron la tabla automáticamente, y la verificación final se realizó mediante consultas que confirmaron la **ausencia de duplicados** y la **recuperación completa del volumen original de datos**.

El principal desafío fue la **falta de documentación o precedentes** sobre cómo realizar un proceso de recuperación de este tipo. Además, la lectura desde el bucket resultó lenta y demandó mantener los scripts en ejecución durante varias horas, incluso con repeticiones por fallas. Una parte fundamental del éxito fue saber identificar a la persona indicada para colaborar: el **manager del equipo de Intake**, quien brindó orientación clave sobre cómo los servicios internos gestionan la sincronización y eliminación de datos. Gracias a su apoyo fue posible no solo comprender el funcionamiento interno de esos servicios, sino también desarrollar una **solución efectiva y completa**.

Desde lo técnico, esta experiencia aportó un conocimiento profundo de **componentes y servicios internos** con los que el equipo de Innovations Lab no suele interactuar directamente. Desde lo personal, fue una **lección de liderazgo y serenidad**: aprender que

incluso ante un incidente crítico, mantener la calma y apoyarse en la colaboración con otros equipos es la mejor manera de alcanzar una solución sólida y segura.

8. Liderazgo del equipo de Quality Engineering en Innovation Lab.

El rol dentro del equipo fue evolucionando progresivamente a medida que el equipo crecía en relevancia y profesionalización. A lo largo del tiempo, las responsabilidades se ampliaron desde la ejecución directa de pruebas hasta la **supervisión general de la calidad de los productos**, la exploración de **nuevas soluciones y automatizaciones**, y la **coordinación de tareas** entre los distintos miembros del equipo. Esta expansión de responsabilidades estuvo acompañada por un crecimiento personal y técnico sostenido, que permitió responder con mayor madurez a las exigencias del área.

El equipo, compuesto por **dos a tres integrantes** en distintos momentos, trabajaba bajo la supervisión directa del **manager de Innovation Lab**, manteniendo una estructura de comunicación constante. La organización interna se basaba en **reuniones periódicas**, con dos *stand-ups* semanales con todo el equipo de Innovation Lab, dos adicionales específicas de QE y reuniones individuales cada viernes con el manager, el *team lead* de desarrollo y cada uno de los Quality Engineers a cargo. Las tareas se gestionaban a través de **tickets en Jira**, los cuales eran **asignados en función de las habilidades, disponibilidad y prioridades** identificadas en estas reuniones.

En el aspecto operativo, el liderazgo se caracterizó por una **participación activa en el desarrollo y automatización de pruebas**, tomando la iniciativa en tareas nuevas o complejas y luego **delegando subtareas más acotadas y bien definidas** a los demás miembros del equipo. Además, se trabajó de forma constante en la **revisión detallada de cada plan de pruebas, Merge Request y reporte de testing manual**, garantizando que los procesos cumplieran con los estándares de calidad y trazabilidad esperados.

El proceso de **capacitación y acompañamiento** de nuevos integrantes fue otro pilar clave. A través de reuniones, sesiones de revisión y análisis conjunto de los flujos de testing, se buscó transmitir conocimientos sobre el funcionamiento interno de Hydrolix, la estructura de código de automatización y las buenas prácticas de documentación.

Los resultados más visibles de esta labor incluyeron una **mejor comunicación interna**, la **reducción de errores recurrentes**, la **expansión de la cobertura de testing** y una mayor claridad en la documentación de cada cambio o incidencia. El liderazgo también contribuyó a fortalecer los lazos entre los equipos internacionales y el grupo local, actuando como un punto de conexión y equilibrio entre culturas y zonas horarias diferentes, lo que favoreció un ambiente de trabajo más colaborativo y ordenado.

Desde lo personal, esta experiencia representó un aprendizaje integral sobre **gestión de prioridades, empatía, planificación y liderazgo en entornos de alta exigencia técnica**. Aprender a mantener la moral del equipo, anticiparse a los desafíos y sostener la calma en

momentos críticos se consolidó como una de las habilidades más valiosas desarrolladas durante esta etapa.

9. Coordinar actividades con equipos distribuidos internacionalmente

La colaboración con equipos distribuidos internacionalmente fue una parte constante del trabajo dentro de Innovation Lab, especialmente debido a la naturaleza global de Hydrolix y la diversidad de roles involucrados en cada proyecto. En numerosas ocasiones, fue necesario **actuar como intermediario en representación del manager del equipo**, particularmente cuando él no estaba disponible o debía atender otros compromisos. En ese contexto, se mantuvieron **reuniones recurrentes y comunicaciones asincrónicas** con integrantes de múltiples áreas —*Solutions Engineers, Customer Success Engineers, Documentation Engineers, Marketing y Site Reliability Engineers*—, muchos de ellos ubicados en **Europa y Asia**, quienes interactuaban con los sistemas del equipo de Quality Engineering para tareas vinculadas a presentaciones a clientes, soporte técnico o validación de producto.

El rol asumido fue el de **enlace técnico y comunicacional**, ofreciendo asistencia directa o redirigiendo cada solicitud hacia la persona o área más adecuada para resolverla. Este trabajo no seguía un cronograma predefinido, sino que surgía de manera espontánea ante las necesidades del momento, requiriendo capacidad de respuesta rápida y criterio para priorizar.

Más allá del aspecto técnico, esta experiencia representó una valiosa oportunidad para fortalecer **habilidades de comunicación, claridad y empatía**. Trabajar con personas de distintos países y disciplinas enseñó la importancia de **adaptar el nivel de detalle y el formato de la información según el interlocutor**, ser paciente ante diferencias culturales o de huso horario, y mantener siempre una actitud **colaborativa y profesional**. Este ejercicio constante de comunicación efectiva resultó clave para mejorar la dinámica entre equipos y garantizar la continuidad del trabajo, incluso en un entorno distribuido y cambiante.

Conclusiones

La práctica profesional supervisada representó una oportunidad para desarrollarme rápidamente en un ambiente profesional y adaptarme a las necesidades cambiantes de mi equipo. A lo largo del proceso pude incorporar nuevos conceptos de programación, testing y tecnologías diversas, lo que me permitió contribuir de manera significativa a la calidad de los productos generados. Mis aportes también tuvieron impacto en la **unidad del equipo**, al facilitar la comunicación y la coordinación entre miembros de distintos subgrupos, fortaleciendo el trabajo conjunto.

El desafío más complejo estuvo en comprender cómo se conectaban entre sí las múltiples piezas del software, dado que el workflow de cada componente era altamente intrincado.

Superar estas dificultades requirió un esfuerzo continuo de aprendizaje, apoyado en la consulta con colegas, la capacidad de pedir ayuda en el momento adecuado, y la autoformación mediante documentación, material audiovisual, inteligencia artificial y práctica constante.

Durante la PPS pude aplicar conocimientos adquiridos en la carrera, especialmente en **programación orientada a objetos, manipulación de datos en Python, bases de datos, protocolos de red y administración de sistemas Linux**, así como también ciertos fundamentos de gestión de proyectos. Más allá de lo técnico, el aprendizaje principal estuvo en reconocer la importancia de la **adaptación al cambio**, la **planificación del trabajo** y la capacidad de aprender de manera autónoma en función de las necesidades del momento.

Al mirarme hoy y compararme con el inicio de la práctica, me encuentro más afianzado en mi capacidad de aprendizaje y resolución de problemas. Si algo destaco como logro personal es haber encontrado mi lugar dentro del equipo como alguien que aporta orden y facilita la resolución de obstáculos, contribuyendo tanto al éxito técnico como a la dinámica humana del grupo.

Planes a Futuro

Con la llegada de nuevos integrantes al equipo, provenientes incluso de una empresa aliada de Ucrania con amplia experiencia, se organizó la transición del framework de testing hacia estos subequipos. El framework fue sometido a un rework para hacerlo más sólido y escalable, y en este proceso participé acompañando la toma de decisiones y manteniendo sincronizaciones frecuentes para explicar las elecciones realizadas en el pasado y analizar, frente a nuevas alternativas, cuál era el mejor camino a seguir.

Actualmente, el foco está puesto en proyectos más ambiciosos, entre los que destaca el nuevo producto de **detección de anomalías**, compuesto por tres partes: *Anomaly Detection*, *Janus* y *RCA*. Cada uno de estos módulos exige aprender y aplicar conceptos más cercanos a los trabajados en la universidad en materias como **Sistemas Inteligentes** y **Procesamiento de Imágenes**, incluyendo manejo de dataframes, inteligencia artificial y estadística. Todo lo desarrollado y aprendido durante la PPS me facilita este trabajo, ya que los entornos y recursos de testing son los mismos, pero ahora aplicados a productos de mayor complejidad, que incorporan incluso el uso de **LLMs y algoritmos basados en estadística**.

De cara al futuro, aún estoy explorando distintos caminos posibles. Los roles de liderazgo y management me resultan atractivos, así como la especialización en **data science**, un área que me interesa profundamente. Al mismo tiempo, me motiva la idea de consolidarme bajo el rol

de **software engineer**, un objetivo personal que también considero como una opción para los próximos años.

Bibliografía

[1] Documentación oficial de Hydrolix. *Hydrolix Streaming Data Lake*. Disponible en: https://docs.hydrolix.io

[2] Documentación oficial de ClickHouse. *Introduction to ClickHouse*. Disponible en: https://clickhouse.com/docs/en/

[3] PySpark Documentation. Disponible en: https://spark.apache.org/docs/latest/api/python/

[4] K9s – Kubernetes CLI management tool. Disponible en: https://k9scli.io/

[5] Documentación oficial de Splunk. *Search Processing Language (SPL)*. Disponible en: https://docs.splunk.com

[6] Splunklib Python SDK. Repositorio oficial. Disponible en: https://github.com/splunk/splunk-sdk-python

[7] Documentación de Databricks Workflows. Disponible en: https://docs.databricks.com/workflows/index.html

[8] Documentación oficial de Python– *concurrent.futures*. Disponible en: https://docs.python.org/3/library/concurrent.futures.html

[9] Documentación oficial de Python – *Classes and OOP*. Disponible en: https://docs.python.org/3/tutorial/classes.html

[10] Selene UI testing framework. Repositorio GitHub. Disponible en: https://github.com/yashaka/selene

[11] Documentación oficial de Docker. Disponible en: https://docs.docker.com

[12] Hydrolix Internal Documentation (Confluence). Documentación de procesos internos, prácticas de desarrollo y arquitectura. Acceso restringido.

[13] Documentación oficial de Apache Spark. https://spark.apache.org/docs/latest/

Agradecimientos

Quiero agradecer en primer lugar a la **Universidad Nacional del Noroeste de la Provincia de Buenos Aires (UNNOBA)** por la formación y el acompañamiento durante toda la carrera. En especial, a **Javier Charne**, por coordinar esta PPS y por haber sido un gran profesor y motivador; y a su mano derecha **Diego Pérez**, por el apoyo constante. También a **Leonardo Esnaola**, **Sandra Serafino** y **Benjamín Cicerchia** por devolverme la pasión por la programación y sentar las bases de lo que hoy es mi camino profesional como *Data Scientist, Data Engineer, AI specialist o cualquiera sea el título que termine teniendo en mi perfil.*

A **Clarolab**, gracias a **Francisco** y **Vero** por hacer que la empresa se sienta como un hogar, y a **Pablo** por ser un gran mentor y una gran persona que me ayudó en muchos momentos de mi corta carrera profesional.

A la gente de **Innovation Lab (Hydrolix, USA)**: a **Alok** (mi manager), **Ethan**, **Val** y **Peter**, que a esta altura ya son como amigos.

A mis compañeros de **Hydrolix** en la oficina: **Gaspar**, **Bruno** y **Walter**, por enseñarme todo lo que sé (y bancarse que los volviera locos varios meses); y a **Benja**, **Silvano**, **Luz**, **Nati**, **Ramiro**, **Facu**, **Ivo**, **Bruno,** **Leo** y **Juan** por ayudarme tanto y hacer que la oficina sea un lugar en el que quiero estar todos los días.

A **Facu**, por ser el mejor compañero que me pudo tocar en Innovation Lab; y a **Lucas**, por ser un gran sucesor.

Finalmente, a mi **familia** y, principalmente, a mis **amigos**: **Rocío**, **Lía**, **Rodrigo**, **Sol**, **Alan**, **Gaspar**, **Luisina**, **Franco** y a todos los que me acompañaron en este camino. Gracias por el apoyo incondicional, la paciencia y el empujón justo cuando más lo necesité.