ESCUELA POLITÉCNICA DE INGENIERÍA DE GIJÓN

GRADO EN INGENIERÍA INFORMÁTICA EN TECNOLOGÍAS DE LA INFORMACIÓN

Lenguajes y Sistemas Informáticos

TRABAJO FIN DE GRADO/MÁSTER Nº ???

Explotación, integración y visualización de múltiples fuentes de datos mediante un Data Lake

Mier Montoto, Juan Francisco

TUTORES:
D. Augusto Alonso, Cristian
D. Morán Barbón, Jesús
D. Vázquez Faes, Eduardo

FECHA: julio 2024

Índice de contenido

Índice de figuras

Índice de tablas

1. Introducción

El proyecto que se presenta en este documento tiene como objetivo la automatización de despliegue de la infraestructura y procesos que permitan hacer un análisis masivo de datos. Para conseguir este objetivo, se analizan las necesidades de la empresa y sus antecedentes, se valoran las tecnologías y herramientas disponibles y se plantea una solución que permita la integración, almacenamiento y análisis de grandes volúmenes de datos, provenientes de múltiples fuentes y en diferentes formatos.

1.1. Antecedentes

Hoy en día, nos encontramos en una era donde la generación y almacenamiento de datos crece exponencialmente [1], un hecho que se ve reflejado en el ámbito empresarial. La diversidad de fuentes y formatos de estos datos introduce una complejidad significativa en su manejo, conocida como *heterogeneidad* [2], siendo las bases de datos, archivos de registros y APIs las fuentes más habituales.

El término *big data* describe este fenómeno de acumulación masiva de datos, cuya magnitud y complejidad sobrepasan las capacidades de los métodos de procesamiento convencionales. El *big data* se caracteriza por tres características principales: volumen, variedad y velocidad - su adecuada gestión y análisis pueden otorgar ventajas competitivas significativas a las empresas, tales como el descubrimiento de patrones ocultos, identificación de nuevas oportunidades de mercado y optimización de procesos de toma de decisiones.

Uno de los procesos que permite la extracción de esta información es la pirámide DIKW, [1] es un modelo que describe la relación entre los datos, la información, el conocimiento y la sabiduría. Según este modelo, los datos son la materia prima de la información, que a su vez es la materia prima del conocimiento, que a su vez es la materia prima de la sabiduría. Una organización sin los procesos adecuados para la gestión y análisis de estos datos, se enfrenta a importantes desafíos, como la dificultad para identificar patrones y tendencias, la toma de decisiones incorrectas y la pérdida de oportunidades de negocio. Por otro lado, una organización que logre extraer información valiosa de sus datos, podrá mejorar su eficiencia, aumentar su competitividad y adaptarse mejor a un entorno empresarial en constante cambio.

La evolución tecnológica ha propiciado el desarrollo de innovadoras herramientas y me-

[1] https://www.statista.com/statistics/871513/worldwide-data-created/
[2] https://www.sciencedirect.com/topics/computer-science/data-heterogeneity

todologías diseñadas para enfrentar estos desafíos. Entre ellas, los *data lakes* (o *lagos de información*) se destacan por su capacidad para consolidar vastos volúmenes de datos heterogéneos, facilitando su posterior análisis y aprovechamiento de manera más efectiva.

Sin embargo, a pesar de que existen herramientas de almacenamiento, el proceso de integración, visualización y análisis de estos datos es una tarea desafiante, ya que requiere de una gran cantidad de recursos y de un tiempo de desarrollo considerable del que, normalmente, no se dispone en el ámbito empresarial.

Con la ingesta masiva de datos, se presentan nuevos problemas a la hora de analizar y obtener información de ellos:

- Sin la necesaria automatización y correcta aplicación de los procesos ETL, al tratarse de un crecimiento exponencial de los datos y, por lo tanto, de la fuerza de trabajo necesaria para manejarla, los resultados del análisis pueden ser incorrectos, lo que deriva en errores y decisiones de negocio equivocadas que impactan negativamente en la empresa.
- La heterogeneidad de los datos, tanto en formato como en origen, dificulta su consolidación y análisis.
- La masificación de información impide el análisis manual de los mismos, requiriendo resúmenes estadísticos o representaciones gráficas como *dashboards* para su correcta interpretación. La visualización de datos es una técnica que permite representar la información de manera visual, para facilitar su análisis y comprensión, una parte vital del proceso de análisis de datos, ya que permite identificar patrones, tendencias y anomalías en los mismos de forma más rápida y sencilla.

1.2. Motivación

Okticket, como el resto de empresas, se enfrenta a la necesidad de gestionar y analizar grandes volúmenes de datos, provenientes de múltiples fuentes y en diferentes formatos. La correcta gestión y análisis de estos datos es fundamental para la toma de decisiones y para la mejora de los procesos internos de la empresa.

En la actualidad, la empresa dispone de una gran cantidad de datos que se encuentran en diferentes formatos y en diferentes ubicaciones, lo que dificulta su análisis y explotación. Por otra parte, se depende de la consulta manual o de servicios de terceros para poder analizar estos datos, lo que supone un coste adicional.

El proyecto surge de la necesidad de la empresa de extraer información y conocimiento de las múltiples y heterogéneas fuentes de datos de las que se disponen, tanto internas (e.g. bases de datos, archivos de registros, APIs, entre otros), como externas (e.g. APIs o datos de webs de terceros, datos de fuentes públicas. . .).

Además del uso interno, la empresa también quiere ofrecer a sus clientes la posibilidad de consultar estos datos de forma visual y sencilla, para que puedan analizarlos y explotarlos de forma autónoma, lo que supondría un valor añadido para los mismos.

1.3. La empresa

Okticket es una startup nacida en Gijón en 2017 cuyo producto principal es un servicio software que escanea automáticamente tickets y notas de gastos, lo que permite reducir los costes y el tiempo que invierten las empresas en contabilizar y manejar los gastos de viaje de los profesionales.

La empresa tienen su suede principal en el Parque Tecnológico de Gijón, aunque cuenta con un número de sedes creciente en varios países, como Francia, Portugal o, más recientemente, México. En esta oficina principal se encuentran los departamentos de ventas y marketing, así como el equipo de desarrollo y soporte.

Okticket es una de las empresas que más crecen tanto del sector como del propio Parque Tecnológico. Debido a este rápido crecimiento, el equipo está en constante desarrollo y cambio, tanto aquí en España como en el resto de sedes. Este crecimiento se refleja en la recepción de un gran número de galardones y reconocimientos. [3] [4] [5] [6]

La parte principal del negocio es el núcleo del software como servicio (Software as a Service en inglés, en adelante *SaaS*), es decir, la aplicación completa tanto para administradores como para empleados. Este SaaS se oferta a empresas de cualquier tamaño, cuyo precio final varía en función del número de usuarios, las características e integraciones que requiera la empresa cliente y el soporte que se ofrezca.

Recientemente se han añadido nuevas propuestas a la cartera de servicios ofertada por Okticket, como la OKTCard - una tarjeta inteligente que gestiona automáticamente los gastos, así como la inclusión de nuevos "módulos" de gestión de gastos y viajes.

Debido a todo este crecimiento, la empresa maneja una gran cantidad de datos importantes que se encuentran en diferentes formatos y en diferentes ubicaciones, lo que dificulta su análisis y explotación. Por otra parte, se depende de la consulta manual o de servicios de terceros para poder analizar estos datos, lo que supone un coste adicional.

[3] Okticket en el especial startups 2023 de Forbes (LinkedIn)
[4] Arcelor y Okticket, premios nacional de Ingeniería Informática (EL COMERCIO)
[5] Okticket recibe el sello Pyme Innovadora (okticket.es)
[6] Okticket, empresa emergente certificada (okticket.es)

1.4. Objetivos

El objetivo del proyecto es la creación de un proceso que permita el despliegue automático de una infraestructura de datos que permita la integración, almacenamiento y análisis de grandes volúmenes de datos, provenientes de múltiples fuentes y en diferentes formatos. La infraestructura de datos debe ser escalable, flexible y robusta, para poder adaptarse a las necesidades cambiantes de la empresa.

La infraestructura de datos debe permitir la integración de datos de múltiples fuentes, tanto internas como externas, y en diferentes formatos, como bases de datos, archivos de registros, APIs, entre otros. La integración de datos debe ser automática y programable, para poder automatizar el proceso de ingestión de datos y reducir el tiempo y los costes asociados.

Pese a que el entregable principal de este proyecto es la creación de una infraestructura, se esperan también otros entregables en forma de herramientas de software, como *scripts*, que faciliten la integración y análisis de los datos, así como la visualización de los mismos.

2. Fundamento teórico

En este capítulo se presentan los conceptos y términos fundamentales que se utilizan en el proyecto, para proporcionar una base teórica sobre la que se desarrolla el trabajo. Se discuten los conceptos fundamentales para el desarrollo del proyecto.

2.1. *Big data*

El término *big data* se refiere a la gestión y análisis de grandes volúmenes de datos que no pueden ser tratados de manera convencional. La evolución natural del progreso tecnológico, la digitalización de la sociedad y la aparición de nuevas tecnologías han propiciado la generación de grandes cantidades de datos en todo el mundo, lo que genera la necesidad de nuevas formas de gestionar y tratar estos datos.

El término *big data* no solo se refiere a la cantidad de datos que se generan, sino a también otras características, a las que se refieren como "las uves del big data". La cantidad de *uves* depende del autor y de la fuente [2, 3], variando desde 3 hasta 7, pero las más comunes son las siguientes:

- **Volumen:** la más básica, la cantidad de datos que se generan y almacenan en un determinado periodo de tiempo. El volumen de datos que se maneja en el *big data* es mucho mayor que el que se maneja en los sistemas tradicionales de gestión de datos, que además se encuentra en aumento constante.
- **Variedad:** se refiere a la diversidad de fuentes y formatos de los datos que se manejan. Los datos pueden provenir de diversas fuentes, como bases de datos, archivos de registros o APIs, pueden estar en diferentes formatos o tener diferentes estructura. Se pueden clasificar de la siguiente manera:
  - **Estructurados:** datos que se encuentran en un formato estructurado, como una base de datos relacional.
  - **Semi-estructurados:** datos que no se encuentran en un formato estructurado, pero que tienen una estructura interna que permite su análisis, como un archivo XML o JSON.
  - **No estructurados:** datos que no tienen una estructura definida, como un archivo de texto o una imagen.

- **Velocidad:** se refiere a la frecuencia con la que se generan y se procesan los datos. En este ámbito, los datos se tratan a una velocidad mucho mayor que en los sistemas tradicionales de gestión de datos. Dicha velocidad puede ser crítica en ámbitos como la bolsa, donde la velocidad de procesamiento de los datos puede ser la diferencia entre obtener beneficios o pérdidas. Según la frecuencia de procesamiento de los datos, los sistemas se pueden clasificar en:

  - **Batch (en lotes):** los datos se procesan en lotes, de manera periódica, como cada hora o cada día. Este tipo de datos, frecuente en aplicaciones que se tratan en Okticket como los notas de viajes o las nóminas, no requieren un procesamiento inmediato.
  - **Streaming (en tiempo real):** los datos se procesan en tiempo real, a medida que se generan. Este tipo de datos, que se puede encontrar en aplicaciones como los sensores o los logs, requieren un procesamiento inmediato si se quiere obtener información relevante sobre los mismos.
  - **Near real-time (casi en tiempo real):** los datos se procesan con un pequeño retraso, de manera que se obtiene información relevante sobre los mismos en un tiempo muy corto.

- **Veracidad:** se refiere a la calidad de los datos que se manejan. La veracidad de los datos es un factor crítico en el ámbito del *big data*, ya que los datos incorrectos o incompletos pueden llevar a decisiones incorrectas y a problemas en la empresa. La veracidad de los datos se puede ver afectada por diferentes factores, como la calidad de los datos, la precisión de los mismos, la integridad de los datos, etc.

2.2. Paradigmas de almacenamiento de datos

En el ámbito del *big data*, existen diferentes paradigmas de almacenamiento de datos que se utilizan para almacenar y analizar grandes cantidades de información. Los tres paradigmas a considerar para este proyecto son los *data warehouses*, los *data lakes* y los *data lakehouses*.

2.2.1. Data warehouse

Un *data warehouse*[1], también conocido en español como almacén de datos, es una base de datos que se utiliza para almacenar y analizar grandes cantidades de datos de manera eficiente. Los almacenes de datos proporcionan acceso rápido y compatible con plataformas de consultas (como SQL) a grandes cantidades de datos, lo que permite a los analistas y a los científicos de datos realizar análisis complejos sobre los datos almacenados.

Todos los datos almacenados en un *data warehouse* se encuentran en un formato común, para lo que se aplican procesos ETL (extracción, transformación y carga) que transforman los datos de diferentes fuentes en un formato común. Esto significa que la información se encuentra en un formato o esquema optimizado y específico, lo que facilita su manipulación y análisis pero limita la flexibilidad al acceso de los datos y genera costes adicionales en el caso de tener que modificar o transferir los mismos para su uso.

2.2.2. Data lake

Los *data lakes*[2] son almacenes de datos que guardan grandes cantidades de datos de manera no estructurada [4]. En el ámbito de una empresa, un *data lake* contiene datos de diferentes fuentes de valor no considerado hasta su análisis, de manera que su explotación posterior y su análisis no depende de una estructuración y transformación compleja, reduciendo los costes de los procesos ETL derivados, una flujo de tareas que se aplican sobre la información para ingestarla. Esto no quiere decir que no se apliquen estos procesos a los datos, sino que se aplican de manera más flexible y básica que en otras estructuras de almacenamiento de datos con esquemas predefinidos, como los *data warehouses*. [5]

A diferencia de los *data warehouses*, los *data lakes* no tienen un esquema definido, lo que permite almacenar datos *heterogéneos*. Esto permite almacenar grandes cantidades de información sin tener que definir un esquema de antemano, lo que puede ser útil en aquellos casos en los que no se conoce la estructura de los datos que se van a almacenar.

[1] `https://aws.amazon.com/es/data-warehouse/`
[2] `https://aws.amazon.com/es/what-is/data-lake/`

Estas características de los *data lakes* hacen que sean más atractivos en el sector empresarial de cara al análisis de información, en contraste con las estructuras planteadas normalmente en el campo de la investigación académica.

Para consultar esta gran cantidad de datos almacenados, se suelen utilizar técnicas de visualización de datos, como los *dashboards*, herramientas de visualización que permiten observar los datos de manera sencilla y eficiente.

2.2.3. Data lakehouse

Los *data lakehouses* son una combinación funcional de los dos paradigmas vistos anteriormente, los *data lakes* y los *data warehouses*. Los *data lakehouses* permiten almacenar datos tanto de manera estructurada como no estructurada, lo que facilita aprovechar la información al contar con una única estructura de bajo coste que ofrece a los usuarios que lo necesiten explorar y analizar los datos según sus necesidades.

2.3. Procesos ETL

Si anteriormente se presentaban los distintos paradigmas de almacenamiento de datos, para su creación y mantenimiento se requieren aplicar unos ciertos procesos que permitan la correcta ingesta y almacenamiento de los datos. Estos procesos se conocen como *procesos ETL*.

Formalmente se definen los procesos ETL [4] como procesos que combinan datos de múltiples fuentes en un único destino, transformando los datos en un formato común. Estos procesos se utilizan para extraer datos de diferentes fuentes, transformarlos en un formato común y cargarlos en un destino común, como puede ser un *data lake*.

Los procesos ETL, fundamentales en el ámbito de la gestión de datos, presentan atributos distintivos que facilitan la integración eficaz de información procedente de diversas fuentes:

- **Adaptabilidad:** los procesos ETL deben de adaptarse a la estructura de los datos de la fuente de origen, ya que dichas fuentes pueden tener diferentes estructuras y tener tipos de datos diferentes (la característica de *heterogeneidad* de los datos que ya se ha mencionado).
- **Escalabilidad:** otra de las características clave de los procesos ETL es que sean escalables, ya que los datos que se muestran en los dashboards suelen ser datos que se generan de manera continua, y por lo tanto los procesos ETL deben ser capaces de procesar grandes cantidades de datos de manera eficiente. En ocasiones, los procesos ETL se pueden realizar en *streaming*, lo que significa que los datos se procesan en tiempo real a medida que se generan.
- **Eficiencia:** los procesos ETL deben ser eficientes, puesto que el tiempo de procesamiento de los datos es un factor vital en el ámbito del *big data*. Los procesos ETL deben ser capaces de procesar grandes cantidades de datos en un tiempo razonable para que los datos estén disponibles en el menor tiempo posible.
- **Fiabilidad:** la fiabilidad es un componente crítico de todo el flujo de datos, ya que estos se utilizan para la toma de decisiones importantes de cualquier empresa. Los procesos ETL deben ser capaces de procesar los datos de manera fiable y consistente, para que los datos que se visualicen y analicen posteriormente sean correctos y fiables.

2.3.1. Funcionamiento

Los procesos ETL se dividen en tres fases principales: *(1) Extraer*, *(2) Transformar* y *(3) Cargar*, como se muestra en el siguiente diagrama:

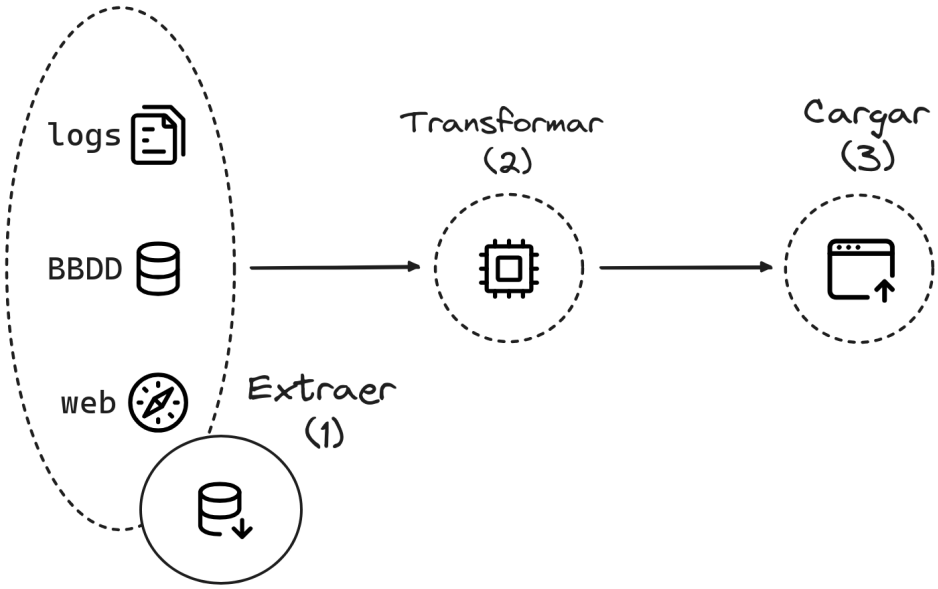

Figura 2.1: Fases de un proceso ETL

Como entrada, se tienen datos presuntamente heterogéneos que no se pueden analizar de manera eficiente. Tras aplicar todos los pasos de las fases anteriores, se obtiene como salida un conjunto de datos corregidos y listos para ser analizados en el destino indicado (en el caso de este proyecto, un *data lake*).

Extracción (1) En este proceso se obtienen los datos de las fuentes de datos, que pueden ser bases de datos, logs, APIs, etc. En esta fase, se pueden aplicar filtros para extraer solo los datos que se necesiten, y se pueden extraer datos de múltiples fuentes *heterogéneas*.

La fase de extracción se puede realizar de dos formas: continua o incremental. Una extracción incremental se realiza de manera periódica, por ejemplo, cada hora, cada día o cada semana, y se extraen los datos que se han generado desde la última extracción. Esto es útil cuando los datos se generan de manera periódica y se necesita mantener actualizada la información. Por otro lado, en una extracción continua se extraen los datos en tiempo real según se van generando. Esto puede ser útil para procesar datos que se generan en tiempo real, como logs o datos de sensores.

Transformación (2) Durante esta fase, se transforman los datos extraídos en la fase anterior, normalmente aplicándoles un proceso de limpieza y transformación a un formato común. En este paso, se pueden aplicar diferentes operaciones a los datos, como la limpieza, la agregación, la normalización, la conversión de formatos, etc.

Uno de los tipos de transformaciones de datos más comunes es la limpieza, que consiste en la revisión y corrección de los datos extraídos, para asegurar que se almacena información correcta y consistente. Durante esta fase se contemplan operaciones más complejas, como pueden ser la agregación de datos, la conversión de formatos, la normalización de datos, el cifrado, etc. La limpieza de datos puede ser una tarea muy sencilla, como la eliminación de caracteres delimitadores, o muy compleja, como la corrección de errores en los datos o la detección de duplicados.

Estos procesos de transformación son vitales cuando el sistema maneja una gran cantidad de datos heterogéneos de múltiples fuentes de manera simultánea, como puede ser el caso de un *data lake* o un *data warehouse*. En el caso del primero, no es necesaria la transformación de los datos a un formato común, pero si otros procesos clave como la limpieza y la normalización de los datos, entre otros.

Carga (3) En este proceso se vuelcan los datos transformados en el destino final. Frecuentemente, los datos se almacenan, dependiendo del paradigma de almacenamiento elegido, en una *data lake*, *data warehouse* o *data lakehouse* para su posterior análisis.

2.3.2. Alternativas

Aunque lo más común es el flujo anteriormente explicado de *extracción*, *transformación* y *carga*, existen algunos flujos y procesos alternativos que evitan algunos de estos pasos, normalmente en casos específicos que se beneficien del cambio:

- **Virtualización de datos:** capa virtual de abstracción que permite acceder a los datos de las fuentes sin necesidad de extraerlos. Esto permite ahorrar espacio de almacenamiento y tiempo de procesamiento, pero suele ser menos eficiente en términos de rendimiento y no es compatible con todas las arquitecturas de datos.

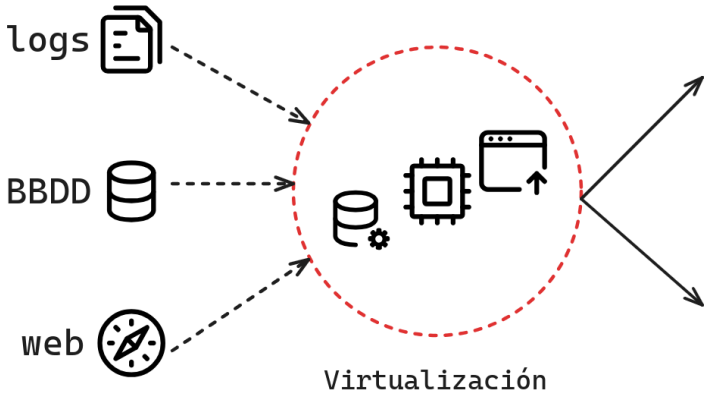

Figura 2.2: Ejemplo de flujo con virtualización

- **Proceso *ELT*[3]:** en lugar de transformar los datos antes de cargarlos en el destino, se cargan los datos en bruto y se transforman en el destino. Funciona bien para grandes conjuntos de datos sin estructura que requieran una carga (o recarga) contínua, aunque, al igual que la virtualización, puede ser menos eficiente o incompatible con algunas arquitecturas de datos, como los *data warehouses*.

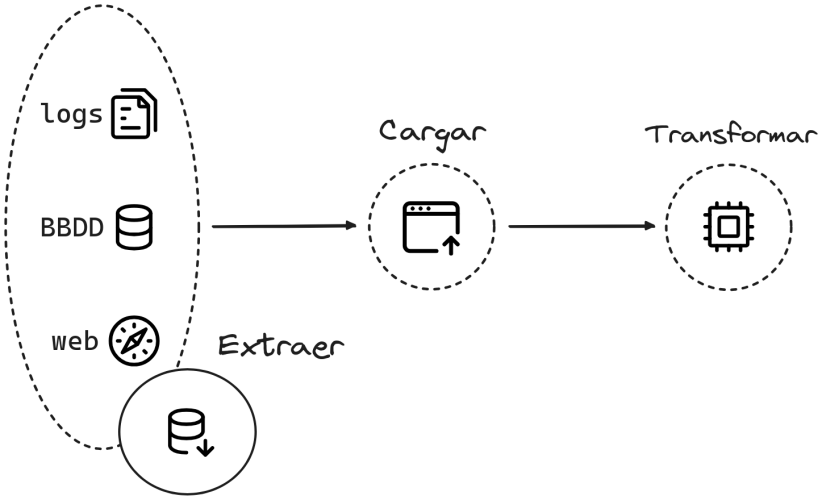

Figura 2.3: Diagrama de flujo de un proceso *ELT*

[3] https://www.ibm.com/topics/elt

2.4. Cuadros de mandos (*dashboards*)

Definición Los cuadros de mandos, en adelante *dashboards*, es un término que se utiliza para referirse a cualquier interfaz gráfica que muestre información relevante de manera visual sobre un proceso o negocio. Aunque el término se utiliza en muchos ámbitos: indicadores comerciales, de producción, de marketing, de calidad, de recursos humanos. . . , en este proyecto se utilizará en el ámbito de la monitorización de sistemas y procesos de negocio.

En el ámbito de deste proyecto, los dashboards reflejan en tiempo real el rendimiento de actividades o procesos de negocio, y se utilizan para tomar decisiones informadas basándose en los mismos. Por ejemplo, el dashboard de una empresa digital puede mostrar desde el rendimiento de la arquitectura en tiempo real hasta el número de ventas conseguidas, y permitir a los directivos tomar decisiones informadas sobre el futuro de la empresa (e.g. necesidad de aumentar la capacidad de los servidores, lanzar una campaña de marketing, etc.).

Características Los dashboards cuentan con una serie de características que los hacen útiles para la toma de decisiones: [4]

- **Visualización de datos:** es la característica fundamental de cualquier dashboard, y aquella que determina su utilidad. La visualización de datos es la ciencia de presentar los datos de manera que se pueda extraer información útil y realizar decisiones informadas sobre ellos. Un buen dashboard cuenta con gráficas, tablas, indicadores, etc. que permiten al usuario entender la información que se está presentando con un conocimiento técnico mínimo.
- **Interactividad y personalización:** un dashboard debe permitir al usuario interactuar con los datos (filtrarlos, ordenarlos, profundizar en ellos...) y ajustar la información que se muestra sobre cada proceso o negocio que se esté evaluando (granularidad de la información). Esta capacidad asegura que el dashboard se adapte tanto a las necesidades actuales como a las evoluciones futuras de lo que se esté analizando.
- **Accesibilidad y portabilidad:** un dashboard debe ser accesible desde una variedad de situaciones y dispositivos, manteniendo su funcionalidad y forma. Aunque normalmente los dashboards se analizan en pantallas grandes, es importante que también se puedan consultar en otras circunstancias, como dispositivos móviles.

Dashboards planteados Para el sistema que se describe, se plantean dos tipos de dashboards diferentes:

- **Dashboards internos**: que reflejan el rendimiento de la plataforma en tiempo real. Estos dashboards están destinados al uso interno de la empresa, y permiten a los empleados monitorizar el rendimiento de la plataforma y tomar decisiones informadas sobre su mantenimiento y evolución.
- **Dashboards externos**: que reflejan el rendimiento de las ventas y permiten a los clientes tomar decisiones informadas sobre su negocio. Estos dashboards están enfocados a los clientes de la empresa, y permite a los mismos obtener información relevante sobre su negocio que tenga Okticket.

2.5. Infraestructura como código

La infraestructura como código (o *IaaS* por sus siglas en inglés) es una práctica que consiste en gestionar la infraestructura de un sistema de manera automática y programática mediante código, en lugar de configuraciones manuales. La infraestructura como código permite gestionar la arquitectura global de un sistema de manera eficiente y escalable, y facilita la creación y el mantenimiento de entornos de desarrollo y producción.

En el ámbito de este proyecto, la infraestructura como código se utiliza para gestionar el despliegue y orquestación de los servicios requeridos para el *data lake*, como los servicios de ingesta o de visualización de datos.

La infraestructura como código es la parte más importante del desarrollo de este proyecto, ya que una buena configuración y toma de decisiones a la hora de desplegar un proyecto de este calibre es vital para el correcto funcionamiento posterior a la hora de ingestar y tratar con datos heterogéneos.

3. Descripción general del proyecto

Esta sección describe el proyecto en términos generales, incluyendo una descripción de los problemas que se pretenden resolver, las partes interesadas en el proyecto y una valoración de las alternativas consideradas.

3.1. Partes interesadas (*stakeholders*)

Las partes interesadas en el proyecto son aquellas personas o entidades que tienen un interés en el mismo, ya sea porque se ven afectadas por el resultado del proyecto, o porque tienen algún tipo de interés en el mismo. Las partes interesadas en este proyecto son las siguientes:

1. **Okticket**: la empresa es la principal parte interesada en el proyecto, ya que es la que se beneficiará directamente de los resultados del mismo, así como de las oportunidades de negocio que se abren con la explotación de los datos. Dentro de la empresa, se pueden identificar dos entidades:
   - **Equipo de desarrollo de la empresa**: el equipo de desarrollo es otra parte interesada en el proyecto, ya que son los encargados de llevar a cabo la implementación del sistema y de garantizar su correcto funcionamiento, además de gestionar el soporte de servicio a nivel técnico.
   - **Equipo de soporte de la empresa**: el sistema planteado ahorraría tiempo al equipo de soporte, ya que les permitiría analizar los datos de forma más eficiente e identificar problemas antes de que tener que resolver las peticiones de los clientes afectados a nivel básico.
2. **Clientes**: los clientes de la empresa también son partes interesadas, puesto que se beneficiarán de los nuevos servicios que se ofrecen, como los dashboards de negocio que se han descrito anteriormente. Estos clientes no son necesariamente los usuarios finales, sino los administradores y gestores de las empresas que utilizan Okticket como herramienta de gestión de gastos.
3. **Investigador y desarrollador (*Mier Montoto, Juan Francisco*):** el desarrollador del proyecto tiene la oportunidad de aplicar los conocimientos adquiridos en el desarrollo de un proyecto real, y de adquirir nuevos conocimientos en el proceso.

3.2. Alternativas existentes

Antes de comenzar la planificación y el desarrollo del proyecto, se consideran varias opciones ya existentes en el mercado que podrían resolver o adaptarse a las necesidades planteadas.

3.2.1. Criterios de evaluación

Los puntos claves a considerar de cada alternativa son los siguientes:

- **Coste:** el coste es un factor clave para evaluar las alternativas, prefiriendo aquellas que sean gratuitas o cuya implementación requiera la mínima inversión posible.
- **Complejidad:** al tratarse de un proyecto con poco conocimiento previo y sin mucha urgencia, se prefiere una solución que sea sencilla y que no requiera demasiado desarrollo.
- **Rendimiento y escalabilidad:** obviamente, a la hora de manejar los volúmenes de datos con los que se están tratando, el sistema debe responder positivamente tanto a la ingesta, tratamiento y visualización como a la escalabilidad del mismo.

Como punto añadido, se valoran de manera positiva las alternativas *open-source*, ya que son más flexibles y permiten personalizar el sistema a los necesidades del proyecto.

3.2.2. Alternativas consideradas

A continuación, se describen brevemente estas alternativas y se comparan sus características.

3.2.2.1. Loggly, Splunk (SaaS de gestión de logs)

Ambas herramientas se centran en la gestión y análisis de logs basada en la nube, permitiendo centralizar, monitorizar y analizar datos de logs en tiempo real. Diseñadas para simplificar la gestión de logs, las dos alternativas ofrecen una interfaz intuitiva y potentes capacidades de búsqueda y visualización que facilitan la identificación y resolución de problemas en los sistemas y aplicaciones.

Figura 3.1: Logo de Loggly ®

Una de las principales ventajas de este tipo de herramientas es su capacidad para integrarse con una amplia variedad de servicios y plataformas, lo que permite a las empresas consolidar sus datos de logs en un único lugar. Además, suelen ofrecer algún sistema de alertas en tiempo real y paneles de control personalizables, lo que encaja muy bien con algunos de los requisitos de este proyecto.

Sin embargo, también presentan algunas desventajas que entran en conflicto con los intereses descritos anteriormente. En primer lugar, al tratarse de servicios enfocados en el tratamiento exclusivo de logs, podrían no acomodar algunos de los requisitos esenciales, como la ingesta de las bases de datos propias de la empresa.

Figura 3.2: Logo de Splunk ®

Otra posible limitación es su sistema de precios; estas herramientas cuentan con una jerarquía de suscripciones que limita mucho su escalabilidad y aumentaría rápidamente los costes de su uso en caso de llegar a depender de ellas.

En resumen, aunque tanto Loggly como Splunk como cualquier otro SaaS por el son soluciones robustas y eficientes para la gestión de logs, sus posibles costes adicionales, junto con sus limitaciones en el análisis de datos de inteligencia de negocio, las convierten en opciones menos atractivas para el desarollo de este proyecto.

3.2.2.2. Loki

Loki es una herramienta de gestión de logs desarrollada por Grafana Labs, diseñada específicamente para ser altamente eficiente y escalable. A diferencia de otras soluciones de gestión de datos, Loki no indexa el contenido completo de estos, sino que se centra en los metadatos, lo que reduce significativamente los requisitos de almacenamiento y mejora el rendimiento.

Figura 3.3: Logo de Loki ®

Ventajas

- **Integración con Grafana:** Una de las principales ventajas de Loki es su integración nativa con Grafana, una popular plataforma de visualización de datos. Esto permite a los usuarios crear paneles de control y alertas basados en los datos de logs de manera sencilla y eficiente.
- **Eficiencia en el almacenamiento:** Al no indexar el contenido completo de los logs, Loki reduce significativamente los requisitos de almacenamiento. Esto lo hace una opción más económica y eficiente en comparación con otras soluciones.
- **Escalabilidad:** Loki está diseñado para ser altamente escalable, lo que permite manejar grandes volúmenes de datos de logs sin comprometer el rendimiento.
- **Simplicidad en la configuración:** La configuración de Loki es relativamente sencilla, especialmente para aquellos que ya están familiarizados con Grafana y Prometheus. Esto facilita su adopción y despliegue en entornos de producción.

Desventajas

- **Limitaciones en la búsqueda:** Al no indexar el contenido completo de los logs, las capacidades de búsqueda de Loki son más limitadas en comparación con otras herramientas como Elasticsearch. Esto puede ser una desventaja en caso de necesitar realizar búsquedas complejas y detalladas.
- **Ecosistema en desarrollo:** Aunque Loki ha ganado popularidad rápidamente, su ecosistema y comunidad de usuarios aún están en desarrollo. Esto puede limitar el acceso a recursos y soporte en comparación con soluciones más maduras.
- **Dependencia de Grafana:** Si bien la integración con Grafana es una ventaja, es un factor limitante para aquellos que no utilicen Grafana, como es el caso de Okticket, al tener que adaptarse a un ecosistema diferente.

En resumen, Loki es una solución eficiente y escalable para la gestión de datos, especialmente adecuada para aquellos que ya utilizan Grafana. Sin embargo, sus limitaciones en la búsqueda y su ecosistema en desarrollo son factores a considerar antes de su implementación.

3.2.2.3. Graylog + Prometheus

Esta combinación de herramientas es una alternativa interesante para la gestión de logs y métricas en entornos de producción. Graylog es una plataforma de gestión de logs de código abierto que permite centralizar, monitorizar y analizar logs de diferentes fuentes. Por otro lado, Prometheus es un sistema de monitorización y alertas de código abierto que se centra en la recopilación de métricas de sistemas y aplicaciones.

Figura 3.4: Logo de Graylog ®

En este documento se tratan de manera combinada porque, aunque son herramientas independientes, su integración es una solución completa que se está volviendo popular en el sector. Graylog se encarga de la gestión de logs, mientras que Prometheus se encarga de la recopilación de métricas y alertas.

La combinación de Graylog y Prometheus ofrece varios beneficios, como la centralización de logs y métricas que facilita un análisis más integral y eficiente de los datos. Además,

la mejora en la monitorización a través de las capacidades avanzadas de Prometheus, integradas con Graylog, mejora la detección de problemas y la respuesta a incidentes en tiempo real. Esta integración también proporciona flexibilidad y escalabilidad, adaptándose a las necesidades específicas de cada entorno de producción. Sin embargo, existen desventajas como la complejidad de configuración al integrar dos sistemas independientes, lo que puede resultar en un mayor esfuerzo en mantenimiento y gestión. Además, al tratarse de herramientas novedosas, supondría una mayor curva de aprendizaje para el equipo de desarrollo a la hora de tratar con las herramientas.

Figura 3.5: Logo de Prometheus ®

3.2.3. Conclusiones

Tras evaluar las alternativas consideradas, se concluye que ninguna de ellas se ajusta completamente a los requisitos y necesidades del proyecto. Si bien todas las alternativas ofrecen capacidades de gestión de logs y métricas, ninguna de ellas proporciona una solución integral que cumpla con los requisitos de integración, visualización y análisis de datos de negocio de Okticket.

Por lo tanto, se considera que la mejor opción es desarrollar una solución personalizada que se adapte a las necesidades específicas de la empresa. Esta solución permitirá a Okticket consolidar y analizar datos de múltiples fuentes, así como crear dashboards de negocio personalizados que faciliten la toma de decisiones y la identificación de oportunidades de negocio.

Desde un principio, la empresa considera que el desarrollo junto a un *stack ELK*, es decir, Elasticsearch, Logstash y Kibana, es la mejor opción para el desarrollo de este proyecto. Aunque se considera que la integración de Prometheus y Grafana es una alternativa interesante, se opta por la solución de *stack ELK* debido a su mayor flexibilidad y capacidad de personalización.

Además de la pila de *Elastic*, se analizará más adelante la posibilidad de integrar otras herramientas y tecnologías que puedan mejorar la eficiencia y escalabilidad del sistema, como Kafka, Spark o Flink, entre otras.

3.3. Descripción del proyecto

4. Planificación del proyecto

La planificación de un proyecto es fundamental para su correcto funcionamiento y desarrollo, dentro de los plazos y costes establecidos. Se presenta un primer apartado de metodología, un segundo apartado con la planificación inicial para posteriormente inferir en base a esta el presupuesto.

4.1. Metodología

En este capítulo se aborda la metodología adoptada para el desarrollo del proyecto, fundamentada en principios ágiles y enfocada en la entrega continua de valor. La elección de *Scrum*, una metodología que permite elaborar productos software de manera incremental, revisando el producto continuamente y adaptándolo a las necesidades del cliente, subraya el compromiso con la adaptabilidad y la mejora continua del producto.

La estructura de este capítulo se organiza en torno a la descripción detallada de la metodología *Scrum*, la visualización de la planificación y las estrategias de comunicación adoptadas. A través de esta metodología, se busca optimizar los recursos disponibles, ajustarse a los plazos establecidos y garantizar la calidad del producto final.

La implementación de *Scrum* se complementa con herramientas de visualización y gestión de proyectos, como los tableros *Kanban*, que facilitan la organización y seguimiento de las tareas. Además, se pone especial énfasis en la comunicación efectiva dentro del equipo de desarrollo y con los stakeholders, asegurando así una alineación constante con los objetivos del proyecto. Existen otras variantes de los tableros *Kanban* que se pueden utilizar para visualizar el progreso de las tareas, pero en este proyecto se ha elegido esta alternativa para facilitar la visualización de las tareas y su estado.

Este enfoque metodológico no solo refleja la planificación y ejecución del proyecto, sino que también establece las bases para una gestión eficaz, adaptativa y orientada a resultados.

4.1.1. Scrum

Para la planificación del proyecto se ha escogido *Scrum*, una metodología "ágil" que se basa en la realización de iteraciones cortas y en la adaptación a los cambios. La metodología *Scrum* se estructura en *sprints* (iteraciones cortas de una duración fija), en las que se llevan a cabo una serie de tareas que se han planificado previamente.

El primer paso de la metodología *Scrum* es la creación de un *product backlog*, una lista ordenada de las tareas a realizar durante el desarrollo del producto, a partir de los requisitos del sistema, que a su vez son una versión refinada de los requisitos iniciales del proyecto. A partir de este *product backlog* se planifican las tareas que se llevarán a cabo en cada *sprint*, de manera que sea posible cumplir con los objetivos del proyecto en el tiempo establecido.

A diferencia de metodologías tradicionales o *en cascada*, *Scrum* permite la adaptación a los cambios y la mejora continua del producto, ya que se revisa y se adapta en cada *sprint* según las necesidades del cliente y del equipo de desarrollo. Por otro lado, *Scrum* se diferencia de otras metodologías ágiles como *XP* en que no se centra tanto en las prácticas de desarrollo, sino en la gestión del proyecto y en la entrega de valor al cliente.

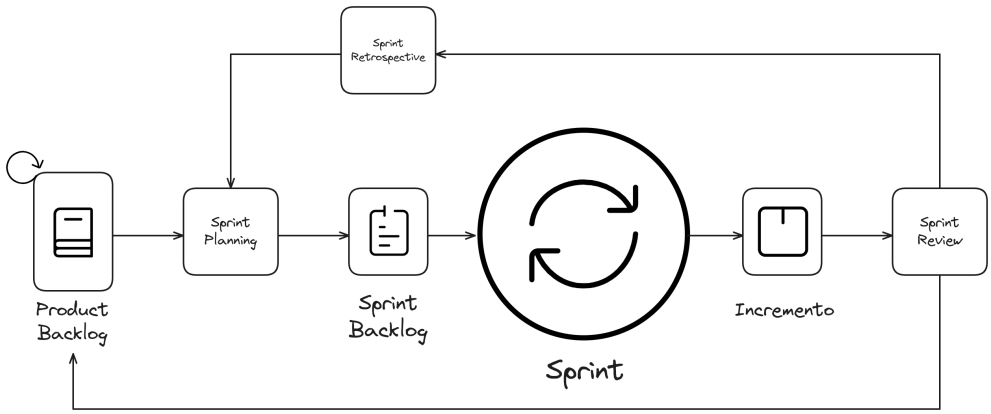

Figura 4.1: Diagrama de la metodología *Scrum*

Roles En *Scrum* se distinguen tres roles principales:

- **Product Owner**: es la persona responsable de definir los requisitos del producto y de priorizar las tareas del *product backlog*. Es el enlace entre el equipo de desarrollo y el cliente, y es el responsable de garantizar que el producto cumple con las expectativas del cliente. En el caso de este proyecto, el *Product Owner* es el director tecnológico de la empresa.
- **Scrum Master**: es la persona responsable de garantizar que el equipo de desarrollo

sigue la metodología *Scrum* y de eliminar los obstáculos que puedan surgir durante el desarrollo del proyecto. El *Scrum Master* es el encargado de organizar las reuniones diarias y de asegurar que el equipo de desarrollo cumple con los plazos y los objetivos del proyecto. En este proyecto, el *Scrum Master* son los tutores académicos del proyecto.

- **Equipo de desarrollo**: es el equipo encargado de llevar a cabo las tareas del *product backlog* y de entregar el producto final. El equipo de desarrollo es autoorganizado y multidisciplinario, y se organiza en torno a las tareas que se van a realizar en cada *sprint*. Para este proyecto, el "equipo" de desarrollo está constituido únicamente por el alumno, que se encarga de todas las tareas de desarrollo y documentación.
- **Stakeholders**: son las partes interesadas en el proyecto, como los clientes, los usuarios finales y los patrocinadores, que desconocen el proceso de desarrollo pero tienen un interés en el producto final y en su correcto funcionamiento.

Estimación En la metodología *Scrum* se pueden utilizar diferentes técnicas de estimación de tareas, como la estimación en puntos de historia, la estimación en horas o la estimación en tallas de camiseta. En este proyecto se ha optado por la estimación en tallas de camiseta, que consiste en asignar a cada tarea una talla que representa su complejidad y su duración. Las tallas de camiseta se suelen representar con letras (XS, S, M, L, XL), que se pueden traducir a puntos de historia siguiendo la secuencia de Fibonacci, es decir, $XS = 1$, $S = 2$, $M = 3$, $L = 5$, $XL = 8$.

La estimación en Scrum es esencial para la planificación de los *sprints* y para la asignación de tareas al equipo de desarrollo. La estimación en tallas se considera óptima para este proyecto, ya que permite una estimación rápida y sencilla de las tareas, al no necesitar una coordinación entre un equipo completo de desarrollo.

Además de la estimación del tamaño de las tareas, también se realiza una estimación sobre la *prioridad* de las mismas, que se representa con siguiendo el equivalente de *GitHub* al sistema de colores de semáforo, donde el rojo (P0) es la máxima prioridad y el verde (P2) la mínima.

4.1.2. Visualización

Para la visualización de la planificación se ha utilizado la herramienta de gestión de proyecto de *GitHub*, que permite múltiples visualizaciones de tareas e *issues* en tableros separados.

- Se utiliza un tablero de *requisitos* al estilo *Kanban* para visualizar los requisitos del proyecto y su estado, siguiendo con la metodología *Scrum*. Un tablero *Kanban* es una herramienta visual que permite gestionar el flujo de de trabajo de un proyecto por "sprints", dividiendo las tareas en columnas y moviéndolas de una columna a otra según su estado.

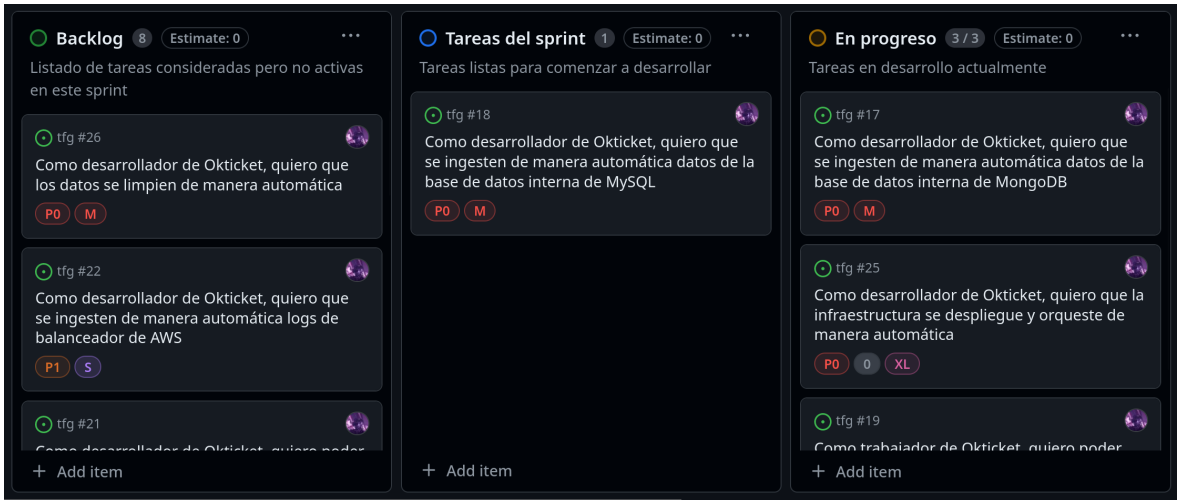

Figura 4.2: Tablero *Kanban* del proyecto

- Adicionalmente, se utiliza un *roadmap* de apartados de la memoria, separado del tablero de desarrollo normal, donde se visualiza su estado y sus fechas límite. Este *roadmap* no está relacionado con la metodología *Scrum*, sino que se ha creado para facilitar la visualización del progreso de cada sección y de la memoria en general.

Figura 4.3: Roadmap de apartados de la memoria

4.1.3. Comunicación

La comunicación con los tutores y con el equipo de desarrollo se considera fundamental para el correcto desarrollo del proyecto. Puesto que el trabajo se desarrolla de manera presencial en la oficina de la empresa, la comunicación con el equipo de desarrollo se realiza de manera frecuente y directa, mientras que la comunicación con los tutores se realiza de manera remota pero igual de frecuente, manteniendo el contacto mediante correo electrónico y Teams para pedir revisiones e informar sobre el estado del trabajo en todo momento.

4.1.4. Herramientas

Con el objetivo de facilitar las tareas de desarrollo y cumplimentar los requisitos por parte de la empresa, se utilizan las siguientes plataformas y herramientas de desarrollo para la fabricación del proyecto:

- **GitHub**: Plataforma de desarrollo colaborativo para el desarrollo del proyecto. Se utiliza para la gestión de tareas, seguimiento de desarrollo, documentación y colaboración.
- **Atlassian suite (*Jira, Bitbucket*)**: Suite de herramientas de gestión de proyectos y desarrollo colaborativo. Se utiliza para el desarrollo y documentación del proyecto de parte de la empresa.
- **Microsoft Teams**: Herramienta de comunicación y colaboración en tiempo real.
- **Microsoft Outlook**: Herramienta de comunicación por correo electrónico.

4.2. Planificación inicial

Como se ha mencionado anteriormente, se utiliza la metodología *Scrum* para la planificación y desarrollo del proyecto. En la figura 4.4 se puede ver el *backlog* de tareas que se planifican en el proyecto.

Figura 4.4: Planificación inicial del proyecto

Las tareas anteriores se clasifican y categorizan según su prioridad y tamaño, haciendo uso de la estrategia de tallas de camiseta como mencionado anteriormente. En el tablero *Kanban* (ver figura 4.2) se puede ver en todo momento el estado de las tareas, su progreso y sus características. El listado de tareas iniciales (ordenado según su prioridad) es el siguiente:

Tarea	Prioridad	Tamaño
Creación de la infraestructura base (técnica)	P0	L
Como desarrollador de Okticket, quiero que la arquitectura se despliegue y orqueste de manera automática	P0	XL
Como desarrollador de Okticket, quiero que se ingesten de manera automática datos de la base de datos interna de MongoDB	P0	M
Como desarrollador de Okticket, quiero que se ingesten de manera automática datos de la base de datos interna de MySQL	P0	M
Como desarrollador de Okticket, quiero que los datos se limpien de manera automática	P0	M
Como trabajador de Okticket, quiero poder ver y consultar datos internos de la empresa	P1	L
Como desarrollador de Okticket, quiero que se ingesten de manera automática logs de balanceador de AWS	P1	S
Como desarrollador de Okticket, quiero poder ver el estado general de la infraestructura	P1	L
Como desarrollador de Okticket, quiero que los datos contengan metadatos que faciliten su filtrado o búsqueda	P2	S
Como trabajador de Okticket, quiero poder ver y consultar datos de empresas cliente	P2	M
Como gestor de una empresa cliente, quiero poder ver información relevante sobre mi empresa que recoja Okticket	P2	L
Como desarrollador de Okticket, quiero poder ingestar datos de APIs externas a la empresa	P2	L
Como desarrollador de Okticket, quiero poder ingestar información de páginas web externas (*scraping*)	P2	XL

Tabla 4.1: Listado de tareas iniciales

Siguiendo la tabla anterior, se pueden planear los *sprints* y asignar las tareas a cada uno de ellos.

4.3. Presupuesto

Para poder llevar a cabo este proyecto, se realiza una estimación del coste total neceario para su desarrollo, que se divide en dos partes: el coste del material, que incluye el coste de los recursos necesarios para el desarrollo del proyecto, y el coste del personal, que incluye el coste de las horas de trabajo del desarrollador.

4.3.1. Presupuesto de material

Puesto que el proyecto se desarrolla en la empresa, se dispone de todos los recursos físicos necesarios para llevar a cabo el proyecto, es decir, que no se incluirá el coste del ordenador o de la conexión a internet en el presupuesto.

Sin embargo, se incluirá el coste de las herramientas y servicios utilizados durante el desarrollo del proyecto, como el coste de las licencias de software, el coste de los servicios en la nube, el coste de las herramientas de desarrollo, etc.

Es importante destacar de que los precios de los servicios en la nube son aproximados y pueden variar en función de la región, el tipo de instancia, el tipo de almacenamiento, etc. Por lo tanto, los precios presentados en este presupuesto son orientativos y pueden variar en función de las necesidades del proyecto. En este caso, se analizan los precios en junio de 2024 en la región de Amazon Web Services (AWS) de *eu-west-3* (París).

Categoría	Ítem	Cantidad	Coste unitario	Coste total
AWS Compute	Fargate	500 vCPU-h/mes	0,04665€/vCPU-h	23,33€
		1000 GB-h/mes	0,00511€/GB-h	5,11€
AWS Storage	EFS	500 GB/mes	0,36€/GB-mes	180,00€
	S3	250 GB/mes	0,0245€/GB-mes	6,13€
AWS Network	VPC	4 NAT Gateways	0,052€/h	149,76€
	ELB	4 ALBs	0,0243€/h	70,00€
AWS Security	IAM	-	Sin cargo	0,00€
	KMS	1 CMK	1,00€/mes	1,00€
Stack KELK	Kafka	-	Licencia gratuita	0,00€
	Elasticsearch	-	Licencia gratuita	0,00€
	Logstash	-	Licencia gratuita	0,00€
	Kibana	-	Licencia gratuita	0,00€
AWS Container	ECS	-	Sin cargo	0,00€
AWS Monitor	CloudWatch	5 métricas	0,30€/métrica-mes	1,50€
	X-Ray	50,000 trazas/mes	4,60€/1M trazas	0,23€
Despliegue	Terraform	-	Licencia gratuita	0,00€
Capacitación	Cursillo online	1	184,00€	184,00€
Subtotal				620,06€
Otros	Support	Plan Basic	Sin cargo	0,00€
	Optimización	-	5 % del subtotal	31,00€
	Contingencia	-	10 % del subtotal	62,00€
Total				713,06€

Tabla 4.2: Propuesta de presupuesto de materiales

4.3.2. Presupuesto de personal

A continuación, se presenta una propuesta de presupuesto de personal para el desarrollo del proyecto, que incluye el coste de las horas de trabajo según cada rol y el coste total del personal.

Rol	**Descripción**	**Horas/mes**	**CU (€/h)**	**CT (€/mes)**
Arquitecto	Diseño de la arquitectura y supervisión	40	60	2.400,00€
Desarrollador	Desarrollo y mantenimiento	160	45	7.200,00€
Administrador	Gestión de sistemas y seguridad	160	50	8.000,00€
DevOps	Infraestructuras y monitorización	80	55	4.400,00€
Subtotal				22.000,00€
Otros	IVA (21 %)			4.620,00€
	Margen (5 %)			1.100,00€
Total				27.720,00€

Tabla 4.3: Propuesta de presupuesto de personal

4.3.3. Presupuesto total

Finalmente, se presenta el presupuesto total del proyecto, que incluye el coste del material y el coste del personal, así como el coste total del proyecto.

Concepto	**Coste (€/mes)**
Presupuesto de materiales	713,06€
Presupuesto de personal	27.720,00€
Subtotal	**28.433,06€**
Beneficio Industrial (15 %)	4.264,96€
Total	**32.698,02€**

Tabla 4.4: Costes combinados de presupuesto y materiales con beneficio industrial

El presupuesto total del proyecto asciende a 32.698,02€ (treinta y dos mil seiscientos noventa y ocho euros con dos céntimos), que incluye el coste del material, el coste del personal y el beneficio industrial, además de sus márgenes.

5. Diseño del sistema

5.1. Estudio de alternativas

En este apartado se explorarán las diferentes alternativas disponibles para el diseño del sistema. Se analizarán las características, ventajas y desventajas de cada opción, con el objetivo de proporcionar una visión clara y fundamentada que permita seleccionar la alternativa más adecuada para el proyecto. Las áreas de estudio incluirán tanto el despliegue de infraestructura como otros aspectos críticos del diseño del sistema, asegurando una evaluación integral y detallada de las posibles soluciones.

5.1.1. Despliegue de infraestructura

A la hora de desplegar la infraestructura de un proyecto, se consideran varias herramientas populares que permiten automatizar este proceso. Entre todas ellas, las más establecidas y atractivas son *Terraform*, *AWS CloudFormation* y *Ansible*.

A continuación, se describen brevemente estas herramientas y se comparan sus características.

5.1.1.1. Alternativas

Terraform *Terraform* es una herramienta de código abierto desarrollada por *HashiCorp* que permite definir y desplegar infraestructura de forma declarativa. *Terraform* permite definir la infraestructura en un archivo de configuración JSON, que describe los recursos que se desean crear y sus dependencias. A partir de este archivo, *Terraform* se encarga de desplegar los recursos en el proveedor de nube especificado, que en el caso de este proyecto es *AWS*.

Figura 5.1: Logo de Terraform ®

AWS CloudFormation *AWS CloudFormation* es un servicio de *Amazon Web Services* similar a *Terraform* que permite definir y desplegar infraestructura en la nube de forma declarativa. *AWS CloudFormation* permite definir la infraestructura o bien mediante un archivo de configuración (en formato JSON o YAML), o bien gráficamente mediante diagramas, un punto muy fuerte a favor de esta alternativa.

Figura 5.2: Logo de AWS CloudFormation ®

Ansible *Ansible* es una herramienta multi-propósito de automatización de tareas entre las que se incluye el despliegue y orquestación de infraestructura. Se trata de una herramienta desarrollada por *Red Hat* que permite definir la infraestructura mediante *playbooks* escritos en YAML, que describen las tareas a realizar y los servidores en los que se deben ejecutar.

Figura 5.3: Logo de Ansible ®

5.1.1.2. Comparación

Comenzando la comparativa por la facilidad de uso de cada herramienta, *Terraform* es la alternativa planteada más fácil de usar, ya que permite definir la infraestructura en un archivo con sintaxis sencilla y desplegarla con un solo comando. Por otro lado, *AWS CloudFormation* es un poco más complejo de usar, ya que requiere definir la infraestructura en un archivo de configuración o en un diagrama, y luego desplegarla mediante la consola de *AWS*. *Ansible* es más complejo de usar, ya que requiere definir la infraestructura mediante *playbooks* y ejecutarlos en los servidores, pero es más flexible y potente que las otras dos herramientas.

Mientras que *Terraform* y *Ansible* son herramientas *multi-cloud*, lo que significa que funcionan con cualquier proveedor de nube, *AWS CloudFormation* es una herramienta específica de *AWS* y solo funciona con sus servicios, lo que puede suponer tanto una ventaja como una desventaja, dependiendo de las necesidades del proyecto. En este caso, la solución se va a desplegar en la nube de *Amazon*, pero a la vez no se requiere el uso de servicios específicos de *AWS*, no se considera una ventaja significativa.

5.1.1.3. Decisión

Ninguna de las alternativas consideradas es claramente superior a las demás, ya que se tratan de herramientas con características y funcionalidades similares y una gran popularidad en la industria. Sin embargo, se decide utilizar *Terraform* para el despliegue de la infraestructura de este proyecto, ya que es la herramienta más “sencilla”, la que mejor se podría

adaptar a las necesidades del proyecto y la única que ya se ha usado en proyectos anteriores dentro de la empresa.

5.1.2. Ingesta de datos

A partir del conjunto de tecnologías seleccionadas en la descripción detallada del proyecto, se consdieran diversas tecnologías, como *Redpanda*, *AWS Glue* y *Kafka* que permitan ingestar datos de todas las fuentes que requieren ser procesadas.

5.1.2.1. Alternativas

Kafka *Kafka* es una plataforma de transmisión de datos distribuida y de código abierto que se utiliza para construir pipelines de datos en tiempo real y aplicaciones de streaming. Desarrollada originalmente por LinkedIn y posteriormente donada a la Apache Software Foundation, *Kafka* se ha convertido en una de las tecnologías más populares para la gestión de flujos de datos en tiempo real.

Figura 5.4: Logo de Kafka ®

Una de las principales ventajas de *Kafka* es su capacidad para manejar grandes volúmenes de datos con alta eficiencia y baja latencia. *Kafka* utiliza un modelo de publicación-suscripción, donde los productores publican mensajes en temas y los consumidores se suscriben a estos temas para recibir los mensajes. Esta arquitectura permite una alta escalabilidad y flexibilidad en la gestión de datos.

Kafka se compone de varios componentes clave:

- **Tópico**: Un tópico es una categoría a la que se envían los mensajes a la que los consumidores están *suscritos*. Los consumidores pueden estar suscritos a uno o varios tópicos, y los productores pueden enviar mensajes a uno o varios tópicos. Los tópicos son la unidad básica de organización de los mensajes en cualquier sistema de mensajería de publicación/suscripción.
- **Productor**: El productor es el componente responsable de crear y enviar mensajes al cluster de Kafka. Está separado del resto de los componentes y produce mensajes de manera asíncrona y rápida.

- **Consumidor**: El consumidor es el componente responsable de leer los mensajes producidos por el productor. Está suscrito a un tópico a través del broker y consume los mensajes.
- **Broker**: El broker es el componente responsable de recibir los mensajes producidos por el productor y enviarlos a los consumidores. Es el intermediario entre los productores y los consumidores.
- **Zookeeper**: Zookeeper es un servicio separado de coordinación distribuida que se utiliza para gestionar y coordinar los brokers de Kafka. Se encarga de mantener la información de los brokers y de los tópicos. Actualmente, este servicio es una dependencia obligatoria de Kafka. [1]

A pesar de sus numerosas ventajas, *Kafka* también presenta algunos desafíos. La configuración y gestión de un clúster de *Kafka* puede ser compleja, especialmente en entornos de producción a gran escala. Además, *Kafka* depende de *Zookeeper* para la coordinación, lo que añade una capa adicional de complejidad en la administración del sistema.

En resumen, *Kafka* es una solución robusta y escalable para la transmisión de datos en tiempo real, ideal para aplicaciones que requieren alta disponibilidad y procesamiento eficiente de grandes volúmenes de datos. Sin embargo, su implementación y gestión requieren un conocimiento profundo de su arquitectura y componentes.

Redpanda *Redpanda* es una plataforma de transmisión de datos en tiempo real que se destaca por su alto rendimiento y baja latencia. Diseñada como una alternativa moderna a *Kafka*, *Redpanda* ofrece una arquitectura simplificada que elimina la necesidad de dependencias externas como *Zookeeper*. Esto no solo reduce la complejidad operativa, sino que también mejora la eficiencia y la escalabilidad del sistema. *Redpanda* es compatible con la API de *Kafka*, lo que facilita la migración de aplicaciones existentes sin necesidad de cambios significativos en el código. Además, su diseño optimizado para hardware moderno permite un procesamiento más rápido y un uso más eficiente de los recursos, lo que la convierte en una opción ideal para aplicaciones que requieren una transmisión de datos rápida y confiable.

Sin embargo, *Redpanda* también presenta algunos puntos en contra. Al ser una tecnología relativamente nueva, su ecosistema y comunidad de usuarios no son tan amplios como los de *Kafka*, lo que puede limitar el acceso a recursos y soporte. Además, aunque la compatibilidad con la API de *Kafka* es una ventaja, puede haber ciertas características y extensiones

[1] Dejará de ser necesario en la versión 4. `https://x.com/coltmcnealy/status/1801987159534264641`

específicas de *Kafka* que no estén completamente soportadas en *Redpanda*. Finalmente, la adopción de una nueva tecnología siempre conlleva riesgos asociados con la estabilidad y el soporte a largo plazo, aspectos que deben ser considerados cuidadosamente antes de su implementación.

AWS Glue *AWS Glue* es un servicio de integración de datos totalmente administrado que facilita la preparación y carga de datos para análisis. Diseñado para trabajar con grandes volúmenes de datos, *AWS Glue* automatiza las tareas de descubrimiento, catalogación, limpieza, enriquecimiento y movimiento de datos entre diferentes almacenes de datos. Una de las principales ventajas de *AWS Glue* es su capacidad para generar automáticamente el código necesario para realizar las transformaciones de datos, lo que reduce significativamente el tiempo y el esfuerzo requeridos para preparar los datos para el análisis. Además, *AWS Glue* es altamente escalable y puede manejar tanto cargas de trabajo por lotes como en tiempo real, lo que lo convierte en una opción versátil para diversas necesidades de integración de datos.

AWS Glue es un servicio administrado, por lo que su uso puede implicar costes adicionales en comparación con soluciones autogestionadas. Además, aunque *AWS Glue* ofrece una gran flexibilidad y potencia, su configuración y optimización pueden requerir un conocimiento profundo de los servicios de *AWS* y de las mejores prácticas de integración de datos. Por último, la dependencia de *AWS Glue* puede limitar la portabilidad de las soluciones de integración de datos a otros proveedores de nube, lo que podría ser un inconveniente en caso de querer migrar el proyecto a otro proveedor.

5.1.2.2. Comparación y decisión

Desde el primer momento, en la empresa se considera Kafka como la opción más sólida junto con el *stack ELK* para desarrollar el proyecto, al tratarse de un estándar en la industria y una solución tanto rápida y escalable como asequible a nivel económico. Por eso, y pese a que las otras alternativas son atractivas para el desarrollo de este proyecto, se decide utilizar Kafka como servicio de ingesta de datos, en consonancia con *Logstash*.

5.2. Arquitectura del sistema

Tras toda la defínción de los requisitos y la valoración de las alternativas disponibles, se plantea la siguiente arquitectura:

5.3. Modelo de datos

6. Implementación

Durante la implementación, se ha seguido la planificación y metodologías anteriormente descritas, dividiendo el proyecto en tareas más pequeñas y manejables, para que se puedan realizar en un periodo de tiempo razonable.

Al seguir la prioridad de las tareas, se realizan primero las tareas más críticas, como la creación de la infraestructura y la ingesta de las fuentes esenciales, y se dejan para más adelante tareas como la visualización para clientes externos o fuentes menos críticas y más complejas, como las APIs de terceros o el *web scraping*.

7. Manual de usuario

8. Resultados

9. Conclusiones y trabajo futuro

El propósito de este capítulo es presentar las conclusiones obtenidas a partir del desarrollo del proyecto, recopilar las dificultades encontradas y proponer líneas de trabajo futuro en vista a la amplicación y mejora del sistema.

Bibliografía

[1] Wikipedia contributors, "Dikw pyramid — Wikipedia, the free encyclopedia." https://en.wikipedia.org/w/index.php?title=DIKW_pyramid&oldid=1211227190, 2024. [Online; accessed 7-April-2024].

[2] Ishwarappa and J. Anuradha, "A brief introduction on big data 5vs characteristics and hadoop technology," *Procedia Computer Science*, vol. 48, pp. 319–324, 2015. International Conference on Computer, Communication and Convergence (ICCC 2015).

[3] S. Sagiroglu and D. Sinanc, "Big data: A review," in *2013 International Conference on Collaboration Technologies and Systems (CTS)*, pp. 42–47, 2013.

[4] J. Mier, "Presentación de datos: dashboards y procesos ETL." Primera entrega de teoría de la asignatura Inteligencia de Negocio, EPI Gijón, curso 23-24, 2023.

[5] P. Khine and Z. Wang, "Data lake: a new ideology in big data era," *ITM Web of Conferences*, vol. 17, p. 03025, 01 2018.

[6] J. Mier, "latexTemplate." https://github.com/miermontoto/latexTemplate, 2024. Plantilla de LaTeX personal para trabajos académicos.

[7] J. Mier, "Minería de anomalías." Segunda entrega de teoría de la asignatura Inteligencia de Negocio, EPI Gijón, curso 23-24, 2024.