ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
(ICAI)

Máster Universitario en Big Data

INGESTA Y PROCESADO DE INPUTS DE INFORMACIÓN PARA LA CREACIÓN DE UN DATA LAKE / WAREHOUSE PARA LA OPTIMIZACIÓN DE RENTABILIDAD EN UNA EMPRESA DE CONSULTORÍA

Autor
Ramón Sánchez Quijada

Dirigido por
Lucía Schmidt Santiago

Coordinado por
Carlos Morrás Ruiz-Falcó

Madrid
Mayo 2025

AUTORIZACIÓN PARA LA DIGITALIZACIÓN, DEPÓSITO Y DIVULGACIÓN EN RED DE PROYECTOS FIN DE GRADO, FIN DE MÁSTER, TESINAS O MEMORIAS DE BACHILLERATO

1º. Declaración de la autoría y acreditación de la misma.

El autor D. Ramón Sánchez Quijada **DECLARA** ser el titular de los derechos de propiedad intelectual de la obra: INGESTA Y PROCESADO DE INPUTS DE INFORMACIÓN PARA LA CREACIÓN DE UN DATA LAKE / WAREHOUSE PARA LA OPTIMIZACIÓN DE RENTABILIDAD EN UNA EMPRESA DE CONSULTORÍA, que ésta es una obra original, y que ostenta la condición de autor en el sentido que otorga la Ley de Propiedad Intelectual.

2º. Objeto y fines de la cesión.

Con el fin de dar la máxima difusión a la obra citada a través del Repositorio institucional de la Universidad, el autor **CEDE** a la Universidad Pontificia Comillas, los derechos de digitalización, de archivo, de reproducción, de distribución y de forma gratuita y no exclusiva, por el máximo plazo legal y con ámbito universal, de comunicación pública, incluido el derecho de puesta a disposición electrónica, tal y como se describen en la Ley de Propiedad Intelectual. El derecho de transformación se cede a los únicos efectos de lo dispuesto en la letra a) del apartado siguiente.

3º. Condiciones de la cesión y acceso

Sin perjuicio de la titularidad de la obra, que sigue correspondiendo a su autor, la cesión de derechos contemplada en esta licencia habilita para:

(a) Transformarla con el fin de adaptarla a cualquier tecnología que permita incorporarla a internet y hacerla accesible; incorporar metadatos para realizar el registro de la obra e incorporar "marcas de agua" o cualquier otro sistema de seguridad o de protección.

(b) Reproducirla en un soporte digital para su incorporación a una base de datos electrónica, incluyendo el derecho de reproducir y almacenar la obra en servidores, a los efectos de garantizar su seguridad, conservación y preservar el formato.

(c) Comunicarla, por defecto, a través de un archivo institucional abierto, accesible de modo libre y gratuito a través de internet.

(d) Cualquier otra forma de acceso (restringido, embargado, cerrado) deberá solicitarse expresamente y obedecer a causas justificadas.

(e) Asignar por defecto a estos trabajos una licencia Creative Commons.

(f) Asignar por defecto a estos trabajos un HANDLE (URL *persistente*).

4º. Derechos del autor.

El autor, en tanto que titular de una obra tiene derecho a:

(a) Que la Universidad identifique claramente su nombre como autor de la misma

(b) Comunicar y dar publicidad a la obra en la versión que ceda y en otras posteriores a través de cualquier medio.

(c) Solicitar la retirada de la obra del repositorio por causa justificada.

(d) Recibir notificación fehaciente de cualquier reclamación que puedan formular terceras personas en relación con la obra y, en particular, de reclamaciones relativas a los derechos de propiedad intelectual sobre ella.

5º. Deberes del autor.

(a) El autor se compromete a:

(b) Garantizar que el compromiso que adquiere mediante el presente escrito no infringe ningún derecho de terceros, ya sean de propiedad industrial, intelectual o cualquier otro.

(c) Garantizar que el contenido de las obras no atenta contra los derechos al honor, a la intimidad y a la imagen de terceros.

(d) Asumir toda reclamación o responsabilidad, incluyendo las indemnizaciones por daños, que pudieran ejercitarse contra la Universidad por terceros que vieran infringidos sus derechos e intereses a causa de la cesión.

(e) Asumir la responsabilidad en el caso de que las instituciones fueran condenadas por infracción de derechos derivada de las obras objeto de la cesión.

6º. Fines y funcionamiento del Repositorio Institucional.

La obra se pondrá a disposición de los usuarios para que hagan de ella un uso justo y respetuoso con los derechos del autor, según lo permitido por la legislación aplicable, y con fines de estudio, investigación, o cualquier otro fin lícito. Con dicha finalidad, la Universidad asume los siguientes deberes y se reserva las siguientes facultades:

- La Universidad informará a los usuarios del archivo sobre los usos permitidos, y no garantiza ni asume responsabilidad alguna por otras formas en que los usuarios hagan un uso posterior de las obras no conforme con la legislación vigente. El uso posterior, más allá de la copia privada, requerirá que se cite la fuente y se reconozca la autoría, que no se obtenga beneficio comercial, y que no se realicen obras derivadas.

- La Universidad no revisará el contenido de las obras, que en todo caso permanecerá bajo la responsabilidad exclusive del autor y no estará obligada a ejercitar acciones legales en nombre del autor en el supuesto de infracciones a derechos de propiedad intelectual derivados del depósito y archivo de las obras. El autor renuncia a cualquier reclamación frente a la Universidad por las formas no ajustadas a la legislación vigente en que los usuarios hagan uso de las obras.

- La Universidad adoptará las medidas necesarias para la preservación de la obra en un futuro.

- La Universidad se reserva la facultad de retirar la obra, previa notificación al autor, en supuestos suficientemente justificados, o en caso de reclamaciones de terceros.

Madrid, a 15 de mayo de 2025

ACEPTA

Fdo.: Ramón Sánchez Quijada

Motivos para solicitar el acceso restringido, cerrado o embargado del trabajo en el Repositorio Institucional:

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
(ICAI)

Máster Universitario en Big Data

INGESTA Y PROCESADO DE INPUTS DE INFORMACIÓN PARA LA CREACIÓN DE UN DATA LAKE / WAREHOUSE PARA LA OPTIMIZACIÓN DE RENTABILIDAD EN UNA EMPRESA DE CONSULTORÍA

Autor
Ramón Sánchez Quijada

Dirigido por
Lucía Schmidt Santiago

Coordinado por
Carlos Morrás Ruiz-Falcó

Madrid
Mayo 2025

Resumen

Un data lake, almacén de datos, para una empresa de consultoría es una herramienta que se utiliza para la ingestión y procesamiento de inputs de información con el objetivo de crear un entorno centralizado y accesible que potencie la optimización de rentabilidad. En esta tesis, se ha investigado y desarrollado un enfoque integral para la implementación de un data lake/warehouse en una empresa de consultoría, con el propósito de mejorar la gestión de datos y facilitar la toma de decisiones estratégicas. El sistema se basa en una arquitectura de microservicios, que ofrece así ventajas de escalabilidad, modularidad y mantenibilidad. A través del análisis de diferentes fuentes de datos y la aplicación de técnicas de extracción, transformación y carga (ETL), se ha logrado consolidar datos en bruto de diversas áreas y departamentos, proporcionando una base sólida para realizar análisis detallados y generar informes personalizados. Además, se han explorado casos de uso específicos, como el seguimiento de la rentabilidad de proyectos y la visualización de datos desglosados, y una tabla centralizada con las cargas por usuario y centro de coste, con el fin de demostrar el valor y las capacidades del data lake en el contexto de una empresa de consultoría. Los resultados obtenidos han demostrado que la implementación de un data lake/warehouse puede mejorar significativamente la eficiencia operativa, la toma de decisiones informadas y la optimización de la rentabilidad en una empresa de consultoría.

Palabras clave: data lake, extracción, transformación y carga (ETL), integración de fuentes de datos, centralización, escalabilidad, análisis de datos, visualización, optimización, arquitectura de datos.

Abstract

A data lake, a data warehouse, for a consulting company is a tool used for the ingestion and processing of information inputs with the aim of creating a centralized and accessible environment that enhances profitability optimization. In this thesis, a comprehensive approach for implementing a data lake/warehouse in a consulting company has been researched and developed, aiming to improve data management and facilitate strategic decision-making. The system is based on a microservices architecture, offering advantages such as scalability, modularity, and maintainability. Through the analysis of different data sources and the application of extraction, transformation, and loading (ETL) techniques, raw data from various areas and departments have been consolidated, providing a solid foundation for detailed analysis and generating customized reports. Additionally, specific use cases have been explored, such as project profitability tracking and the visualization of disaggregated data, or a centralized table with user loads and cost centers, in order to demonstrate the value and capabilities of the data lake in the context of a consulting company. The obtained results have shown that the implementation of a data lake/warehouse can significantly enhance operational efficiency, informed decision-making, and profitability optimization in a consulting company.

Keywords: data lake, extraction, transformation and loading (ETL), data source integration, centralization, scalability, data analysis, visualization, optimization, data architecture.

A aquellos
que me quieren en el éxito y en el fracaso.

Success is not the key to happiness.
Happiness is the key to success.
If you love what you are doing,
you will be successful.
ALBERT SCHWEITZER

Agradecimientos

Quisiera aprovechar este espacio para expresar mi más sincero agradecimiento a todas las personas que han estado a mi lado y han brindado su apoyo incondicional a lo largo de este camino. En primer lugar, quiero agradecer a mi familia, especialmente a mis padres y abuelos, por su amor, comprensión y por siempre esperar lo mejor de mí. Su constante aliento y sacrificio han sido fundamentales en mi desarrollo personal y académico.

También quiero expresar mi gratitud hacia mis amigos de toda la vida. Ellos han sido mi refugio en momentos difíciles, siempre encontrando la manera de hacerme evadir de mis problemas. Su amistad y apoyo han sido un verdadero regalo que valoro enormemente.

No puedo dejar de mencionar a mis amigos del máster, quienes han compartido conmigo esta etapa de aprendizaje y crecimiento. Sin su compañerismo, motivación y colaboración, este año no hubiese sido lo mismo. Juntos hemos enfrentado desafíos, celebrado logros y creado recuerdos que atesoraré siempre.

Agradezco también a mis compañeros de piso y amigos de la carrera, por su comprensión durante los momentos de mayor presión. A lo largo de estos años, han estado ahí para brindarme su ayuda y apoyo en momentos académicos y personales. Su presencia constante y disposición para escucharme han sido de gran valor.

A todas estas personas, gracias por estar presentes en mi vida, por su confianza, amistad y por siempre estar ahí cuando los necesito. Su influencia positiva ha dejado huellas imborrables en mi camino y estaré siempre agradecido.

Por último, agradecer a Cloud District por brindarme la valiosa oportunidad de realizar mi Trabajo de Fin de Máster con ellos. Gracias por proporcionarme un entorno en el que pude aprender y crecer profesionalmente. Durante estos meses de colaboración, el equipo de Cloud District siempre estuvo dispuesto a ofrecer su apoyo y conocimientos. Agradezco profundamente a todos los miembros del equipo de Cloud District por su paciencia, orientación y por dejarme trabajar en proyectos desafiantes y significativos.

Índice general

Índice de figuras

Acrónimos

ICAI	Insitituto Católico de Artes e Industrias
PFC	Proyecto Fin de Carrera
TFM	Trabajo de Fin de Master
AWS	Amazon Web Services
SqL	Squad Leader
SQL	Lenguaje de Consulta Estructurado
UI	Interfaz de Usuario
API	Interfaz de Programación de Aplicaciones
CRM	Gestión de Relaciones con el Cliente
ERP	Planificación de Recursos Empresariales
S3	Simple Storage Service
BI	Business Intelligence
JSON	JavaScript Object Notation
CSV	Valores Separados por Comas
SSM	Systems Manager Parameter Store
SGBD	Sistema de Gestión de Bases de Datos
RDS	Servicio de Base de Datos Relacional
YAML	YAML Ain't Markup Language
SSH	Secure Shell
IDE	Entorno de Desarrollo Integrado
VSCode	Visual Studio Code
HTTP	Protocolo de Transferencia de Hipertexto
DM	Delivery Manager
N/A	No Aplica
EC2	Elastic Compute Cloud
RAM	Memoria de Acceso Aleatorio
CPU	Unidad Central de Procesamiento
IoT	Internet de las Cosas
XML	Lenguaje de Marcado Extensible

Capítulo 1

Introducción

El volumen de datos generados y puestos a disposición para el análisis en el mundo empresarial actual ha crecido exponencialmente. Para poder ofrecer a sus clientes servicios de alta calidad, las consultoras deben gestionar y extraer valor de esta ingente cantidad de información. En este sentido, se sugiere el desarrollo de un «Data Lake» como una posible solución.

Primero, un data lake (lago de datos) ofrece una ubicación central para almacenar datos recopilados de varias fuentes. Esta centralización facilita a los consultores el acceso y la búsqueda de información, evitando la dispersión de datos entre varios sistemas y aumentando la efectividad de la obtención de datos pertinentes. Además, un lago de datos proporciona flexibilidad y escalabilidad, lo que le permite administrar grandes volúmenes de datos estructurados y no estructurados. Esta flexibilidad es fundamental en un entorno empresarial en constante cambio y donde el volumen de datos generados es cada vez mayor.

Otra característica notable es la capacidad de realizar un análisis en profundidad y obtener información importante de los datos guardados en el lago de datos. Como resultado, se mejora la toma de decisiones y los clientes reciben más valor de los proyectos y recomendaciones de los consultores.

De manera similar, la implementación de un data lake simplifica la integración de datos de varias fuentes. Las empresas de consultoría utilizan con frecuencia una variedad de fuentes de datos, incluidas bases de datos internas, datos de clientes y datos externos. La integración de estos datos se facilita con el data lake, que también facilita el procesamiento y el análisis [1].

El desarrollo de un data lake también promueve la productividad y el trabajo en equipo dentro de la empresa consultora. Los consultores pueden acceder a los datos necesarios para los proyectos de manera rápida y efectiva, y se facilita el

intercambio de datos entre equipos de trabajo. Esto fomenta el trabajo en equipo y el intercambio de conocimientos, lo que eleva el nivel de los servicios prestados.

Esta tesis se desarrolló como un proyecto interno en Cloud District. Sin embargo, otras empresas de la industria podrían utilizar las estrategias y los procedimientos investigados en este proyecto. Si bien los requisitos y desafíos únicos de Cloud District son el enfoque principal de este estudio, otras organizaciones que buscan maximizar su rentabilidad mediante la implementación de un data lake/warehouse pueden adaptar y utilizar estos conocimientos.

Este data lake ha sido implementado exitosamente en la empresa Cloud District, una empresa que tiene 12 años de experiencia, como parte de su estrategia de transformación digital. Se desarrolló con el objetivo de centralizar y administrar de manera efectiva cantidades masivas de datos de diversas fuentes internas y externas de la empresa, que con el tiempo llegan en un volumen cada vez mayor. La empresa puede obtener una visión completa y unificada de sus datos consolidando esta información en un único repositorio, lo que facilita la toma de decisiones basadas en datos y fomenta la creación de soluciones novedosas.

Los aspectos técnicos, metodológicos y prácticos del establecimiento de un almacén de datos se cubrirán en las secciones siguientes. Se analizarán los problemas relacionados con la adquisición y el procesamiento de las entradas de datos, así como las tácticas para garantizar la disponibilidad y la calidad de los datos. Adicionalmente, teniendo en cuenta las características y requerimientos únicos de otros negocios de la industria, se desarrollarán recomendaciones prácticas para la correcta implementación de esta solución tecnológica.

1.1. Motivación

En un entorno empresarial altamente competitivo y en constante evolución tecnológica, donde la información se actualiza a diario, existe una necesidad apremiante de crear un data lake/warehouse para la optimización de la rentabilidad en una empresa de consultoría. El éxito de estas organizaciones ahora depende en gran medida de su capacidad para administrar su información de manera efectiva. Con el fin de producir información útil para respaldar la toma de decisiones estratégicas, este proyecto de tesis analizará cómo la implementación de un lago o almacén de datos puede mejorar la calidad y la accesibilidad de los datos utilizados en los procesos de consultoría. También pretende ofrecer a las empresas consultoras un manual útil para la aplicación eficiente de esta solución tecnológica, potenciando su éxito financiero y comerciabilidad.

1.2. Objetivos

1.2.1. Objetivo general

El objetivo del presente proyecto es la habilitación de un nuevo entorno de gestión de los datos producidos en Cloud District y su representación gráfica en forma de cuadros de mandos. Convertir los datos en bruto en elementos de juicio que permitan la toma informada de decisiones.

Esta representación contará asimismo con la funcionalidad de analítica de datos que permitan la extracción de la información relevante que se presentará al usuario.

1.2.2. Objetivos específicos

Se ha propuesto seguir una serie de objetivos más específicos y así concretar ciertas acciones que ayudarán a guiar el trabajo realizado buscando siempre cumplir con el objetivo principal. De esta manera, también se facilitará la evaluación del avance del proyecto, ya que se medirá la efectividad de cada una de las partes. Son los siguientes:

Para recopilar datos de varias fuentes de datos, primero se desarrollará una arquitectura de microservicios en AWS. Estos datos se utilizarán para crear los conjuntos de datos específicos requeridos para dar al usuario la información.

El proceso de carga y transformación de datos se llevará a cabo automáticamente sin intervención del usuario. Para asegurar que los procesos se completen correctamente y brindar la información pertinente en caso de errores, también se implementarán sistemas de registro y trazabilidad. Esto hará que sea más fácil tomar las acciones correctivas apropiadas.

Asegurarse de que se pueda acceder a los datos casi en tiempo real mientras se completan con éxito las tareas de carga y transformación es otro objetivo crucial del proyecto. Esto brindará a los usuarios acceso constante a datos actualizados y confiables.

Adicionalmente, se buscará estandarizar los inputs (entradas); es decir, se desarrollarán estándares y formatos comunes para los datos recolectados, facilitando su integración y aplicación. También se mejorará la calidad de los datos, con la implementación de procedimientos y medidas para garantizar su precisión, consistencia e integridad.

Finalmente, se concentrará en consultas e informes utilizando el lenguaje SQL y herramientas como Amazon QuickSight, lo que permitirá a los usuarios realizar

análisis avanzados y obtener información valiosa para la parte más administrativa. Además, se utilizarán herramientas no code/low code como UI Bakery para visualizar los datos, lo que facilitará la creación de representaciones visuales interactivas y fáciles de usar para cualquier cliente final.

1.3. Estructura del documento

Esta sección muestra las diferentes partes de este proyecto, para facilitar así su lectura:

- **Introducción:** Esta sección proporciona una breve descripción general del proyecto, establece el contexto y determina los objetivos generales y específicos para optimizar la rentabilidad en una empresa de consultoría a través del desarrollo de un lago/almacén de datos.
- **Estado del arte:** En esta sección se analiza el estado actual de los data lake/warehouse en las empresas de consultoría. Se analizan las tendencias, las mejores prácticas y las tecnologías utilizadas para procesar información de entrada para maximizar la rentabilidad.
- **Descripción del trabajo:** En esta sección se detallan las metodologías utilizadas para diseñar, implementar y administrar el almacén de datos. Se detallan las etapas del proyecto desde la identificación de los datos pertinentes hasta el diseño de la arquitectura y la implementación de los procesos de ingesta y transformación de datos.
- **Experimentación:** En este capítulo se presentan los resultados de los experimentos y pruebas realizados para validar la eficacia y la eficiencia del data lake en la optimización de la rentabilidad de una firma de consultoría. Los datos obtenidos se analizan y se evalúa el cumplimiento de los objetivos.
- **Gestión del proyecto:** Esta sección describe la gestión del proyecto, que incluye la planificación, la asignación de recursos, el seguimiento del progreso y la resolución de problemas. Se analizan los obstáculos que surgen durante la implementación y se presentan soluciones para superarlos.
- **Conclusiones:** Este capítulo resume los hallazgos y conclusiones principales del proyecto. Se discuten las implicaciones de los resultados obtenidos y se presentan sugerencias para futuras investigaciones y mejoras en el ámbito de los data lake/warehouse en empresas de consultoría.

- **Anexos:** Aquí se incluirá toda la información que no ha podido ser incluida en capítulos anteriores, como un manual de instalación y un manual de usuario.

Capítulo 2

Estado del arte

Este capítulo proporciona un análisis exhaustivo de las diversas opciones y estrategias que se pueden utilizar en cada componente del trabajo. Este análisis incluye la recopilación y el procesamiento de los datos de entrada.

En términos de entrada de datos, existen numerosas opciones para recopilar información de fuentes internas y externas. Estas opciones consisten en la extracción de información de bases de datos relacionales, incorporación de datos de sistemas CRM y ERP, recopilación de información a través de API y servicios web, y recopilación de información a través de técnicas de web scraping para datos estructurados y no estructurados. Al desarrollar y poner en práctica el sistema de recopilación de datos, es importante tener en cuenta las ventajas y desventajas específicas de cada una de estas opciones [2].

Para procesar las entradas de datos y preparar los datos para un mayor análisis, se puede utilizar una variedad de metodologías y herramientas. Estas opciones consisten en procesos de limpieza y normalización de datos, identificación y corrección de valores atípicos y datos faltantes, empleo de métodos de aprendizaje automático y minería de datos para encontrar patrones y tendencias, e implementación de algoritmos para el enriquecimiento de los datos, agregando información adicional de fuentes externas. Los métodos y recursos más efectivos serán elegidos en base a los detalles de los datos y los objetivos de la empresa consultora para la optimización de la rentabilidad.

En cuanto a la arquitectura y las tecnologías empleadas, existen diversas opciones para llevar a cabo la implementación de un data lake o data warehouse. Una opción es utilizar servicios en la nube como Amazon S3 (Simple Storage Service) y Google BigQuery, así como métodos basados en sistemas de procesamiento y almacenamiento distribuido como Hadoop y Apache Spark [3] [4]. Para facilitar

el acceso y la integración de los datos almacenados, también es posible utilizar tecnologías de virtualización de datos y modelos de datos como el esquema de estrella o copo de nieve (Snowflake schema) [5]. La elección de la arquitectura y las tecnologías adecuadas dependerá de la cantidad de datos, la complejidad de las consultas necesarias y las limitaciones presupuestarias de la empresa consultora.

Para crear consultas e informes que permitan el análisis de los datos almacenados en el data lake, se puede utilizar una variedad de métodos y herramientas. Una de estas opciones es presentar los resultados de manera efectiva utilizando herramientas de visualización de datos low code/no code como QuickSight, UI Bakery, Tableau o Power BI, que permiten a los usuarios crear consultas e informes sin necesidad de conocimientos de programación.

2.1. Comparativa de arquitecturas y plataformas cloud

A pesar de que Amazon Web Services (AWS) ha sido la plataforma seleccionada para la implementación de este proyecto, el actual ecosistema de soluciones cloud para data warehousing ofrece múltiples alternativas igualmente válidas y consolidadas. Esta sección presenta una comparativa técnica y estratégica de varias de las plataformas más utilizadas para el desarrollo de data lakes y data warehouses modernos: AWS (con Redshift), Google Cloud Platform (con Big-Query), Microsoft Azure (con Synapse Analytics) y Snowflake [21][22][23][24].

El objetivo de este análisis es contrastar capacidades, evaluar ventajas e identificar casos de uso ideales para cada solución, lo que también puede servir como guía para futuras migraciones o ampliaciones del sistema.

2.1.1. Plataformas analizadas

- **Amazon Web Services (AWS)** – Redshift como motor analítico y S3 como almacenamiento central [21].
- **Google Cloud Platform (GCP**) – BigQuery como solución serverless orientada a análisis masivo [22].
- **Microsoft Azure** – Synapse Analytics como solución integrada de DWH + Big Data [23].

- **Snowflake** – Plataforma nativa en la nube, multi-cloud, con separación de almacenamiento y computación [24].

2.1.2. Criterios de comparación

Se han seleccionado los siguientes criterios como ejes de evaluación:

- **Arquitectura:** Modelo técnico y componentes clave.
- **Facilidad de uso:** Curva de aprendizaje, integración con otras herramientas.
- **Coste:** Modelo de pricing, coste de almacenamiento y consultas.
- **Escalabilidad y rendimiento:** Capacidad de crecer con la demanda sin pérdida de eficiencia**.**
- **Soporte parra herramientas BI:** Compatibilidad con soluciones como Tableau, Power BI, QuickSight, etc.
- **Gobierno del dato:** Seguridad, roles, acceso, auditoría [25][26].

Plataforma	Arquitectura	Facilidad de uso	Coste	Escalabilidad	BI compatible	Gobierno del dato
AWS Redshift	Clásica con clusters y S3	Media	Medio	Alta	QuickSight, Tableau	IAM, S3 policies
BigQuery	Serverless, desacoplado	Alta	Bajo (por uso)	Muy alta	Looker, Data Studio	IAM, etiquetas
Azure Synapse	Integrado con Azure ecosystem	Alta	Medio-alto	Alta	Power BI	Azure AD, Purview
Snowflake	Multi-cloud, escalado auto	Alta	Variable por uso	Muy alta	Tableau, Power BI, etc.	RBAC, Time Travel

Figura 2.1: Tabla comparativa de arquitecturas

2.1.3. Análisis comparativo y justificación de la elección

Aunque la plataforma seleccionada para este proyecto ha sido Amazon Web Services (AWS), es relevante destacar que el actual ecosistema tecnológico ofrece múltiples alternativas igualmente válidas para la construcción de data warehouses modernos. Entre ellas, destacan Google Cloud Platform (GCP) con BigQuery, Microsoft Azure con Synapse Analytics y Snowflake, una plataforma independiente y multi-cloud. El análisis comparativo entre estas opciones resulta fundamental no solo para contextualizar la elección realizada, sino también para abrir la puerta a futuras decisiones estratégicas en caso de que las necesidades técnicas o presupuestarias evolucionen.

Cada una de estas soluciones presenta ventajas particulares que responden a distintos enfoques de negocio, capacidades técnicas del equipo, y niveles de madurez digital en las organizaciones. Por ejemplo, BigQuery, la propuesta de Google, se caracteriza por su enfoque completamente serverless, lo cual elimina la necesidad de aprovisionar y gestionar infraestructuras [22]. Esto la convierte en una opción especialmente atractiva para análisis masivos de datos, ya que permite lanzar consultas sobre grandes volúmenes sin necesidad de configuración previa, lo que se traduce en agilidad y eficiencia operativa. Además, su integración nativa con Looker Studio y otras herramientas de Google facilita la construcción rápida de paneles y visualizaciones.

Microsoft Azure Synapse, por su parte, destaca por su integración completa con el ecosistema Microsoft [23]. Para organizaciones que ya trabajan con herramientas como Power BI, Excel o Azure Data Factory, Synapse permite una transición natural hacia un entorno de analítica avanzada sin salir del mismo marco tecnológico. Su arquitectura híbrida, que combina capacidades de big data con motor SQL clásico, lo convierte en una solución potente y flexible para entornos corporativos con fuerte dependencia de soluciones Microsoft.

En cuanto a Snowflake, se trata de una de las plataformas más innovadoras y ampliamente adoptadas en los últimos años [24]. Su arquitectura desacoplada permite escalar almacenamiento y computación de forma independiente, lo cual optimiza costes y mejora la eficiencia. Además, al ser una plataforma nativa en la nube pero disponible sobre AWS, Azure y GCP, ofrece una versatilidad inigualable en entornos multi-cloud. Su modelo de uso basado en pago por consumo, unido a una experiencia de usuario altamente simplificada, la posiciona como una opción ideal para equipos que buscan alto rendimiento sin una inversión operativa significativa.

No obstante, la elección de AWS en este proyecto responde a una combinación de factores estratégicos y técnicos. En primer lugar, el equipo de desarro-

llo ya contaba con experiencia consolidada en su ecosistema, lo cual permitía aprovechar de forma inmediata sus ventajas sin necesidad de procesos de formación adicionales. AWS proporciona además un alto grado de flexibilidad en la configuración de la infraestructura, aspecto clave para adaptar la arquitectura del data lake a las particularidades operativas de Cloud District. Herramientas como Amazon S3, Redshift, Lambda, QuickSight o Systems Manager Parameter Store han demostrado una capacidad robusta para orquestar el almacenamiento, procesamiento y visualización de datos de forma integrada [21][25].

La posibilidad de definir la infraestructura como código mediante Terraform y automatizar procesos con funciones Lambda ha permitido una implementación modular, escalable y fácilmente mantenible, en línea con los principios de la arquitectura de microservicios adoptada en el proyecto. En definitiva, aunque otras plataformas ofrecen alternativas igualmente válidas, AWS ha sido la opción más adecuada para este caso concreto por su equilibrio entre potencia técnica, familiaridad del equipo, capacidad de integración y flexibilidad operativa.

Capítulo 3

Descripción del trabajo

3.1. Introducción

Para cumplir con los objetivos previamente planteados, se implementará la integración de los inputs de cuatro aplicaciones: Factorial, Teamwork, Holded y Hubspot, en un entorno de AWS (Amazon Web Services). La elección de AWS como plataforma se debe a sus características destacadas, como escalabilidad, flexibilidad, confiabilidad y seguridad, aunque se entrará más en detalle en la próxima sección.

En cuanto a las salidas estandarizadas, una vez tratados estos inputs, se visualizarán en dos aplicaciones específicas: AWS QuickSight para la parte administrativa (se crearán los cuadros de mandos para la operación de los usuarios) y UI Bakery para la presentación final al cliente. QuickSight permite generar informes y análisis detallados para la toma de decisiones internas, mientras que UI Bakery ofrece una interfaz amigable y personalizable para mostrar los resultados de manera atractiva a los clientes, lo que facilita la optimización de la rentabilidad y desarrollar estrategias efectivas en una empresa de consultoría.

3.2. Análisis del sistema

En esta sección, se llevará a cabo un análisis detallado del sistema implementado. Para empezar, hay que tener en cuenta unas cuantas funcionalidades de AWS que han hecho que se escoja como núcleo de este proyecto.

La utilización de AWS en primera instancia, fue por mayor control y experiencia por todas las personas encargadas en desarrollar este data lake, a parte, que tiene

una configuración más extensa, que significa que a diferencia de por ejemplo Google o Azure, que ofrecen unos ajustes ya predefinidos, en AWS se tiene más flexibilidad ya que te deja hacer la configuración desde cero.

AWS proporciona una infraestructura en la nube robusta y altamente disponible, permitiendo el almacenamiento y procesamiento eficiente de grandes volúmenes de datos. Además, ofrece una amplia gama de servicios y herramientas para el procesamiento y análisis de datos, lo que facilita la implementación de un data lake eficiente y optimizado. Al optar por AWS, una empresa puede aprovechar la capacidad de escalar de manera dinámica los recursos de cómputo y almacenamiento según las necesidades cambiantes. Además, la infraestructura segura de AWS garantiza la protección de los datos y la continuidad del negocio [6]. Se diseñará y desplegará la infraestructura necesaria para la correcta operación de los servicios desarrollados. Esta configuración se describirá en Terraform.

El data lake es un almacén de datos en bruto que se obtienen de diferentes orígenes. Estos datos son procesados posteriormente para generar los Data Marts (que son bases de datos diseñadas para satisfacer las necesidades de análisis y consulta de un área de negocio específica o de usuarios concretos, proporcionando una visión detallada y optimizada de los datos relacionados con ese tema en particular [7]) convenientes a su aplicación en los cuadros de mandos específicos para cada cliente/usuario/área.

Dada la complejidad del problema así como la urgencia de su implementación, se aborda desde un punto de vista progresivo. En la primera fase, o bootstrap (configuración) inicial, se realizarán los desarrollos necesarios para cubrir la necesidad actual de las áreas de dirección, marketing y finanzas en cuanto a la gestión de los proyectos. En el capítulo de conclusiones y líneas futuras se exponen posibles áreas donde seguir trabajando.

Terraform

Para la creación del data lake, Terraform, que implementa lo que se conoce configuración como código, se ha utilizado para diseñar y desplegar la infraestructura necesaria para garantizar el correcto funcionamiento de los servicios desarrollados.

Con otras palabras, Terraform se utiliza como registro de todo lo que se programa por consola, y la entrada, lo que se ha configurado, se compara con lo que en este caso se tiene desplegado en AWS y ejecuta las diferencias a la vez que se almacena el log (registro).

Con Terraform, se ha definido y configurado la infraestructura en AWS, incluyendo la creación de instancias de servidores, redes, bases de datos, y otros recursos

como las lambdas, etc.

Además, Terraform también ha sido utilizado para programar los conectores que se encargan de recopilar la información generada desde diferentes fuentes de datos y luego insertarla en el data lake, donde será procesada posteriormente.

En relación a la base de datos, Terraform ha sido utilizado para desarrollar el esquema y automatismos necesarios, como los triggers (disparadores), que permiten una organización eficiente de los datos insertados, así como garantizar la trazabilidad y anonimización de los mismos. Un par de ejemplos de variables que se han configurado en Terraform son: cada cuanto tiempo se actualiza la base de datos, que en este caso será cada dos horas, o por ejemplo cuantos días se actualizan cada vez que se recarga la base de datos, que en este caso serán noventa días, ya que si se realiza algún cambio en el pasado siempre y cuando esté dentro de estos límites, serán incluidos en la tabla, este parámetro se suele limitar por razones de tiempo de carga, ya que al ser tan escalable, llegará un momento en el que nuestra base de datos pese demasiado para ser cargada al completo cada dos horas.

Obtención de datos

Para continuar, se hablará de la obtención de datos. La obtención de los datos se desarrolla en base a funciones lambda. Cada una de estas funciones es un conector con el origen de datos y su misión es obtener los datos de dicho origen y almacenarlo en forma de fichero en el bucket (contenedor) de S3.

Inicialmente se contempla la creación de dos tipos de fichero: JSON o CSV, y este será el formato en el que se almacenarán los datos obtenidos. Cuando el tiempo lo permita, se añadirá un tercer formato, Parquet, en preparación a la arquitectura final.

Estos ficheros ejecutarán un trigger en S3 que ejecutará una función lambda encargada de cargar el contenido de los ficheros en la base de datos. Estos *ingestadores* (procesadores de datos) deben conocer los datos de origen y el destino de dichos datos en la base de datos del data lake.

Las funciones lambda se nombran con el siguiente esquema:

- Conectores: lambda_${PROVEEDOR}${DESCRIPTOR} (Ej: lambda_tw_timelog)
- Ingestadores: lambda${PROVEEDOR}_ingest (Ej: lambda_tw_ingest)

En la primera iteración, el data lake consiste en una serie de tablas, vistas y procedimientos almacenados que se utilizan para condensar los datos procedentes de

los diferentes orígenes. Se toma esta decisión por la facilidad en su implementación y el volumen de datos esperado, que tiene cabida en este tipo de arquitectura.

El esquema de la base de datos y su evolución se mantiene y gestiona desde un fichero de configuración ubicado en el directorio database/Liquibase en el repositorio en Bitbucket.

En este repositorio de Bitbucket también encontramos una carpeta para cada aplicación desde donde se obtienen los datos. En todas ellas se incluyen:

- **event.json:** Este archivo generalmente se utiliza en el contexto de funciones de AWS Lambda. Contiene datos de eventos simulados que se pueden utilizar para probar y depurar la función Lambda. El archivo event.json contiene información que representa un evento específico que se envía a la función Lambda para su procesamiento.

- **helpers.js:** Este archivo es un módulo JavaScript que contiene funciones o utilidades adicionales que se utilizan en el proyecto. Puede incluir funciones comunes que se utilizan en varios archivos del proyecto para evitar la repetición de código y promover la reutilización. Estas funciones pueden realizar tareas específicas o proporcionar funcionalidades adicionales para simplificar el desarrollo. En el caso de este proyecto se ha usado frecuentemente para hacer consultas SQL.

- **index.js:** Este archivo generalmente es el punto de entrada principal de una aplicación o servicio. Contiene la lógica principal y la estructura del programa. En el contexto de un servidor web, en este caso es el archivo con el que se hacen las llamadas a las APIs correspondientes de los campos necesarios para cada aplicación. Es el archivo que se ejecuta cuando se inicia la aplicación.

- **package.json:** Este, es un archivo de configuración utilizado por el sistema de administración de paquetes de Node.js (npm). Contiene información sobre el proyecto, sus dependencias, scripts (códigos) de construcción, versiones y más. Entre otras cosas, el archivo package.json se utiliza para instalar y administrar las dependencias del proyecto, definir comandos personalizados y configurar ciertas características del proyecto.

Otro archivo clave que se encuentra en cada una de las diferentes carpetas es el .env, este es un archivo de configuración que se utiliza para definir variables de entorno en una aplicación, en este proyecto es un archivo común para todas las aplicaciones, ya que en el se encuentran las variables de entorno de todas ellas. Estas variables suelen contener información sensible o configuraciones específicas

del entorno, como claves API, contraseñas de bases de datos u otras variables personalizadas. En nuestro caso, que las lambdas han sido escritas con JavaScript (como se explicará más adelante), existe una librería llamada dotenv, que al meterla en el código, comprueba si se tiene un fichero .env en el directorio, lo carga en memoria, haciendo que se añadan esas variables de entorno a nuestro sistema.

Este tipo de variables, en AWS, se definen en una estructura que se llama SSM (Systems Manager Parameter Store). El SSM es un servicio de administración de parámetros que permite almacenar y recuperar configuraciones y secretos de forma segura en la nube de AWS. Al utilizar el SSM, uno puede almacenar sus variables de entorno de forma centralizada y acceder a ellas desde diferentes servicios y aplicaciones dentro de tu entorno de AWS. Esto facilita la gestión y el mantenimiento de las configuraciones de las aplicaciones, ya que se puede actualizar o modificar los valores de las variables en un solo lugar sin necesidad de modificar directamente los archivos de configuración en cada instancia o servidor. Además, el SSM ofrece características de seguridad como la encriptación de datos y el control de acceso para garantizar la protección de tus variables sensibles [8].

El data lake se implementa en el bootstrap sobre un SGBD PostgreSQL en una instancia AWS RDS. Gracias a esto, previo a la creación de los paneles de control de los datos en QuickSight o UI Bakery, se pueden hacer comprobaciones en aplicaciones como por ejemplo PgAdmin. Sobre esta instancia se desarrolla el esquema mediante migraciones utilizando la herramienta Liquibase desde la que se mantiene su evolución. Se utiliza Liquibase como herramienta de migración para mantener la base de datos actualizada al tiempo que permite la trazabilidad de las acciones realizadas.

Liquibase

Liquibase es una utilidad para gestionar de manera rápida y controlada cambios en el esquema de la base de datos. Se utiliza en el proyecto como mecanismo estándar de gestión del esquema, lo que implica que cualquier cambio se realizará a través de *'changesets'* que se aplicarán mediante Liquibase [9].

Cada base de datos que se gestiona con Liquibase tiene su propio directorio. En el directorio raíz, se encuentra un subdirectorio llamado *'changelog'* que contiene los archivos de cambios. Cada archivo de cambios se nombra siguiendo el formato "{dbname}000000-changelog-{descripción}.sql" y se recomienda generarlos en formato YAML.

El archivo liquibase.properties en el directorio de cada base de datos contiene la configuración de Liquibase, incluyendo el archivo raíz (root.xml) que 'incluye' los archivos que definen los changesets.

Los archivos de cambios (changelog) contienen los cambios que se aplicarán al esquema de la base de datos. Pueden estar en formato YAML o SQL, y se recomienda documentar los cambios con comentarios y siempre incluir un *'rollback'*.

Además, también se encuentra la gestión de lógica almacenada, donde se mantiene un changelog separado dedicado a la lógica almacenada, utilizando la opción *runOnChange="true"* en los changesets para asegurar que los cambios se implementen cuando se modifica un archivo.

Se describen los comandos de Liquibase utilizados en el proyecto, como la inicialización, la aceptación de cambios como realizados, la comprobación del estado de las migraciones, la adición de cambios al esquema, la corrección de errores de *'checksum'* y la sincronización del changelog con la base de datos.

Finalmente, el script llamado migrate.sh, se utiliza para verificar la conexión con la base de datos en Amazon RDS y levantar un túnel SSH si es necesario antes de ejecutar Liquibase .

3.2.1. Descripción de las características funcionales

Las capacidades y características específicas que se espera que el sistema de data lake posea incluyen las siguientes:

- **Ingestión de datos:** El sistema debe de ser capaz de recibir y procesar datos de las diversas fuentes que previamente se han mencionado. Debe contar con mecanismos para garantizar la integridad y calidad de los datos durante la ingestión. Hay que tener en cuenta que en un futuro el número de aplicaciones de la que el data lake bebe los datos puede crecer, por lo que tiene que ser fácilmente escalable.
- **Procesamiento de datos:** El sistema debe ser capaz de procesar los datos de manera eficiente y escalable. Esto implica realizar transformaciones, cálculos, filtrado, agregación u otras operaciones necesarias para obtener la información deseada. Puede requerir el uso de técnicas de procesamiento en tiempo real o en lotes (batches), según los requisitos del sistema.
- **Almacenamiento de datos:** El sistema debe proporcionar un almacenamiento adecuado para los datos procesados, en este caso, utilizando diversas capacidades y características en AWS. Esto incluye servicios como por ejemplo Amazon S3, que ofrece almacenamiento escalable y duradero de objetos. Estos servicios permiten almacenar, procesar, consultar y visualizar datos de manera eficiente y segura, brindando flexibilidad y escalabilidad para adaptarse a los requisitos específicos del sistema. Además, AWS ofrece una amplia

gama de opciones de almacenamiento confiables y escalables que se adaptan a diferentes tipos y volúmenes de datos, y cuentan con mecanismos de respaldo y recuperación ante fallos, garantizando una gestión eficiente de la información en la nube.

- **Consulta de información:** El sistema debe permitir realizar consultas eficientes sobre los datos almacenados. Debe proporcionar un lenguaje de consulta o una interfaz que permita expresar consultas complejas y obtener resultados de manera rápida. Puede incluir capacidades de indexación, optimización de consultas y soporte para consultas bajo demanda (ad hoc).
- **Visualización de información:** El sistema debe ser capaz de presentar la información de manera clara y comprensible para los usuarios. Puede incluir la generación de informes, gráficos, tablas o cualquier otra forma de representación visual de los datos procesados. Debe contar con opciones de personalización y capacidad de exportar los resultados en diferentes formatos.

Ahora se pasará a explicar las interacciones entre los distintos componentes del sistema y cómo se espera que funcionen en conjunto para lograr los objetivos del proyecto:

- **Comunicación:** Los componentes se comunican entre sí para intercambiar información y datos necesarios para llevar a cabo sus funciones. La comunicación más frecuente ha sido mediante APIs (Interfaces de Programación de Aplicaciones) utilizadas sobretodo para la ingesta de datos.
- **Integración:** Los componentes se integran para formar un sistema coherente. Esto puede implicar la combinación de funcionalidades individuales para lograr una funcionalidad más amplia, la interconexión de subsistemas o la interoperabilidad entre diferentes servicios. El caso más claro utilizado en este proyecto ha sido el uso de aplicaciones como UI Bakery para mostrar los datos, y estos a su vez poder ser modificados y que se actualicen automáticamente en la base de datos.
- **Coordinación:** Los componentes deben coordinarse entre sí para asegurar una ejecución adecuada de las tareas y una sincronización eficiente de las operaciones. Esto ha incluido la sincronización de eventos, la gestión de flujos de trabajo o la coordinación de recursos compartidos. Se ha llevado a cabo con la infraestructura programada en Terraform, la cual incluye los diferentes parámetros de despliegue de las aplicaciones de las que obtenemos los datos, para optimizar y unificar los procesos.

- **Dependencias:** Los componentes pueden depender unos de otros en términos de funcionalidad o recursos. Algunos componentes pueden requerir la salida de otros componentes como entrada para realizar sus tareas. Estas dependencias deben ser gestionadas y coordinadas adecuadamente para asegurar un funcionamiento correcto del sistema. No todos los datos obtenidos de las aplicaciones de las que antes se hablaba proveían unos datos útiles y limpios, sin embargo, en combinación con otros datos o con alguna modificación, estos pasaban a ser utilizables. Para una mayor y mejor implementación se decidió unificar estos datos limpios en nuevas tablas para evitar equivocaciones.

3.2.2. Entorno operacional

En esta sección, se proporcionará una descripción detallada del entorno en el cual el sistema estará operativo. Esto incluirá información sobre los requisitos técnicos necesarios, como el sistema operativo, la infraestructura de red y los servidores, así como cualquier otro componente necesario para el funcionamiento adecuado del sistema. También se especificarán los requisitos de software, como las versiones de los programas y herramientas requeridas, y cualquier configuración adicional necesaria.

Dentro del entorno operativo del sistema, es importante mencionar algunos componentes clave utilizados en el desarrollo y despliegue del proyecto. Para la gestión del código fuente, se ha utilizado Bitbucket, una plataforma del grupo Atlassian de control de versiones que permite un seguimiento preciso de los cambios realizados en el código y facilita la colaboración entre los miembros del equipo. Jira y Confluence, también de Atlassian, por su parte, han sido empleados como sistemas de seguimiento de problemas y tareas, permitiendo una gestión eficiente de las tareas pendientes y proporcionando un flujo de trabajo organizado [10].

En cuanto a la herramienta de desarrollo, se ha optado por VSCode, un entorno de desarrollo integrado (IDE) ampliamente utilizado y altamente configurable. Su elección se basa en su amplia compatibilidad con lenguajes de programación, su extensa biblioteca de extensiones y su capacidad para aumentar la productividad durante el proceso de desarrollo.

El proyecto se ha desarrollado en JavaScript (JS) principalmente por la experiencia previa del equipo en este lenguaje, a parte, debido a su versatilidad y amplia adopción en el ámbito del desarrollo web y de aplicaciones. JavaScript ofrece una gran cantidad de *frameworks* (y bibliotecas que permiten desarrollar aplicaciones eficientes y escalables, lo que lo convierte en una elección adecuada para alcanzar los objetivos del proyecto.

Un aspecto relevante del sistema es la necesidad de consultar el archivo .env configurado en el SSM (Systems Manager) a través de las funciones de AWS Lambda. En este contexto, se utiliza Terraform para facilitar esta consulta. Terraform consulta las variables definidas en el SSM y las introduce en el archivo .env durante el proceso de construcción de la lambda. De esta manera, las variables de entorno requeridas por la lambda se encuentran disponibles sin necesidad de incluir directamente el archivo .env en sí mismo. El enfoque de utilizar variables de entorno, y consultarlas desde el SSM, permite una gestión más eficiente y segura de la configuración del sistema. En resumen, es como si las variables de entorno se establecieran manualmente en la consola de la lambda, pero los valores definidos se leen desde el SSM, lo que garantiza una mayor flexibilidad y facilidad de administración en el entorno de ejecución.

3.3. Diseño del sistema

En esta sección, se abordará el diseño del sistema, donde se detallarán las decisiones arquitectónicas, tecnológicas y de infraestructura que se han tomado para construir y desplegar el sistema de manera eficiente y robusta. El diseño del sistema es fundamental para garantizar que cumpla con los objetivos del proyecto y se ajuste a las necesidades de la empresa.

3.3.1. Arquitectura del sistema

El sistema se basa en una arquitectura de microservicios, que ofrece ventajas en términos de escalabilidad, modularidad y mantenibilidad. El sistema se compone de varios microservicios independientes que se comunican entre sí mediante interfaces bien definidas. Cada microservicio se encarga de una funcionalidad específica y puede ser desarrollado, desplegado y escalado de forma independiente, lo que facilita la evolución y el mantenimiento del sistema.

La comunicación entre los microservicios se realiza mediante una capa de API, utilizando protocolos estándar como HTTP. Esto permite una comunicación eficiente y flexible entre los diferentes componentes del sistema, facilitando la integración y el intercambio de datos.

Además, para garantizar la disponibilidad y el rendimiento del sistema, se ha implementado un enfoque de alta disponibilidad y escalabilidad. Los microservicios se despliegan en un entorno de nube utilizando servicios gestionados como AWS, el cual simplifica la administración y permite la escalabilidad automática de los contenedores que ejecutan los microservicios.

En cuanto a la persistencia de datos, se utiliza en este proyecto tan solo bases de datos relacionales, que es lo que requieren los microservicios. Las bases de datos relacionales, como PostgreSQL, se emplean en funcionalidades que requieren transacciones y relaciones complejas para garantizar la integridad y coherencia de los datos.

La arquitectura del sistema se despliega en la nube de AWS, aprovechando los servicios y herramientas que esta plataforma ofrece. Entre ellos, se utilizan servicios de almacenamiento como Amazon S3 para el almacenamiento de archivos estáticos.

3.3.2. Descripción general del sistema

El sistema se representa visualmente en la figura 3.1, donde se puede observar su estructura y las cuatro secciones principales que lo componen. Estas secciones son: 'Empleados y Facturación', 'AWS Boundary', 'Proyectos' y 'Visualización'. Una característica destacada es que la caja que representa Amazon Web Services (AWS) se muestra en color azul, lo cual indica que es el sistema central del cual dependen todos los demás. Por otro lado, las demás secciones representan sistemas externos que no son indispensables para el funcionamiento básico del data lake.

Las secciones de 'Empleados y Facturación' y 'Proyectos' son responsables de proporcionar los datos de entrada al sistema (se puede apreciar que de estas secciones surgen flechas etiquetadas como 'consume'). Estas aplicaciones son las fuentes de datos de las cuales nuestro data lake se alimentará y procesará la información.

En cuanto a AWS, una vez que los datos han sido procesados, se puede observar una única flecha que se dirige hacia la sección de 'Visualización'. Esta sección representa un sistema externo encargado de proporcionar una interfaz tanto para los usuarios finales como para los administradores, donde podrán visualizar y gestionar la información de manera intuitiva y eficiente.

La representación gráfica del sistema ayuda a comprender su estructura y las interacciones entre las distintas secciones, destacando la importancia de AWS como sistema central y resaltando las funciones de adquisición de datos, procesamiento y visualización en las diferentes etapas del proceso.

3.3.3. Descripción de componentes

Para la consecución de los objetivos se desarrollará una arquitectura de microservicios en AWS, que recogerán la información contenida en los distintos orígenes

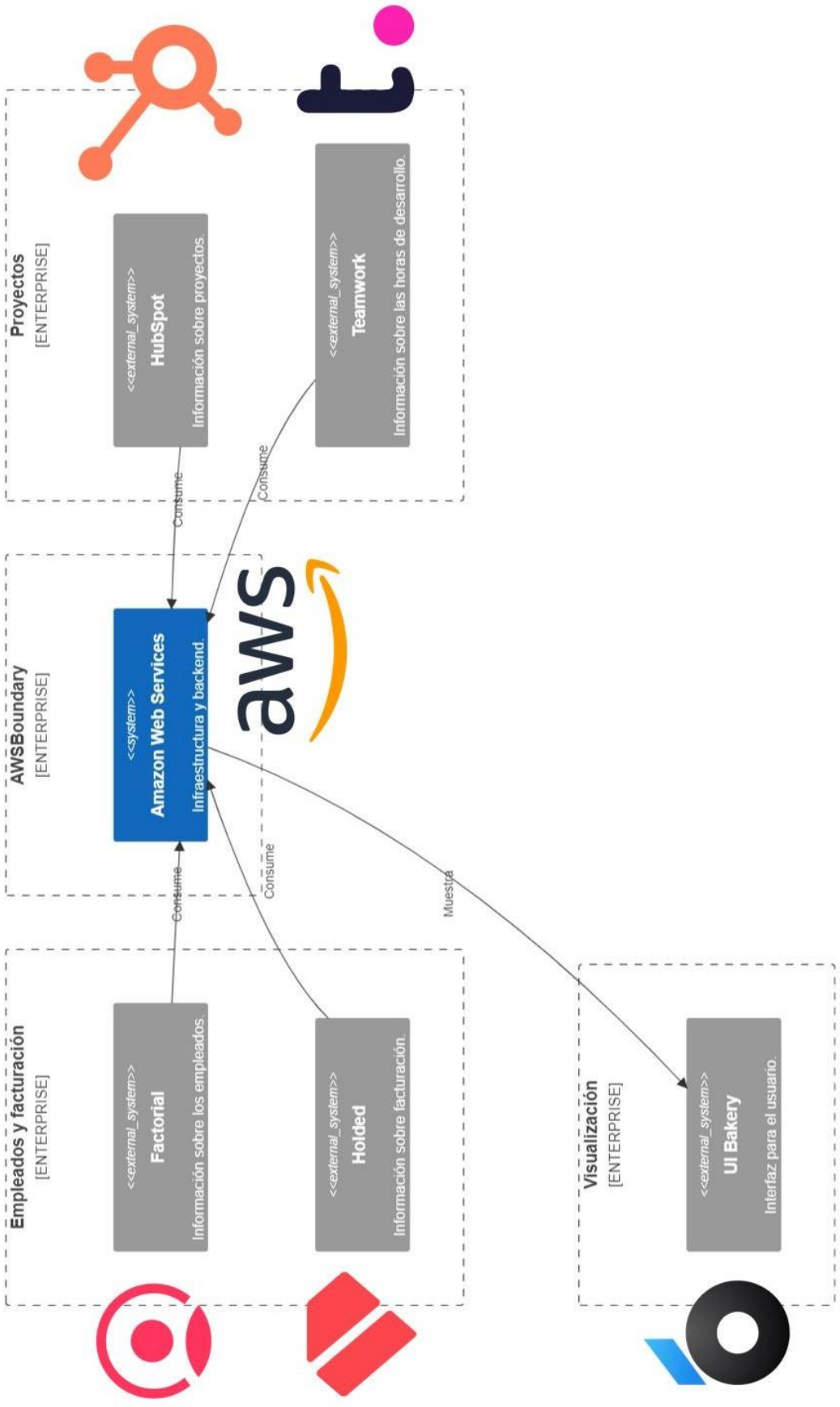

Figura 3.1: Diagrama de los componentes del sistema.

de datos, a partir de los cuáles se generarán los datasets específicos necesarios para la presentación de la información al usuario.

Amazon Web Services (AWS) es una plataforma de servicios en la nube líder en el mercado, proporcionada por Amazon. Ofrece una amplia gama de servicios y soluciones en la nube, que incluyen almacenamiento, computación, bases de datos, redes, análisis, inteligencia artificial, machine learning, Internet de las cosas (IoT) y mucho más. AWS permite a las organizaciones implementar, escalar y administrar sus aplicaciones y recursos de infraestructura de manera eficiente y segura, sin la necesidad de mantener una infraestructura física en sus propias instalaciones. Con la escalabilidad y flexibilidad inherentes a AWS, las empresas pueden adaptarse rápidamente a las demandas cambiantes, reducir los costos operativos y acelerar la innovación. Además, AWS cuenta con una amplia red global de centros de datos, lo que garantiza un alto nivel de disponibilidad y rendimiento para los servicios alojados en la plataforma [11].

El data lake que se pretende desarrollar tiene que integrar los inputs de las cuatro aplicaciones clave que se mencionaban previamente: Factorial, Teamwork, Holded y Hubspot. A continuación, se proporciona una pequeña descripción de cada de estas aplicaciones y su potencial contribución a una empresa de consultoría:

- **Factorial:** Factorial es una aplicación de recursos humanos y gestión de talento. Permite la administración de manera efectiva los datos de los empleados, como salarios, vacaciones, ausentismo e informes, y analiza la productividad y el rendimiento de los empleados. La integración de Factorial en un data lake proporciona información valiosa sobre el capital humano de esa empresa, lo que permite tomar decisiones de gestión del talento y optimizar los recursos [12].

- **Teamwork:** Teamwork es una herramienta de gestión de proyectos y colaboración en equipo. Facilita la planificación, seguimiento y coordinación de tareas y proyectos, así como la comunicación y colaboración entre los miembros del equipo. Al incorporar los datos de Teamwork en el data lake, se puede obtener una visión global de los proyectos en curso, identificar áreas de mejora en la eficiencia y productividad del equipo, y generar informes para evaluar el rendimiento y la rentabilidad de los proyectos [13].

- **Holded:** Holded es una plataforma integral de gestión empresarial que abarca áreas como la contabilidad, facturación, inventario y CRM (Customer Relationship Management). La integración de Holded en el data lake permitiría un seguimiento más preciso de las transacciones financieras, el análisis de la rentabilidad de los clientes, la optimización de los flujos de trabajo contables

y una mejor comprensión de la situación financiera global de la empresa de consultoría [14].

- **Hubspot:** HubSpot es una herramienta de marketing y automatización de ventas que ayuda a las empresas a gestionar y nutrir sus relaciones con los clientes. Proporciona funcionalidades como la gestión de leads (un 'lead' es alguien que ha proporcionado información de contacto y ha manifestado algún grado de interés en lo que la empresa ofrece), la creación de campañas de marketing, el seguimiento de interacciones y el análisis de datos relacionados con el comportamiento del cliente. La integración de HubSpot en el data lake permitiría una visión completa del ciclo de vida del cliente, la identificación de oportunidades de venta y la personalización de estrategias de marketing, lo que contribuiría a mejorar la rentabilidad y la eficacia de las actividades comerciales de la empresa de consultoría [15].

Dentro de la parte de visualización de los datos, se han utilizado dos programas, QuickSight para la parte más administrativa y UI Bakery para el cliente final como se expuso más arriba:

- **QuickSight:** QuickSight es una herramienta de visualización de datos desarrollada por Amazon Web Services (AWS). Permite crear informes interactivos, paneles de control y visualizaciones de datos de manera rápida y sencilla. Con QuickSight, los usuarios pueden conectar a diversas fuentes de datos, explorar y analizar los datos, y compartir los resultados de manera colaborativa. Ofrece capacidades de exploración intuitivas y funciones avanzadas de visualización para facilitar la comprensión de los datos y la toma de decisiones basadas en ellos [16].

- **UI Bakery:** UI Bakery es una plataforma de desarrollo y diseño de interfaces de usuario sin código (no-code). Permite a los usuarios crear aplicaciones y páginas web visualmente atractivas sin necesidad de programación. Con UI Bakery, es posible arrastrar y soltar elementos visuales, personalizar la apariencia y el comportamiento de los componentes, y conectarlos con fuentes de datos para mostrar información dinámica. Esta herramienta es especialmente útil para la creación rápida de interfaces de usuario interactivas y personalizadas, lo que facilita la presentación final de los datos de manera atractiva a los clientes y usuarios finales [17].

Se diseñará y desplegará la infraestructura necesaria para la correcta operación de los servicios desarrollados. Esta configuración se describirá en Terraform. Terraform es una herramienta de código abierto desarrollada por HashiCorp que

se utiliza para automatizar y gestionar la infraestructura como código. Es compatible con una amplia variedad de proveedores de servicios en la nube, incluyendo Amazon Web Services (AWS). Con Terraform, puedes definir y desplegar recursos en AWS, como instancias EC2, grupos de seguridad, balanceadores de carga y más, utilizando un enfoque declarativo. Esto significa que puedes describir tu infraestructura deseada en archivos de configuración, lo que facilita la creación y gestión de recursos de forma reproducible y escalable. Terraform te proporciona una forma eficiente y flexible de trabajar con AWS, permitiéndote automatizar y mantener tu infraestructura de manera consistente y controlada [18].

Se utiliza Liquibase como herramienta de migración para mantener la base de datos actualizada al tiempo que permite la trazabilidad de las acciones realizadas.

Liquibase es una herramienta de código abierto utilizada en el desarrollo de software para el control y gestión de cambios en la estructura de una base de datos. Permite a los equipos de desarrollo automatizar y rastrear los cambios en el esquema de la base de datos a lo largo del tiempo, lo que garantiza la integridad y consistencia de los datos en diferentes entornos. Liquibase utiliza archivos XML, YAML o SQL para definir los cambios en la estructura de la base de datos, como la creación de tablas, la modificación de columnas o la inserción de datos iniciales. Con su enfoque basado en cambios y la posibilidad de deshacer o volver a aplicar cambios específicos, Liquibase facilita la colaboración entre los equipos de desarrollo y mejora la calidad y el control de versiones de las bases de datos en el ciclo de vida del software [19].

3.4. Definición de requisitos

En esta sección, se detallan los requisitos del sistema, clasificándolos en funcionales y no funcionales, con el objetivo de establecer los criterios y las características necesarias para su correcto funcionamiento y desempeño.

3.4.1. Requisitos funcionales

Los requisitos funcionales describen las funcionalidades y las acciones específicas que el sistema debe ser capaz de realizar. Estos requisitos se centran en las tareas y las acciones que el sistema debe llevar a cabo para satisfacer las necesidades de los usuarios y lograr los objetivos del proyecto. Algunos ejemplos de requisitos funcionales pueden incluir:

- El sistema debe permitir al usuario visualizar datos de proyectos para una

toma de decisiones en base a los mismos

- El sistema debe ser restrictivo, es decir, no todos los usuarios finales pueden acceder a los mismos datos.
- El sistema debe proporcionar la funcionalidad de búsqueda para permitir a los usuarios buscar y filtrar información relevante.
- El sistema debe permitir a los usuarios crear, editar y eliminar elementos de la base de datos manualmente, tanto si se hace desde la aplicación input o si se hace desde el dashboard de visualización.
- El sistema debe de ser capaz de actualizar la base de datos y dejar solo el último registro en caso de que este haya sido editado.
- El sistema debe de poder garantizar que al menos los datos desde el 1 de enero de 2023 son correctos.

3.4.2. Requisitos no funcionales

Los requisitos no funcionales se centran en los aspectos de rendimiento, seguridad, usabilidad y otros atributos del sistema que no están relacionados directamente con las funcionalidades específicas. Estos requisitos establecen los estándares y las características que el sistema debe cumplir para proporcionar una experiencia de usuario satisfactoria y garantizar su correcto funcionamiento. Algunos ejemplos de requisitos no funcionales pueden incluir:

- Rendimiento: El sistema debe responder de manera rápida y eficiente, con tiempos de carga y procesamiento mínimos.
- Seguridad: El sistema debe garantizar la protección de los datos y la confidencialidad de la información de los usuarios.
- Usabilidad: El sistema debe ser intuitivo y fácil de usar, con una interfaz clara y accesible.
- Escalabilidad: El sistema debe ser capaz de adaptarse y crecer para manejar un aumento en la cantidad de usuarios y la carga de trabajo.
- Disponibilidad: El sistema debe estar disponible en todo momento, con un tiempo de inactividad mínimo.

En resumen, estos requisitos proporcionan una base sólida para el diseño y la implementación del sistema, asegurando que cumpla con las expectativas y los estándares establecidos [20].

3.5. Casos de uso

En esta sección se presentarán y describirán un par de los casos de uso clave identificados para el sistema. Los casos de uso representan las interacciones y funcionalidades específicas que el sistema debe ser capaz de realizar para satisfacer los requisitos y necesidades de los usuarios y las partes interesadas.

Cada caso de uso describe una tarea o función particular que el sistema debe llevar a cabo, y proporciona detalles sobre los actores involucrados, los pasos a seguir y las condiciones necesarias para su ejecución exitosa. Además, se especificarán los resultados esperados o las salidas obtenidas al completar cada caso de uso.

El análisis y diseño de los casos de uso se ha realizado en estrecha colaboración con los usuarios (en el caso de este proyecto han sido usuarios internos de la empresa) y las partes interesadas, con el objetivo de comprender y documentar las principales interacciones del sistema. Esto nos permite tener una visión clara y detallada de cómo el sistema será utilizado en diferentes situaciones y cómo se espera que funcione en la práctica.

En esta sección, es importante mencionar que los datos presentados en las figuras y ejemplos no corresponden a información real, sino que han pasado por un proceso de anonimización para proteger la privacidad y confidencialidad de los datos. Se ha aplicado este procedimiento de manera rigurosa para garantizar que toda la información mostrada sea completamente anónima y no se pueda rastrear a individuos, organizaciones o cualquier otra entidad relacionada.

3.5.1. Tabla de cargas por centro de coste

El primer caso de uso fue propuesto por los SqL (Squad Leaders), quienes identificaron la necesidad de contar con una tabla interactiva que muestre las cargas por usuario y por centro de coste. Este caso de uso tiene como objetivo brindar a los SqL una herramienta que les permita visualizar de manera clara y concisa la distribución de cargas de trabajo entre los diferentes usuarios y centros de coste, y que se pudiese modificar fácilmente las cargas así como añadir tanto nuevas cargas como usuarios a un centro de coste.

La tabla interactiva proporcionará información detallada sobre las cargas asignadas a cada usuario, permitiendo a los SqL realizar un seguimiento preciso de las tareas asignadas y su distribución equitativa. Además, podrán acceder tanto como a los datos actuales como a datos históricos y realizar consultas dinámicas para obtener estadísticas relevantes sobre la carga de trabajo en diferentes períodos de

tiempo. Estos aspectos son claves para el calculo de los costes de cada centro de coste.

Con esta funcionalidad, los SqL podrán tomar decisiones informadas y equilibrar la asignación de tareas de manera eficiente, asegurando que los recursos estén adecuadamente distribuidos y maximizando la productividad del equipo. Además, la tabla interactiva facilitará la identificación de posibles cuellos de botella o desequilibrios en la carga de trabajo, lo que permitirá tomar medidas correctivas de manera oportuna.

El desarrollo de este caso de uso se llevará a cabo en estrecha colaboración con los SqL y otros usuarios clave, garantizando que la tabla interactiva cumpla con sus necesidades y expectativas. Se realizarán iteraciones y pruebas continuas para asegurar que la funcionalidad sea intuitiva, fácil de usar y proporcione información relevante de manera clara y precisa.

En la figura 3.2 se muestra la pestaña 'Actual' donde aparecen las asignaciones en el mes que aun no ha terminado. Las diferentes funcionalidades que le pueden ver a simple vista son:

- Un buscador con todos los resultados, el cual se ha hecho con diferentes páginas para no aglomerar todos los registros en una misma página. Este está ordenado por orden alfabético.
- Filtrados tanto por nombre como 'employee arn' (identificador único de cada usuario) y centro de coste (cost center).
- Si se selecciona una fila, y se pulsa el botón azul 'Finalizar registro', se le asignará un 'date_end' con el último día del mes actual, ya que las cargas de los centros de coste se suelen revisar mensualmente, y se verá tan solo en la pestaña 'Historial'.
- Si se selecciona una fila, y se pulsa el botón rojo 'Eliminar registro', se eliminará el registro, y este no quedará almacenado en el historial.
- Para modificarle la carga a un usuario, tan solo habría que seleccionar su fila y en la caja de abajo donde pone 'Carga', modificar su número, y seguidamente darle al botón azul de 'Actualizar'.
- Con el botón de 'Añadir', en vez de modificar el registro seleccionado, se creará uno nuevo con la información que se haya modificado en las cajas de abajo (solo se puede modificar la carga y el centro de coste).
- El botón verde 'Nuevo registro', al pulsarlo, pondrá en blanco todas las cajas de su derecha, pudiendo escribir en todas ellas, y a continuación para guardar

la información se clickeará en ‘Añadir’. A este registro, no se le asigna una fecha fin, por lo que siempre aparecerá en la pestaña ‘Actual’.

- Existe la opción de exportar los datos a un archivo CSV pulsando en la esquina inferior derecha de la tabla (en la flecha hacia abajo).

Es importante destacar que siempre que se haga una acción, antes de realizarse, se desplegará un *‘warning’* para ver si se quiere proseguir o no, ya que lo cambios como borrar registros, son irreversibles.

Otro punto importante es que hay que tener cuidado de no duplicar filas, ya que más tarde, las acciones que le apliques a una, la otra también se verá afectada.

Continuando con la pestaña ‘Historial’, como se puede apreciar en la figura 3.3, aparecerán tanto los registros que hayan terminado como los que no. Las funcionalidades que ofrece esta tabla son:

- Un buscador con todos los resultados, con diferentes páginas, ordenado de menor a mayor por su número de ‘employee arn’.
- Filtrado por nombre, identificador y centro de coste
- Clickeando en el icono de la basura, se podrá eliminar definitivamente el registro.

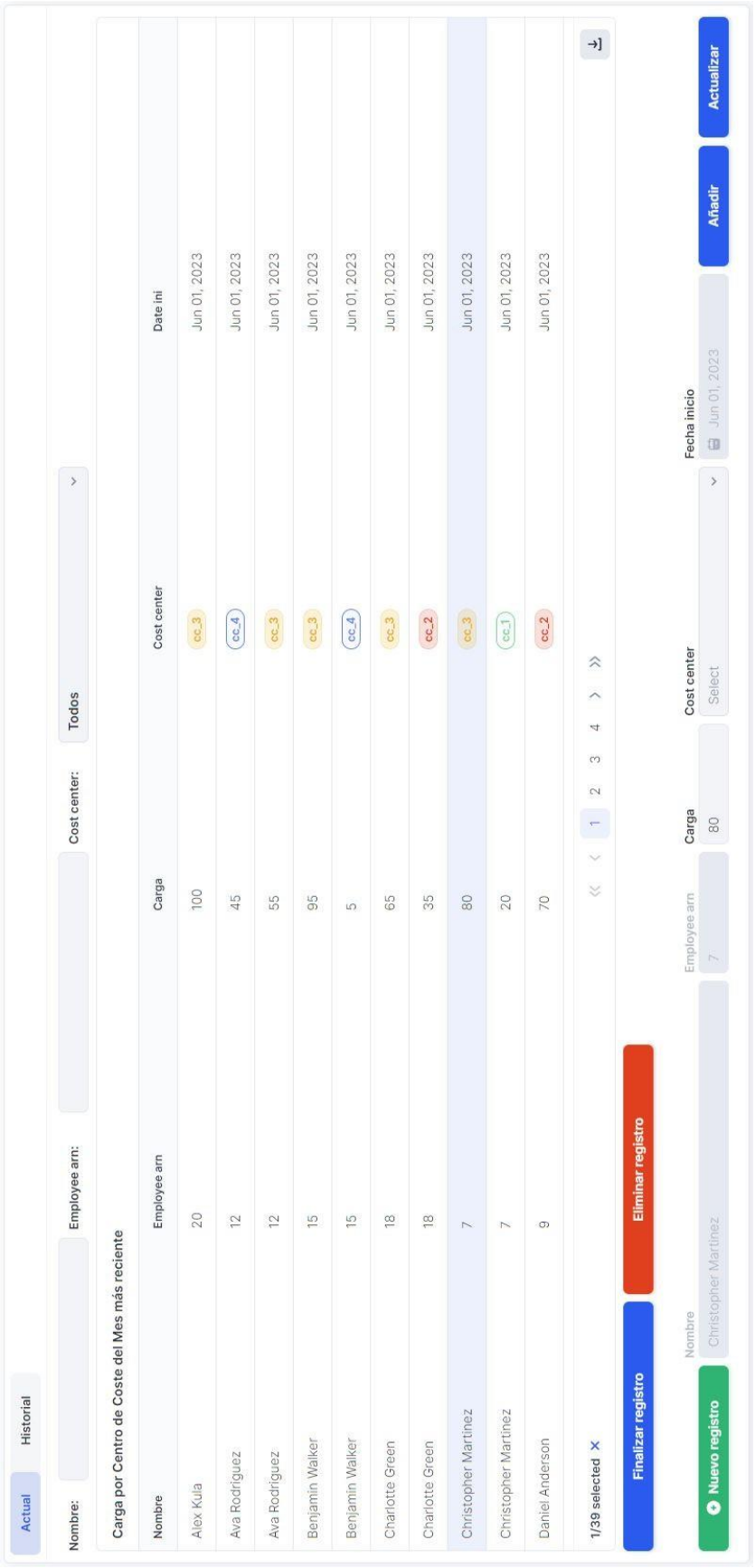

Figura 3.2: Carga por usuario y por centro de coste en el mes actual.

Figura 3.3: Carga por usuario y por centro de coste histórico.

3.5.2. Panel de control de proyectos

En esta visualización, se ha desarrollado una interfaz específica que permite visualizar de manera clara y concisa la información relevante sobre los proyectos. Esta visualización fue diseñada y desarrollada en respuesta a las solicitudes y necesidades tanto de los socios como de los trabajadores del departamento de ventas y finanzas de la empresa.

El panel de control de proyectos proporciona una visión general de los proyectos en curso, permitiendo a los usuarios acceder rápidamente a datos clave como el estado del proyecto, el progreso, los hitos importantes, los recursos asignados y el rendimiento financiero. Además, se pueden generar informes y análisis detallados sobre el desempeño de los proyectos, lo que facilita la toma de decisiones informadas y estratégicas.

La visualización en el panel de control de proyectos es intuitiva y amigable, diseñada para brindar una experiencia de usuario óptima. Los usuarios podrán personalizar las vistas, aplicar filtros y explorar los datos de manera interactiva, permitiéndoles obtener información precisa y actualizada sobre los proyectos en tiempo real.

A continuación, se presentará una descripción detallada del funcionamiento y características del panel de control de proyectos. Este componente del sistema proporciona una visualización exhaustiva y en tiempo real de la información relevante para la gestión de proyectos en la empresa.

Figura 3.4: Panel de control de proyectos (primera parte).

En la figura 3.4 se puede apreciar que es una simple tabla en la que se muestran los diferentes proyectos que están activos en este momento en la empresa, indicando del mismo modo quienes son los socios, DMs (delivery managers) y SqLs que están a cargo de los mismos. Se puede hacer un filtrado tanto por nombre de

proyecto como por nombre de los diferentes cargos, para acceder más prontamente al proyecto requerido.

Una vez seleccionado el proyecto se desplegaran diferentes informaciones acerca del mismo, todas ellas descargables. Un ejemplo sería lo que se ve en la figura 3.5, que es un desglose de todos los ingresos que se han hecho por cada centro de coste involucrado en ese proyecto por mes, y se acumula todo en la gráfica de la derecha, también teniendo en cuenta el mes.

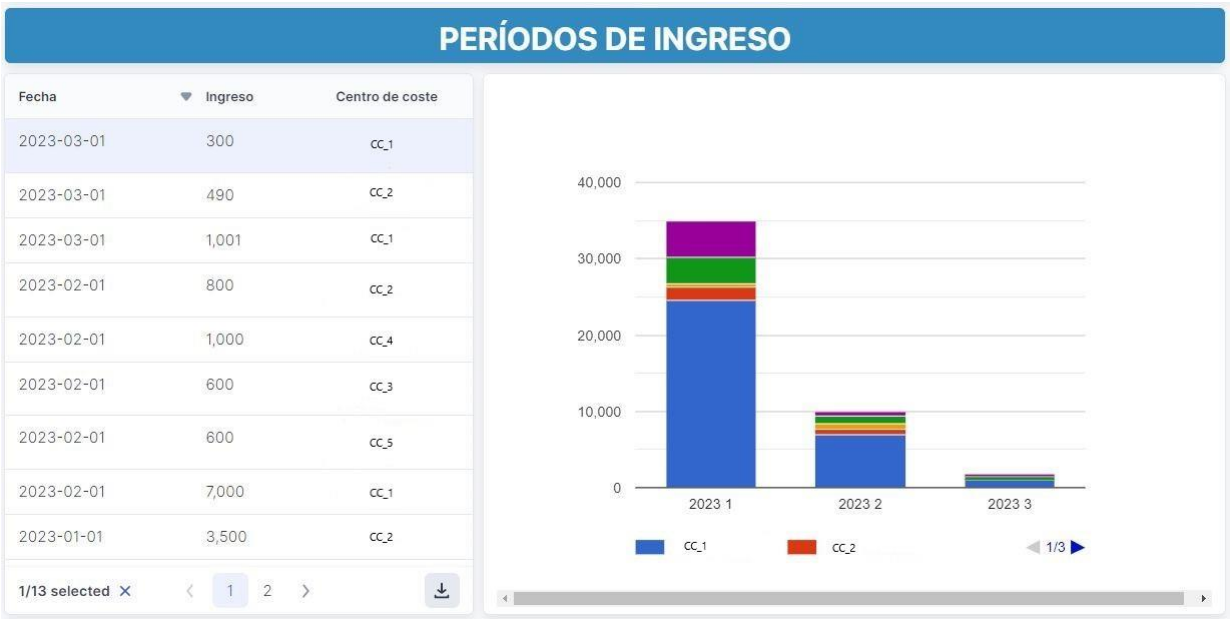

Figura 3.5: Panel de control de proyectos (segunda parte).

Otro ejemplo destacado de información desglosada se muestra en la figura 3.6. Esta representación visual proporciona un resumen completo de los gastos totales, beneficio, ingreso acumulado, rentabilidad actual, ingresos mensuales y rentabilidad mensual del proyecto. La figura ofrece una instantánea clara y concisa de estos indicadores clave, lo que permite una rápida evaluación del rendimiento financiero y la rentabilidad del proyecto en diferentes períodos. La visualización detallada de estos datos facilita la comprensión y el análisis de la salud financiera del proyecto, lo que a su vez respalda la toma de decisiones informadas y estratégicas. Hay que tener en cuenta que los ingresos mensuales y rentabilidad mensual dependen del mes que se haya seleccionado, si ese mes no tiene datos aparecerá un 'N/A'.

Además, en el panel de control se incluye un desglose detallado de los gastos del proyecto, así como una sección dedicada a las facturas. En esta sección, se muestra de manera clara y organizada la información sobre las facturas asociadas al proyecto, incluyendo el monto total facturado hasta la fecha, los pagos recibidos y el

saldo pendiente por cobrar. Esta visión completa de las transacciones financieras relacionadas con el proyecto permite un seguimiento preciso de los ingresos generados y de los pagos pendientes, brindando una perspectiva clara de la situación financiera del proyecto.

Figura 3.6: Panel de control de proyectos (tercera parte).

En la figura 3.7, se presenta un desglose contable acumulado que brinda información detallada sobre los aspectos financieros del proyecto. Este desglose incluye la totalidad de las horas registradas por los empleados para este proyecto, así como los gastos asociados a dichas horas y cualquier otro gasto adicional relacionado con la ejecución del proyecto. Se muestra un subtotal que representa la suma de los gastos directos del proyecto, es decir, los gastos relacionados directamente con su ejecución. Además, se incluyen los gastos indirectos, como por ejemplo, los salarios de personal que trabaja exclusivamente en proyectos internos de la empresa. Por último, se presenta un total general que engloba tanto los gastos directos como los indirectos, brindando una visión global de los recursos financieros asignados al proyecto.

Por último, en la figura 3.8, se presenta un panel con un desglose contable mensual que permite seleccionar un rango de meses dentro del último año (en este caso, 2023) para obtener datos específicos. Esta funcionalidad brinda la posibilidad de analizar la rentabilidad de un mes en particular, identificar el mes en el que se dedicaron más horas al proyecto y realizar otras evaluaciones similares. Los datos mostrados en las secciones superiores de la figura son los mismos que se presentan en la figura 3.7, pero teniendo en cuenta el periodo seleccionado, al igual que los gastos del proyecto y las facturas. Sin embargo, en la parte izquierda, se muestra un gráfico de sectores que representa la distribución de los ingresos según los diferentes centros de coste. Esta visualización proporciona una perspectiva clara sobre cómo se distribuyen los ingresos en relación con los diferentes aspectos del proyecto.

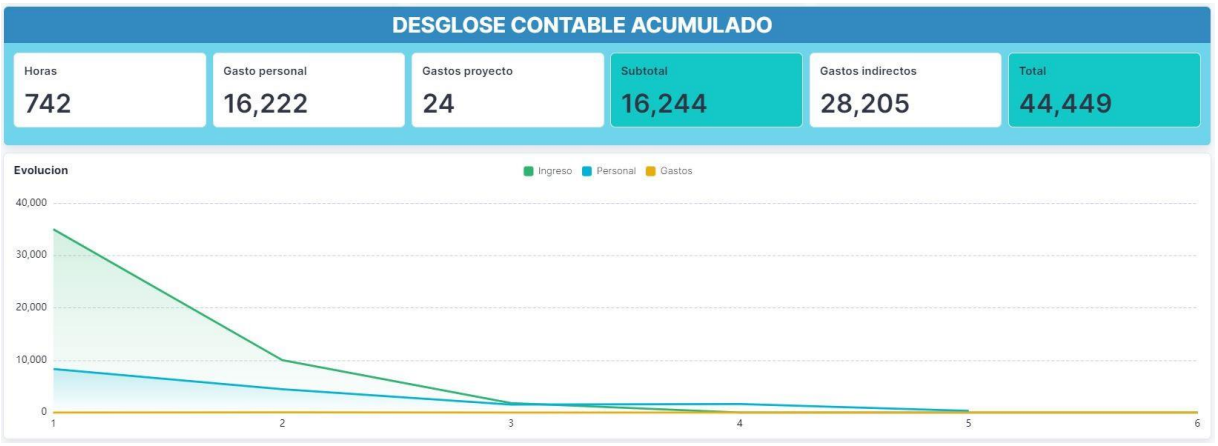

Figura 3.7: Panel de control de proyectos (cuarta parte).

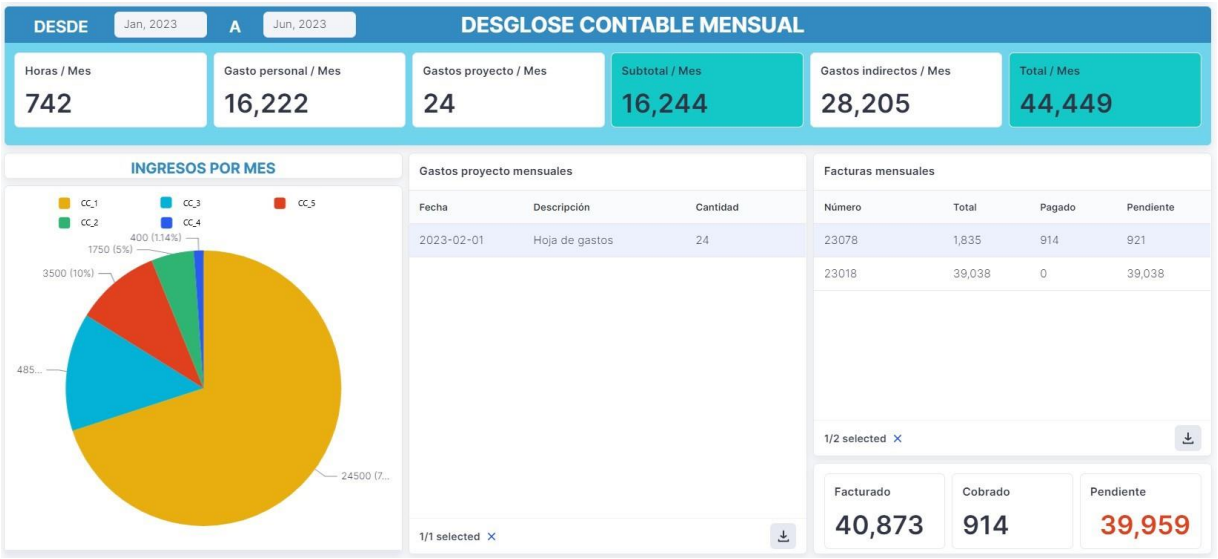

Figura 3.8: Panel de control de proyectos (quinta parte).

Capítulo 4

Experimentación

Para este proyecto en particular, se ha seguido una estrategia de experimentación directa para evaluar y validar el sistema. En este enfoque, se han seleccionado a un grupo de usuarios llamados *beta testers*, quienes han sido responsables de comprobar la precisión y validez de los datos presentados en los tableros de control del data lake. Los *beta testers* han verificado que la información mostrada en los paneles de control sea correcta y coincida con los datos de origen.

Además, se ha llevado a cabo una verificación exhaustiva de todas las funcionalidades del sistema de forma personalizada. Esto implica que se han realizado pruebas manuales y pruebas de funcionamiento en todas las características y funcionalidades del sistema para garantizar que no haya errores o malfuncionamientos. Se ha comprobado que los cambios realizados en el sistema se reflejen correctamente en la base de datos subyacente, asegurando así la integridad de los datos.

La experimentación directa en este proyecto ha permitido una evaluación detallada y minuciosa del sistema, asegurando que los datos sean precisos y que todas las funcionalidades operen sin errores. Esta estrategia ha involucrado la participación activa de los *beta testers* y SqLs y pruebas exhaustivas por parte del equipo de desarrollo, garantizando así la calidad y el correcto funcionamiento del sistema en su totalidad.

4.1. Identificación y Mejoras a través de la Experimentación Directa

Durante las pruebas y el mantenimiento realizado para garantizar el correcto funcionamiento del sistema, se identificaron ciertos problemas que requirieron

atención. Los hallazgos incluyeron:

- Se encontraron registros de horas introducidas en proyectos que no aparecían en la aplicación Teamwork.
- Se detectaron errores en los nombres de algunos proyectos.
- Se identificaron casos en los que diferentes proyectos compartían el mismo identificador.

Estas inconsistencias se originaron debido a un problema en la programación del sistema en relación con las actualizaciones de datos una vez que se habían introducido en las distintas aplicaciones. En el caso de las lambdas de Teamwork, se estableció que los cambios se subirían a la base de datos cada dos horas. Sin embargo, si un dato era introducido y luego eliminado antes de las dos horas, y se creaba uno nuevo con datos correctos, ambos registros coexistían en el sistema. Una situación similar se presentaba en aplicaciones como Hubspot, donde si se ingresaba incorrectamente el nombre de un proyecto y luego se corregía, el registro permanecía con el nombre incorrecto en la base de datos después de dos horas.

Para abordar este problema, se decidió realizar una limpieza completa de la base de datos y comenzar de nuevo con una base de datos limpia. No se perderían datos, ya que los datos correctos se conservaban en las diferentes aplicaciones y solo era necesario volver a cargarlos. A partir de ese punto, se programó una actualización del sistema cada dos horas, eliminando y descargando nuevamente los datos de los últimos tres meses en la lambda de Teamwork (debido a limitaciones de capacidad de procesamiento, se estableció este límite). De esta manera, se aseguraba que al menos los datos de los últimos tres meses fueran siempre correctos.

Sin embargo, es importante tener en cuenta que esta solución no resuelve el problema si se modifica un registro más allá de los tres meses en Teamwork u otra aplicación. Por lo tanto, se ha programado periódicamente una limpieza y actualización general de toda la base de datos para abordar estas situaciones y mantener la integridad de los datos.

Para la aplicación de Hubspot, ya que la carga de datos es mucho menor, se implementó la misma medida solo que en vez de cada tres meses, con toda la base de datos.

Capítulo 5

Gestión del proyecto

En el desarrollo de este proyecto, se ha asumido la tarea de continuar el trabajo que previamente había sido iniciado por otros integrantes del equipo. Al unirse al proyecto, se llevó a cabo un proceso de revisión exhaustivo para comprender el estado actual del código, la arquitectura y los requisitos del sistema. Se realizaron esfuerzos para familiarizarse con la lógica y estructura existentes, así como con las decisiones de diseño que se habían tomado anteriormente.

Es importante mencionar que, dado que no se inició el proyecto desde cero, se ha tenido que adaptar y trabajar con el código existente. Durante este proceso, se han identificado y abordado posibles desafíos relacionados con la coherencia del código, la integración de nuevas funcionalidades y la solución de errores previamente identificados.

Se ha aplicado un enfoque riguroso para garantizar que el trabajo realizado sea coherente con los estándares de calidad y buenas prácticas. Se han realizado pruebas exhaustivas, tanto unitarias como de integración, para asegurar el correcto funcionamiento del sistema y minimizar cualquier impacto negativo debido a la incorporación de nuevo código.

A lo largo del proyecto, se ha mantenido una comunicación abierta y colaborativa con el equipo anterior (que ahora seguía en segundo plano), lo que ha permitido resolver dudas, abordar problemas y garantizar una transición suave en el desarrollo. Además, se han documentado cuidadosamente los cambios realizados y se ha mantenido un registro claro de las decisiones tomadas.

5.1. Descripción de las fases del proyecto

El proyecto se ha dividido en las siguientes fases principales:

- **Fase previa a la incorporación:** En esta etapa inicial, se desarrolló un código base que establecía conexiones con las API de las diversas aplicaciones relevantes, así como la arquitectura del sistema, definiendo los componentes, las interacciones y las interfaces entre ellos, con el fin de explorar el potencial del data lake. Durante esta fase, se llevaron a cabo pruebas iniciales para comprender la viabilidad y la disponibilidad de los datos provenientes de las aplicaciones. El objetivo principal era evaluar la factibilidad técnica del proyecto y determinar si la extracción y procesamiento de los datos eran adecuados para lograr los objetivos establecidos. Esta fase proporcionó información valiosa sobre la accesibilidad de los datos y sirvió como punto de partida para el desarrollo posterior del sistema.

- **Fase de análisis y diseño:** En esta fase se recopilaron y analizaron los requisitos del sistema, así como los objetivos y alcance del proyecto, tal y como se especifica en el apartado anterior. Se llevó a cabo un estudio detallado de las tecnologías y herramientas adecuadas para la implementación del sistema.

- **Fase de desarrollo:** Durante esta etapa, se llevó a cabo la ampliación y mejora del código existente, aprovechando los cimientos proporcionados por los desarrollos previos. Se crearon diferentes tablas y vistas que unificaron e interpretaron los inputs provenientes de las diversas fuentes de datos. Se realizó un análisis exhaustivo de los requerimientos y se identificaron campos de interés adicionales que no habían sido considerados inicialmente. Se realizaron ajustes en las consultas y en la lógica de extracción de datos, asegurando así una mayor precisión y relevancia en la información obtenida. Además, se establecieron relaciones y conexiones entre las diferentes tablas y vistas, permitiendo una visión integrada de los datos.

- **Fase de visualización de datos:** En esta etapa del proyecto, se puso un fuerte énfasis en la visualización de los datos obtenidos, con un enfoque particular en la integración con UI Bakery. El principal objetivo fue presentar la información extraída del data lake de manera efectiva y comprensible, a través de interfaces intuitivas y amigables para los usuarios. La fase de visualización se llevó a cabo en respuesta a las solicitudes y requerimientos planteados por los SQLs (Squad Leaders), quienes expresaron su deseo de visualizar los datos de manera específica. Se trabajó en estrecha colaboración

con ellos para comprender sus necesidades y expectativas, y se implementaron soluciones que cumplieran con sus requisitos.

- **Fase de pruebas:** En esta etapa se realizaron pruebas exhaustivas del sistema en diferentes escenarios para verificar su correcto funcionamiento. Se identificaron y corrigieron los errores encontrados, asegurando así la estabilidad y confiabilidad del sistema.
- **Fase de despliegue:** Durante esta fase, se realizó la preparación del entorno de producción para asegurar un despliegue exitoso del sistema. Se llevaron a cabo pruebas finales exhaustivas para confirmar el correcto funcionamiento de todas las funcionalidades en el entorno de producción, verificando que el sistema respondiera de manera óptima y sin errores. Además, se realizaron tareas de asignación de permisos específicos a los empleados de la consultora, permitiendo el acceso adecuado a los diferentes paneles de control y recursos del sistema de acuerdo con las necesidades y roles de cada usuario.
- **Fase de mantenimiento:** Esta fase se dedica a garantizar el correcto funcionamiento continuo del sistema. Se realizan actualizaciones, correcciones de errores y mejoras según las necesidades y los requisitos de los usuarios.

5.1.1. Diagrama de Gantt

En esta sección, se presenta el diagrama de Gantt correspondiente al proyecto en cuestión, como se puede ver en la figura 5.1, brindando una visión general de la secuencia temporal de las actividades planificadas y su interrelación.

5.2. Presupuesto

El presente proyecto ha sido desarrollado internamente por la empresa, y ya que según las necesidades, este proyecto tendría una duración u otra, implicó que no contara con un presupuesto fijo asignado específicamente para este fin. Sabiendo esto, el proyecto ha sido financiado de manera indirecta a través de los beneficios generados por la empresa en sus relaciones con otros clientes y proyectos. Esta modalidad de financiamiento ha permitido aprovechar los recursos y capacidades existentes dentro de la organización para llevar a cabo el desarrollo y la implementación del proyecto. A lo largo del proceso, se ha realizado una gestión responsable de los recursos disponibles, buscando optimizar la relación costo-beneficio y garantizar la viabilidad económica del proyecto en el contexto de las actividades regulares de la empresa.

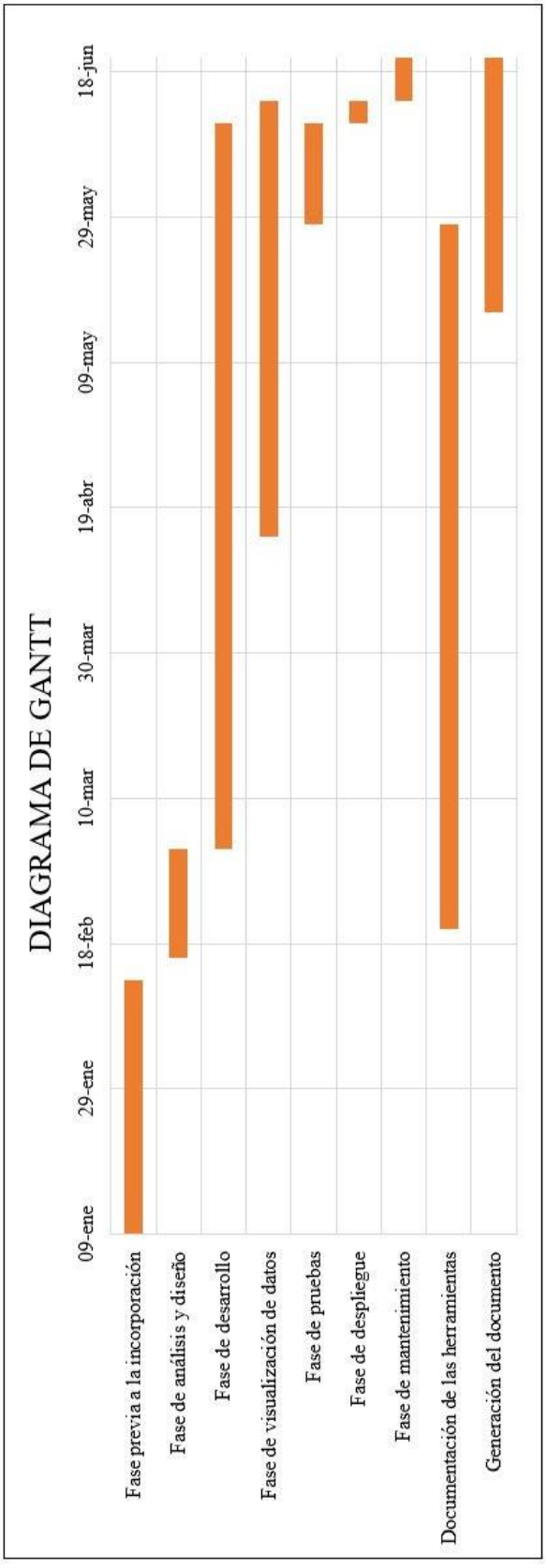

Figura 5.1: Diagrama de Gantt.

Capítulo 6

Conclusiones y trabajos futuros

6.1. Conclusiones generales

Tras la implementación exitosa del data lake en el proyecto, se han obtenido conclusiones altamente positivas. La valoración por parte de los usuarios finales ha sido muy satisfactoria, destacando la calidad visual de las visualizaciones y la notable optimización del tiempo.

El data lake ha demostrado ser una herramienta eficiente y efectiva para la gestión de datos en el proyecto. Los usuarios han destacado su capacidad para proporcionar información relevante de manera clara y concisa, lo que les ha permitido tomar decisiones informadas de manera más ágil y eficiente.

Además, el data lake ha sido especialmente valorado por su capacidad para centralizar y unificar la información de diferentes fuentes, lo que ha facilitado el acceso y análisis de los datos de manera integral. Esto ha permitido a los usuarios obtener una visión más completa y detallada de la situación del proyecto, lo que ha resultado en una mejor toma de decisiones y una mayor eficiencia en la gestión de recursos.

Sin embargo, se han enfrentado ciertas dificultades durante el proceso. Una de ellas se presentó al tratar de asegurar la consistencia de los datos. Como se detalló en el capítulo de experimentación, surgieron desafíos al eliminar un registro existente y reemplazarlo por uno nuevo en la base de datos, lo que resultaba en la persistencia de ambos. Otra casuística fue la duplicidad de identificadores únicos de los proyectos, generado por meter los datos de estos a mano. Para abordar estas situaciones, se implementó una solución que implicaba borrar periódicamente todos los datos de la base de datos y actualizarla con la información más reciente.

Otro desafío fue la comunicación con personas externas al proyecto, como los SqLs. En algunos casos, estas personas no estaban familiarizadas con las capacidades del data lake y solicitaban acciones que requerían un equipo más experimentado y con mayores recursos para su implementación adecuada. Esta situación puso de manifiesto la necesidad de una comunicación efectiva y una gestión adecuada de las expectativas con los colaboradores externos.

Además, se encontró otra complicación relacionada con la disponibilidad de ciertos campos dentro de una aplicación por encontrarse en fase de desarrollo. Estos campos no podían descargarse mediante la API, lo que requería la extracción manual de los datos correspondientes. Esta tarea adicional consumió más tiempo del previsto, generando un impacto en la eficiencia del proceso. Se identificó la necesidad de mejorar la coordinación con los equipos responsables de las aplicaciones en desarrollo para garantizar la disponibilidad y accesibilidad adecuada de los datos requeridos. Tras comunicar esta preocupación, se obtuvo la respuesta de que los equipos estaban trabajando en solucionar este problema. Por lo tanto, se considera esta línea de acción como una propuesta para el futuro, con el objetivo de optimizar la obtención de datos y minimizar la necesidad de extracciones manuales. Esta colaboración más estrecha permitirá una integración más fluida entre las aplicaciones y el data lake, mejorando así la eficiencia y precisión de los procesos.

6.2. Conclusiones referentes a los objetivos

En cuanto a las conclusiones relacionadas con los objetivos establecidos, se ha logrado con éxito la transformación de los datos desde su estado bruto a operacional, lo que ha permitido su utilización efectiva en la toma de decisiones y la extracción de información relevante para los usuarios finales.

La implementación de la arquitectura de microservicios en AWS también ha sido exitosa. Gracias a la programación y automatización del despliegue de las lambdas, los procesos de carga y transformación de los datos se ejecutan de manera totalmente automática. Esto ha simplificado considerablemente el trabajo de los usuarios y ha asegurado que los datos estén siempre accesibles, estandarizados y consistentes.

En relación a la visualización de los datos, se ha realizado un notable trabajo en los casos de uso identificados. Es importante destacar que, en última instancia, se decidió unificar la mayoría de las visualizaciones en la plataforma de UI Bakery en lugar de QuickSight. Esta decisión fue tomada desde la dirección, considerando la funcionalidad de UI Bakery que permite asignar permisos individualizados a

los diferentes dashboards, restringiendo así su acceso a usuarios administrativos y garantizando un mayor control en la seguridad de los datos.

6.3. Trabajos futuros

En esta sección, se presentan algunas áreas de mejora y posibles trabajos futuros relacionados con el data lake, considerando su escalabilidad, flexibilidad y potencial de aplicación en diversas áreas.

Una primera área de enfoque sería la traducción del código de las lambdas de Java Script a Python. Python es un lenguaje de programación ampliamente utilizado y cuenta con una extensa documentación, lo que facilitaría su mantenimiento y evolución.

Además, teniendo en cuenta las futuras actualizaciones de las aplicaciones de las cuales se extraen datos para el data lake, se ha identificado la oportunidad de unificar servicios proporcionados por estas aplicaciones. Esta consolidación permitiría simplificar los códigos y reducir los costos asociados a las suscripciones de las aplicaciones, sin comprometer la funcionalidad del sistema.

Otro caso de uso interesante, aunque pendiente debido a restricciones de tiempo, es la creación de un optimizador de cargas. Esta aplicación o panel de control consideraría la tabla de cargas por centro de coste y empleado, y proporcionaría recomendaciones sobre la asignación óptima de recursos humanos entre proyectos. Teniendo en cuenta factores como el costo por hora de cada trabajador, las horas requeridas por cada proyecto y la carga individual de los empleados, se podrían tomar decisiones informadas para mejorar la rentabilidad de los proyectos. Esta funcionalidad facilitaría el trabajo de los Squad Leaders al automatizar cálculos y ajustes manuales.

Por último, se sabe que el data lake tiene el potencial de ofrecer a cada departamento de la empresa una interfaz y utilidades altamente personalizadas que se adapten a sus necesidades específicas. En el pasado, el desarrollo de herramientas empresariales personalizadas suponía un costo elevado, con programadores que cobraban tarifas considerables. Sin embargo, en la actualidad, el acceso a estas soluciones se ha vuelto más accesible y asequible para las empresas. Esta accesibilidad plantea la posibilidad de ampliar el enfoque del data lake para abarcar a todos los empleados de la empresa, no solo limitándolo a funciones administrativas. En lugar de utilizar múltiples aplicaciones dispersas, el data lake puede convertirse en una interfaz única y centralizada para cada empleado, que reúna todas las funciones necesarias. Esto simplificaría las tareas de recursos humanos y ahorraría tiempo y esfuerzo, al proporcionar a los usuarios una experiencia integrada y

eficiente.

Apéndice A

Manual de instalación

En esta sección, se proporcionará una guía paso a paso para la instalación/uso del software de UI Bakery, aplicación base de este proyecto donde se exponen los centros de mando útiles para el usuario. A continuación, se detallarán los pasos necesarios para llevar a cabo la instalación.

A.1. Requisitos del sistema

Antes de comenzar con la instalación, es importante saber que hay tres maneras de acceder a la aplicación: descargándola, accediento online, o a través de infraestructuras cloud como Azure o Kubernetes.

Verifica que el sistema cumpla con los requisitos mínimos necesarios para ejecutar el software desde el navegador. Son los siguientes:

- Sistema operativo compatible: UI Bakery es compatible con Windows, macOS y Linux. Asegúrate de que el sistema operativo de tu dispositivo cumpla con los requisitos de compatibilidad.
- Conexión a Internet: Se requiere una conexión estable a Internet para acceder a la aplicación y utilizar todas sus funcionalidades. Asegúrate de tener acceso a Internet durante la instalación y el uso de UI Bakery.
- Navegador web actualizado: UI Bakery es una aplicación basada en web, por lo que necesitarás un navegador web moderno y actualizado para acceder a ella. Se recomienda utilizar Google Chrome, Mozilla Firefox, Microsoft Edge o Safari en sus versiones más recientes.

- Hardware: Aunque no hay requisitos de hardware específicos para UI Bakery, se recomienda tener un dispositivo con un procesador y memoria RAM adecuados para un rendimiento óptimo. Si tu dispositivo cumple con los requisitos mínimos para ejecutar un navegador web moderno, debería ser suficiente para utilizar UI Bakery.

A.2. Instalación del software

En caso de querer hacerlo en desde una aplicación física, también existe la opción de descargar la aplicación. Para instalar y utilizar UI Bakery, sigue estos pasos:

1. Descarga: Si eliges descargar UI Bakery, visita el sitio web oficial de UI Bakery y busca la opción de descarga. Haz clic en el enlace de descarga correspondiente a tu sistema operativo.
2. Instalación: Una vez descargado el archivo de instalación, ejecútalo y sigue las instrucciones del asistente de instalación. Asegúrate de leer y aceptar los términos y condiciones antes de continuar.
3. Configuración inicial: Durante el proceso de instalación, se te puede pedir que realices alguna configuración inicial, como establecer una contraseña de administrador o proporcionar información básica del sistema. Completa estos pasos según sea necesario.
4. Inicio de sesión: Después de la instalación, se te pedirá que inicies sesión en UI Bakery. Ingresa tus credenciales de usuario proporcionadas por el administrador del sistema y haz clic en 'Iniciar sesión' para acceder a la aplicación.
5. Uso de la aplicación: Una vez que hayas iniciado sesión correctamente, podrás utilizar UI Bakery para acceder a los centros de mando y visualizaciones disponibles. Explora las diferentes secciones y opciones de la aplicación para acceder a la información deseada y realizar las operaciones necesarias.

Si deseas utilizar UI Bakery a través de infraestructuras cloud como Azure o Kubernetes, existen un par de condiciones adicionales que habría que seguir:

- Creación de una máquina virtual o cluster: En Azure, crea una máquina virtual o una instancia de Kubernetes según tus requisitos. En el caso de Kubernetes, puedes crear un clúster de Kubernetes donde se ejecutará UI

Bakery. Si utilizas otra plataforma cloud, sigue los pasos proporcionados por esa plataforma para crear una instancia adecuada.

- Configuración de la máquina virtual o cluster: Una vez que hayas creado la máquina virtual o el cluster, configura la instancia según las necesidades de UI Bakery. Esto puede incluir la selección de la imagen o plantilla de sistema operativo adecuada, asignación de recursos como CPU y memoria, configuración de almacenamiento y configuración de la red.

Recuerda seguir las instrucciones específicas proporcionadas por el equipo de desarrollo de UI Bakery, ya que pueden haber variaciones en el proceso de instalación dependiendo de las versiones y configuraciones específicas del software.

Apéndice B

Manual de usuario

En esta sección, se proporcionará una guía para que los usuarios finales puedan utilizar el proyecto sin necesidad de tener conocimientos técnicos. Se explicará cómo acceder a la aplicación UI Bakery y cómo operar desde allí para aprovechar al máximo las visualizaciones y los datos centralizados. A continuación, se detallarán los pasos necesarios para utilizar el proyecto.

B.1. Acceso a la aplicación UI Bakery

Para acceder al proyecto y comenzar a utilizarlo, sigue estos pasos:

1. Abre tu navegador web preferido (como Google Chrome, Mozilla Firefox o Microsoft Edge).
2. Ingresa la URL de la aplicación UI Bakery en la barra de direcciones del navegador. Por ejemplo, podría ser 'https://cloud.uibakery.io/dev/clouddistrict/' o cualquier otra dirección proporcionada específicamente para el proyecto, ya que cada empresa tendría su url específica.
3. Presiona 'Enter' o haz clic en el botón 'Ir' para acceder al sitio web de la aplicación.
4. Espera a que la página cargue completamente. Una vez cargada, se mostrará la página de inicio de la aplicación UI Bakery.
5. Si se requiere iniciar sesión, ingresa tus credenciales de usuario proporcionadas por el administrador del sistema. Estas credenciales pueden ser un

nombre de usuario y una contraseña o cualquier otro método de autenticación utilizado en el proyecto.

6. Una vez que hayas iniciado sesión correctamente, se te redirigirá a la interfaz principal de la aplicación UI Bakery, desde donde podrás operar con las visualizaciones y los datos centralizados.

B.2. Operación desde la aplicación UI Bakery

Una vez dentro de la aplicación UI Bakery, podrás aprovechar las visualizaciones y los datos centralizados siguiendo estos pasos:

- Explora las diferentes secciones o pestañas disponibles en la interfaz de la aplicación. Estas secciones pueden estar organizadas en categorías relevantes para el proyecto y pueden incluir paneles de control, gráficos, tablas u otras representaciones visuales de los datos.
- Haz clic en las diferentes opciones o elementos interactivos dentro de las visualizaciones para obtener más información o navegar a vistas específicas. Por ejemplo, podrías hacer clic en un gráfico para ver detalles adicionales, ajustar filtros o seleccionar diferentes opciones de presentación.
- Utiliza las opciones de navegación o búsqueda proporcionadas en la interfaz para acceder a información específica o para explorar diferentes aspectos de los datos. Esto puede incluir la capacidad de filtrar los datos por categorías, fechas u otras variables relevantes.
- Interactúa con los elementos interactivos dentro.

Bibliografía

[1] “Data Warehouse: ¿qué es y cómo utilizarlo?” 2022. Formation Data Scien- ce | Datascientest.com. January 10, 2022. https://datascientest.com/es/data- warehouse-que-es-y-como-utilizarlo.

[2] Pursell, Shelley. 2021. “Sistemas de información en las empresas: tipos, funciones y ejemplos.” Hubspot.es. December 8, 2021. https://blog.hubspot.es/marketing/sistema-informacion.

[3] “Cómo Construir Un Data Lake.” n.d. Google Cloud. https://cloud.google.com/architecture/build-a-data-lake-on-gcp?hl=es-419.

[4] “Lago de datos | implementaciones.” n.d. Amazon.com. https://aws.amazon.com/es/solutions/implementations/data-lake-solution/.

[5] “Snowflake para data lake.” n.d. Snowflake.com. https://www.snowflake.com/es/data-cloud/workloads/data-lake/.

[6] “Data lakes and Analytics” n.d. Amazon.com. https://aws.amazon.com/es/big- data/datalakes-and-analytics/.

[7] “What is a Data Mart?” n.d. Amazon.com. https://aws.amazon.com/es/what- is/data-mart/.

[8] “¿Qué es AWS Systems Manager?” n.d. Amazon.com. https://docs.aws.amazon.com/es_es/systems-manager/latest/userguide/what-is-systems-manager.html.

[9] “The Liquibase Community.” 2020. Liquibase.org. April 14, 2020. https://www.liquibase.org/.

[10] Atlassian. n.d. “Bitbucket Overview.” Bitbucket. https://bitbucket.org/product/guides/getting-started/overview.

[11] “What is AWS?.” n.d. Amazon.com. https://aws.amazon.com/es/what-is/data-mart/.

[12] “All-in-One Human Resources (HR) Software - FactorialHR.” n.d. Factorialhr.com. https://factorialhr.com/.

[13] “Project and Team Management Software.” n.d. Teamwork.com. https://www.teamwork.com/.

[14] “Holded the Smart Management Software for SMEs.” n.d. Holded.com.. https://www.holded.com/.

[15] “Qué es HubSpot y cómo funciona.” n.d. Hubspot.es. https://www.hubspot.es/products.

[16] “QuickSight” n.d. Amazon.com. https://aws.amazon.com/es/quicksight/.

[17] “Build Internal Tools Faster than Ever.” n.d. Uibakery.Io. https://uibakery.io/.

[18] “Terraform by HashiCorp.” n.d. Terraform by HashiCorp. https://www.terraform.io/.

[19] “Introduction to Liquibase.” n.d. Liquibase.com. https://docs.liquibase.com/concepts/introduction-to-liquibase.html.

[20] P N Sawadogo, E Scholly, C Favre, E Ferey, S Loudcher, and J Darmont. «metadata systems for data lakes: models and features,» de new trends in databases and information systems. In ADBIS 2019, volume 1064, pages 440–451. Springer, Cham, 2019.

[21] Amazon Web Services, Amazon Redshift - Data Warehouse in the Cloud. AWS Documentation, 2023. Disponible en: https://aws.amazon.com/redshift/

[22] Google Cloud, BigQuery: Serverless, highly scalable, and cost-effective cloud data warehouse. Google Cloud Documentation, 2023. Disponible en: https://cloud.google.com/bigquery

[23] Microsoft Azure, What is Azure Synapse Analytics?. Microsoft Learn, 2023. Disponible en: https://learn.microsoft.com/en-us/azure/synapse-analytics

[24] Snowflake Inc., Snowflake Documentation. Snowflake Official Docs, 2023. Disponible en: https://docs.snowflake.com/

[25] Raj, P., & Dutt, A., Big Data Analytics. Packt Publishing, 2019. Capítulo 6: “Modern Data Warehousing Architectures”.

[26] Armbrust, M., Ghodsi, A., Xin, R., & Zaharia, M. (2021). Lakehouse: A New Generation of Open Platforms that Unify Data Warehousing and Advanced Analytics. Proceedings of the CIDR Conference. Disponible en: https://www.cidrdb.org/cidr2021/papers/cidr2021_paper17.pdf