A close-up of a logo

Description automatically generated

**Sprint 2**

**Grupo 2: María Carreño Nin de Cardona, Raquel Fernández Esquinas, Teresa X. Garvía Gallego e Isabel V. Morell Maudes**

**Tecnologías de Procesamiento Big Data**

**3º Grado en Ingeniería Matemática e Inteligencia Artificial**

# Índice

[Introducción 3](#_Toc177138896)

[Metodología 4](#_Toc177138897)

[Resultados 5](#_Toc177138898)

[Conclusión 6](#_Toc177138899)

# Introducción

# Este sprint supone la segunda práctica del proyecto final de la asignatura Tecnologías de Procesamiento Big Data sobre las criptomonedas. En este sprint del proyecto, daremos un paso crucial para guardar y leer los metadatos relacionados con los datos históricos almacenados en S3 mediante la creación de un crawler utilizando el servicio de AWS Glue Data Catalog y AWS Glue Crawler.

El objetivo principal de este sprint es desarrollar un script en Python que automatice tanto la creación del catálogo de datos, como la configuración del crawler.

Dado que la seguridad es un pilar fundamental para el cliente, el código subido no debe contener ningún tipo de credencial o información sensible.

Este proceso permitirá mejorar la gobernanza de los datos almacenados en S3, asegurando una gestión eficiente y segura de los metadatos necesarios para el análisis en futuras fases del proyecto.

# Metodología

Para completar la segunda historia de usuario hemos utilizado Glue de AWS que es un servicio de integración de datos ‘serverless’ que facilita la detección, preparación y la combinación de los datos para analística, machine learning y desarrollo de aplicaciones.

Comenzamos creando un Glue Data Catalog para almacenar y organizar los metadatos de las tablas que hicimos en el sprint interior desde la página web. Para ello, utilizamos una política de glue que permita acceder a todos sus servicios y otra de lectura de s3 que nos permitirá leer los datos almacenados en el bucket que creamos en el Sprint 1.

Después, hemos hecho un script de Python llamado create\_bbdd.py que utiliza la librería boto3 para conseguir la funcionalidad pedida. Comenzamos creando un cliente de glue con boto3.client que recibe el servicio de “glue” y las diferentes credenciales necesarias. Éstas se encuentran en un fichero llamado credenciales.py que tenemos almacenado en local con el fin de no hacerlas públicas. El cliente comienza creando una base de datos con nombre “trade\_data\_imat3a02” y luego por cada crypto se crea una carpeta y por cada año se crea una table. Para ello, guardamos los strings con las direcciones de las carpetas en s3 y creamos un crawler al que pasamos todos enlaces. Así conseguimos crear la base de datos con las diferentes tablas con los metadatos de los diferentes ficheros del buckets de s3.

Esta sección describe en detalle cómo se ha implementado la solución, abordando las decisiones técnicas, las tecnologías utilizadas y el proceso de desarrollo. Incluye:

1. **Descripción del entorno de desarrollo**: Herramientas utilizadas: p.ej Python, librerías, etc., así como el entorno de ejecución.
2. **Diseño de la solución**: Describe el diseño y funcionamiento de la arquitectura del sistema. Puede incluirse texto, diagramas, u otros recursos visuales que ayuden a comunicar la solución de manera efectiva.
3. **Pruebas realizadas:** Explica cómo se realizaron las pruebas para verificar el correcto funcionamiento del sistema.

# Resultados

En primer lugar, decidimos hacer a mano la creación de la database, el datalog y el crawler en Glue de AWS, siguiendo el tutorial con el fin de afianzarnos más con el tema.

Una vez que ya interiorizamos qué teníamos que hacer, hicimos un programa de Python que nos permitía crear todo lo anterior con solo ejecutarlo, como ya hemos explicado anteriormente.

A la hora de crear el crawler se nos pedía la creación de una política, donde tuvimos que crear un rol y asociárselo a esa política, donde se le permitía al usuario poder hacer cualquier operación de lectura en el bucket. Para la creación de la política también añadimos las de Glue.

Como resultados, podemos observar las siguientes creaciones que hemos hecho.

En primer lugar, vemos como el rol que hemos creado para la creación del crawler está asociado a las 2 políticas mencionadas anteriormente, la de Glue y la de lectura de los buckets de s3.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

La política final sería la que observamos en la imagen de debajo, la de trading-view que usaremos a la hora de crear el crawler

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Una vez que ya tenemos las políticas y el rol definido podemos crear el crawler. Pero antes, creamos el datalog donde se va a almacenar el formato de la estructura de los datos del bucket. (trade\_data\_imat3a02).

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

TERMINARRRRRRRRR

En esta sección se presentan los resultados obtenidos durante la ejecución del sprint, demostrando cómo la solución implementada resolvió el problema planteado. Debe incluir:

1. **Descripción de los resultados**: Describe los resultados obtenidos a partir de la implementación.
2. **Pantallazos de la ejecución (IMPORTANTE)**: Incluye capturas de pantalla que muestren la ejecución del sistema en una terminal o entorno de pruebas.
3. **Discusión de los resultados**: Comparar los resultados obtenidos con los esperados. Este apartado pretende responder a preguntas como: ¿Se comporta el sistema de la manera prevista? ¿Qué factores han afectado a cada resultado? ¿Ha habido algún comportamiento inesperado del sistema? ¿Por qué?

# Conclusión

La conclusión debe resumir los principales hallazgos y aprendizajes obtenidos durante el sprint, así como destacar la relevancia de la solución implementada. Incluye:

1. **Resumen del proceso**: Recapitula brevemente el proceso seguido desde la identificación del problema hasta la obtención de los resultados.
2. **Principales logros**: Destaca los logros más importantes, como el correcto funcionamiento del sistema, en base a los resultados obtenidos.