

# Patrón Log Aggregation

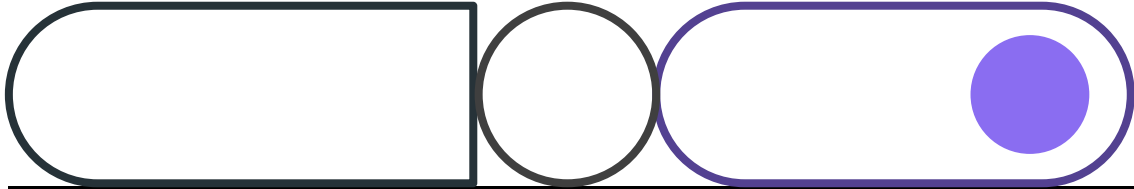
# Índice

- 01** Qué es Log Aggregation
- 02** Stack ELk
- 03** Implementación en Spring Boot
- 04** Index Lifecycle Management

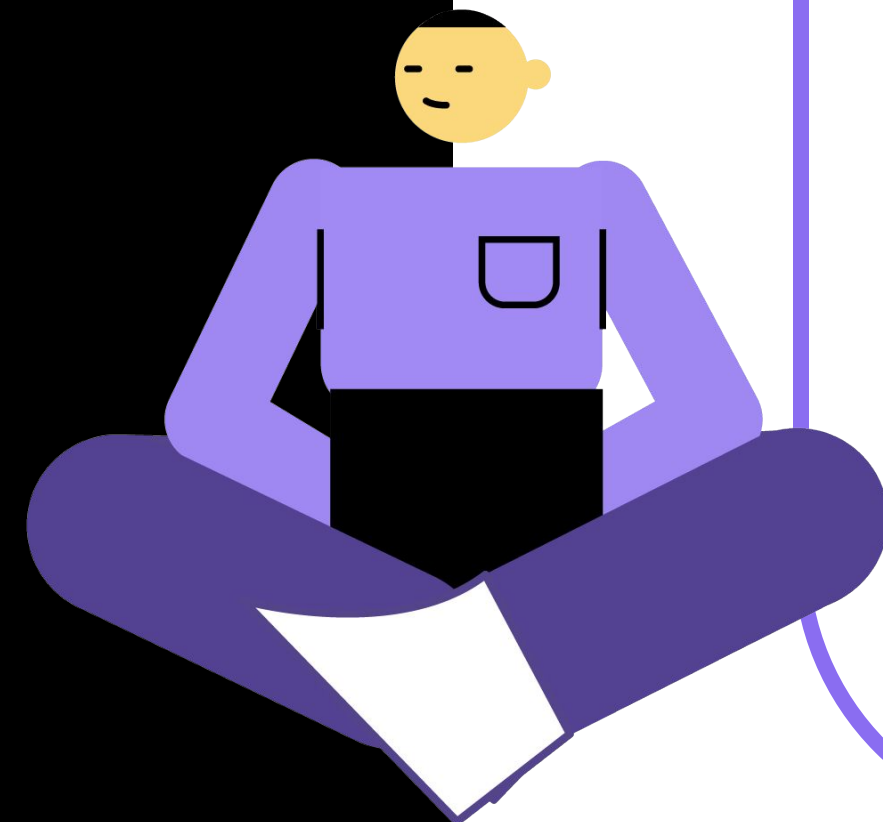


01

# Qué es Log Aggregation

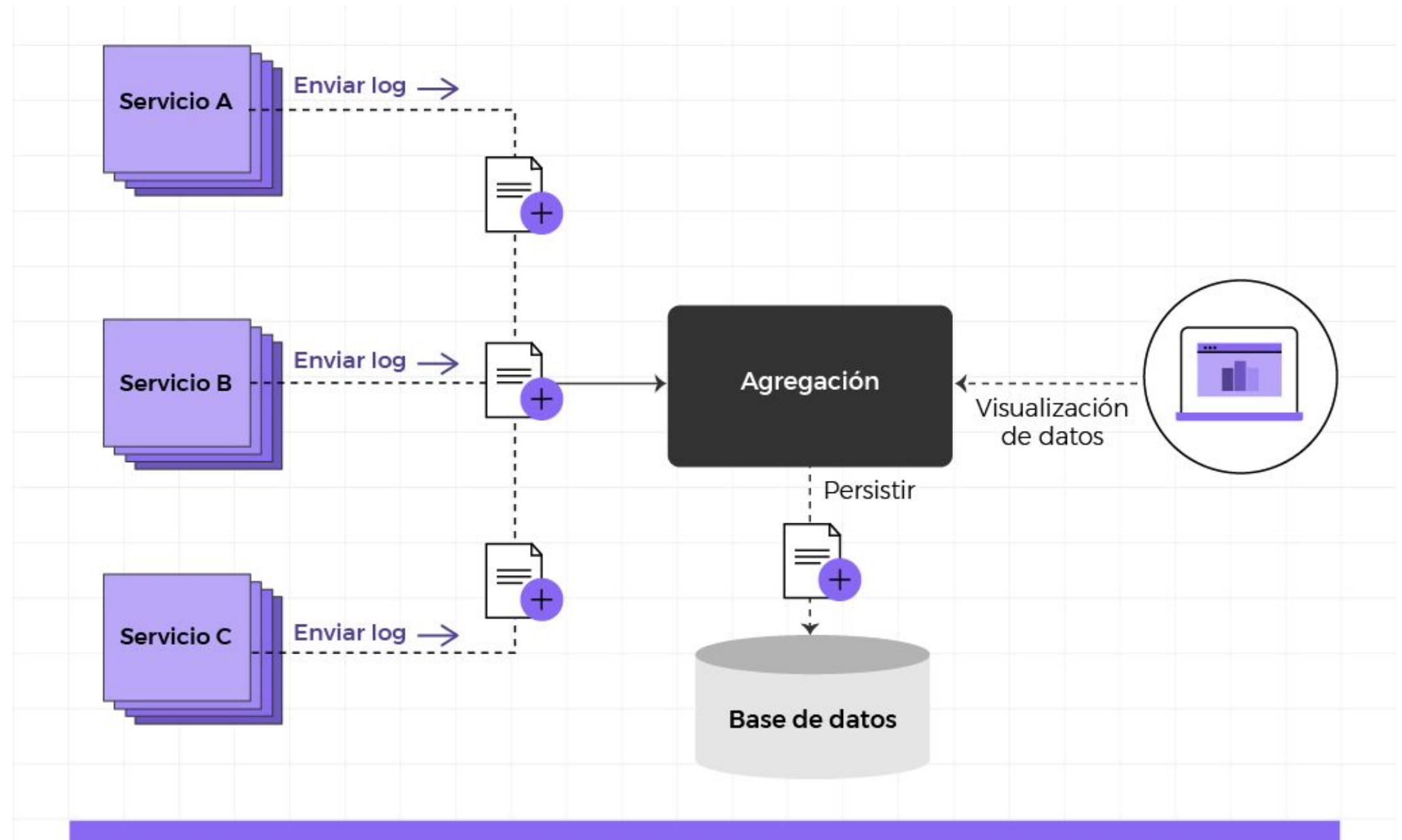


En el desarrollo de software,  
los logs son el mapa del  
tesoro para encontrar y  
solucionar problemas.

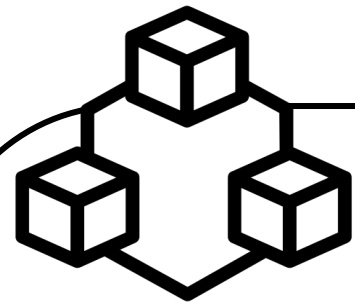


# Qué es Log Aggregation

- Patrón de diseño que permite centralizar los logs en una misma base de datos, que se podrá consultar cuando sea necesario.
- Viene a resolver el problema de la trazabilidad de ejecución de un servicio en sistemas distribuidos.

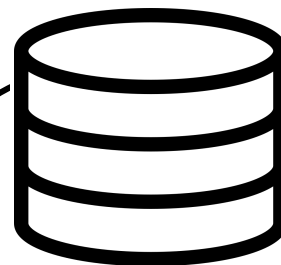


# Componentes



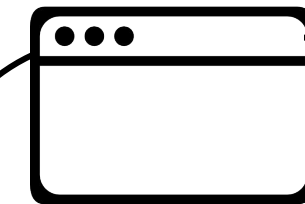
## Agregación

Cada microservicio genera logs y son absorbidos por el componente de agregación, que analiza los logs, los transforma y los envía al componente de persistencia.



## Persistencia centralizada

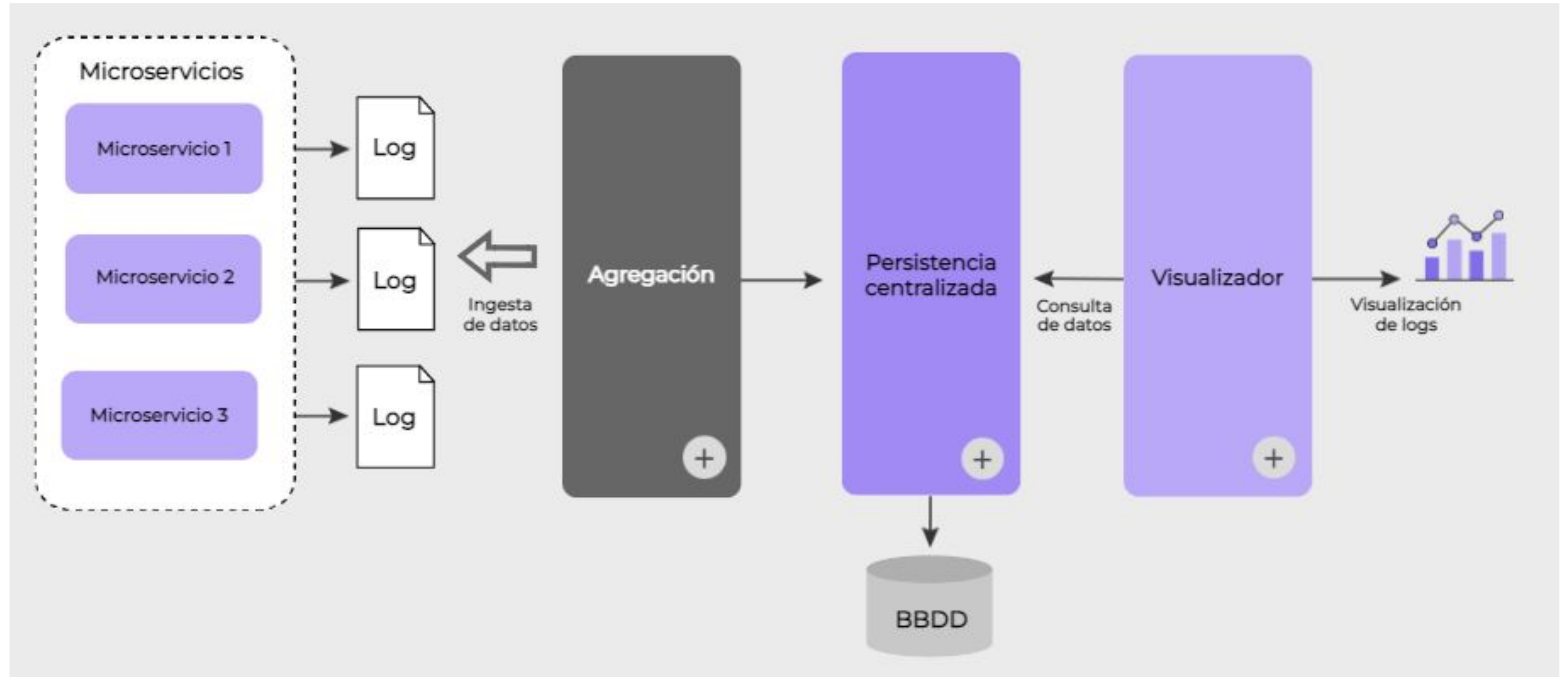
Este componente tomará el input recibido y lo almacenará en un repositorio de información (base de datos) centralizado.



## Visualizador de logs

Se pueden analizar los logs para determinar un patrón de negocio, como las APIs más invocadas de nuestra solución, o bien, determinar la causa de un error.

# Diagrama Log Aggregation



02

# Stack ELK



# Qué es

Es un stack de tecnologías (Elasticsearch, Logstash y Kibana) para ser utilizado en la gestión de agregación de logs implementando una solución para cada uno de los componentes del patrón Log Aggregation.



## Logstash

Es un pipeline que toma la información de un input —los logs de nuestra aplicación—, los transforma de acuerdo al criterio de filtro y envía esa información a Elasticsearch.



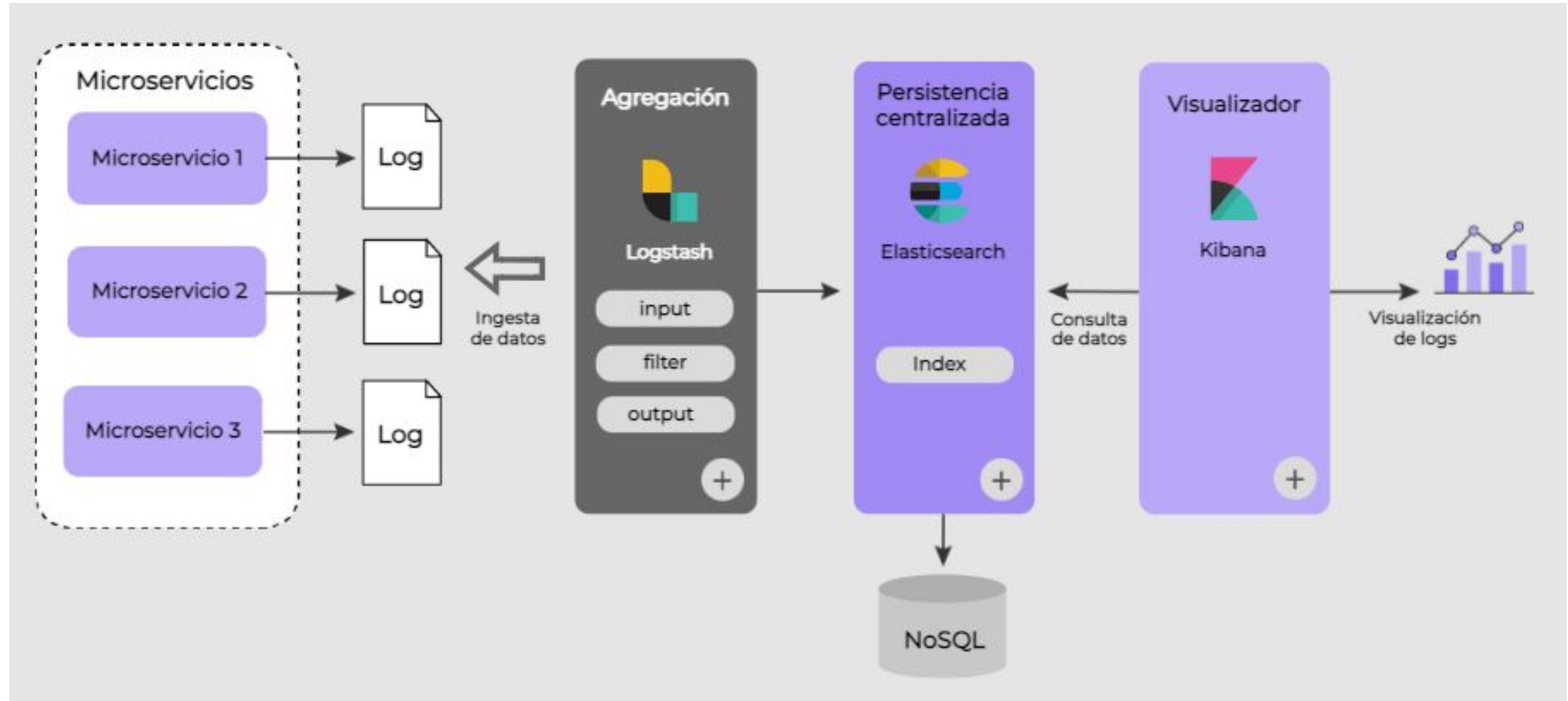
## Elasticsearch

Motor de búsqueda y análisis distribuido basado en Apache Lucene. Se utiliza para análisis de registros, búsqueda de texto completo, análisis empresarial, entre otros usos.



## Kibana

Este componente tomará el input recibido y lo almacenará en un repositorio de información (base de datos) centralizado.



03

# Implementación en Spring Boot

# Preparativos

Primero, vamos a descargar cada una de las herramientas. En esta guía utilizaremos la **versión 8.9.2** del stack ELK.

Seleccionamos el link de descarga según nuestro sistema operativo.

- [Logstash](#)
- [Kibana](#)
- [Elasticsearch](#)

Luego, editaremos algunas configuraciones de estos archivos.



# Configuraciones - Elasticsearch

En el archivo **elasticsearch.yml** dentro del directorio **/elasticsearch-8.9.2/config** agregamos la siguiente configuración:

- Desactivamos la seguridad (para evitar problemas con SSL en localhost) y el enrolamiento de nodos.

```
xpack.security.enabled: false  
xpack.security.enrollment.enabled: false
```

- Nombre de host al que Elasticsearch vinculará su servidor HTTP. Si se deja en blanco o con **0.0.0.0** solo escuchará las peticiones de **localhost**.

```
http.host: 0.0.0.0
```

Para ejecutarlo, abrimos una terminal en **/elasticsearch-8.9.2** y ejecutamos **./bin/elasticsearch**

[Link al archivo de configuración](#)

# Configuraciones - Kibana

En el archivo **kibana.yml** dentro del directorio **/kibana-8.9.2/config** agregamos la siguiente configuración:

- Todo el archivo estará comentado, salvo esta línea, que especifica la dirección del clúster de Elasticsearch al que Kibana se conectará:

```
elasticsearch.hosts: ["http://localhost:9200"]
```

Para ejecutarlo, abrimos una terminal en **/kibana-8.9.2** y ejecutamos **./bin/kibana**

[Link al archivo de configuración](#)

# Configuraciones - Logstash

En el archivo **logstash.conf** dentro del directorio **/logstash-8.9.2/config** agregamos la siguiente configuración:

```
input {
  file {
    path => "/ruta/directorio/de/logs/*.log"
    start_position => "beginning"
    sincedb_path => "/dev/null"
  }
}

output {
  stdout {
    codec => rubydebug
  }

  elasticsearch {
    hosts => ["http://localhost:9200"]
    index => "nombre-indice"
  }
}
```

- path tendrá la ruta de la carpeta donde se almacenarán los logs.
- start\_position indica desde dónde debe comenzar la lectura de los archivos de registro.
- hosts especifica los nodos a los que Logstash debe enviar los datos procesados.
- index será para indicar a qué índice se envían los logs. Pueden ser dinámicos.

Para ejecutarlo, abrimos una terminal en **/logstash-8.9.2** y ejecutamos **./bin/logstash -f ruta/a/logstash.conf**

[Link al archivo de configuración](#)



Dentro de nuestros microservicios no tendremos que agregar ninguna dependencia adicional. Únicamente utilizar un logger como puede ser **log4j**.

```
@RestController
public class PaymentController {

    2 usages
    private PaymentService paymentService;

    1 usage
    Logger log = LoggerFactory.getLogger(PaymentController.class);

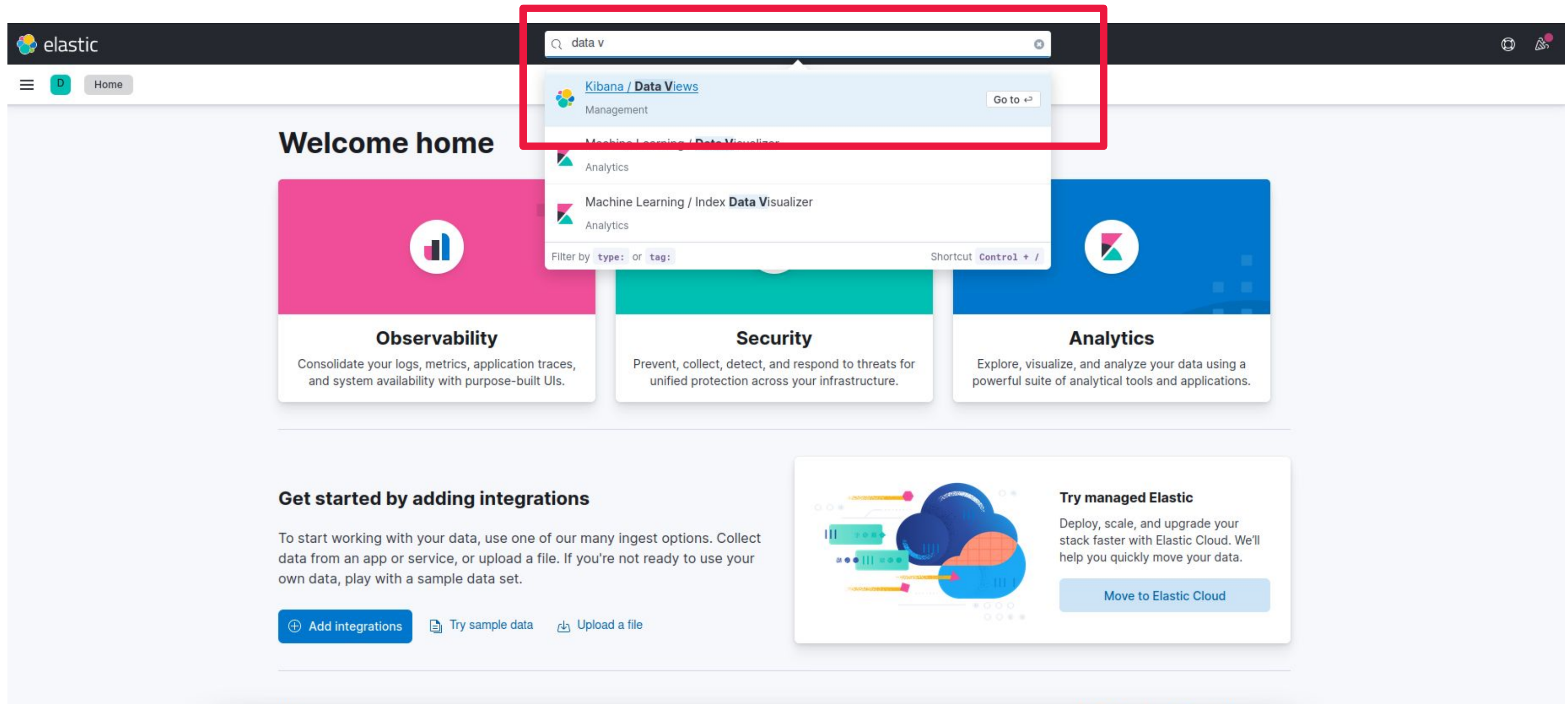
    no usages  EFFM00
    @GetMapping()
    public Payment getPayment(String id) {
        log.info("Intentando obtener payment con id: "+ id);
        return new Payment(id, amount: 999.99f, clientId: "178482", ownerId: "838283");
    }
}
```

Y en el application.properties especificar dónde se guardarán esos logs. Será la misma ruta que indicamos en Logstash.

```
logging.file.name=../logs_microservicios/payment-service.log
```



¡Listo! Cuando tengamos nuestras herramientas en ejecución, vamos a acceder al link de Kibana (<http://localhost:5601>), y en la barra de búsqueda escribiremos **data view**.



En Data View, la interfaz puede variar, según tengamos o no datos en Elasticsearch, pero lo importante es buscar la opción que diga “**create data view**” o similar. Al clickearlo se abrirá un modal para crear una vista.

**Create data view**

Name

Index pattern

example-\*

Timestamp field

Select a timestamp field

Show advanced settings

× Close

Save data view to Kibana

Your index pattern can match 1 source.

logs-generic-default Data stream

Rows per page: 10

Tendremos que seleccionar un índice del cual se tomarán los datos y opcionalmente un nombre.



Puede ocurrirnos que no veamos ningún índice disponible. Si es así, tendremos que crearlo manualmente.

Antes de crear nuestro data view en Kibana, debemos crear un index en Elasticsearch. Podemos hacerlo a través de su API y un curl:

**curl -X PUT "localhost:9200/nombre-de-mi-indice-000001?pretty"**

- El parámetro “?pretty” es opcional y sirve para formatear el output de la petición.
- La secuencia de números no es requerida pero es una convención en la nomenclatura de índices para llevar una secuencia.
- Debe ser el mismo nombre que indicamos en la configuración de Logstash.

Si recargamos la web de Kibana, ya podremos visualizar el índice y crear nuestro data view 🎉

**Create data view**

Name

Index pattern

example-\*

Timestamp field

Select a timestamp field

Show advanced settings

Your index pattern can match 2 sources.

logs-generic-default	Data stream
nombre-de-mi-indice-000001	Index

Rows per page: 10



Ya podemos empezar a generar logs en nuestros servicios, y los vamos a visualizar clickeando en le menú hamburguesa de la esquina superior izquierda > dentro del desplegable de Analytics > Discover.

The screenshot shows the Elastic Discover interface. At the top, there's a search bar with the text "Find apps, content, and more." and a search icon. Below the search bar, there's a navigation bar with a hamburger menu icon, a "Discover" button, and a "Save" button. The main area is divided into three sections: a left sidebar, a top filter bar, and a main content area. The left sidebar has a search bar for field names and a list of available fields: @timestamp, @version, event.original, host.name, log.file.path, and message. There are also empty fields and meta fields sections. The top filter bar has a search bar for KQL syntax and a "Refresh" button. The main content area shows 114 hits. A message box at the top of the main content area says "Get the best look at your search results" and "Take the tour". Below this, there's a table of log entries. The first entry is a warning log from the payment-service, mentioning a BeanPostProcessorChecker. The second entry is a warning log from the payment-service, mentioning a LoadBalancerEurekaAutoConfiguration. The third entry is a warning log from the payment-service, mentioning a LoadBalancerEurekaAutoConfiguration. The fourth entry is a warning log from the payment-service, mentioning a LoadBalancerEurekaAutoConfiguration. The fifth entry is a warning log from the payment-service, mentioning a LoadBalancerEurekaAutoConfiguration.

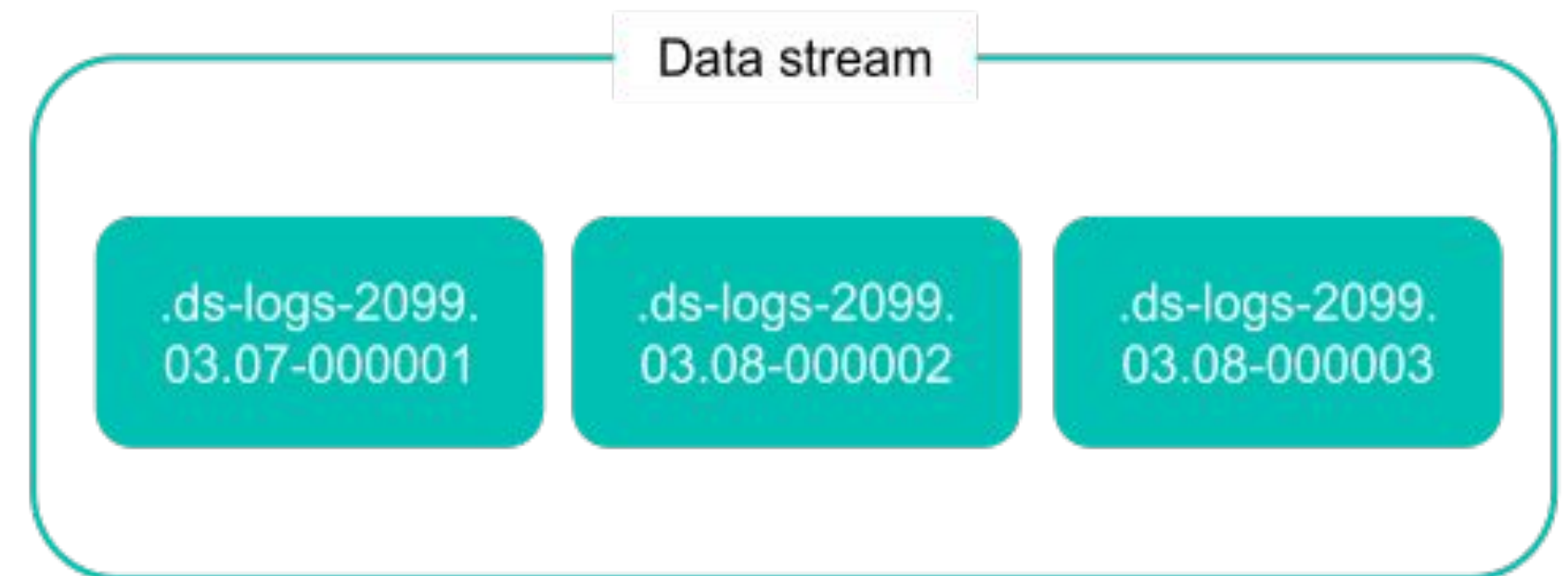
¡Y listo! Ya tenemos un manejo organizado de los logs de nuestros microservicios.

04

# Index Lifecycle Management

# Data stream, índices e ILM

- Data stream es una forma que tiene Elasticsearch de gestionar conjuntos de datos generados continuamente, por ejemplo registros de eventos, métricas, etc.
- Un índice es un segmento temporal dentro de este registro continuo y es la forma en que se fragmentan los logs almacenados por un data stream.
- Los data stream e índices forma parte del index lifecycle management (ILM o ciclo de vida del índice) que permite gestionar bajo qué condiciones mutan los índices. Ejemplo: activar un nuevo índice cuando un índice alcanza un cierto tamaño, crear un nuevo índice cada cierto tiempo.
- Los data stream por lo general se manejan automáticamente con su propia nomenclatura y comportamiento.

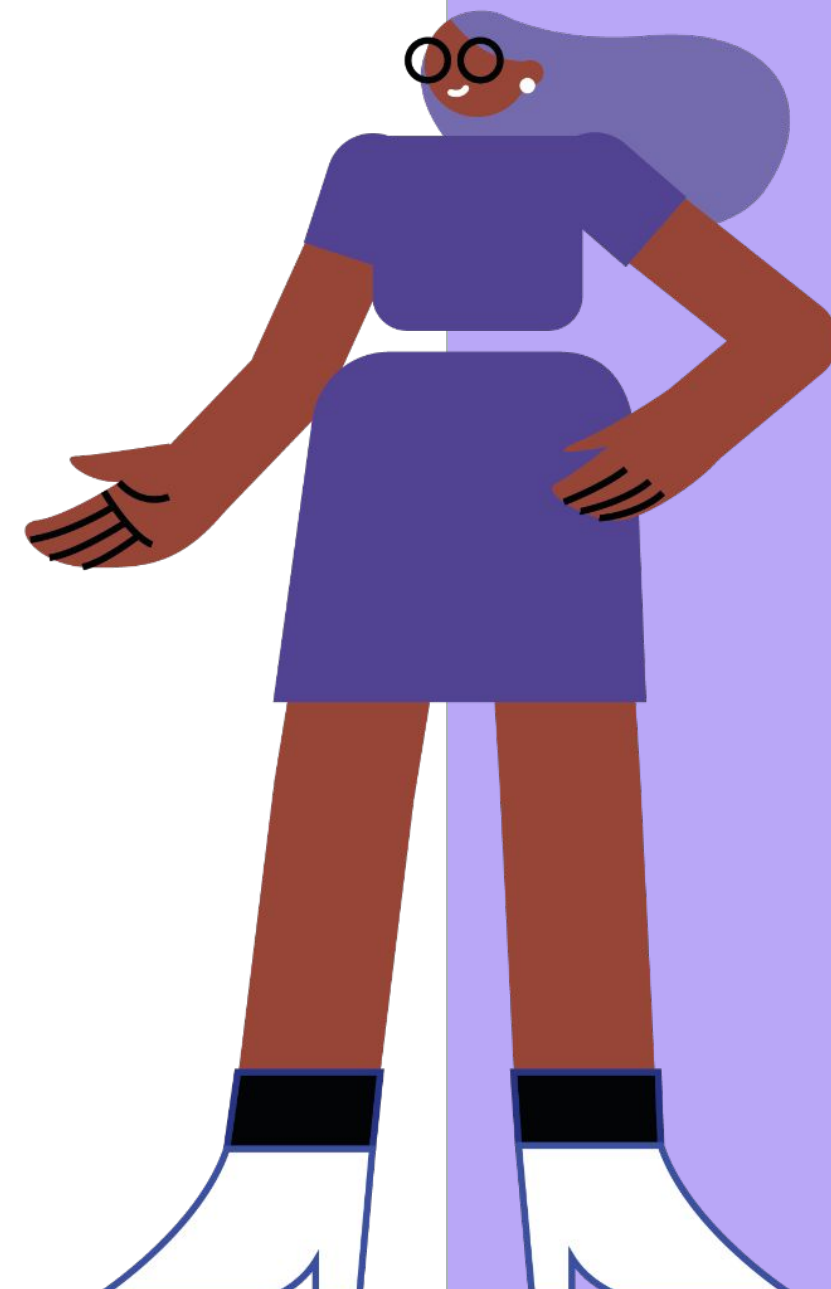


# Conclusiones

---

La utilización del patrón de agregación de logs junto con el stack ELK (Elasticsearch, Logstash, y Kibana) ofrece una solución poderosa y efectiva para la gestión y análisis de logs en entornos de sistemas distribuidos.

El stack ELK facilita la búsqueda, correlación y resolución de problemas, permitiendo a los equipos de operaciones y desarrollo tomar decisiones informadas y mejorar la eficiencia operativa. Con Elasticsearch como motor de búsqueda, Logstash para la ingestión y transformación de datos, y Kibana para la visualización intuitiva, el stack ELK se destaca como una herramienta valiosa para la monitorización y troubleshooting en entornos modernos de tecnología.



¡Muchas gracias!