

# Proyecto de Fin de Bootcamp

Por Jose Luis Zaragoza Ortega

## Contenidos

---

1. Arquitectura del sistema
2. Definición de la arquitectura cloud
3. Descripción del ciclo de vida

## Arquitectura del sistema

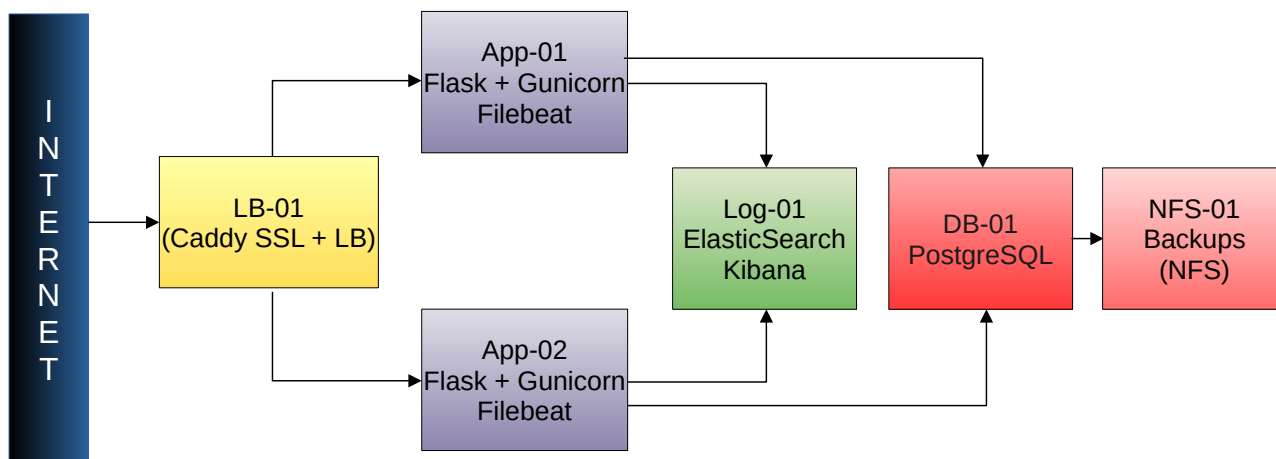
---

El sistema contiene los siguientes elementos:

- Un load balancer configurado con Caddy para garantizar que los certificados TLS se aprovisionen automáticamente y los mantenga renovados sin ninguna intervención por nuestra parte.
- N instancias de la aplicación suministrada. Cada instancia contendrá una copia de Filebeat para mantener los logs.
- Un host con una copia de ElasticSearch y Kibana, el cual almacenará los logs.
- Una base de datos (PostgreSQL).
- Un NFS para backups diarios.

Esta implementación no es perfecta, ya que entre otras cosas contiene un SPoF al solo tener un único load balancer, pero he tenido que realizar algún compromiso para poder enviar una entrega funcional a tiempo.

### *Diagrama de la Implementación*



# Definición de la arquitectura cloud

Primero definiremos un inventario de recursos de AWS:

## Red

VPC	CIDR 10.0.0.0/16
Subred publica	10.0.1.0/24
Subred privada	10.0.2.0/24

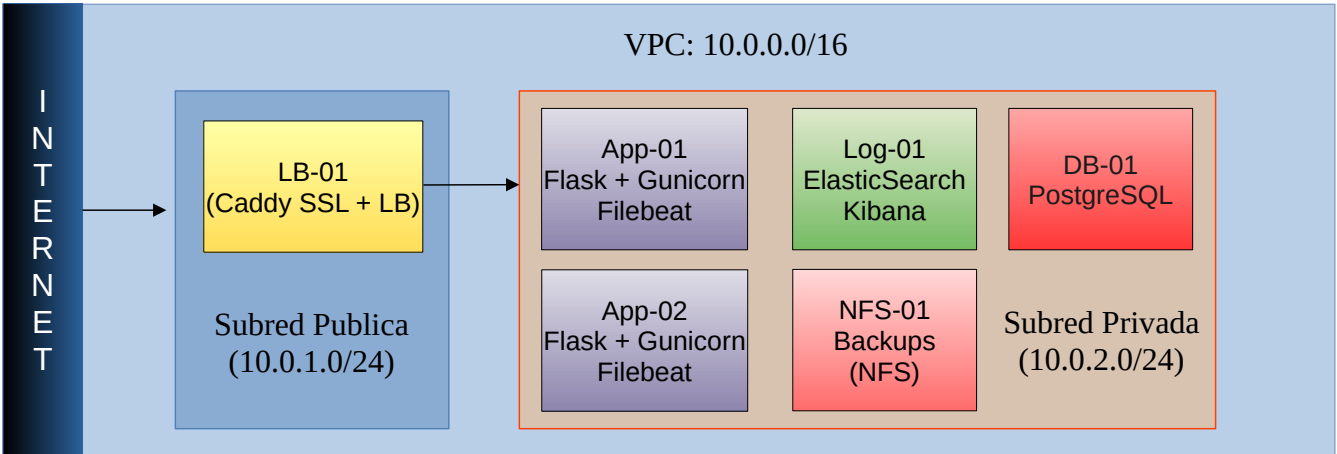
## EC2

LB-01	T3.Micro
APP-n	T3.Micro
DB-01	T3.Small
NFS-01	T3.Micro (50GB EBS)
LOG-01	T3.Medium

## Security Groups

SG-LB	INBOUND: 80, 443 (internet)
SG-APP	INBOUND: 5000 (SG-LB)
SG-DB	INBOUND: 5432 (SG-APP)
SG-NFS	INBOUND: 2049 (SG-SB)
SG-LOG	INBOUND: 9200 (SG-APP), 5601 (admin)

## Diagrama del Cloud



# Ciclo de vida del software

---

Optaremos por el modelo Github Flow con el siguiente modelo de ramas:

- Main: Siempre desplegable a producción.
- Feature-X: Desarrollo de nuevas funcionalidades.

## ***Flujo***

1. Creación de rama Feature desde Main.
2. Desarrollo del Feature.
3. Abrir Pull Request.
4. Paso de tests.
5. Code review.
6. Merge a Main.
7. Despliegue.

## ***Entornos***

- Local: Desarrollo con Docker Compose para ramas Feature.
- Producción: Despliegue de ramas main en el entorno de AWS.

El desarrollo se realizará puramente en el entorno local gracias a un container de Docker. Este container contendrá una serie de herramientas para ejecutar las pruebas y interactuar con el repositorio principal. Además, contendrá un container separado para la base de datos.

## ***Pruebas***

- Pytest: Pila de pruebas unitarias.
- Flake8: Linting, verificación de la calidad del código.

## ***Estrategia de despliegue***

El software del servidor será desplegado en nuestros EC2 en forma de una aplicación dockerizada. Enviaremos los cambios a una instancia a la vez para asegurarnos de que no se degrada el servicio (rolling update).

## ***Monitorización***

Se realizará un seguimiento de la aplicación con el stack de Filebeat + Elasticsearch + Kibana, y también se realizarán health checks automáticos con Caddy.