

A photograph of a server room with rows of server racks. The racks are illuminated with blue light, and the floor has a grid pattern. The perspective is looking down a long aisle towards a bright doorway at the end.

# SERVERLESS

Ismael Alba Areces (UO270176)  
Lucía Gutiérrez Búa (UO250755)  
Rubén Pardiño Iglesias (UO271397)

# Índice



¿Qué es Serverless?

¿Por qué Serverless?

¿Cuándo utilizar Serverless?

¿Cómo se utiliza Serverless?

Resumiendo

Referencias

# ¿Qué es Serverless?

- FaaS (Functions-as-a-Service)
- BaaS (Backend-as-a-Service)

**Serverless  
Computing**

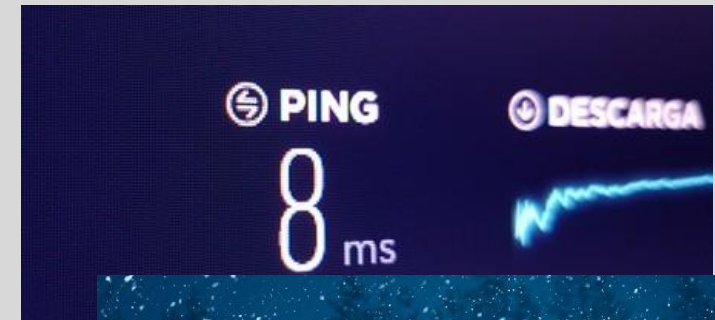


# ¿Por qué Serverless?

- Razones principales por las que los desarrolladores optan por Serverless:
  - Reducción de costes
  - Desarrolladores centrados en la lógica de negocio
  - Escalabilidad de las aplicaciones
  - Otras como velocidad para lanzar el producto al mercado o mejoras de rendimiento
- **Mayores ventajas de Serverless:**
  - Arquitectura basada en eventos
  - Reducción de costes
  - Velocidad de desarrollo
  - Flexibilidad para escalar la aplicación
  - Rendimiento de la aplicación

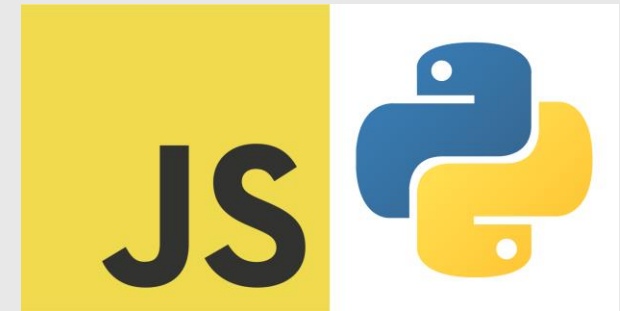
# ¿Cuándo utilizar Serverless?

- “Serverless no funciona en aplicaciones que necesiten pocas latencias”
  - Un 39% tienen un alto tráfico de peticiones
- “Cold starts”
  - 32% tienen latencia en todas sus funcionalidades
  - 28% tienen latencias parciales
- “Son inadecuadas para tareas que manejen una gran cantidad de datos”
  - 69% utilizan un volumen de datos <10MB



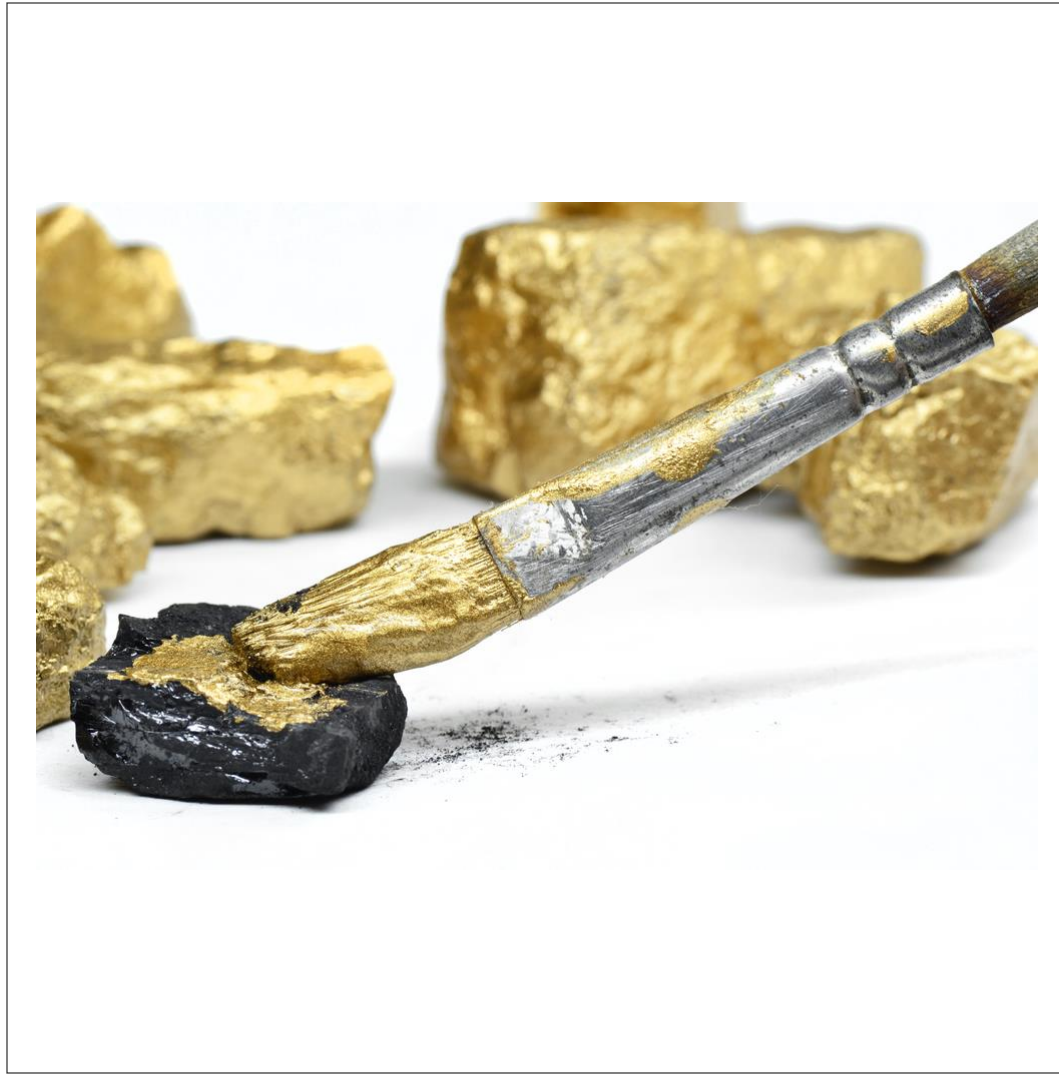
# ¿Cómo se utiliza Serverless?

- ¿Cuál proveedor a utilizar? ¿Qué lenguaje utilizar? ¿Javascript, Python?... ¿Ruby?
  - Dependiendo del lenguaje de programación podemos encontrarnos con que un arranque en frío de la función tarde demasiado para él cliente.
  - Proveedores como AWS que optimizan las funciones guardándolas en "prewarm".
- Backend
  - Qué hacer con los datos.
  - Bases de datos descentralizadas y ya administradas por el proveedor del servicio en la nube, (DynamoDB).



**DynamoDB**





# No es oro todo lo que reluce...

- Rendimiento
- Límites de recursos
- Monitoreo y depuración
- Seguridad e intimidad
- Dependencia de un proveedor

# RESUMIENDO

## Motivators



47% Save Costs

34% Built-in scalability

34% No operations

## Application Type



42% Core functionality

39% Utility functionality

16% Scientific workload

## Deployment Platform



80% AWS

10% Microsoft Azure

8% Private Cloud

## Programming Languages



42% JavaScript

42% Python

12% Java

## Integrated BaaS Solutions



61% Storage

48% Database

38% Messaging



# Referencias

- <https://martinfowler.com/articles/serverless.html> ,© Martin Fowler
- <https://arxiv.org/pdf/2009.08173.pdf>, © 2020 IEEE
- <https://squadex.com/insights/what-is-serverless/>, JANUARY 17, 2019|IN ARTICLE|BY YEVHEN DUMA



PREGUNTAS