# Explicar o conceito de Serverless. (Overview).

## O que é?

* + - Trata-se de um modelo de desenvolvimento em nuvem, sem a necessidade de gerenciar uma infraestrutura.

## Em que ajuda?

* + - Elimina a necessidade de gerenciar uma infraestrutura e sua escalabilidade.
    - Custo mais baixo, seguindo o modelo de “pague pelo uso”.
    - Escalabilidade automática baseada em parâmetros de configuração.

## Em que atrapalha?​

* + - Dependência de terceiros, tendo que seguir as regras do fornecedor do serviço (Amazon, Microsoft e Google).
    - Sem controle livre do hardware fornecido.
    - Custo alto para serviços de uso constante.
    - “Cold Start”, em situações em que a aplicação fique ociosa, existe a demora de acesso, pois os serviços são iniciados.

# Overview sobre as frentes estudadas.

## lambda

### O que é?

* + - * Lambda se trata de um serviço da AWS que permite efetuar a execução de código sem a necessidade de prover e gerenciar um servidor. É Utilizado a arquitetura de função como serviço (Function as a service - FaaS).
      * Algumas alternativas fora da AWS são Google Cloud Functions, Azure Functions, Netlify Functions e [Serverless Framework](https://www.serverless.com/framework/docs/).

### Em que ajuda?

* + - * Elimina a necessidade de gerenciar infraestrutura;
      * Escalabilidade continua, Lambda executa o código em paralelo, sendo cada processo iniciado individualmente;
      * Custo, sendo cobrado apenas pelo tempo de utilização, eliminando custos de máquina ociosa;
      * Performance consistente, Lambda permite otimizar a escalabilidade vertical por código, por exemplo determinando a quantidade de RAM utilizada em certo código.

### Em que atrapalha?​

* + - * Estado do processo, uma vez criado para realizar um processamento, este estado não é preservado para futuras chamadas, necessitando a utilização de uma base de dados para armazenar estados;
      * Negação de serviço (Denial of Service - DoS), Lambda possui um limite de 1000 execuções concorrentes por conta, em situações de teste, enquanto produção esteja ativa, existe o risco de causar DoS em produção;
      * Limite de duração, por padrão, Lambda pode abortar um processo caso esteja em execução por mais de 5 minutos, podendo cancelar processo como conversão de vídeo;
      * Latência de inicialização, Alguma situações podem demorar para iniciar o processo, principalmente em funções utilizando JVM (Java Virtual Machine);
      * Entrega, como cada processo deve se encontrar em seu próprio arquivo, a entrega deve ser individual, ou seja, em situação onde exista 20 funções, cada uma deve ser construída e entregue individualmente.

## EventBridge

### O que é?

* + - * Trata-se de um barramento de eventos *serverless*, utilizado em aplicações orientadas por eventos.
      * Ao utilizar eventos, é retirada a necessidade de comunicação direta entre APIs.

### Em que ajuda?

* + - * Utilizar na criação de sistema orientado por evento: com o *EventBridge*, os Serviços de destino não precisam saber da origem do evento por estarem publicados diretamente.
      * Permite a utilização de esquemas dos eventos gerados, mantendo padronizado.
      * Mensagens podem ser arquivadas e reutilizadas.

### Em que atrapalha?​

* + - * A ordem de mensagens não é mantida.
      * Existem alternativas como SNS e SQS, que são serviços de mensageria mais simples.

## Aurora serverless

### **O que é?**

* + - * Trata-se de um banco de dados relacional na nuvem compatível com bancos populares como MySQL e PostgreSQL.

### **Em que ajuda?**

* + - * Compatível com MySQL e PostgreSQL;
      * Mais rápido que MySQL e PostgreSQL;
      * Disponibilidade, recuperando de falhas de armazenamento de forma transparente;
      * Seguro.

### **Em que atrapalha?**​

* + - * Tecnologias de compatibilidade atrasadas em comparação aos bancos mais populares;
      * Sem possibilidade de desenvolvimento online;
      * Instâncias pequenas e médias possui número de conexões e recursos limitados;

# Cenários em que esta tecnologia seria recomendada (ou que não seria recomendada). ​

## Lambda

* + - Recomendado para operações simples, como processamento de dados.
    - Não recomendado para operações complexas envolvendo vários modelos e regras de negócio.

## EventBridge

* + - Recomendado para comunicação entre serviços de forma assíncrona.
    - Não recomendado quando ordem seja importante, ou para eventos simples.

## Aurora

* + - Recomendado para websites onde o fluxo pode variar.
    - Não recomendado para uso interno ou com muitas conexões.

# Falácias, boas práticas, práticas não recomendadas e erros comuns aos utilizar a tecnologia​

## Falácias

* + - O usuário não precisa se preocupar com a segurança da aplicação e dados.
    - São mais difíceis de serem utilizados.

## Boas práticas

* + - Aplicar as regras de segurança sugeridas pelos serviços
    - Adicionar serviços de monitoramento para verificação da saúde e custos dos serviços
    - Controle de funcionamento de serviço, como desligamento de serviços e redução de escala.

# Referências para quem quiser se aprofundar no assunto

* + Documentações da AWS estão repleta de informações sobre como utilizar as ferramentas, além de vários crash courses presente pela internet.
  + [https://aws.amazon.com/pt/eventbridge](https://aws.amazon.com/pt/eventbridge/)
  + <https://aws.amazon.com/pt/rds/aurora>
  + <https://aws.amazon.com/pt/lambda/>
  + [A crash course on Serverless with AWS - Building APIs with Lambda and Aurora Serverless - DEV Community](https://dev.to/adnanrahic/a-crash-course-on-serverless-with-aws---building-apis-with-lambda-and-aurora-serverless-3fn3)