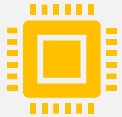
Conhecendo ferramentas serverless na AWS

Leandro Damascena – AWS Certified

O que é serverless?



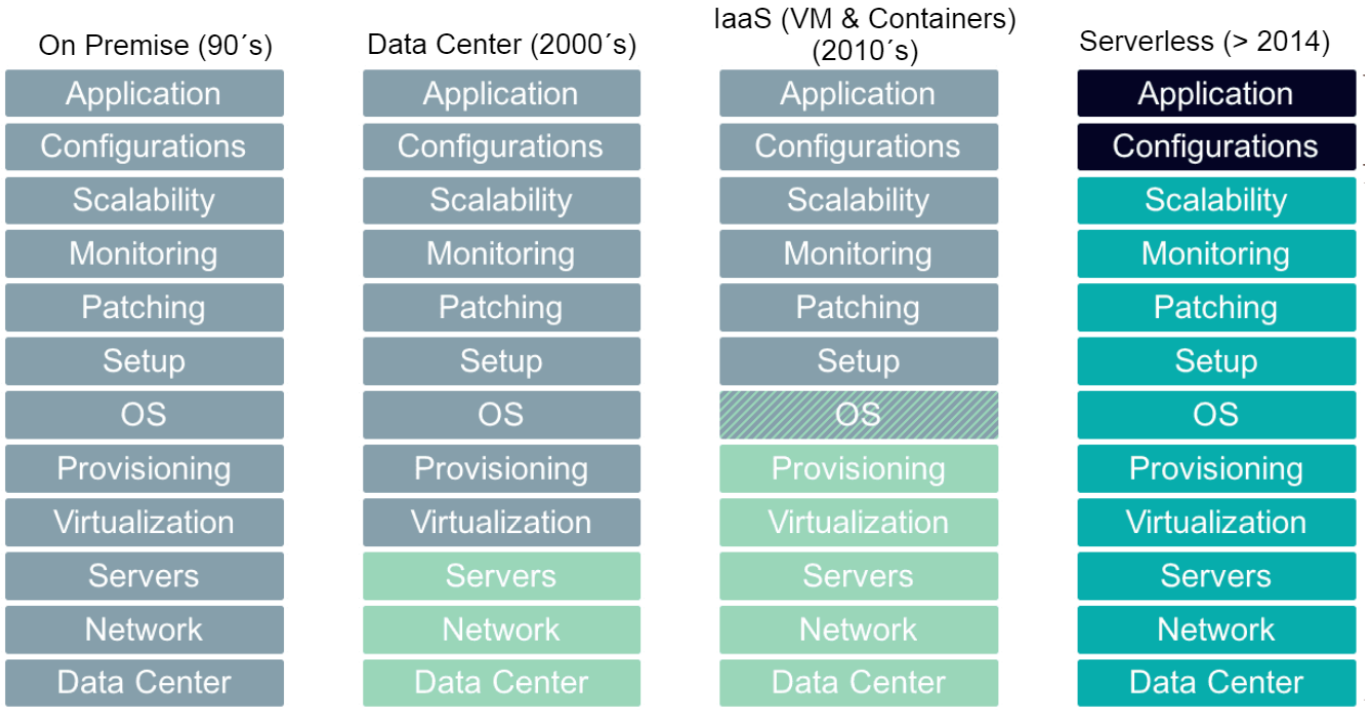
É uma arquitetura nativa de computação em nuvem que permite utilizar diversos recursos computacionais sem se preocupar com o gerenciamento da infraestrutura, escalabilidade e alta disponibilidade.

Uma arquitetura serverless vai além de apenas FaaS (*Function as a Service*).

Diminui o tempo de provisionamento da infraestrutura.

Custa apenas o que for utilizado no período... Mas é preciso ter cuidado!

A evolução - do metal ao serverless



E que benefícios tenho ao adotar uma arquitetura serverless?

Flexibilidade de alocação de novos recursos e capacidade computacional.

Redução do risco de falhas de hardware.
Redução dos riscos de falhas de segurança do sistema operacional e runtime.

Possibilidade de utilizar as melhores práticas para a construção de sistemas escaláveis, desacoplados, com tolerância a falhas e mais resilientes.

Redução do *Time to Market* e maior produtividade da equipe.

Comparativo de alguns fatores entre as arquiteturas

	ON-PREM	VM	CONTAINERS	SERVERLESS
Tempo de provisionamento	Semanas/Meses	Minutos	Segundos/Minutos	Milissegundos
Cobrança	CapEx	Horas	Minutos / Horas	Blocos de gb / milissegundos

E como podemos nos beneficiar desta arquitetura na AWS?!

Conhecendo as ferramentas serverless da AWS e algumas possibilidades de aplicação!

Ferramentas serverless AWS

Computação:
Lambda, Lambda@Edge, Fargate

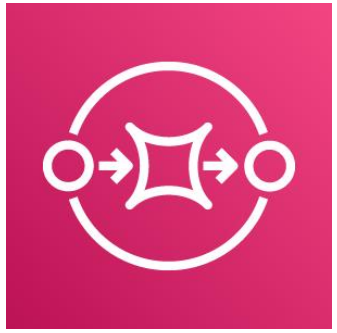
Storage:
S3, EFS

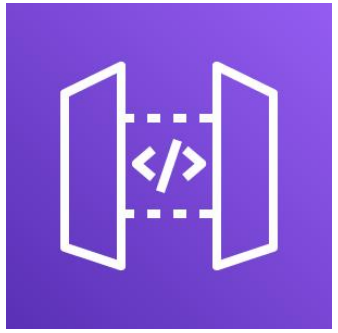
Data Storage:
DynamoDB, Aurora Serverless, RDS Proxy

API:
API Gateway

Aplicação:
SNS, SQS, AppSync, EventBridge

Outras:
IoT Tools, Kinesis, Athena, Step Functions, Cognito, Rekognition, entre outras.





O que iremos detalhar:

AWS Lambda
Amazon S3
Amazon API Gateway
Amazon DynamoDB
Amazon SQS
Amazon SNS

AWS Lambda

- Diversos runtimes:
  - Python
  - Go
  - Node.js
  - Java
  - Ruby
  - .NET Core

- Paga apenas pelo tempo de execução;

- Gerenciamento de recursos como uso de memoria, timeout e concorrência;

- Escalabilidade quase que infinita;

- Integração com diversos outros serviços da AWS através de Events;

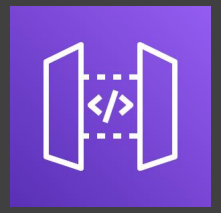
- Gerenciamento de versões e alias;

- Execução de códigos de eventos do Cloudfront através do Lambda@Edge;

- 100% integrado ao IAM.

Amazon S3

- Serviço de armazenamento de objetos (não é um FileSystem) com diversas classes de armazenamento;
- Baixíssimo custo para armazenar objetos;
- Performance e escalabilidade quase que infinitos. Podendo, inclusive, hospedar sites estáticos/imagens e integrá-los ao CloudFront;
- Durabilidade de 99,999999999% (11 9s) dos objetos;
- Versionamento de arquivos e MFA para deleção de arquivos;
- Politicas de *lifecycle* para redução de custos de arquivos pouco acessados;
- Configuração de politica de acessos restritas a usuários, grupos ou todos;
- Serviço em conformidade com padrões como HIIPA, PCI-DSS, entre outros;
- Politica de Vault Lock e WORM;
- Integração com o AWS Lambda, SQS e SNS.



Amazon API Gateway

- Serviço de publicação de APIs;
- APIs Rest, HTTP e WebSocket;
- Compatibilidade com Swagger e OpenAPI;
- Total controle sobre os diversos métodos e stages das APIs;
- Integração com o AWS Lambda para a criação de um robusto conjunto de API;
- Criação de authorizers para controle de acesso a API;
- Mapping templates de request e response para transformações de dados;
- Planos de uso para controle de chamadas por cliente/aplicação;
- Cache das chamadas mais frequentes.

Amazon DynamoDB

- Banco de dados NoSQL;
- Pode ser usado no modelo Key / Value ou Documento;
- Armazenamento e escalabilidade quase que ilimitados;
- Redundância dos dados;
- Suporte a transações ACID;
- Possibilidade de utilizar modo provisionado ou escalável de recursos;
- Tabelas globais para menor latência;
- Suporte a query e scan;
- Integração com o AWS Lambda;
- Definição de TTL para expiração automática de registros.

Amazon SQS

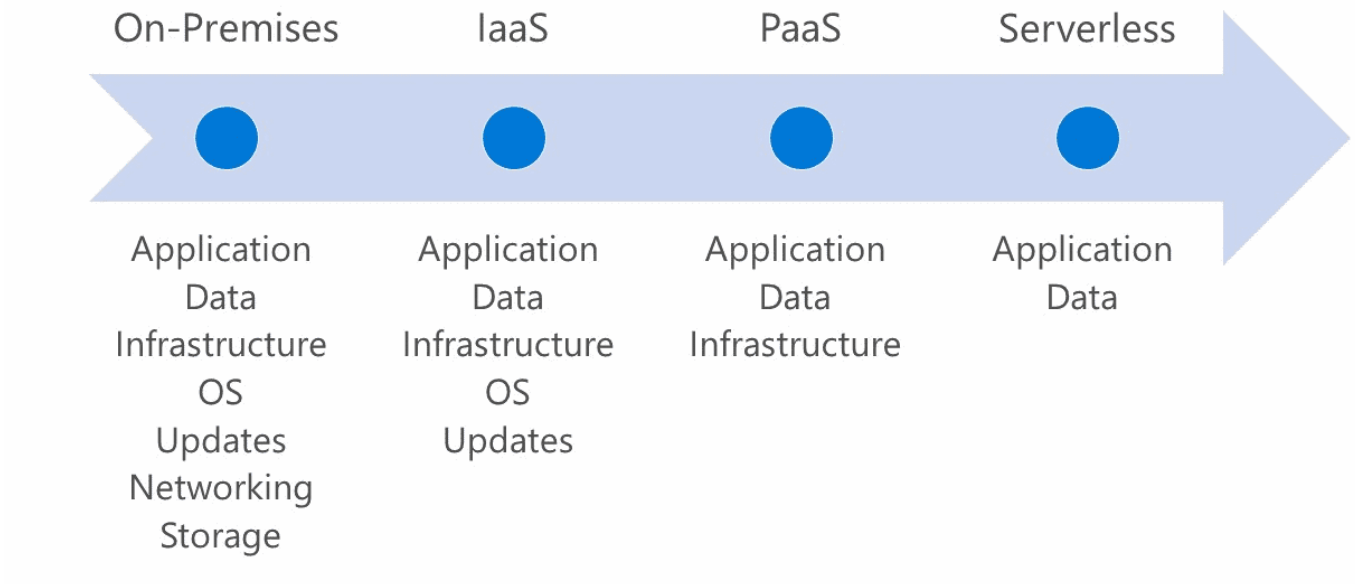
- Serviço de gerenciamento de filas e mensagens;
- Alto throughput no envio de mensagens;
- Redundância e escalabilidade infinitas;
- Filas FIFO com garantia de entrega apenas uma vez e na ordem;
- Possibilidade de criptografar dados na fila;
- Integração com o AWS Lambda.

Amazon SNS

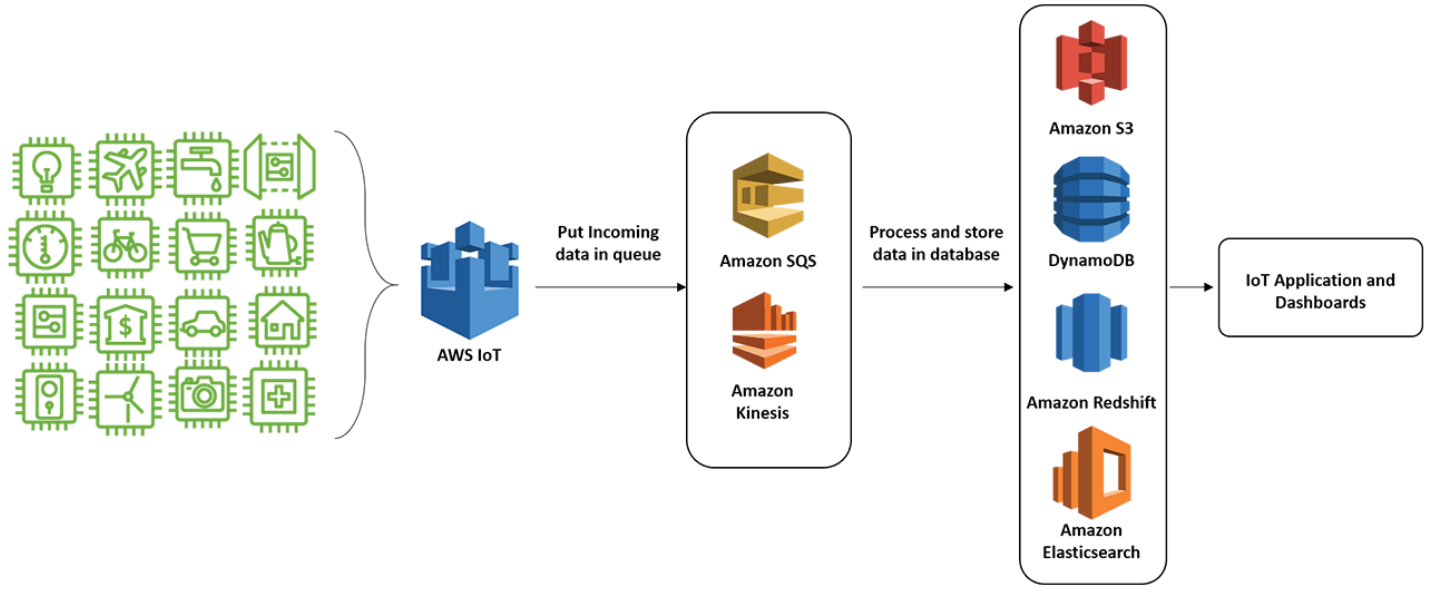
- Serviço de envio de notificação;
- Possibilidade de enviar notificações para email, http endpoints, SMS, entre outros;
- Envio de notificações para 1 tópico com diversas assinaturas;
- Integração com o AWS Lambda;
- Baixo custo no envio de notificações.

E como transformo estas ferramentas em soluções reais?

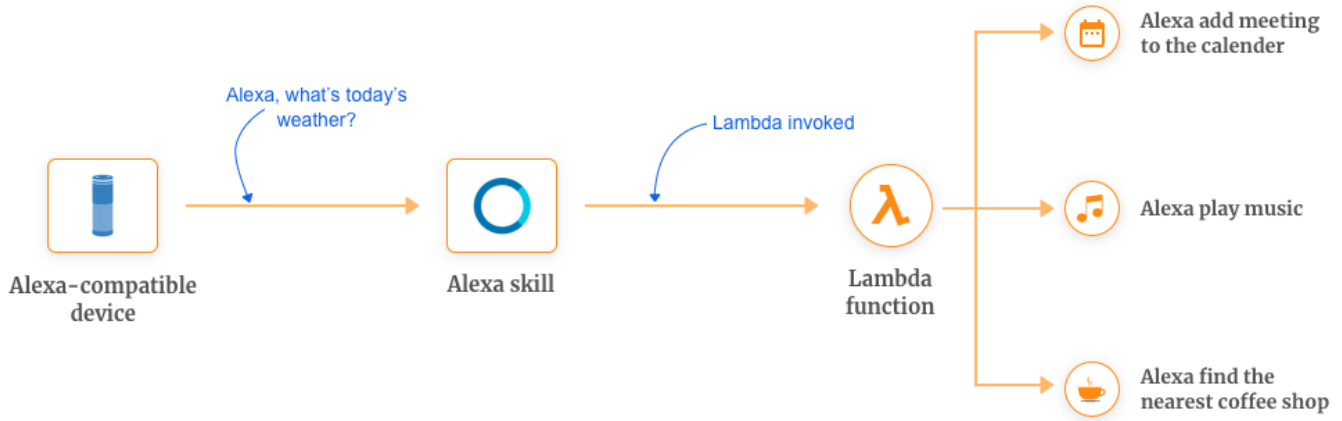
Em um pipeline de análise de dados



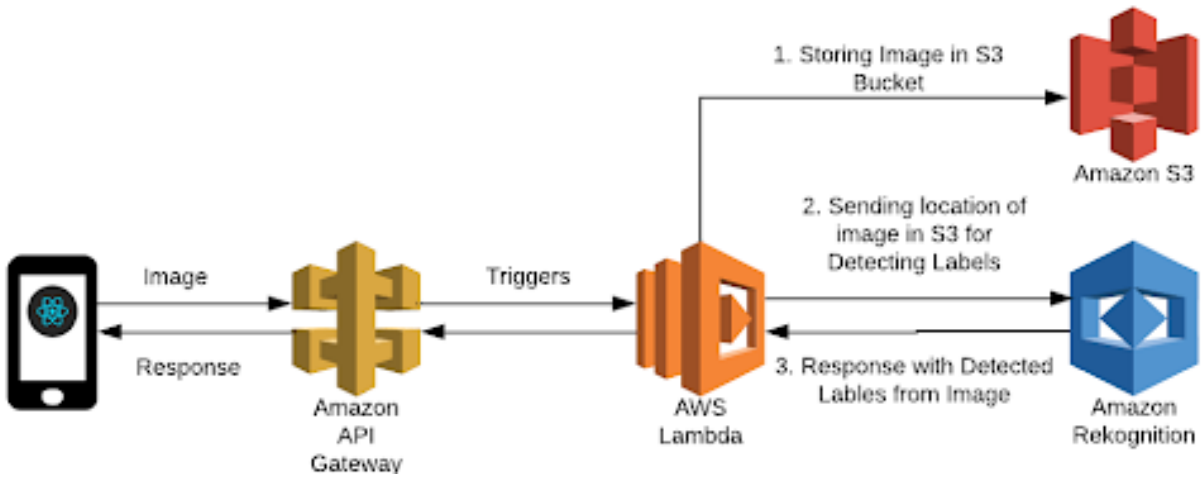
Em um gateway de dispositivos IoT's



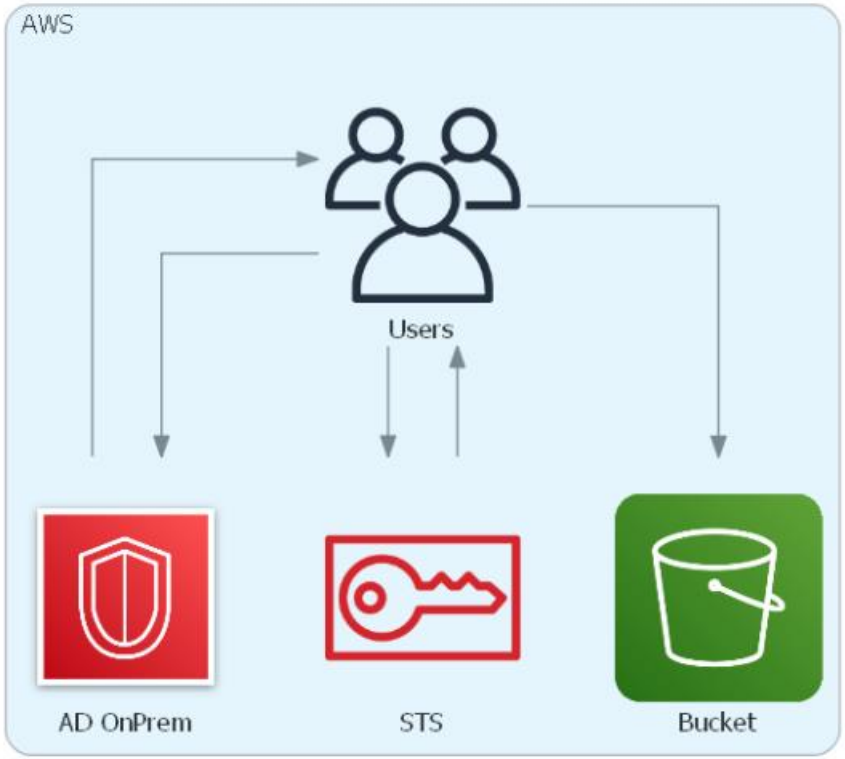
Em uma skill da Alexa



Em um sistema de reconhecimento de faces



Em um file sharing



S3 Bucket

E serverless serve pra qualquer aplicação? Posso usar em tudo?

A resposta é não!
É preciso ter cuidado e fazer a melhor escolha.

Em alguns casos o uso de serverless, especialmente em funções, não tem vantagens:

- Complexas regras de negócios;
- Funções que demorem muito para executar;
- Quando você precisa ter total controle do ambiente;
- Custom runtimes que precisem de muita adaptação;
- Quando a aplicação requer uso de GPU.

Boas praticas para o uso de arquiteturas serverless.

Adote algumas boas praticas para o uso de arquiteturas serverless:

- Segurança SEMPRE em primeiro lugar. Lembre-se que no ***Shared Model Responsibility*** você é responsável pela segurança da sua aplicação. Evite roles com muita permissão e utilize uma role por função, use e aplique SEMPRE o conceito *least privilegie;*

- Utilize funções Async e configure DLQ para elas;

- Otimize sempre seu código. Tire tudo o que for desnecessário para rodar a aplicação, lembre-se que você paga por tempo de execução;

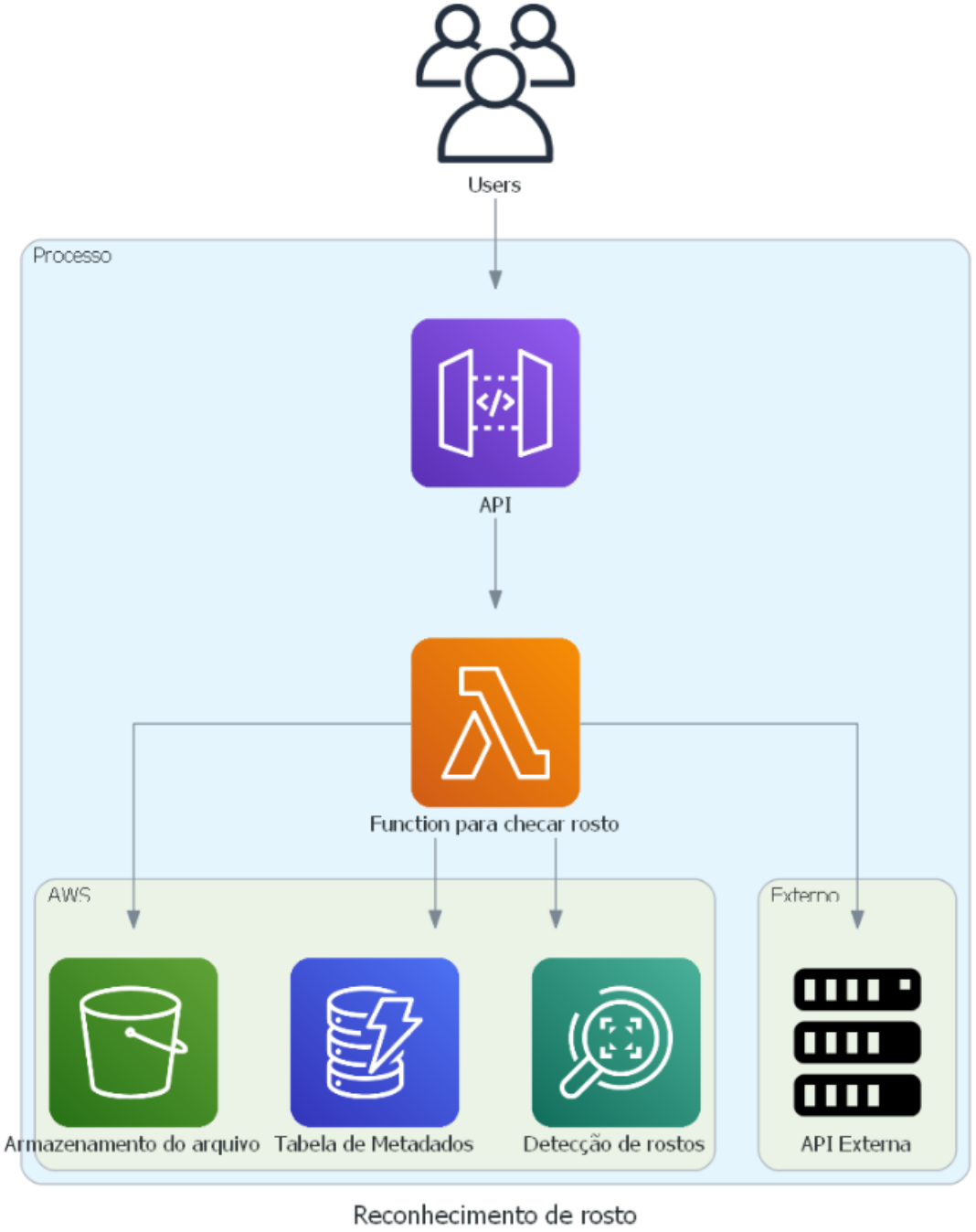
- Monitore -> Corrija -> Evolua ... Monitore -> Corrija -> Evolua ... Monitore -> Corrija -> Evolua;

- Automatize os deploys;

- Leia o guia *Well-Architected Framework* da AWS.

Hands-on:
Reconhecimento de rostos utilizando o framework AWS SAM.

Hands-on de Reconhecimento de rostos



CUSTOS

Precisamos falar sobre este tema!

Cuidados para não ter um susto com a conta.

Adote algumas boas praticas antes de migrar para uma arquitetura serverless:

1. Desenhe sua arquitetura para ter certeza dos serviços que irá utilizar e desacatar o que não for necessário;
2. Utilize a calculadora de custos da AWS;
3. Todas as contas tem direito a um período de *Free Tier*, faça uso disso;
4. Tente comprar um ou mais pacotes de *Saving Plans;*
5. Não tenha medo de identificar que serverless fica inviável para implantar na sua empresa e que container ou EC2 são melhores opções;
6. Crie budgets reports e receba alertas por email;
7. Configure TAGS em toda sua arquitetura, dessa forma fica fácil identificar o uso por recurso e ajustar o que for necessário;
8. Crie contas separadas para desenvolvimento e produção, monitore elas;
9. Lembre-se que o dinheiro da sua empresa/cliente é o seu dinheiro também.

Links:

1. AWS Calculator - https://calculator.aws/#/
2. Lambda x EC2 - https://servers.lol/
3. Serverless costs - https://cost-calculator.bref.sh/
4. AWS Free Tier - https://aws.amazon.com/pt/free/

Custos do nosso hands-on.

Sem levar em consideração o benefício do *Free Tier* da sua conta, o custo aproximado seria o seguinte:

VALORES APROXIMADOS	
Requisições por mês	**Custo (em dólares)**
1000	$ 1,20
10000	$ 13,50
100000	$ 106,00
1000000	$ 1.090,00
10000000	$ 11.800,00

OBRIGADO!

REPOSITÓRIO DO MEETUP

https://github.com/awsugce/meetups-projects

Links interessantes e contatos.

Frameworks serverless:
- Middy (Node.js) - https://github.com/middyjs/middy
- Chalice (Python) - https://github.com/aws/chalice
- Zappa (Python) - https://github.com/Miserlou/Zappa
- Sparta (Go) - http://gosparta.io/
- Bref (PHP) - https://bref.sh/
- Claudia.js (Javascript) - https://claudiajs.com/
- AWS SAM - https://docs.aws.amazon.com/serverless-application-model/

Geral AWS:
- AWS Well Architected Framework - https://d1.awsstatic.com/whitepapers/architecture/AWS_Well-Architected_Framework.pdf

Bons repositórios Github
- Lambda Layers, utilitários, segurança, monitoramento, runtimes - https://github.com/mthenw/awesome-layers
- Diagramas como código - https://github.com/mingrammer/diagrams
- Tudo sobre AWS - https://github.com/donnemartin/awesome-aws
- Guia sobre AWS - https://github.com/open-guides/og-aws
- Segurança na AWS - https://github.com/toniblyx/my-arsenal-of-aws-security-tools
- Cloud Discovery - https://github.com/Cloud-Architects/cloud-discovery
- FAQ AWS - https://github.com/Dreadstar22/AWS-FAQ/blob/master/FAQ.md

Contatos
- Email: leandro.damascena@gmail.com
- Github: https://github.com/leandrodamascena
- Whatsapp: Mande-me email que envio :D