# AWS Lambda Serverless Workshop

Nov. 2017

Alex Lossing

Tech Lead | alossing@palo-it.com

# CONTENT

# 1

## WHAT IS SERVERLESS

# Serverless Architectures

**No more server ?**

**Serverless computing** is a cloud computing execution model in which the **cloud provider dynamically manages the allocation of machine resources**. [...] It is a form of **utility computing**.
Serverless computing still requires servers. The name "serverless computing" is used because the **server management and capacity planning decisions are completely hidden** from the developer or operator.
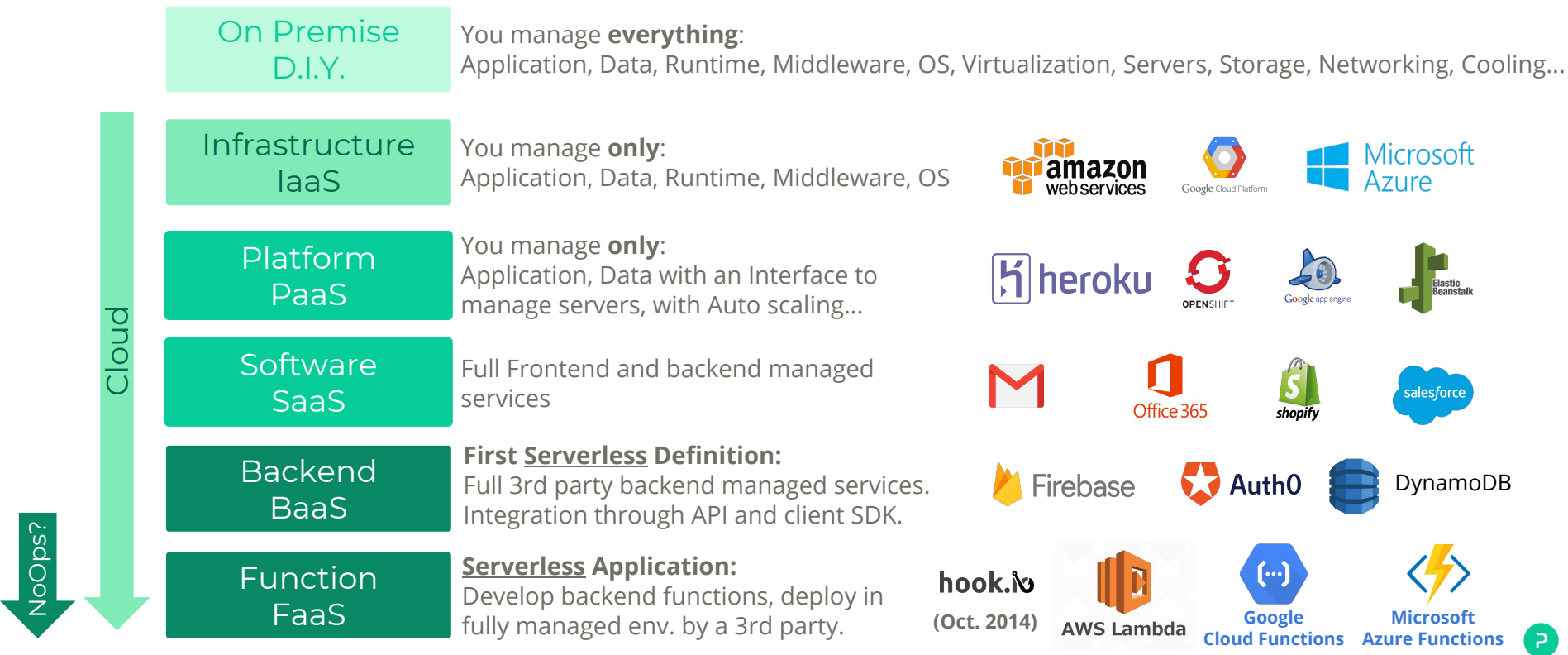
**Source: Wikipedia**

A **serverless architecture** approach replaces long-running virtual machines with **ephemeral compute power** that **comes into existence on request** and **disappears immediately after use**.

**Source: ThoughtWorks Technology Radar**

# Serverless Architectures

## The evolution of XaaS

| | | |
|---|---|---|
| **On Premise D.I.Y.** | You manage **everything**: Application, Data, Runtime, Middleware, OS, Virtualization, Servers, Storage, Networking, Cooling... | |
| **Infrastructure IaaS** | You manage **only**: Application, Data, Runtime, Middleware, OS | amazon web services · Google Cloud Platform · Microsoft Azure |
| **Platform PaaS** | You manage **only**: Application, Data with an Interface to manage servers, with Auto scaling... | heroku · OPENSHIFT · Google app engine · Elastic Beanstalk |
| **Software SaaS** | Full Frontend and backend managed services | M · Office 365 · shopify · salesforce |
| **Backend BaaS** | **First Serverless Definition:** Full 3rd party backend managed services. Integration through API and client SDK. | Firebase · Auth0 · DynamoDB |
| **Function FaaS** | **Serverless Application:** Develop backend functions, deploy in fully managed env. by a 3rd party. | hook.io (Oct. 2014) · AWS Lambda · Google Cloud Functions · Microsoft Azure Functions |

Cloud

NoOps?

# Serverless (FaaS)

## Principale

- You develop a function and deploy a function.

- No management of server systems and applications servers.

- Function are deployed on request base and are Ephemere, thus stateless.

  => You can save state/sessions in a central storage like a NoSQL database.

- Execution Duration and Memory are dependent on third-party provider. (Remember, you don't

  manage the server anymore).

  => If you need more resources, you can launch multiple lambdas following master/worker

  pattern.

- Cold Start: Latency to take in account. => You can "warm-up" your lambda

# Serverless Architectures

**Example of a Full Serverless Architecture**
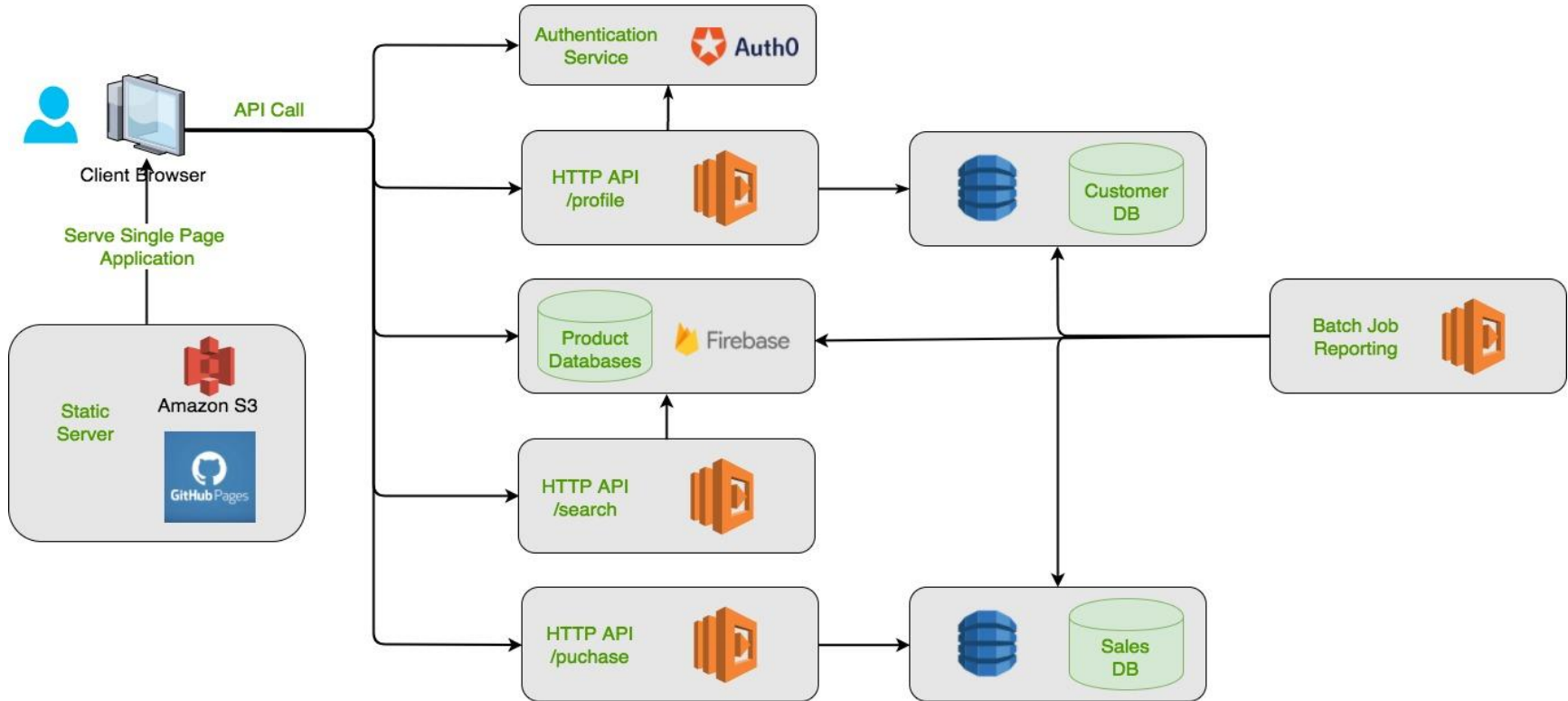
# Serverless

## Benefits

- Reduce Operational Cost
- Reduce Development Cost (BaaS)
- Scaling
- Zero Administration
- Time to market, experimentation
- "Greener". Mutualize resources
- By nature, you build a more decouple application (Microservice).
- Event-driven, Asynchronous architecture. Can still work synchronously for HTTP call.

## Challenges

- Vendor control, "Black Box". No server access.
- Vendor limitations (cold start, duration, memory...)
- Vendor "lock-in"
- Multitenancy and Regulatory Compliance
- Security concerns
- Development environment setup.
- Testing. Unit testing easy, Integration testing harder
- Stateless Architecture
- Event-driven, Asynchronous architecture
- Distributed/Microservice Architecture challenges (Data Consistency, DevOps, Monitoring, Deployment...)

# Serverless Architectures

**No More Ops ?**

**Serverless doesn't imply "No Ops".**

- **"Sys Admin"** role may not be needed anymore depending how far you go with serverless.

- Ops is also **monitoring, deployment, security, networking, production debugging, scaling**...

- Serverless architecture with FaaS are **distributed architectures** and come with their associated challenges. **DevOps** culture is important. Deployment, monitoring, operations and maintenance are shared responsibilities.

# 2

AWS Lambda

# AWS Lambda

**AWS Lambda** lets you run code **without provisioning or managing servers**. You **pay only for the compute time you consume** [...]. With Lambda, you can run code for virtually any type of application or backend service - all with zero administration. Just upload your code and Lambda takes care of everything required to **run and scale** your code **with high availability**. You can set up your code to automatically **trigger** from other AWS services or call it directly from any web or mobile app.

# AWS Lambda

**Language supported**

# AWS Lambda

**Event-Base triggering**

All Lambda can invoke manually or triggered through and event:



Amazon DynamoDB

Amazon S3

Amazon SNS

Amazon Kinesis

Amazon SES

Scheduled Events

Amazon Cognito

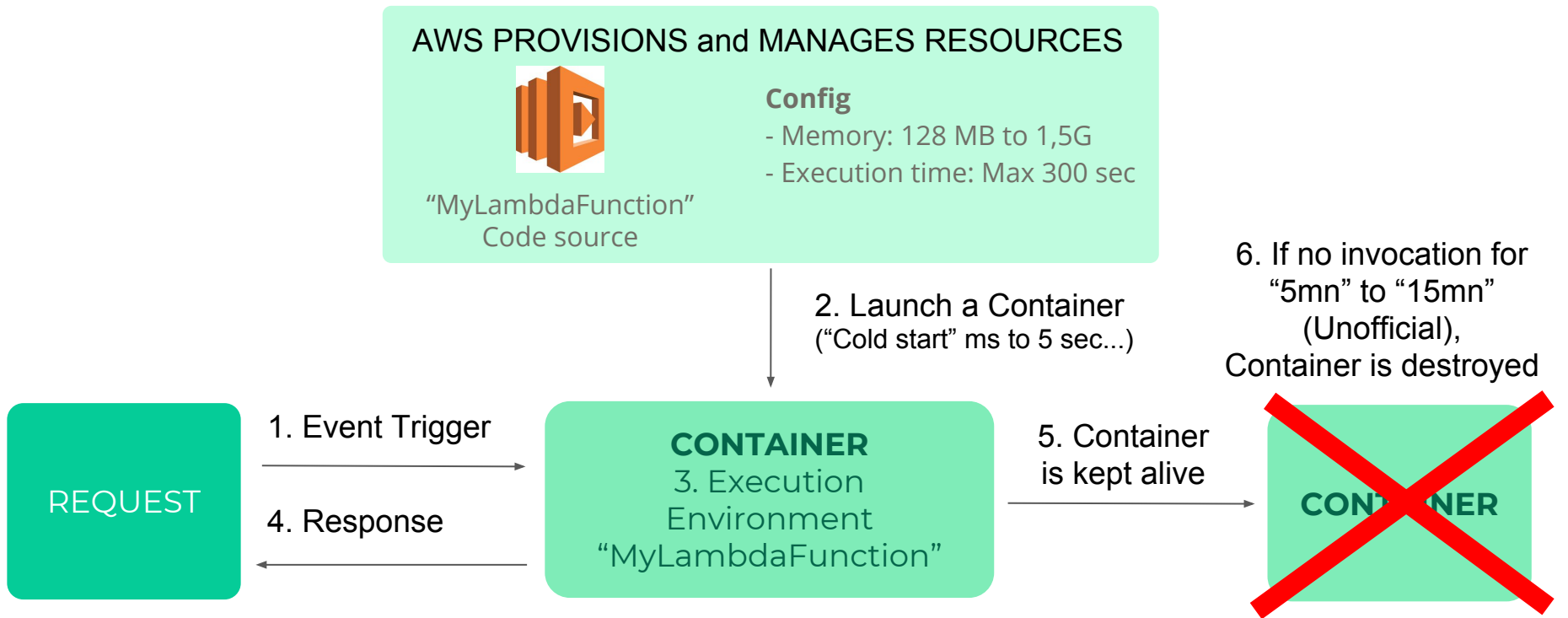AWS SDKs via Amazon API Gateway

Amazon CloudWatch

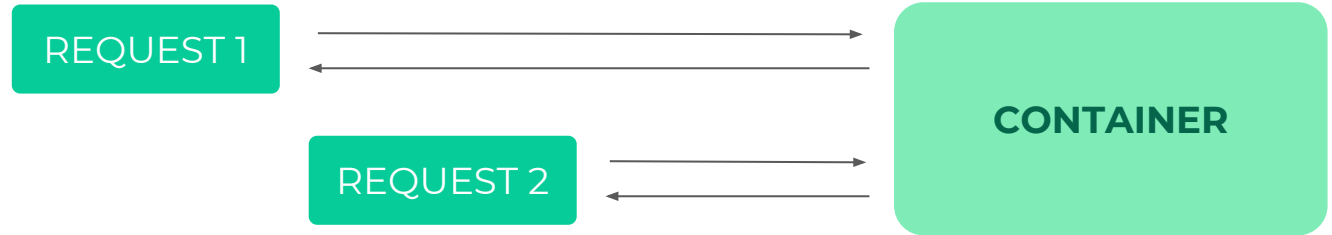Amazon Echo: Alexa Skills
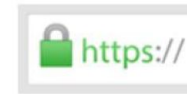
HTTPS via API Gateway

# AWS Lambda
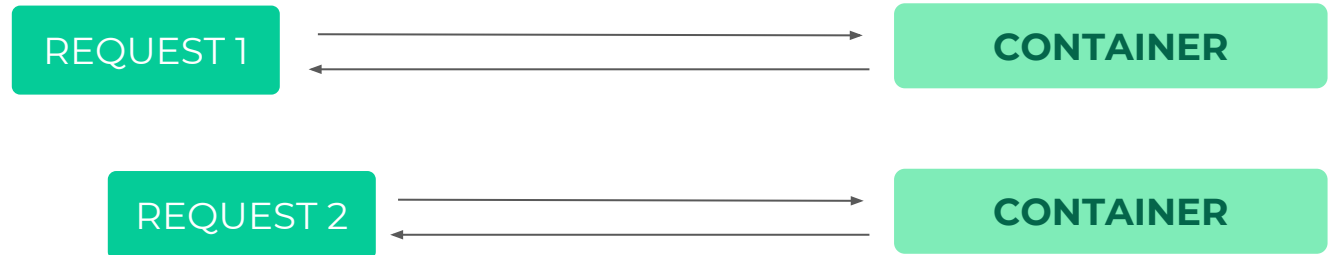
**Behind the scene: THE CONTAINER MODEL**

AWS PROVISIONS and MANAGES RESOURCES

**Config**
- Memory: 128 MB to 1,5G
- Execution time: Max 300 sec

"MyLambdaFunction"
Code source

2. Launch a Container
("Cold start" ms to 5 sec...)

6. If no invocation for
"5mn" to "15mn"
(Unofficial),
Container is destroyed

REQUEST

1. Event Trigger

4. Response

**CONTAINER**
3. Execution
Environment
"MyLambdaFunction"

5. Container
is kept alive

CONTAINER

# AWS Lambda

**Behind the scene: Multiple requests**

Sequential Requests:

REQUEST 1

REQUEST 2

CONTAINER

Parallel Requests:

REQUEST 1

CONTAINER

REQUEST 2

CONTAINER

# AWS Lambda

**Cost**

**Number of requests:**

- $0.20 per 1 million requests thereafter ($0.0000002 per request)

**Duration:**

- Duration is calculated from the time your code begins executing until it returns or otherwise terminates, rounded up to the nearest 100ms.

- The price depends on the amount of memory you allocate to your function.

- You are charged $0.00001667 for every GB-second used.

**Free-tiers:** 1M free requests per month and 400,000 GB-seconds of compute time per month.

# DEMO

AWS Lambda "Hello World"

http://docs.aws.amazon.com/lambda/latest/dg/getting-started-create-function.html

# 3 Serverless.com Framework

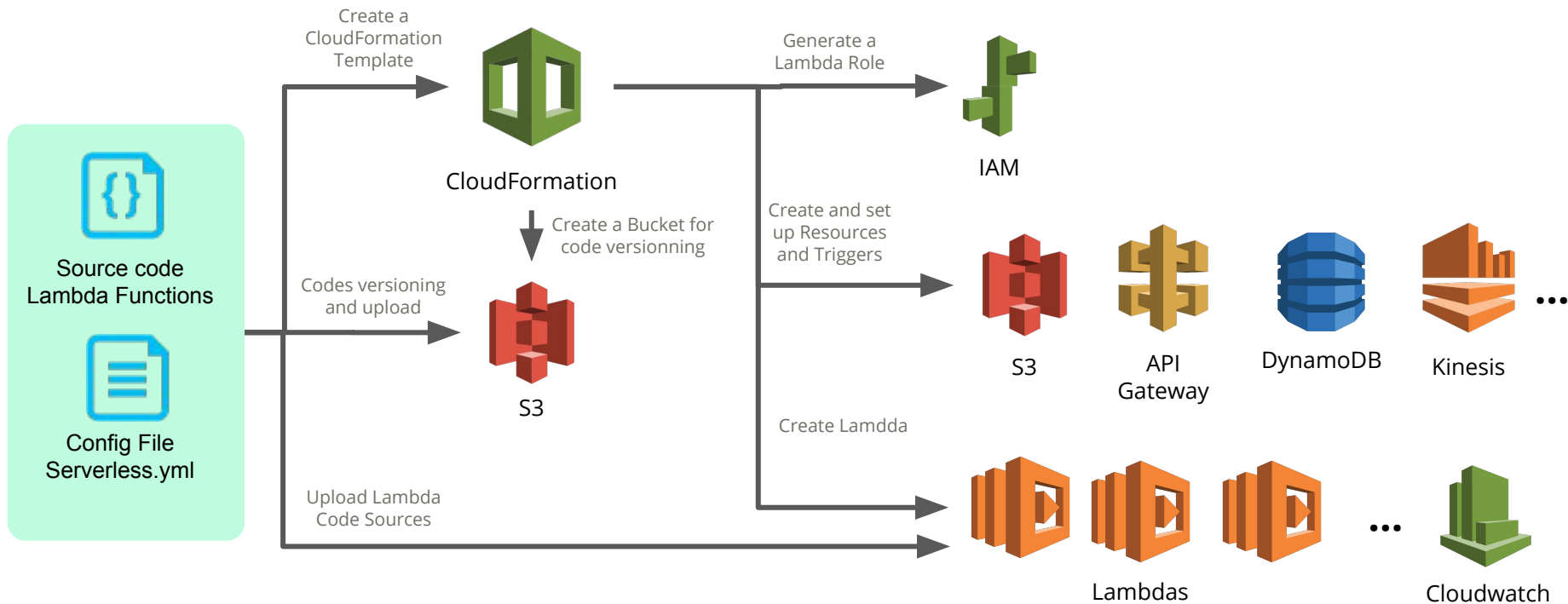# Serverless.com

**Framework**

"Serverless is your toolkit for deploying and operating serverless architectures."

Manage you serverless architecture as an Application:

# Serverless.com

## AWS Lambda

# 4 Workshop

# Workshop: Serverless.com with AWS Lambda

https://github.com/lossingalex/aws-serverless-exercise

- AWS Credentials Setup

- 4 Exercises using Serverless.com framework with AWS Lambda:
    - Exercise 1: Hello World Lambda
    - Exercise 2: Hello World through AWS Http Gateway Event
    - Exercise 3: Create Rest API TODO App with DynamoDB (Add TODO task and fetch list of tasks)
    - Exercise 4: Add a list of tasks from a CSV file using S3 Event with Lambda

# THANK YOU

*Any questions? Feel free to ask!*

# References

https://www.thoughtworks.com/radar/techniques/serverless-architecture

https://assist-software.net/blog/cloud-offering-comparison-between-iaas-paas-saas-baas

https://martinfowler.com/articles/serverless.html

http://docs.aws.amazon.com/lambda/latest/dg/lambda-introduction.html

https://serverless.com/

# CONTACT



## Alex Lossing - Tech Lead

📍 26A Circular Road, Singapore 049382

✉ alossing@palo-it.com

PALO IT

*Feel free to reach out to me if needed!*

# CONTACT

## France

21 rue de Cléry, Schoolab,75002 Paris

+33(0)1 76 54 38 16

france@palo-it.com

## Hong Kong

WeWork 20/F 535 Jaffe Road, CW, Hong Kong

+852 3905 4001

hongkong@palo-it.com

## Singapore

26A Circular Road, Singapore 049382

+65 6220 9908

singapore@palo-it.com

## Australia

50 Bridge Street, Sydney NSW 2000

australia@palo-it.com

## Mexico

Calle Moliere 50, 11560, Mexico CDMX

+52(1) 55 4000 1282

mexico@palo-it.com

PALO IT