UNIVERSITY OF JYVÄSKYLÄ

Lecture 6: Serverless

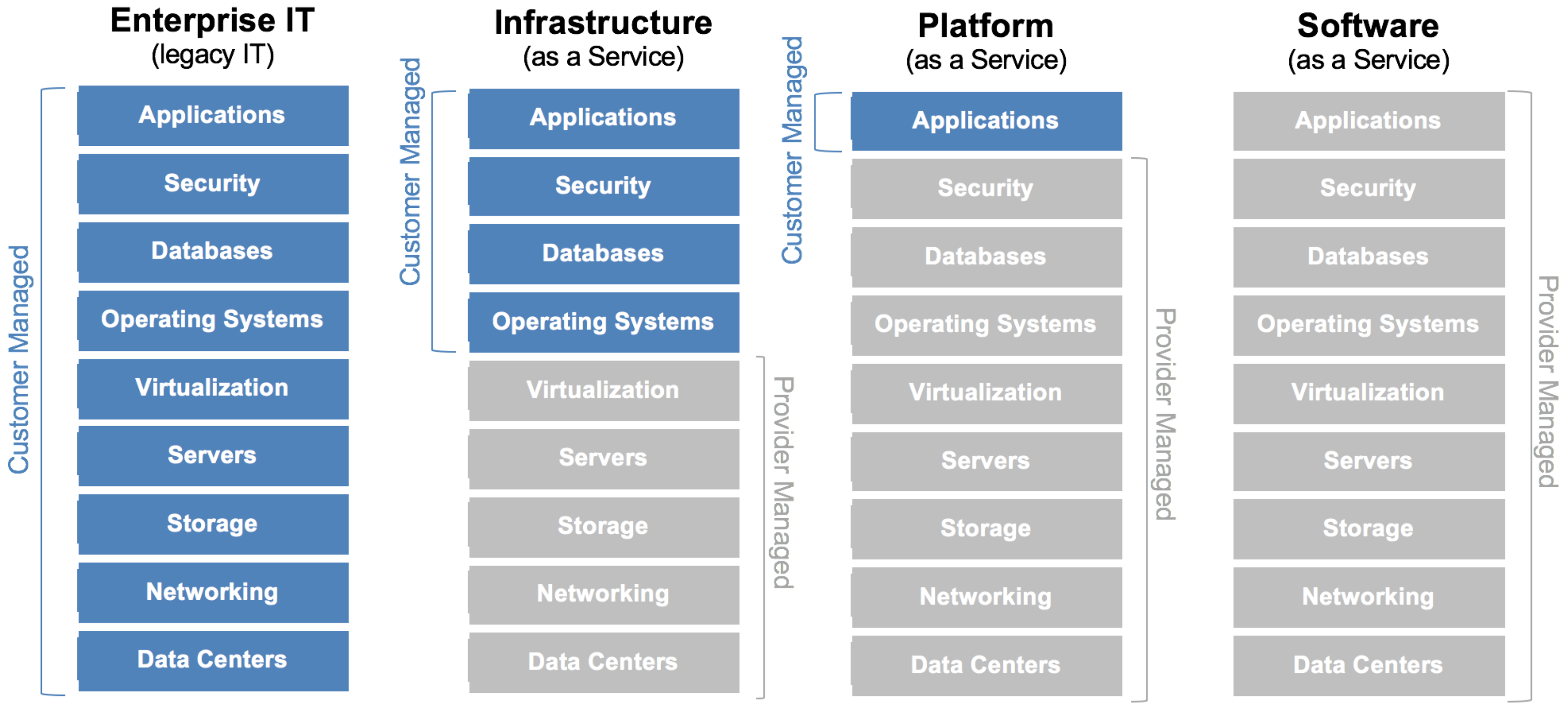
TIES4560 SOA and Cloud Computing
Autumn 2023

University of Jyväskylä

Khriyenko Oleksiy

Cloud Computing

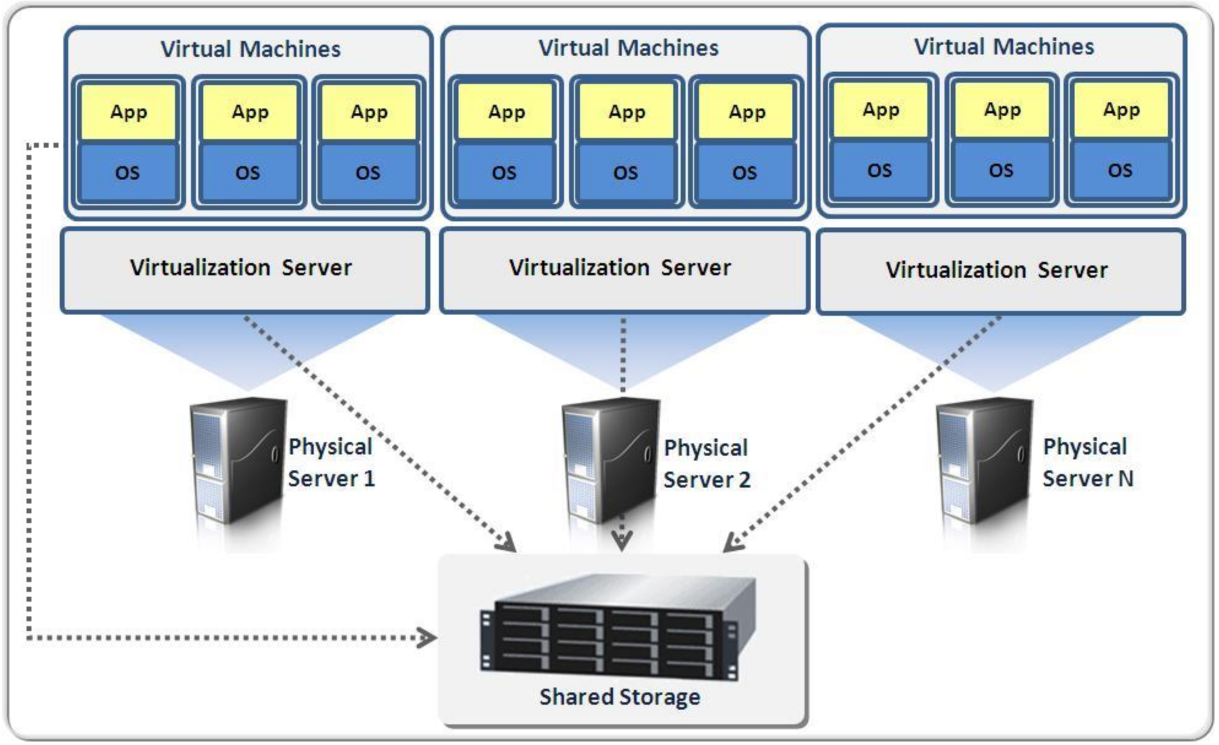
Understand IaaS, CaaS, PaaS, FaaS and SaaS…



Links:

- *https://www.ivanteong.com/blog/cloud-computing.html*

Cloud Computing



Virtualization is one of the main components behind cloud computing that enables the abstraction of the underlying physical resources as a set of multiple logical virtual machines (VMs).

- *partitioning* supports running many applications and operating systems in a single physical system;
- *isolation* ensures boundaries between the host physical system and virtual containers;
- *encapsulation* enables packaging virtual machines as complete entities to prevent applications from interfering with each other.

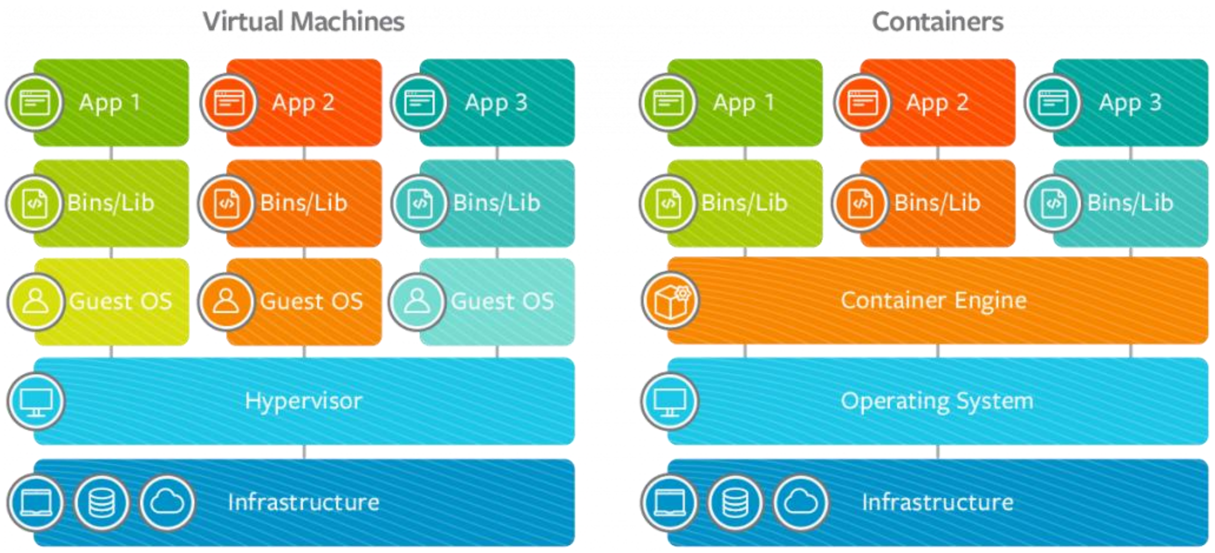
However, minimal sharing VM's own set of resources between the host system lead into:

- *high memory and storage requirements* (as each VM requires a full OS image in addition to the actual application files)
- *launch times measured in minutes* (since VM also has to go through the standard OS boot process on startup)

Cloud Computing

Containerization *is alternative lightweight solution that allows significantly smaller deployments and fast launch times ranging from less than a second to a few seconds.*

The host shares a kernel with container applications, therefore enables hosting hundreds of containers simultaneously.



- *lack of strong isolation (comparably to VM's). However, processes are still isolated from each other and the host inside containers via namespaces.*
- *lack of ability to run a different OS per deployment*
- *packaging applications and related dependencies into standardized container images to ease development efficiency and interoperability*

Pricing model concerns…

per instance per hour - the consumer is charged for the duration that an application is hosted on a VM or a container.

- *idle time is not taken into account - the consumer ends up paying for the whole hour even if actual computation took seconds…*
- *resource under-utilization - continuously hosting of non-executing applications is problematic on the provider side as well. The problem of under-utilization boils down to elasticity and resource management issues…*
- *energy efficiency - data centers spend on average only 6% to 12% of the electricity powering servers that do computation, the rest is used to keep servers idling for redundancy…*

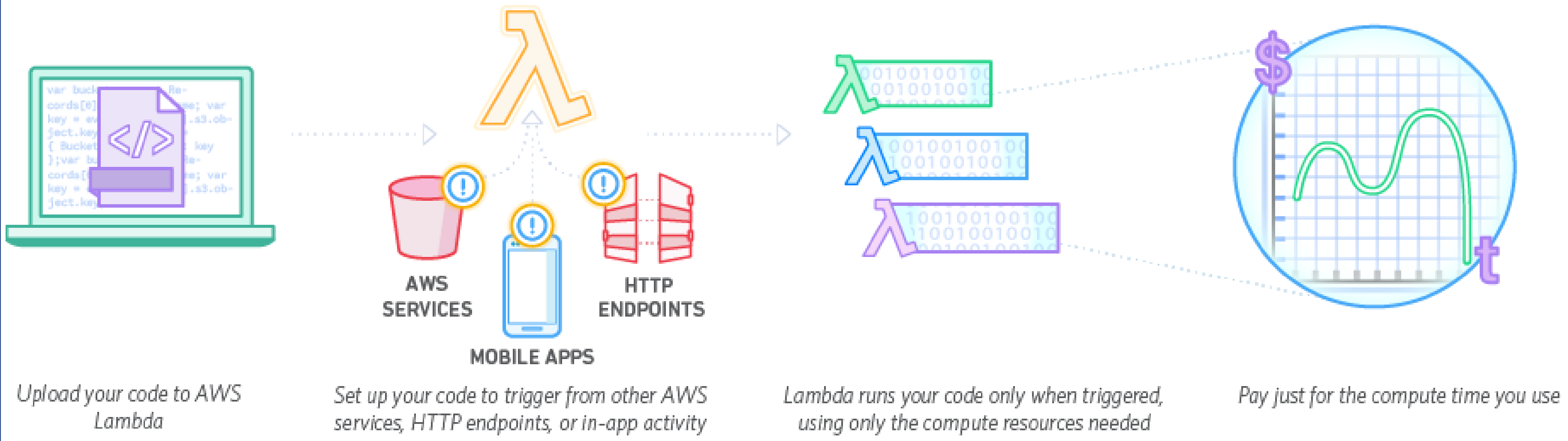
Serverless

Build more, manage less.
Serverless lets you focus on driving business value.

Serverless *is a form of cloud computing that allows users to run event-driven and granularly billed applications, without having to address the operational logic.*

*It means that developers **don't** have to **worry about servers** or any other **infrastructure issues** or **operational detail**, instead, to **focus on the code** to expose different functions. This programming model is **perfect match for Microservices**, **mobile**, **IoT**, etc.*

Function as a Service (FaaS)


AWS SERVICES
MOBILE APPS
HTTP ENDPOINTS
$
t
Upload your code to AWS Lambda
Set up your code to trigger from other AWS services, HTTP endpoints, or in-app activity
Lambda runs your code only when triggered, using only the compute resources needed
Pay just for the compute time you use

Serverless

Build more, manage less.
Serverless lets you focus on driving business value.

- *Event-driven*: interactions with serverless applications are designed to be short-lived, allowing the infrastructure to deploy serverless applications to respond to events only when needed.
- *Decoupled computation and storage*: the storage and computation scale separately and are provisioned and priced independently. In general, the storage is provided by a separate cloud service and the computation is stateless.
- *Executing code without managing resource allocation*: instead of requesting resources, the user provides a piece of code and the cloud automatically provisions resources to execute that code.
- *(Almost) no operational logic*: operational logic, such as resource management and auto-scaling, is delegated to the infrastructure, making those concerns of the infrastructure operator. However, parameters such as *memory reservation size, maximum parallelism and execution time* are still left for the user to configure *(e.g. AWS Lambda has a maximum execution duration of 15 minutes and a maximum memory allocation of 3008 MB).*
- *Granular billing*: the user of a serverless model is charged only when the application is actually executing *(measured execution time is typically 100 millisecond increments).*
- *Paying in proportion to resources used instead of for resources allocated*: billing is by some dimension associated with the execution, such as execution time, rather than by a dimension of the base cloud platform, such as size and number of VMs allocated.

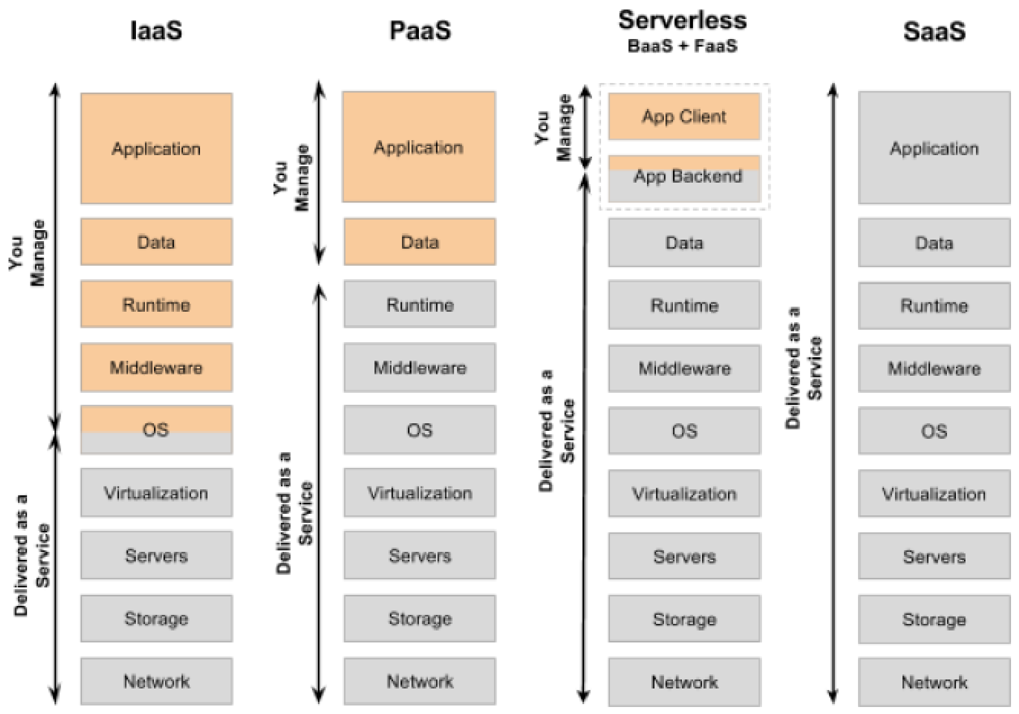
Links:

- *https://jyx.jyu.fi/handle/123456789/64836*

Serverless

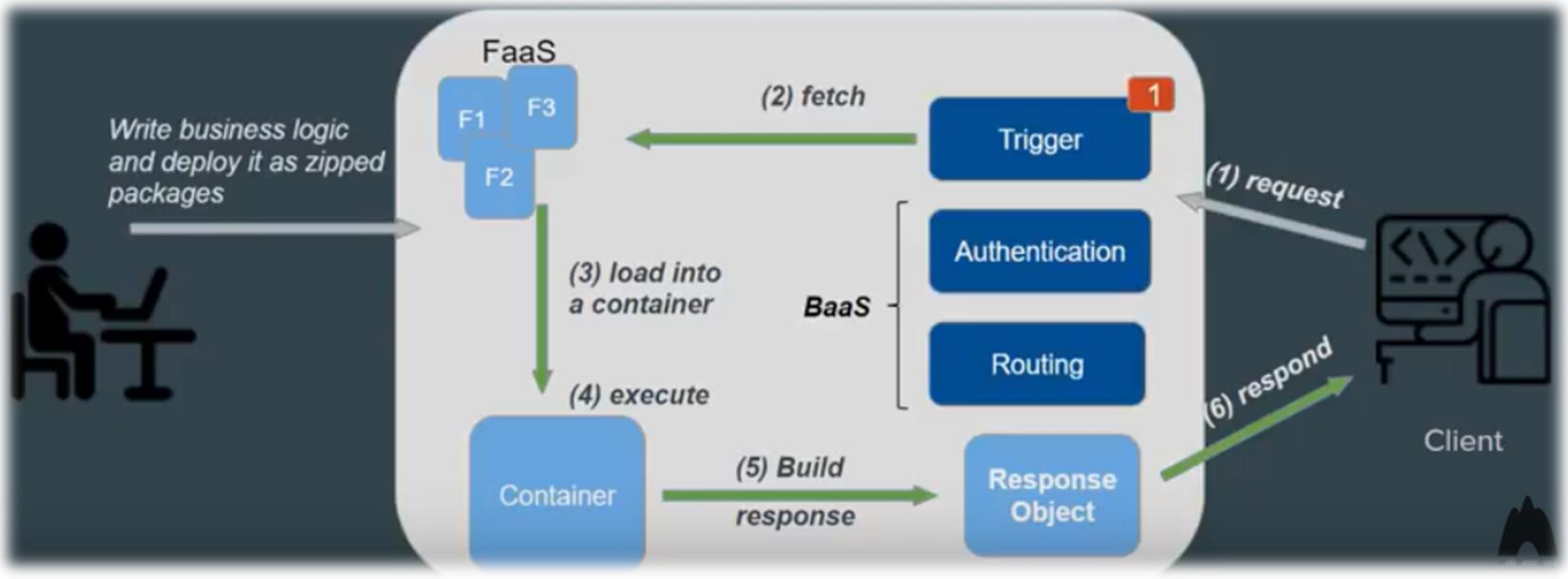
Build more, manage less.
Serverless lets you focus on driving business value.

Serverless architectures *refer to applications that significantly depend on third-party services (**Backend as a Service** or **BaaS**) or on custom code that's run in ephemeral containers (**Function as a Service** or **FaaS**), the best known vendor hosts of which currently are AWS Lambda, Azure –, Google –, IBM – and Alibaba Cloud Functions.*



Serverless

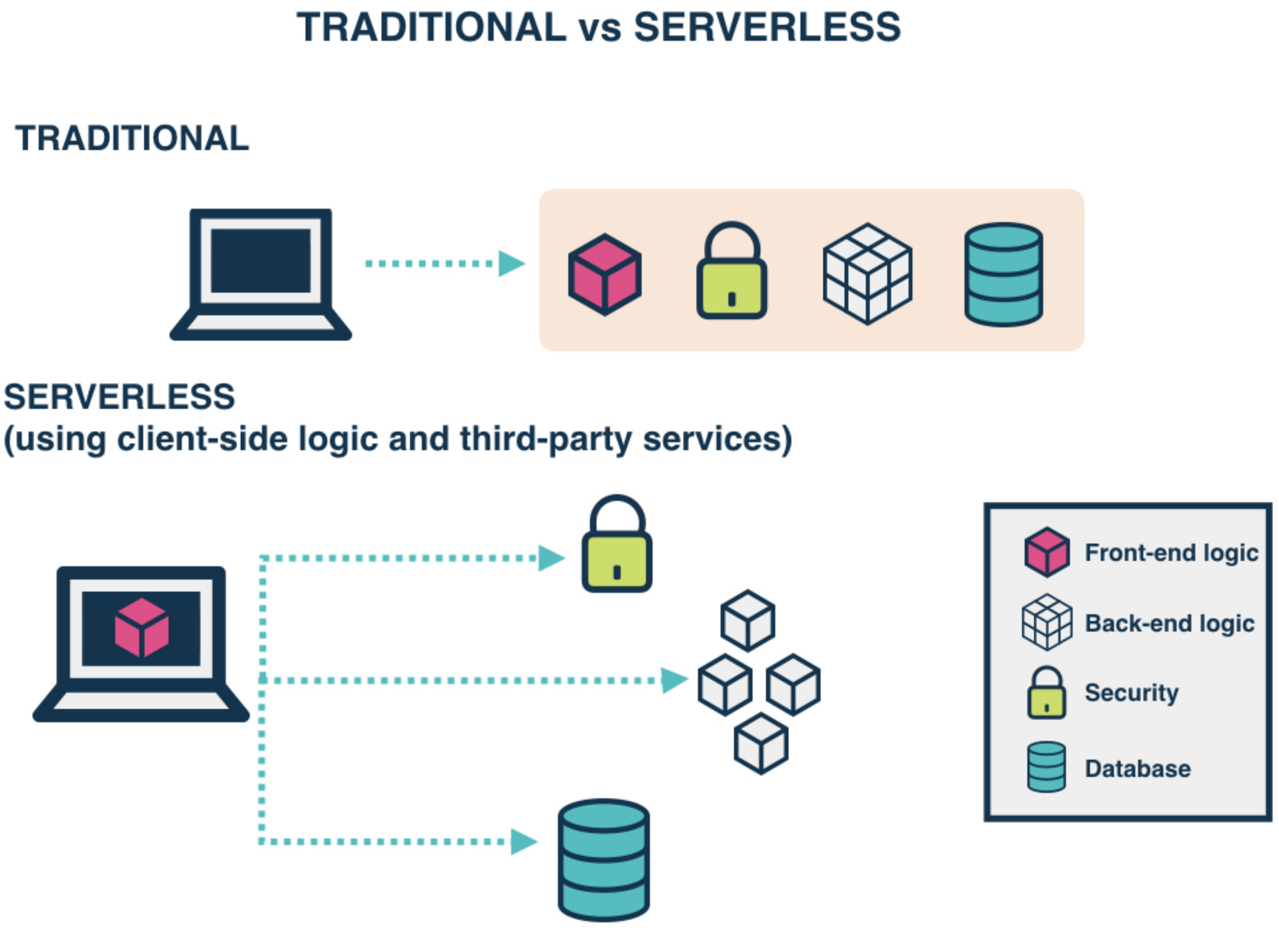
Build more, manage less.
Serverless lets you focus on driving business value.



Links:
- *https://www.youtube.com/watch?v=RzsaM6kL1FU*
- *https://www.youtube.com/watch?v=uMCtcZ46gns*

Serverless

Build more, manage less.
Serverless lets you focus on driving business value.


TRADITIONAL vs SERVERLESS
TRADITIONAL
SERVERLESS
(using client-side logic and third-party services)
Front-end logic
Back-end logic
Security
Database

Links:
- *https://www.gocd.org/2017/06/26/serverless-architecture-continuous-delivery/*

Serverless

Build more, manage less.
Serverless lets you focus on driving business value.

Benefits of Serverless:

- **Automatic Scaling** *helps to forget about provisioning & managing the server related issues. Serverless applications scale with demand…*
- **No Server Management**. *Deploy your code and let cloud provider run and scale it for you.*
- **Pay-per execution**. *Never pay for idle. Serverless applications charge you only when they run the service…*
- **Event-driven**. *Code can be triggered from both internal cloud service and external, as well as called directly from any web, mobile, or backend application via HTTP request.*
- **Low Overhead**. *Serverless teams prototype faster, get to market faster, and spend more time working on new ideas.*

Drawbacks of Serverless:

- **Problems due to third-party API system**. *Vendor control, multitenancy problems, vendor lock-in and security concerns are some of the problems due to the use of 3rd party APIs.*
- **Lack of operational tools**. *The developers are dependent on vendors for debugging and monitoring tools. Debugging Distributed Systems is difficult and usually requires access to a significant number of relevant metrics to identify the root cause.*
- **Architectural complexity**. *Decisions about how small (granular) the function should be, takes time to assess, implement and test. There should be a balance between the number of functions should an application call. Nested function call could lead to double billing. Statelessness might be an issue for large system implementation.*
- **Implementation drawbacks**. *The units of integration with Serverless FaaS (i.e. each function) are a lot smaller than with other architectures and therefore we rely on integration testing a lot more than we may do with other architectural styles. You may need to deploy a FaaS artifact separately for every function in your entire logical application.*
- **Startup latency**. *In case of "cold start", caused by some time elapsed since previous execution and the host container instance has been deprovisioned, the platform has to launch a new container, set up the runtime environment and start a fresh function host process.*

Links:
- *https://jyx.jyu.fi/handle/123456789/64836*
- *https://www.datadoghq.com/knowledge-center/serverless-architecture/*

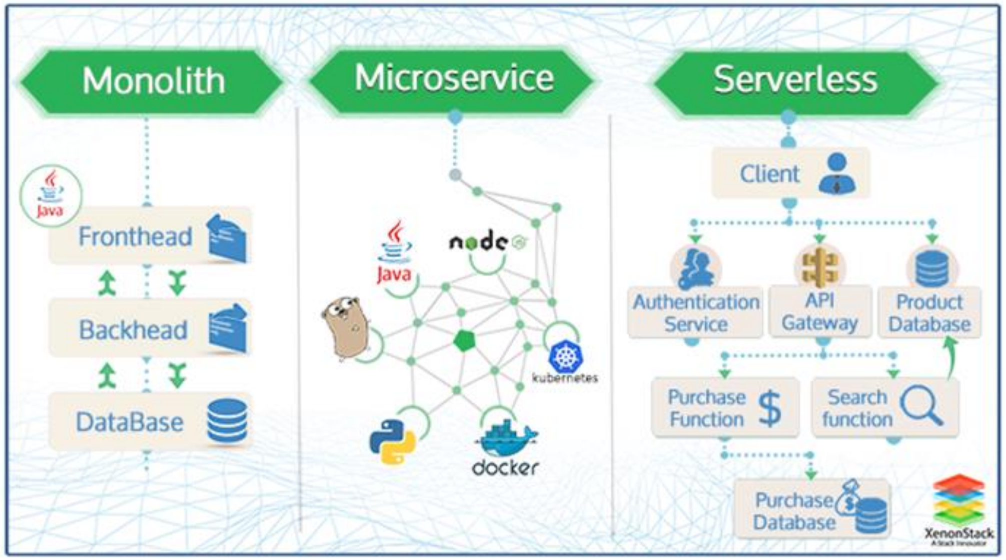
Serverless

FaaS vs PaaS…

Platform as a Service (PaaS) products offer many of the same benefits as Serverless (FaaS). They do eliminate the need for management of server hardware and software…

*The primary difference is in the **way you compose and deploy**, and therefore the **scalability of the application**.*

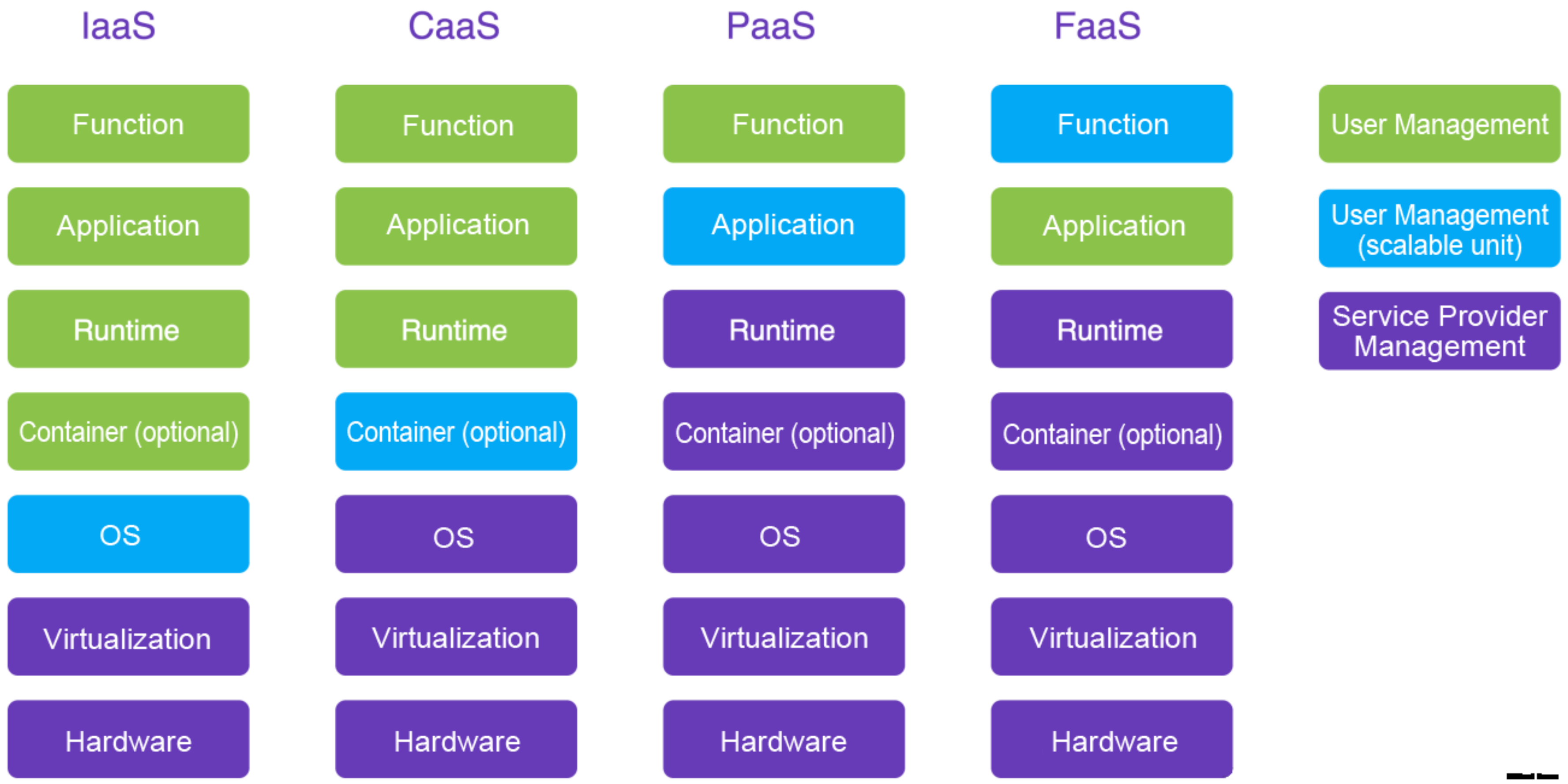
*With **PaaS**, application is deployed as a single unit and is developed in the traditional way using some kind of web framework. Scaling is only done at the entire application level. You can decide to run multiple instances of your application to handle additional load.*



*With **FaaS**, application is composed into individual, autonomous functions. Each function is hosted by the FaaS provider and can be scaled automatically as function call frequency increases or decreases. This becomes a very cost-effective way of paying for compute resources. **You only pay for the times that your functions get called**, rather than paying to have your application always on and waiting for requests on so many different instances.*

Cloud Computing

Understand IaaS, CaaS, PaaS, FaaS and SaaS…



IaaS	CaaS	PaaS	FaaS	
🟩 Function	🟩 Function	🟩 Function	🟦 Function	🟩 User Management
🟩 Application	🟩 Application	🟦 Application	🟩 Application	🟦 User Management (scalable unit)
🟩 Runtime	🟩 Runtime	🟪 Runtime	🟪 Runtime	🟪 Service Provider Management
🟩 Container (optional)	🟦 Container (optional)	🟪 Container (optional)	🟪 Container (optional)	
🟦 OS	🟪 OS	🟪 OS	🟪 OS	
🟪 Virtualization	🟪 Virtualization	🟪 Virtualization	🟪 Virtualization	
🟪 Hardware	🟪 Hardware	🟪 Hardware	🟪 Hardware	

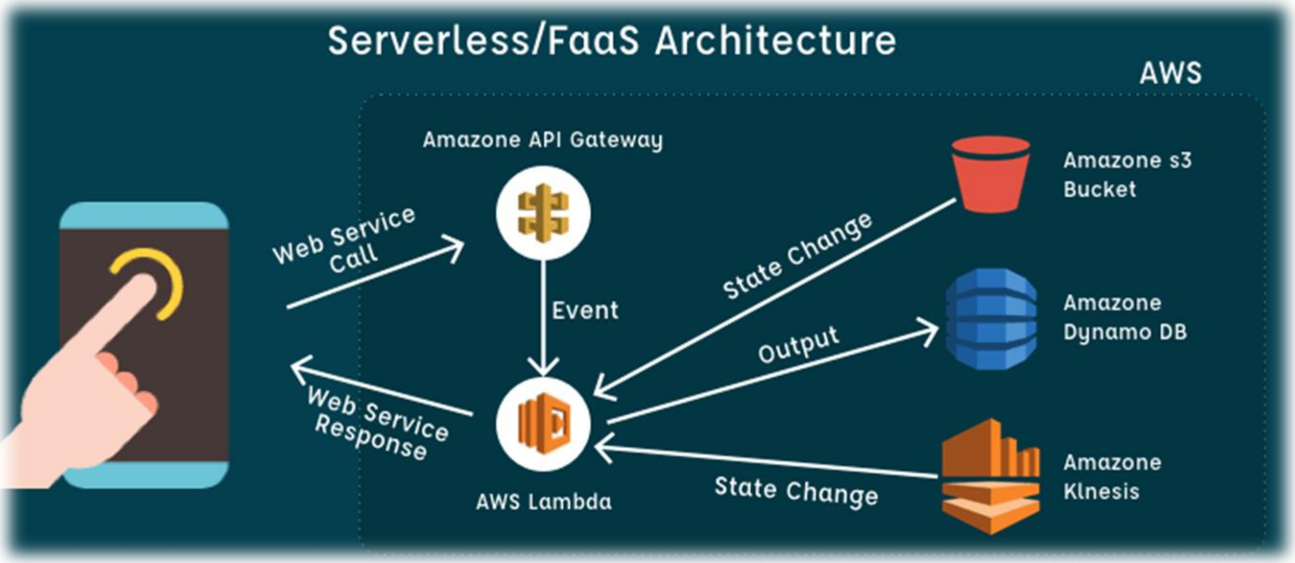
AWS Lambda

Getting Started with ***AWS Lambda***

https://aws.amazon.com/lambda/resources/#Getting_Started

Triggers. *Lambda function could be triggered by:*

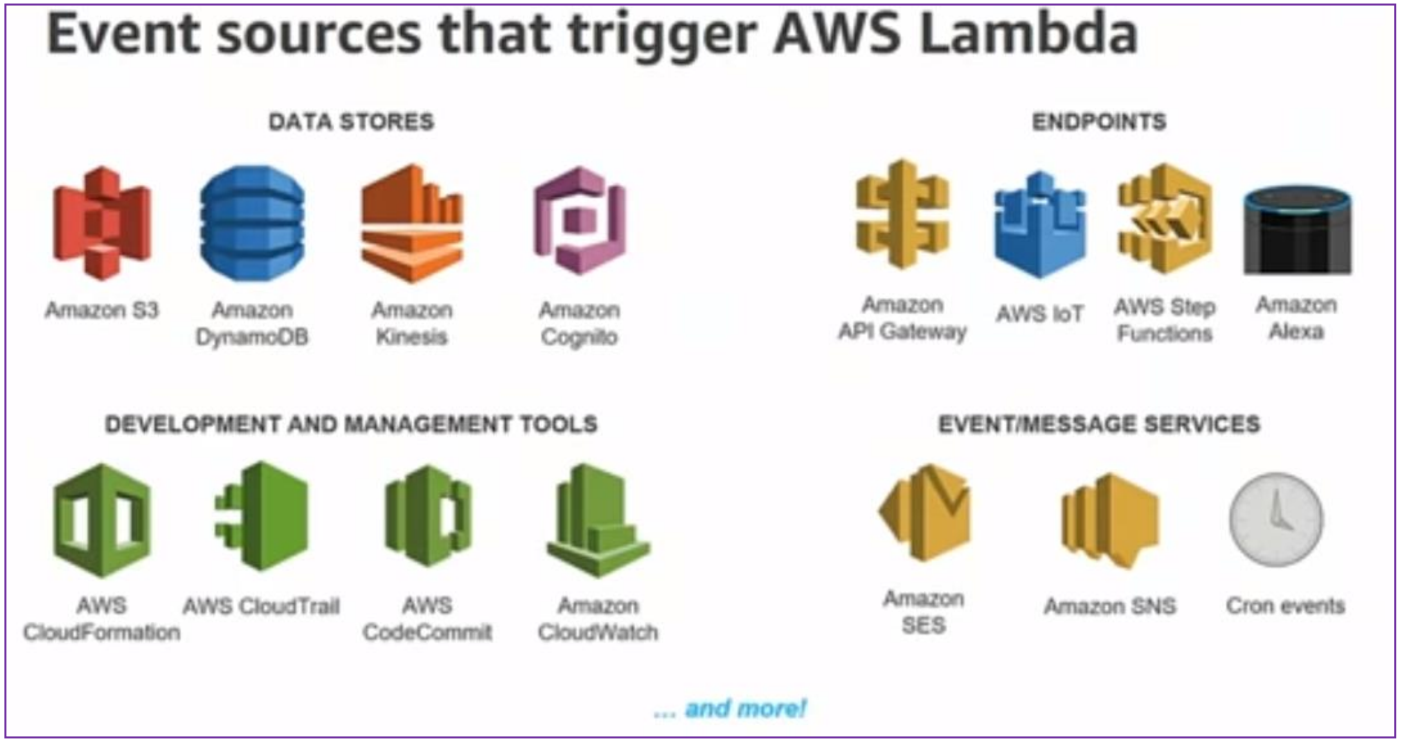
- *events from AWS services:*
  - ***S3*** *(object storage)*
  - ***DynamoDB*** *(Database with dynamically updated schema)*
  - ***CloudWatch*** *(Events, Alarms, Metric filters)*
  - ***SNS*** *(simple notification service)*
  - ***SQS*** *(simple Queue Service)*
  - ***AWS IoT***
  - *etc.*
- *HTTP calls:*
  - ***API Gateway***



AWS Lambda

Getting Started with ***AWS Lambda***

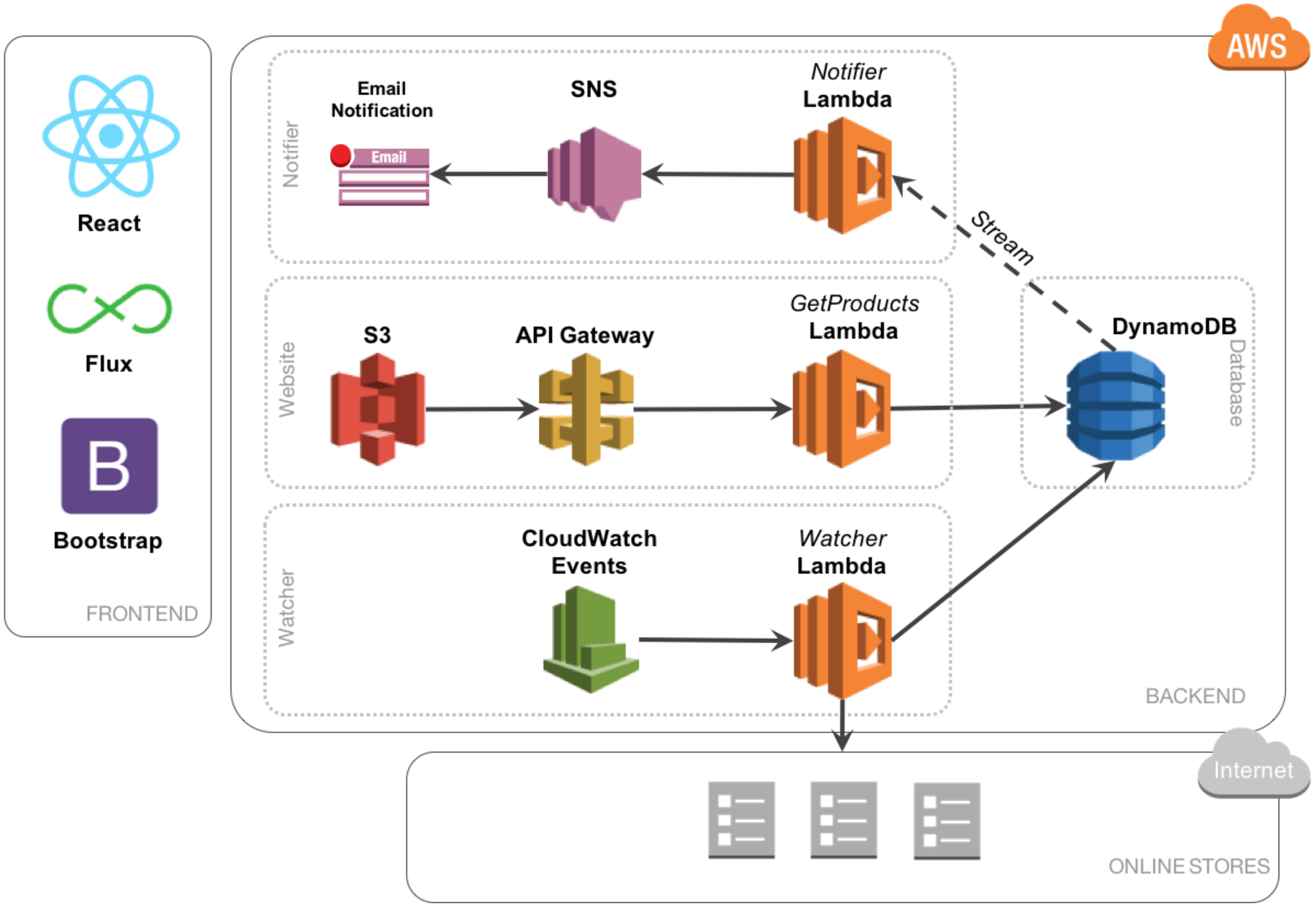
https://aws.amazon.com/lambda/resources/#Getting_Started



Links:

- *https://www.youtube.com/watch?v=WbHw14hF7lU*

Example architecture



Use-Cases

Image Thumbnail Creation:

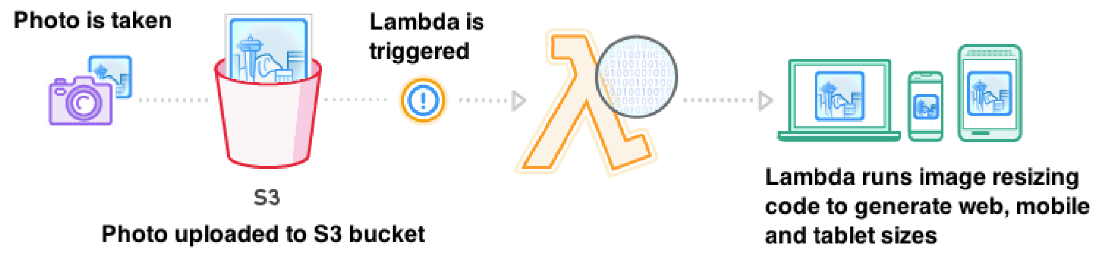
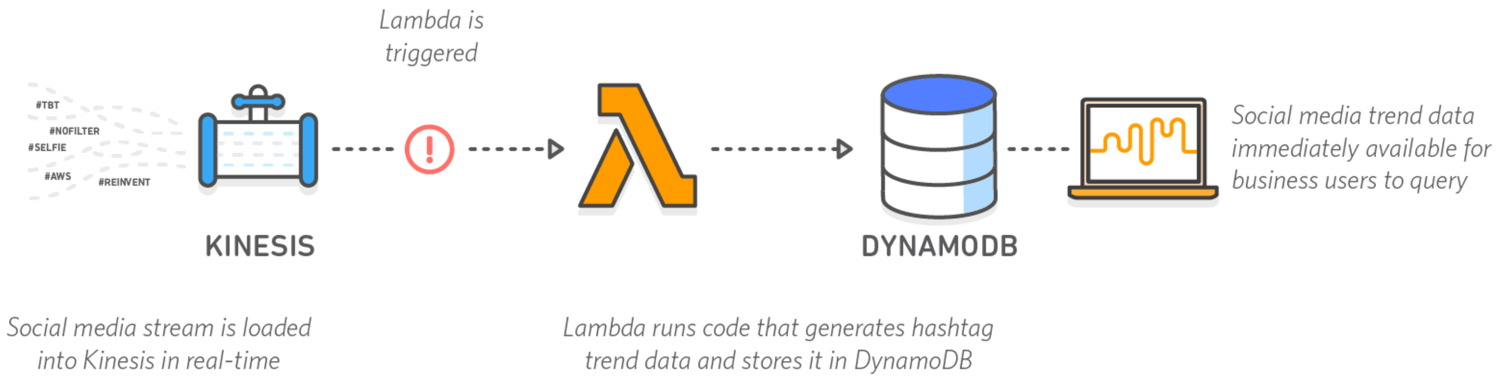


Photo is taken
Lambda is triggered
S3
Photo uploaded to S3 bucket
Lambda runs image resizing code to generate web, mobile and tablet sizes

Analysis of Streaming Social Media Data:

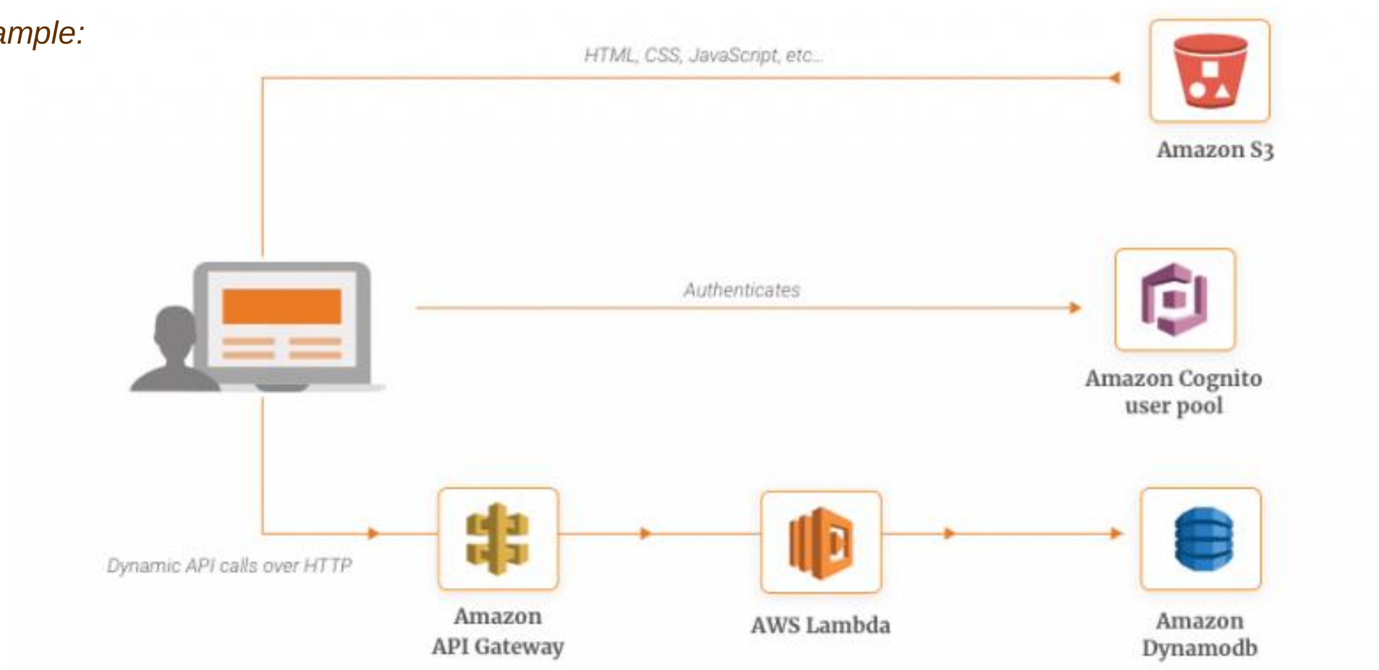

Lambda is triggered
#TBT
#NOFILTER
#SELFIE
#AWS
#REINVENT
KINESIS
DYNAMODB
Social media trend data immediately available for business users to query
Social media stream is loaded into Kinesis in real-time
Lambda runs code that generates hashtag trend data and stores it in DynamoDB

Links:

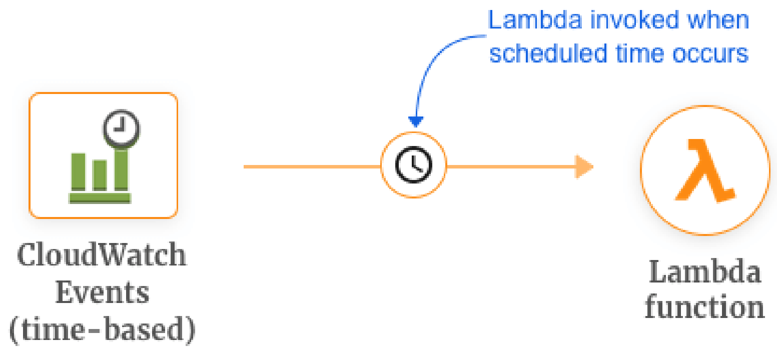
- *https://aws.amazon.com/lambda/*
- *https://dzone.com/articles/4-use-cases-of-serverless-architecture*
- *https://www.youtube.com/watch?v=p6LQ-ELvZe0*
- *https://www.simform.com/serverless-examples-aws-lambda-use-cases/*

Use-Cases

Website Example:



CRON Jobs Example:

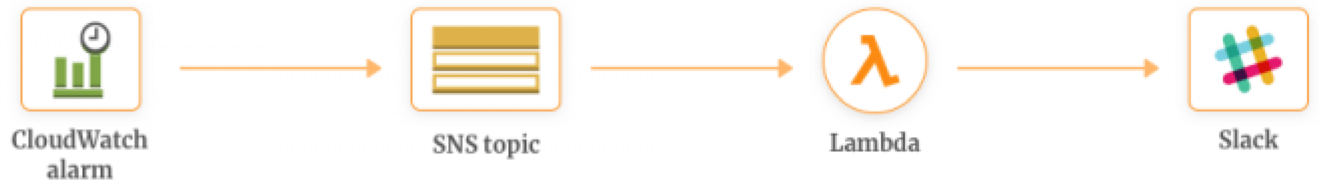

Lambda invoked when scheduled time occurs
CloudWatch Events (time-based)
Lambda function

Links:

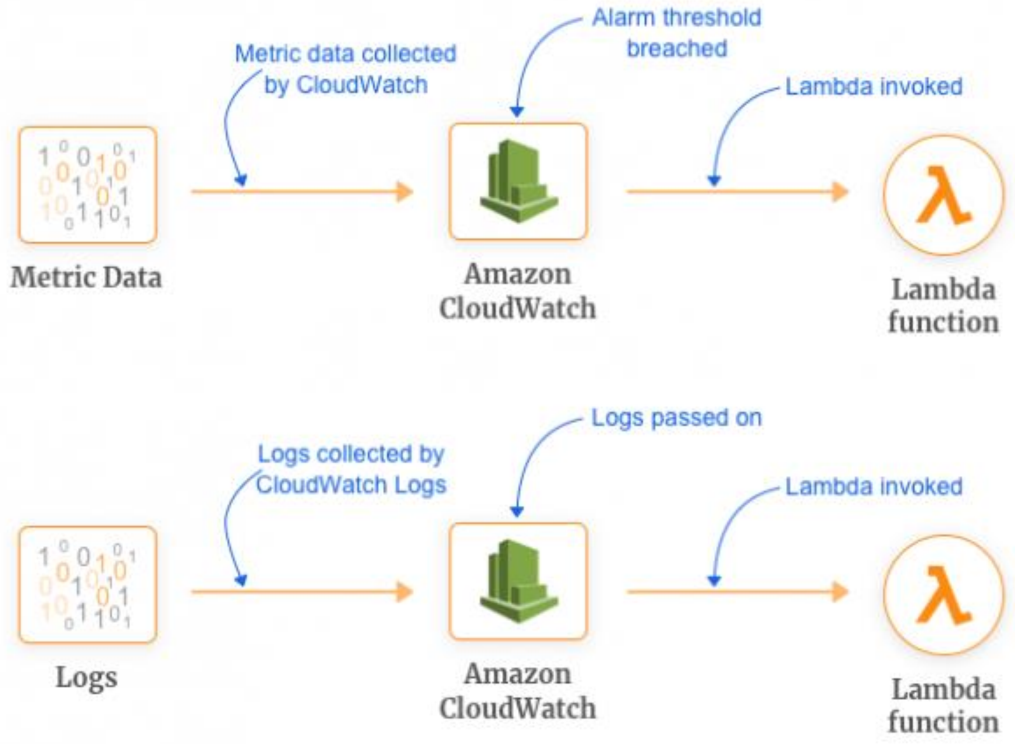
- *https://www.simform.com/serverless-examples-aws-lambda-use-cases/*
- *https://aws.amazon.com/getting-started/projects/build-serverless-web-app-lambda-apigateway-s3-dynamodb-cognito/*

Use-Cases

Real-time Notifications :



Efficient Monitoring:

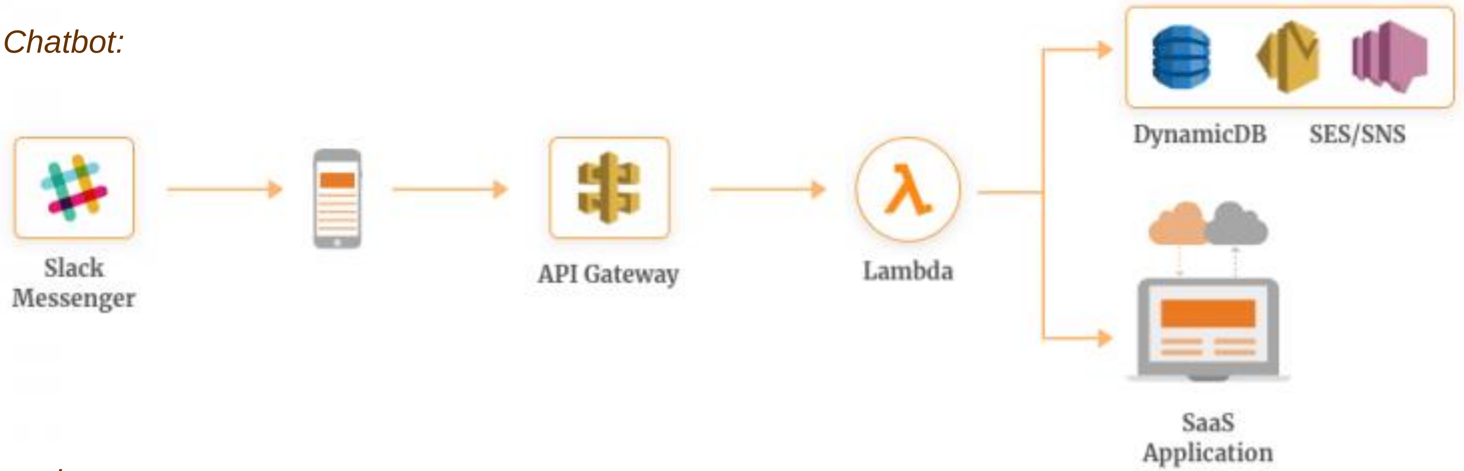


Links:

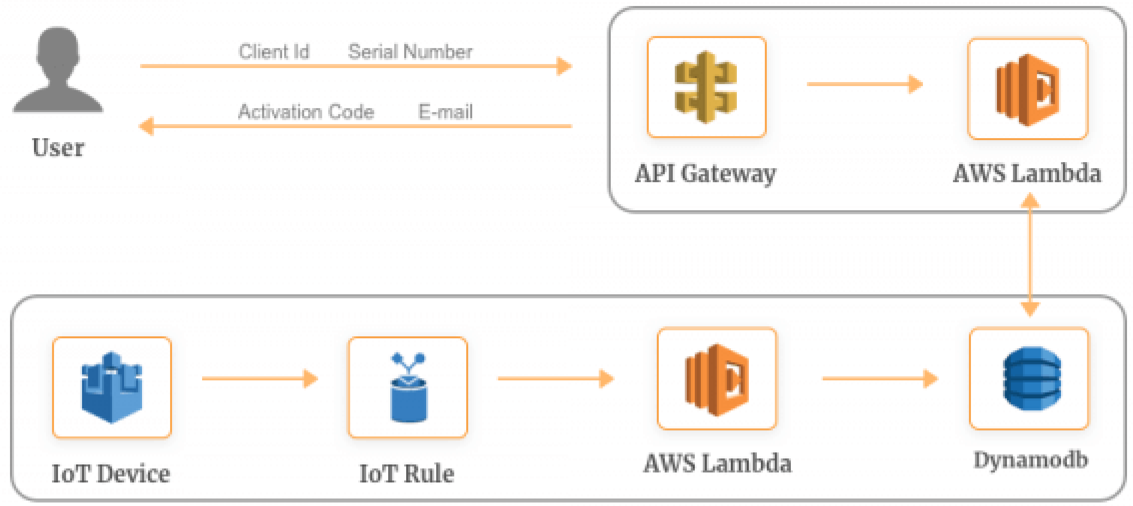
- *https://www.simform.com/serverless-examples-aws-lambda-use-cases/*

Use-Cases

Building Serverless Chatbot:



Serverless IoT Backend:



Links:

- *https://www.simform.com/serverless-examples-aws-lambda-use-cases/*

Azure Function

Getting Started with ***Azure Function***

https://azure.microsoft.com/en-us/services/functions/

Microsoft Azure

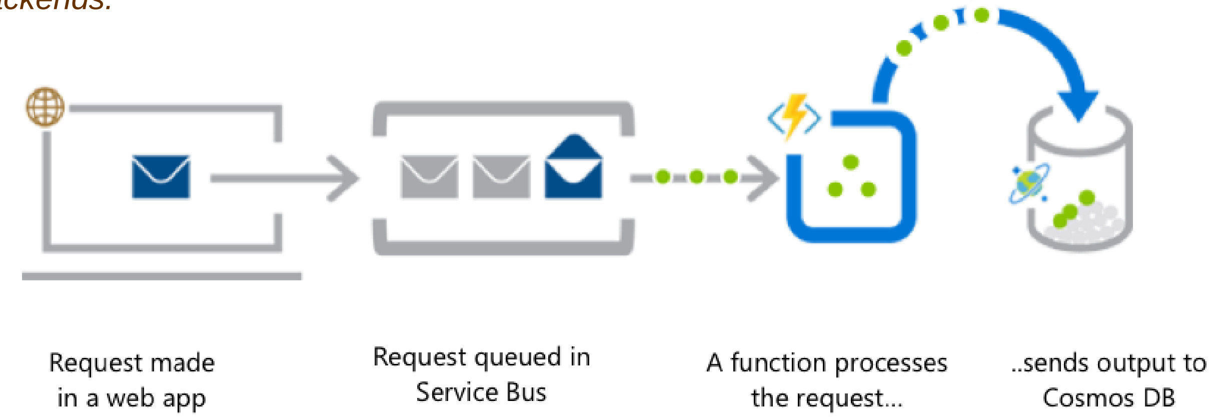
Triggers. *Azure Function could be triggered by* ***Trigers*** *(HTTPTrigger, TimerTrigger, CosmosDBTrigger, BlobTrigger, QueueTrigger, EventGridTrigger, EventHubTrigger, ServiceBusQueueTrigger, ServiceBusTopicTrigger):*

- *events from Azure and 3rd-party services:*
  - ***Azure Cosmos DB***
  - ***Azure Event Hubs***
  - ***Azure Event Grid***
  - ***Azure Notification Hubs***
  - ***Azure Service Bus*** *(queues and topics)*
  - ***Azure Storage*** *(blob, queues, and tables)*
  - ***On-premises*** *(using Service Bus)*
  - ***Twilio*** *(SMS messages)*
- *direct* ***HTTP*** *calls.*

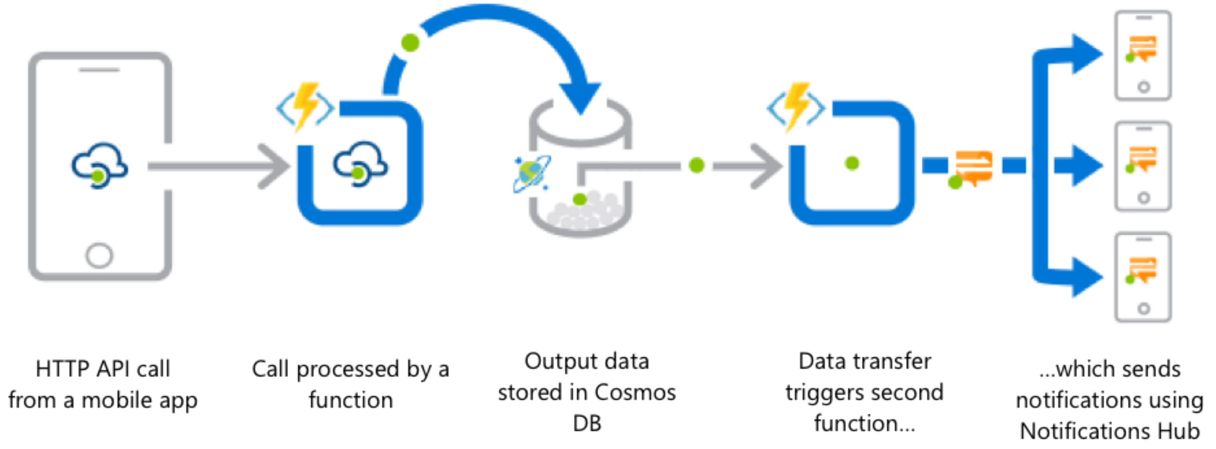
Links:
- *https://docs.microsoft.com/en-us/azure/azure-functions/functions-overview*

Use-Cases

Web application backends:


Request made in a web app
Request queued in Service Bus
A function processes the request...
..sends output to Cosmos DB

Mobile application backends:

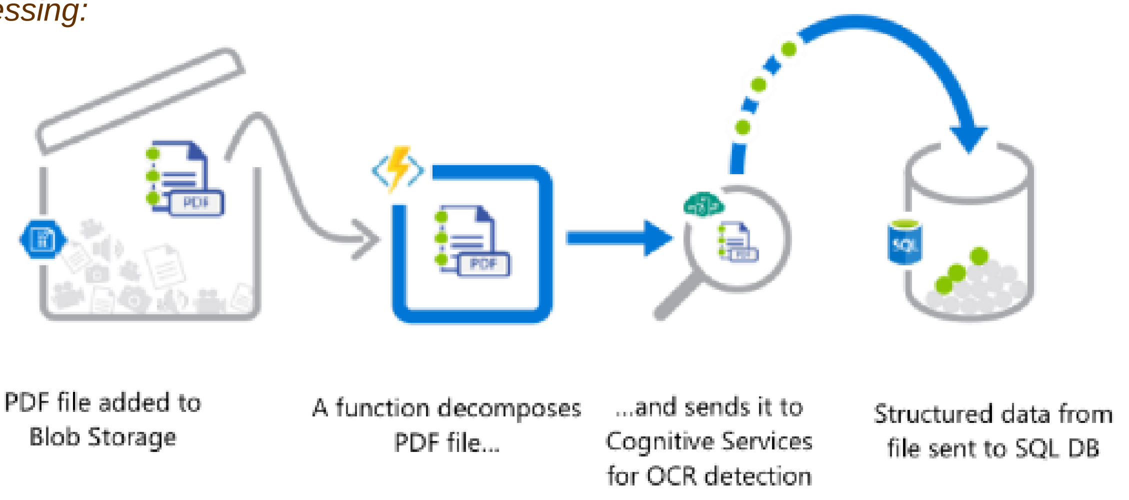

HTTP API call from a mobile app
Call processed by a function
Output data stored in Cosmos DB
Data transfer triggers second function...
...which sends notifications using Notifications Hub

Links:

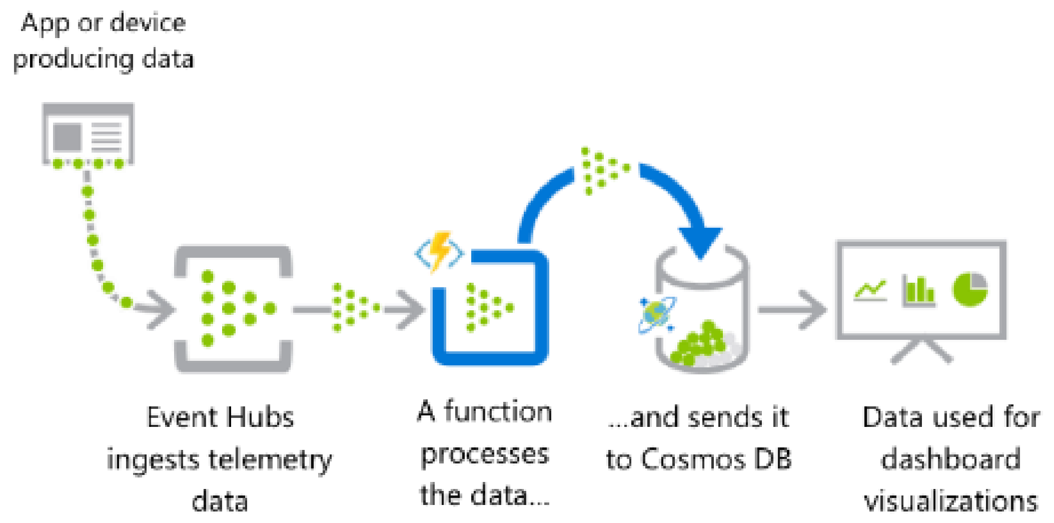
- *https://azure.microsoft.com/en-us/services/functions/*

Use-Cases

Real-time file processing:


PDF
PDF file added to Blob Storage
A function decomposes PDF file...
...and sends it to Cognitive Services for OCR detection
SQL
Structured data from file sent to SQL DB

Real-time stream processing:

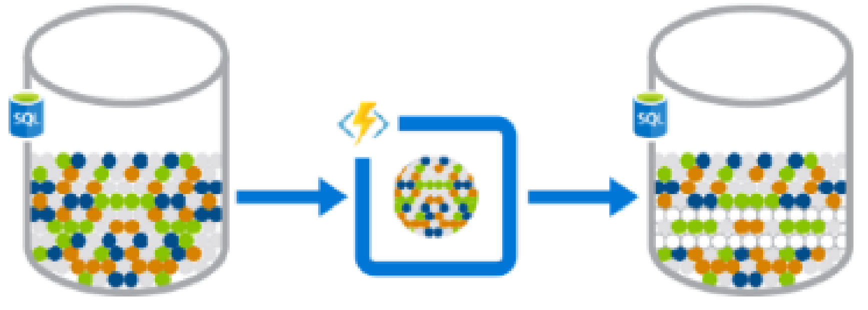

App or device producing data
Event Hubs ingests telemetry data
A function processes the data...
...and sends it to Cosmos DB
Data used for dashboard visualizations

Links:

- *https://azure.microsoft.com/en-us/services/functions/*

Use-Cases

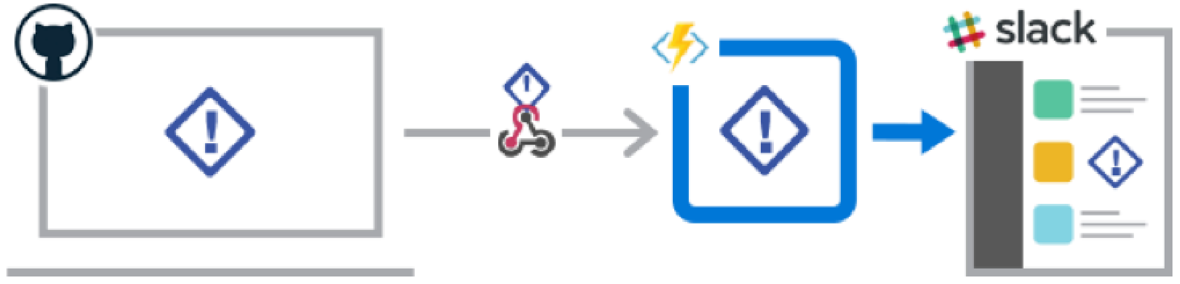
Automation of scheduled tasks:



A function cleans a database every 15 minutes...

...deduplicating entries based on business logic

Extending SaaS Applications:



Issue created in GitHub...

...which triggers a webhook call

...which is processed by a function...

...by posting the issue details to Slack

Links:

- *https://azure.microsoft.com/en-us/services/functions/*

Use-Cases

Azure Cosmos DB Trigger:

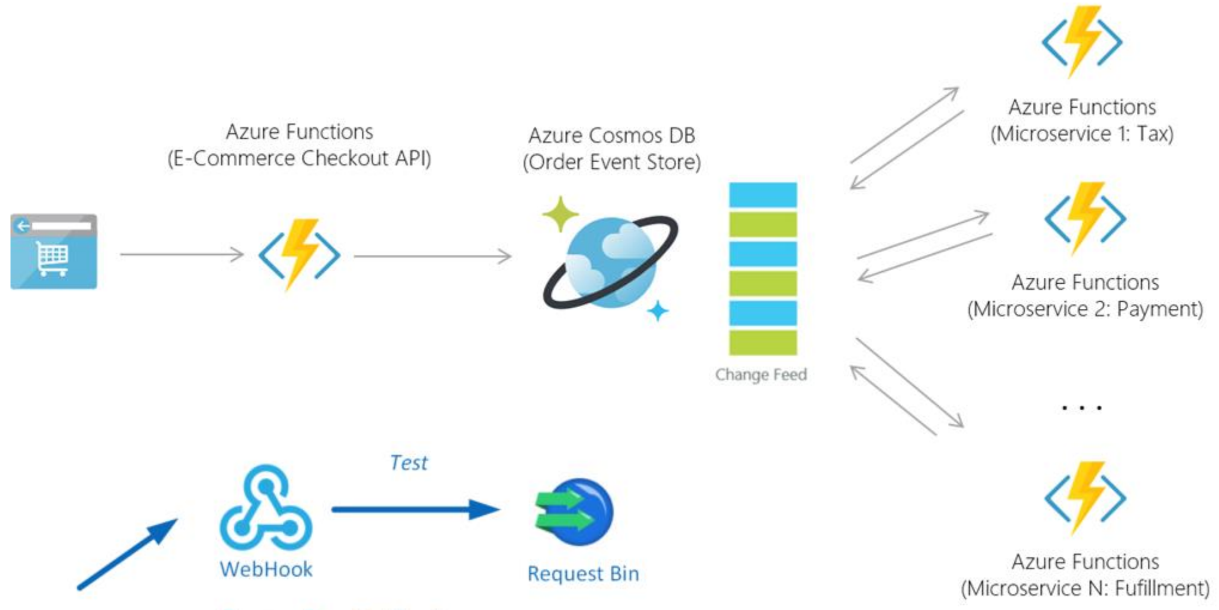
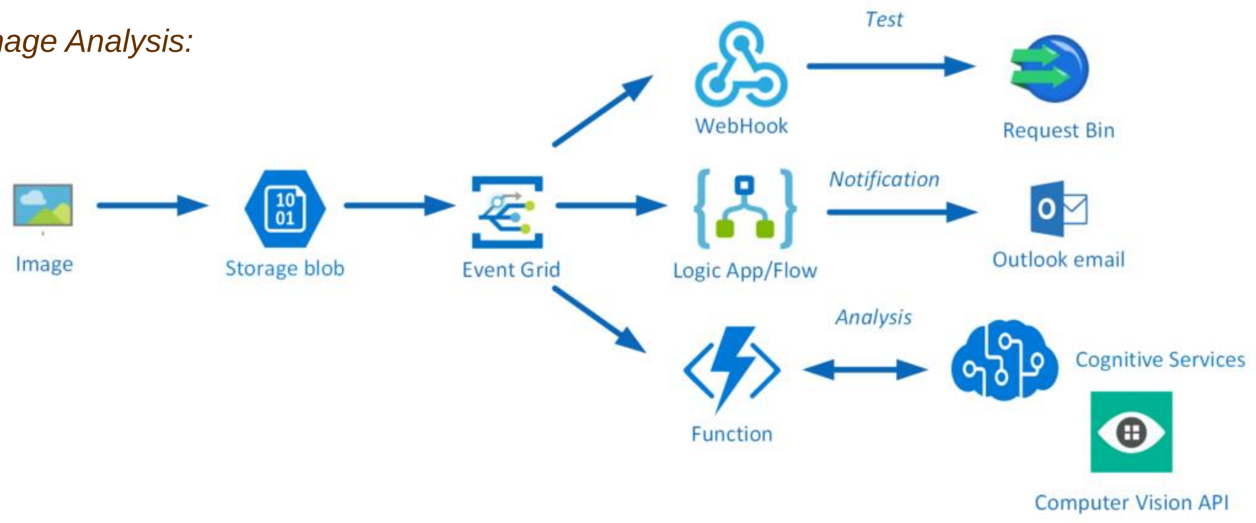


Image Analysis:



Links:

- *https://azure.microsoft.com/ja-jp/blog/serverless-for-all-developers-bringing-azure-functions-to-linux-mac-planet-scale-nosql-real-time-analytics-and-productivity-apps/*
- *https://blogs.biztalk360.com/route-azure-storage-events-multiple-subscribers-event-grid/*

Google Cloud Function

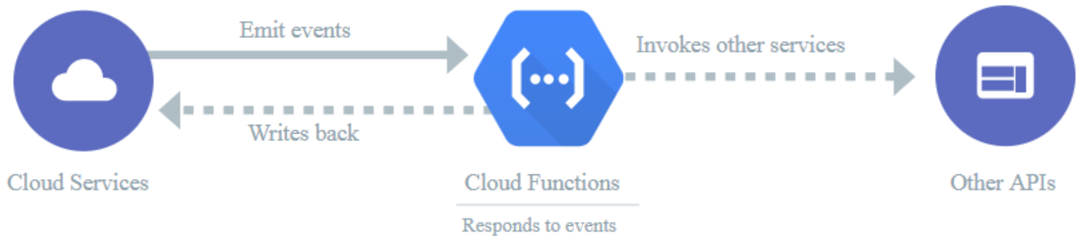
Getting Started with ***Google Cloud Function***

https://cloud.google.com/functions/

Google Cloud Platform

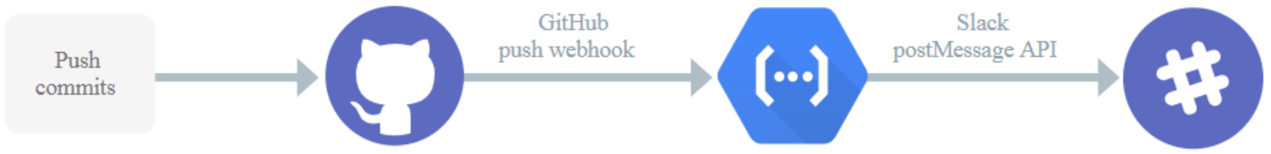
Triggers. *Google Cloud Function could be triggered by:*

- *events from cloud services:*
  - ***Cloud Pub/Sub****. Function can be invoked in response to messages published to Cloud Pub/Sub – a globally distributed message bus.*
  - ***Cloud Storage****. Function can be invoked in response to change notifications from Cloud Storage such as addition, update, or deletion.*
  - ***Stackdriver Logging****. Function can be invoked in response to log changes in Stackdriver Logging that allows you to store, search, analyze, monitor, and alert on log data and events from GCP and AWS.*
  - ***Firebase****. Function can be invoked in response to mobile-related events from Firebase such as changes to data in the Real-time Database, new user sign-ups via Auth, and conversion events in Analytics.*
- *direct* ***HTTP*** *calls.*

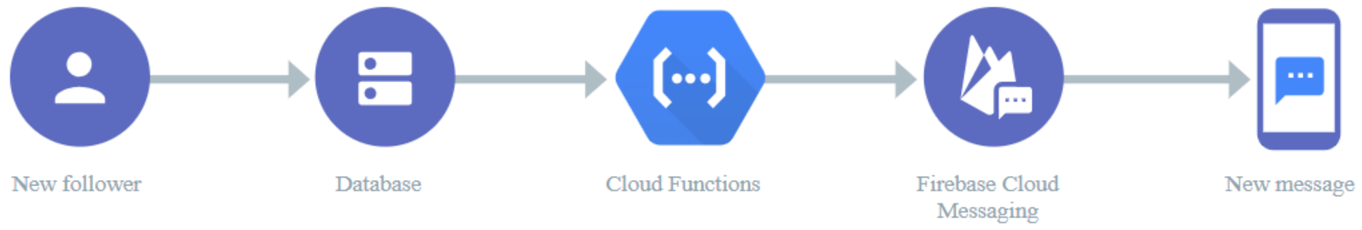


Use-Cases

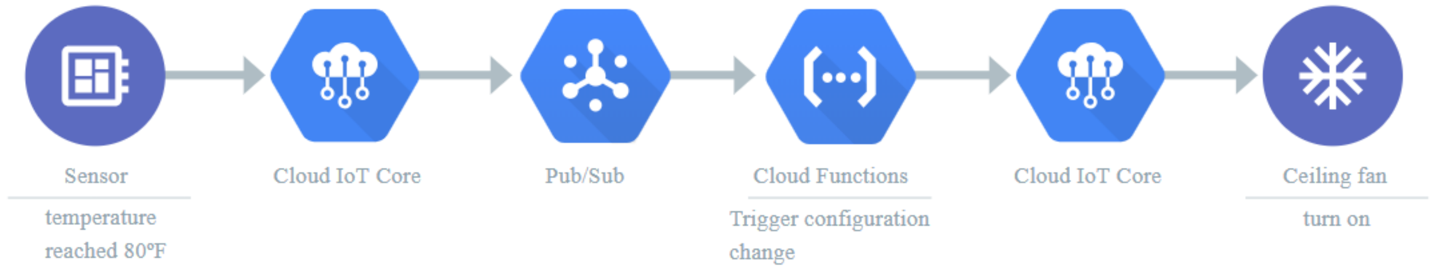
Post a comment on Slack channel in response to a GitHub commit:



Send notifications about new followers:



Update device configuration:

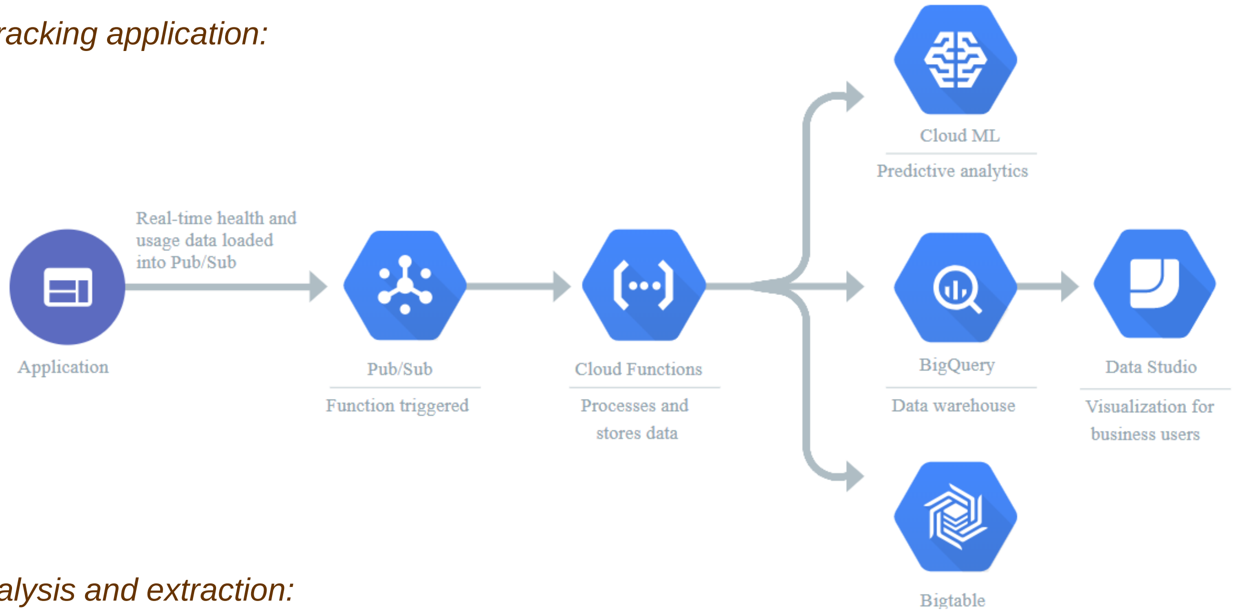


Links:

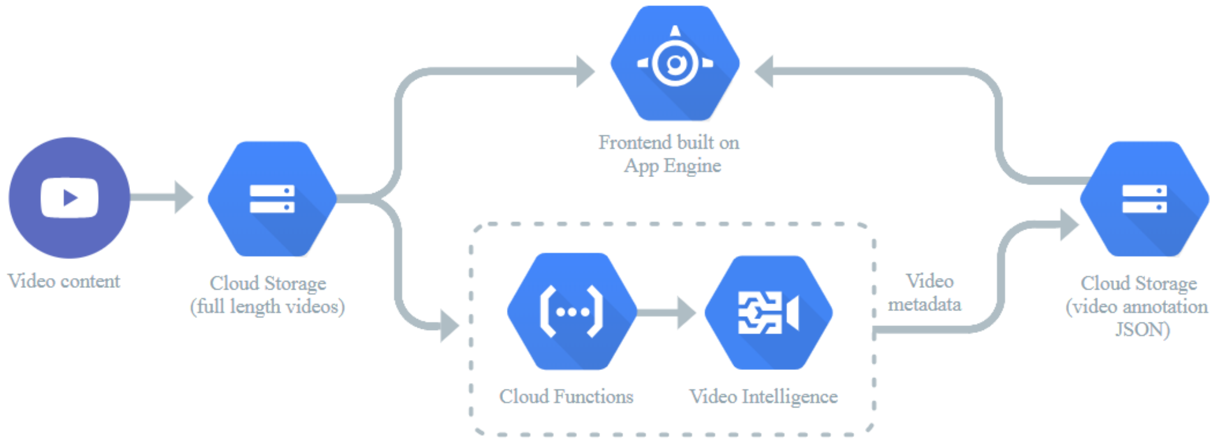
- *https://cloud.google.com/functions/use-cases/*

Use-Cases

Quality-of-service tracking application:



Video metadata analysis and extraction:

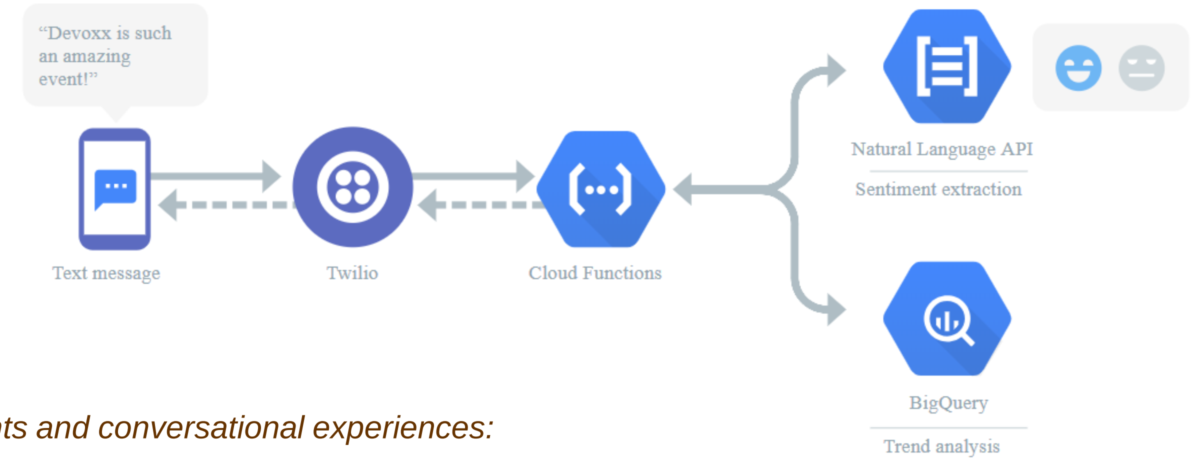

Frontend built on App Engine
Video content
Cloud Storage (full length videos)
Cloud Functions
Video Intelligence
Video metadata
Cloud Storage (video annotation JSON)

Links:

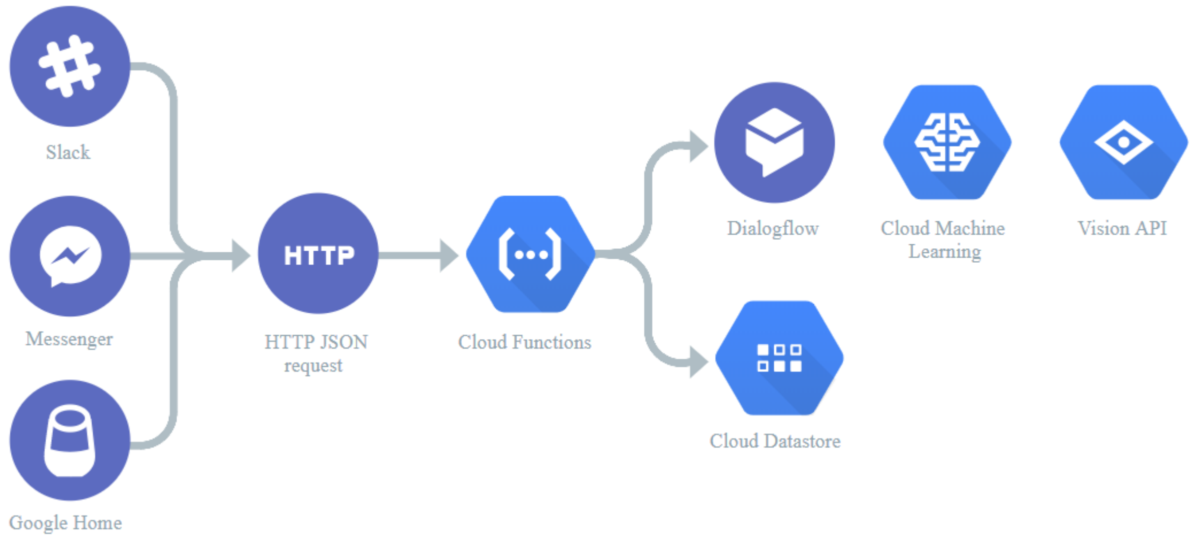
- *https://cloud.google.com/functions/use-cases/*

Use-Cases

Text message sentiment analysis:



Virtual assistants and conversational experiences:



Links:

- *https://cloud.google.com/functions/use-cases/*

IBM Cloud Function

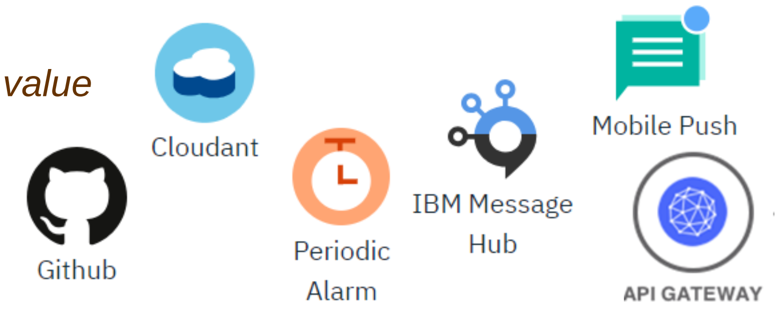
*Getting Started with **IBM Cloud Function***

https://www.ibm.com/cloud/functions

IBM Cloud

IBM Cloud Function (action) could be triggered by events including:

- *changes to database records,*
- *IoT sensor readings that exceed a certain threshold value*
- *activities in the Message Hub*
- *new code commits to a GitHub repository*
- *Periodic Alarm*
- *simple **HTTP requests** from web or mobile apps.*

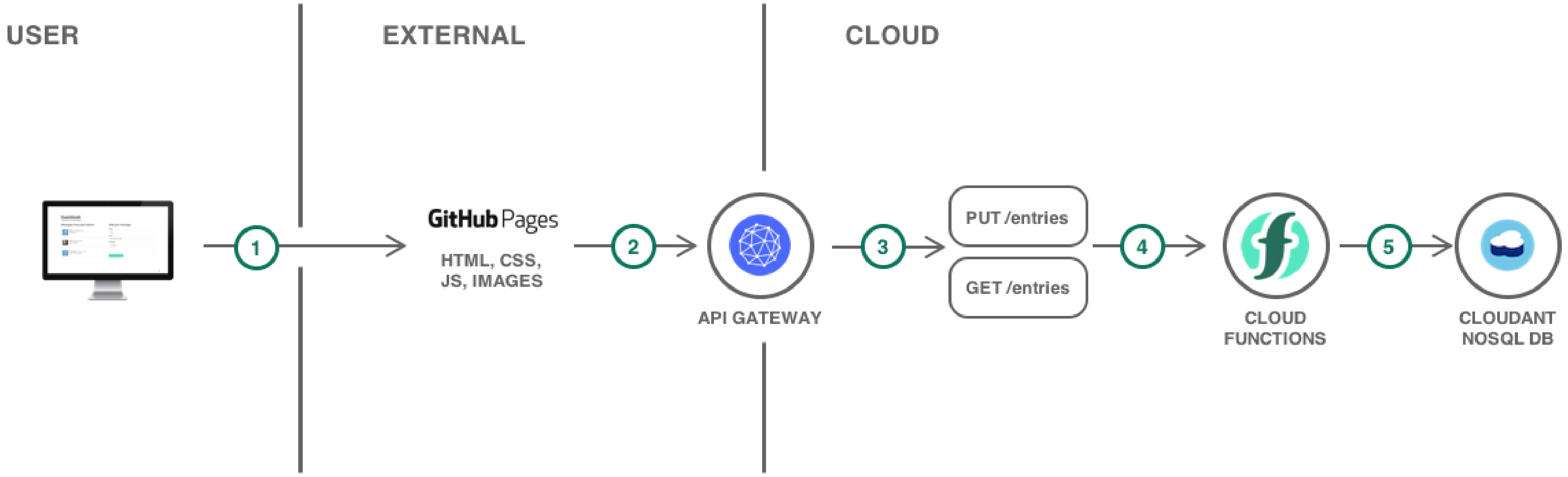


*Events from external and internal event sources are channeled through a trigger. **Triggers** are a named channel for a class of events.*
***Rules** allow actions to react to these events with the appropriate set of rules, it's possible for a single trigger event to invoke multiple actions, or for an action to be invoked as a response to events from multiple triggers.*
***Feed** is a convenient way to configure an external event source to fire trigger events that can be consumed by Cloud Functions. For example, a Git feed might fire a trigger event for every commit to a Git repository.*

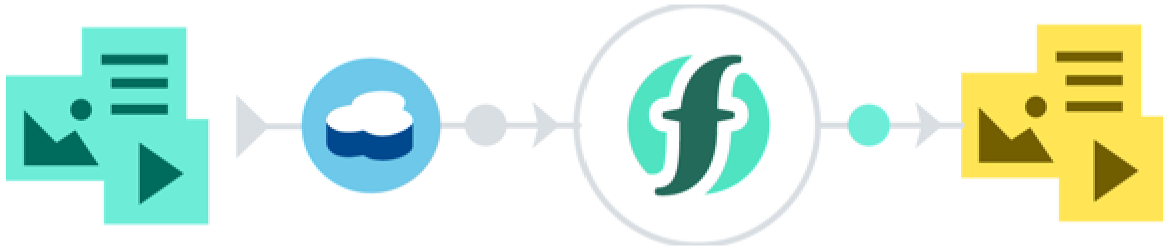
Links:
- *https://cloud.ibm.com/docs/openwhisk/openwhisk_about.html#openwhisk_about*
- *https://cloud.ibm.com/docs/openwhisk/openwhisk_triggers_rules.html#openwhisk_triggers_create*
- *https://cloud.ibm.com/docs/openwhisk/openwhisk_feeds.html#openwhisk_feeds*

Use-Cases

API Gateway integration:



Triggering IBM Cloud Functions on Cloudant data changes:

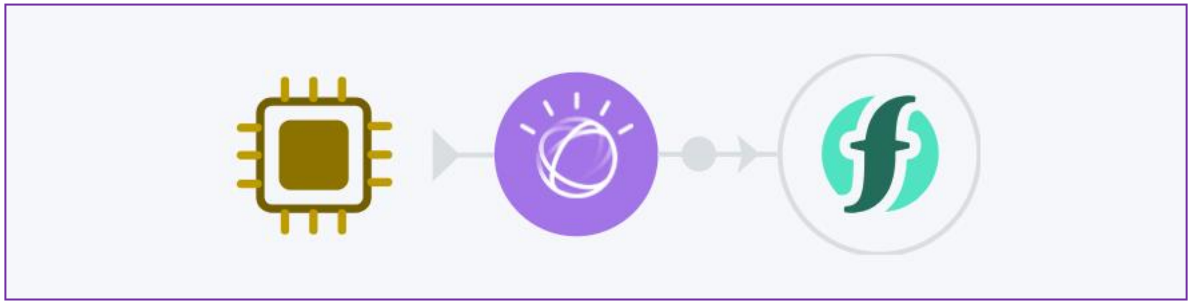


Links:

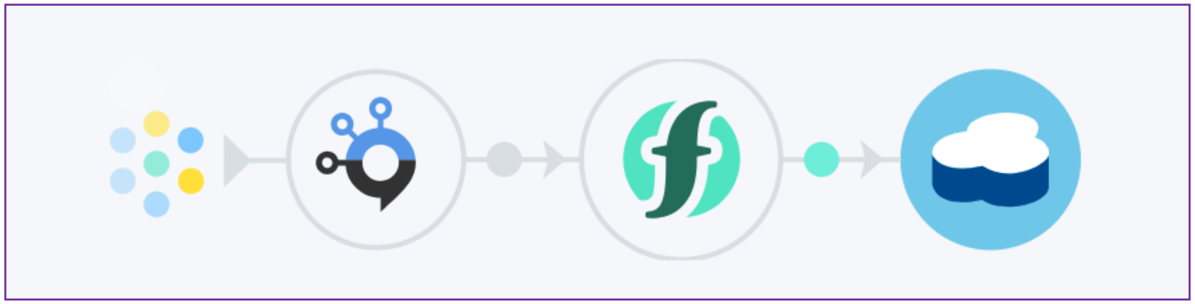
- *https://cloud.ibm.com/functions/*

Use-Cases

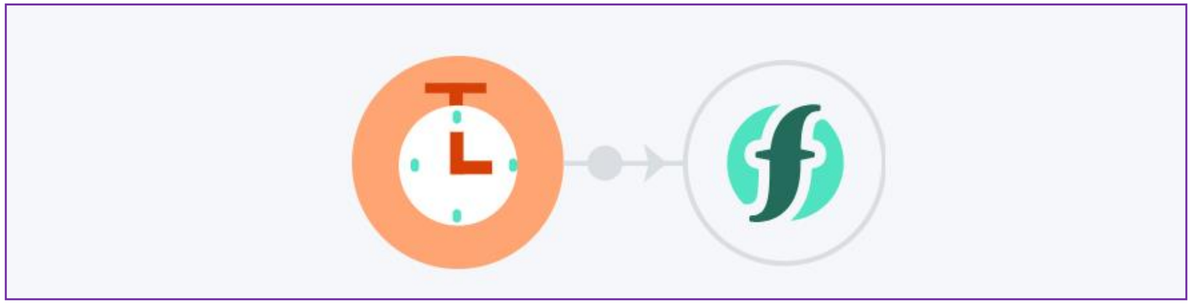
IoT Ready:



Event Stream Processing:



Scheduled Tasks:

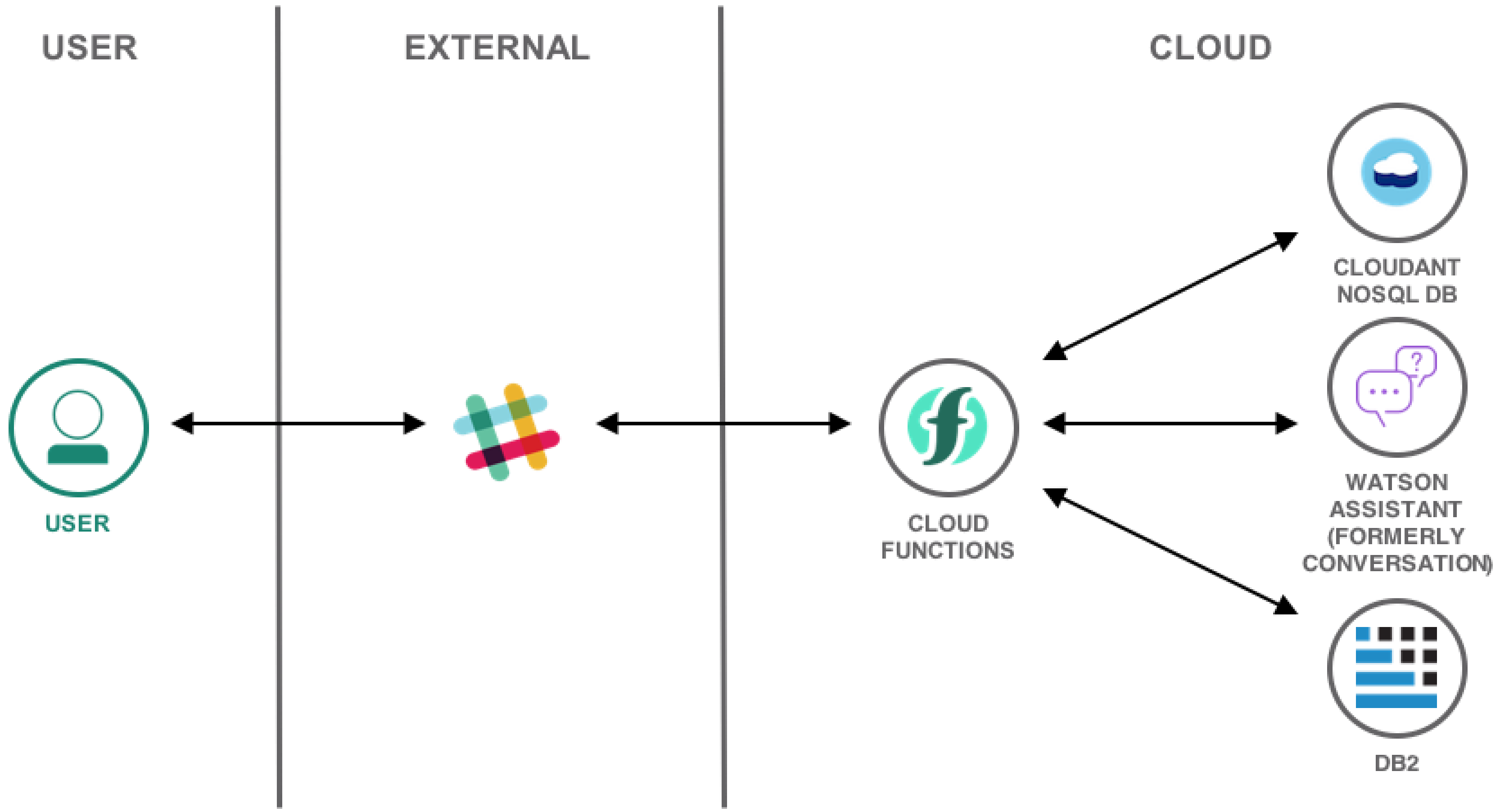


Links:

- *https://cloud.ibm.com/functions/*

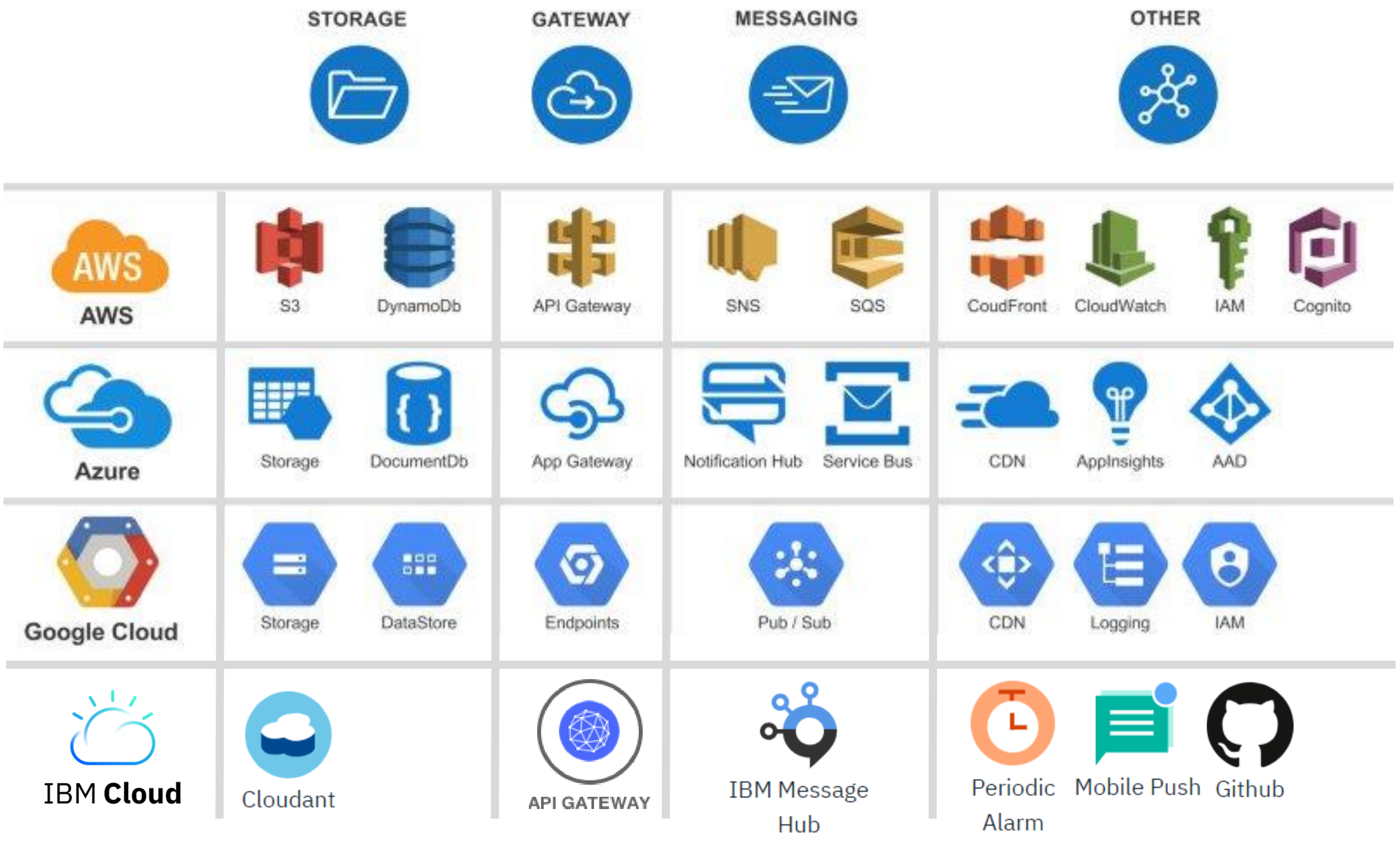
Use-Cases

Conversational Scenarios (database-driven Slackbot):



Links:
- *https://cloud.ibm.com/functions/*

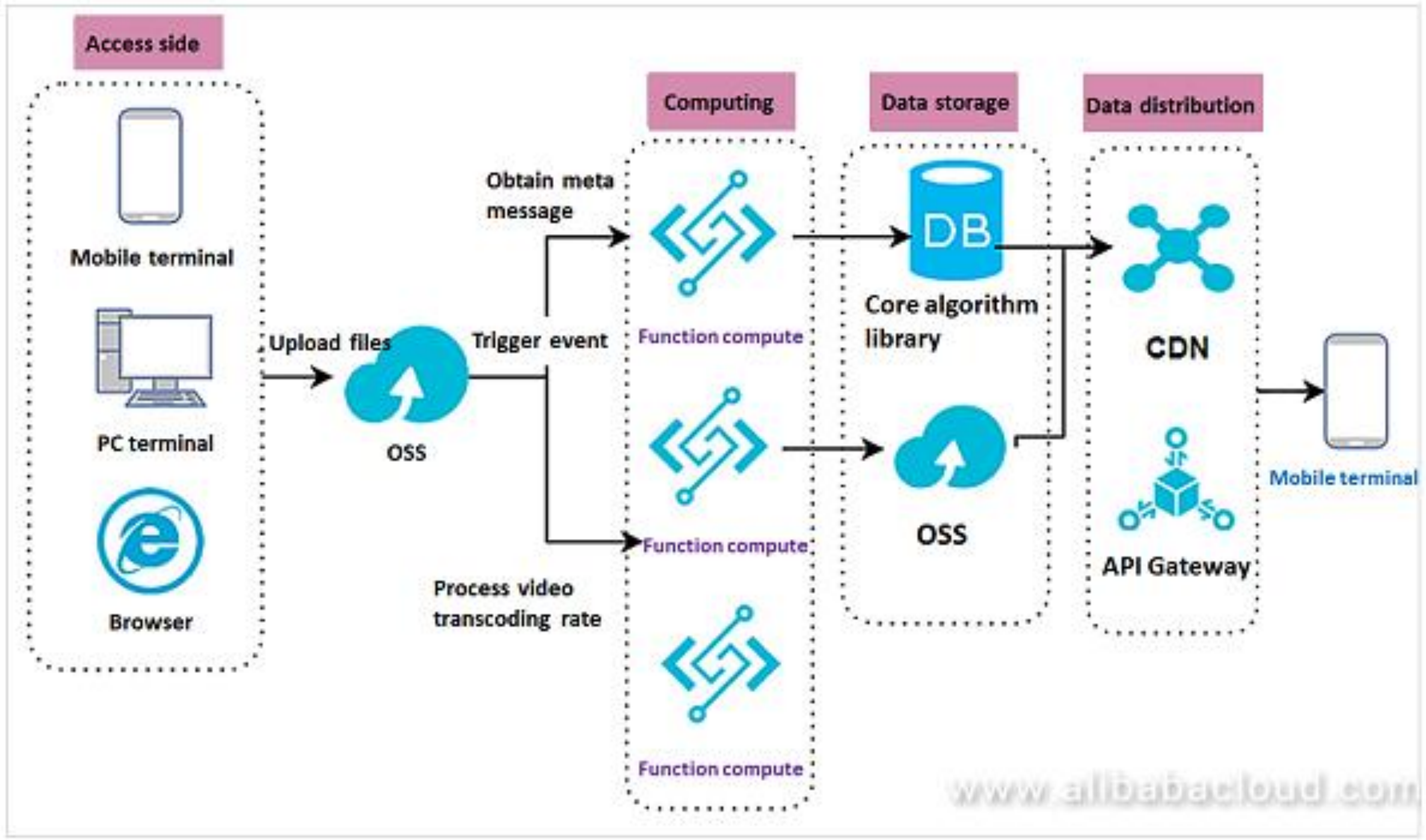
Cloud Triggers



	STORAGE	GATEWAY	MESSAGING	OTHER
AWS	S3, DynamoDb	API Gateway	SNS, SQS	CoudFront, CloudWatch, IAM, Cognito
Azure	Storage, DocumentDb	App Gateway	Notification Hub, Service Bus	CDN, AppInsights, AAD
Google Cloud	Storage, DataStore	Endpoints	Pub / Sub	CDN, Logging, IAM
IBM Cloud	Cloudant	API GATEWAY	IBM Message Hub	Periodic Alarm, Mobile Push, Github

Use-Cases

Obtain and transmit the object metadata, and to synchronize multiple transcoding rates and store the processed video:

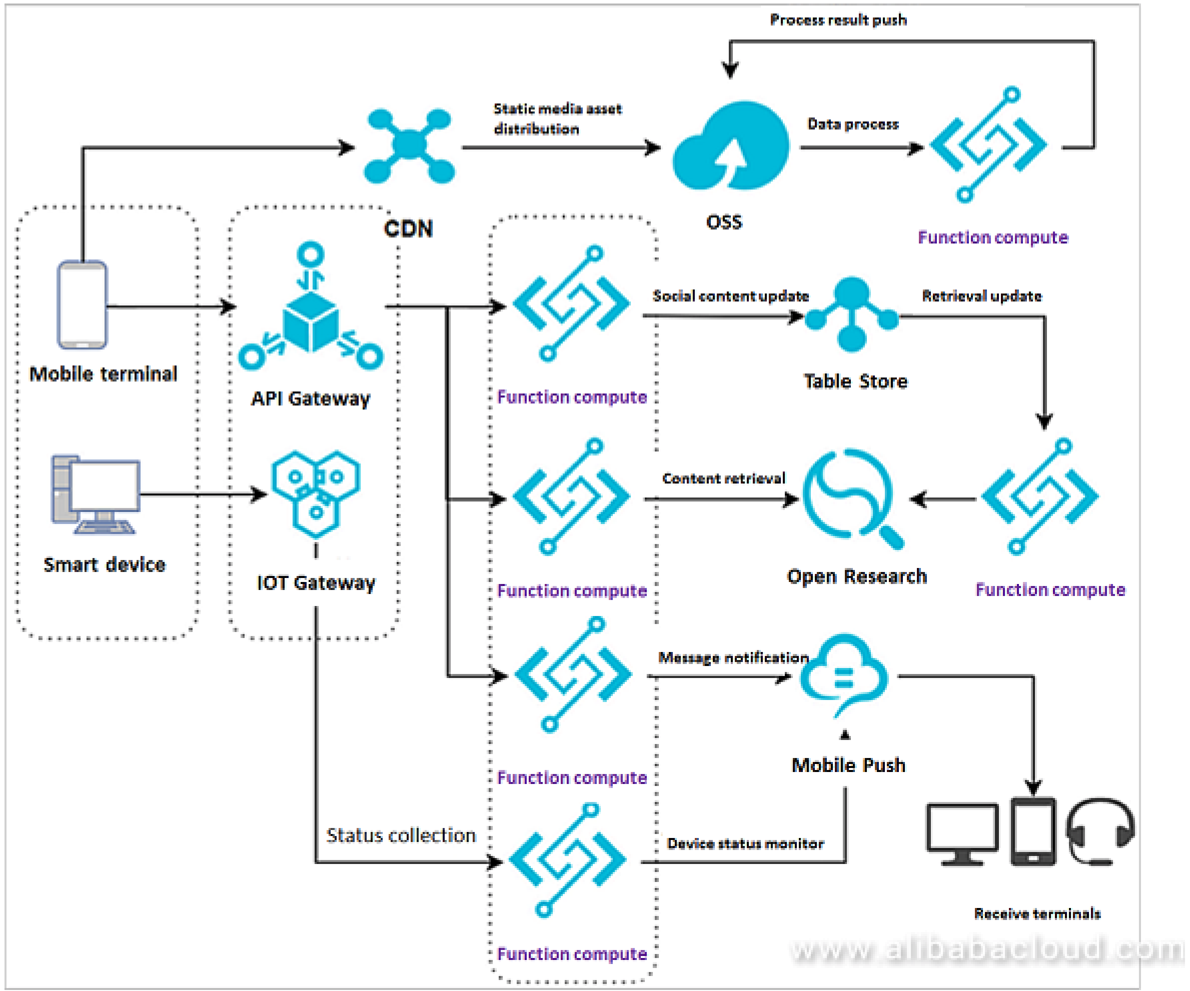


Links:

- *https://dzone.com/articles/4-use-cases-of-serverless-architecture*

Use-Cases

IoT Data Processing:

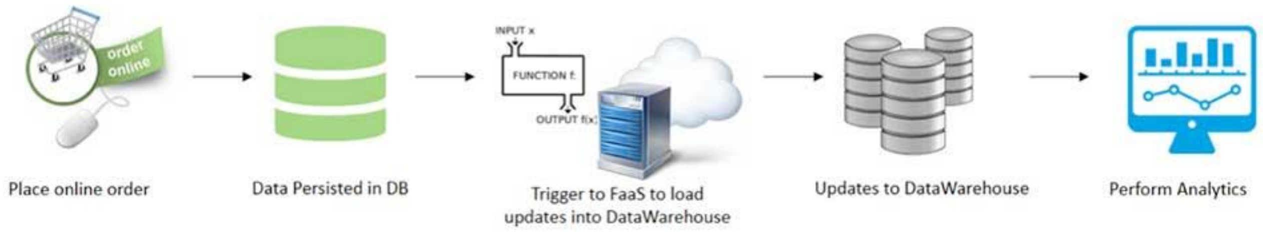


Links:

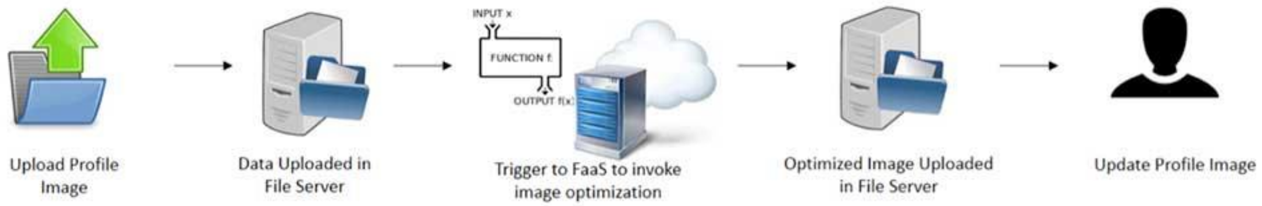
- *https://dzone.com/articles/4-use-cases-of-serverless-architecture*

Use-Cases

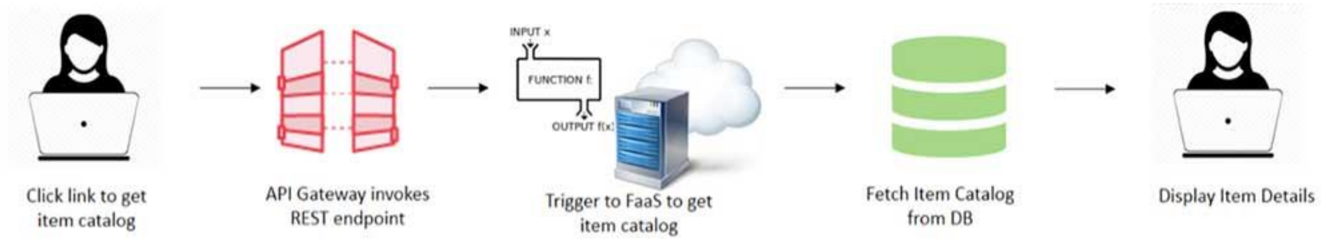
Analytical Reporting of Online Orders:



User Profile Updates:


Upload Profile Image
Data Uploaded in File Server
INPUT x
FUNCTION f:
OUTPUT f(x)
Trigger to FaaS to invoke image optimization
Optimized Image Uploaded in File Server
Update Profile Image

Fetch Latest Item Catalog:

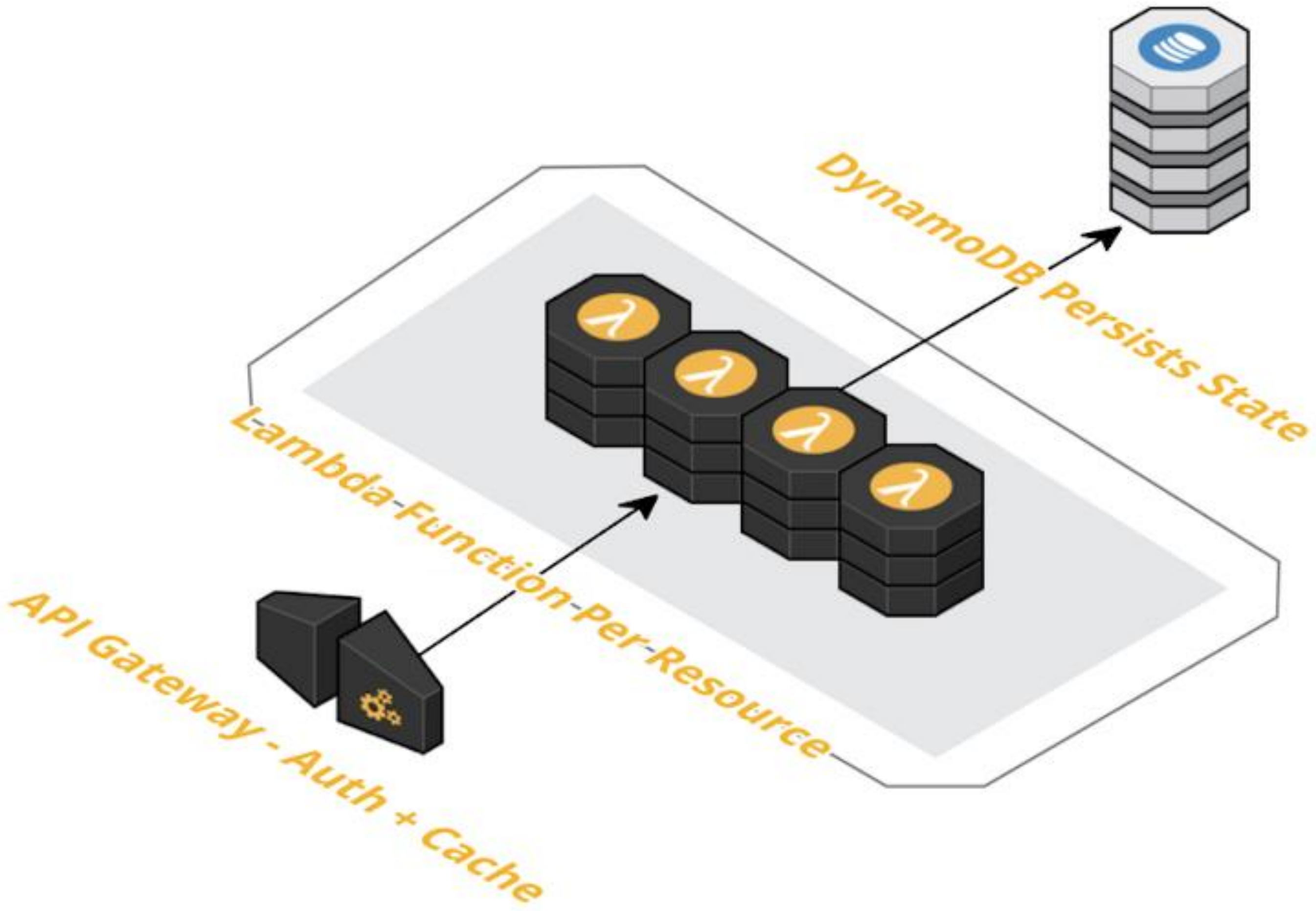

Click link to get item catalog
API Gateway invokes REST endpoint
INPUT x
FUNCTION f:
OUTPUT f(x)
Trigger to FaaS to get item catalog
Fetch Item Catalog from DB
Display Item Details

Links:

- *https://www.jeremydaly.com/making-the-case-for-serverless-use-cases/*

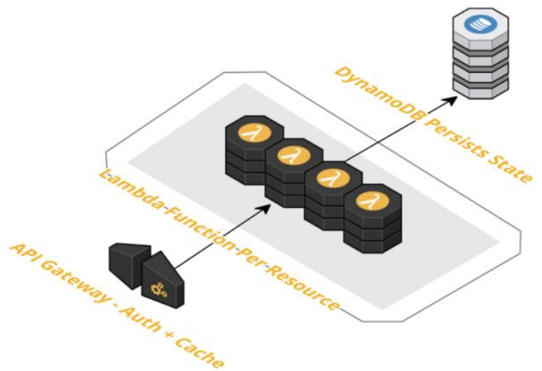
Demo

Go REST with AWS Lambda



Demo

Go REST with AWS Lambda



- *Create **DynamoDB** instance. Provide <name> and <partition key field> of the table to store the resources.*
- *Create **API Gateway** and REST API. Provide <API name> and create corresponding resources and methods. Integrate it with corresponding Lambda Function (will be created next).*
- *Create **Lambda Function**. Provide <name>, chose a runtime (e.g. Python 3.6) and set the Role (e.g. create new role from the template(s)… e.g. "Simple microservice permissions" role). Write simple business logic to test the API Gateway…*
- *Test **API Gateway**. Provide a body of the request in JSON format…*
- *Extend the business logic to store data in DB. Relevant documentation regarding DynamoDB support in Python (https://boto3.amazonaws.com/v1/documentation/api/latest/index.html).*
- *Deploy API. Create/chose a stage of the deployment.*
- *Test API. Use API client (e.g. Postman)*

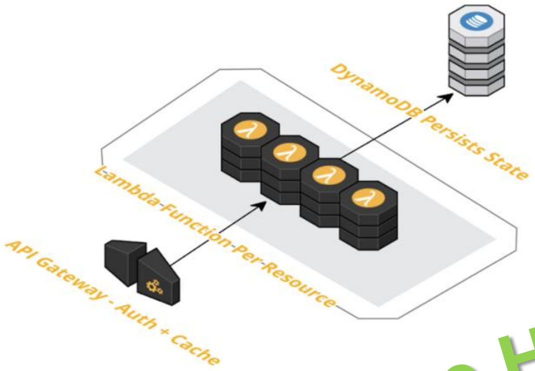
```
import json

def lambda_handler(event, context):
    # TODO implement
    # message = 'Hello from Lambda!'
    message = str(event)
    return {
        "statusCode": 200,
        "body": json.dumps(message)
    }
```

```
{
    "PUBLICATION_ID":"12345",
    "Title":"Publication 001"
}
```

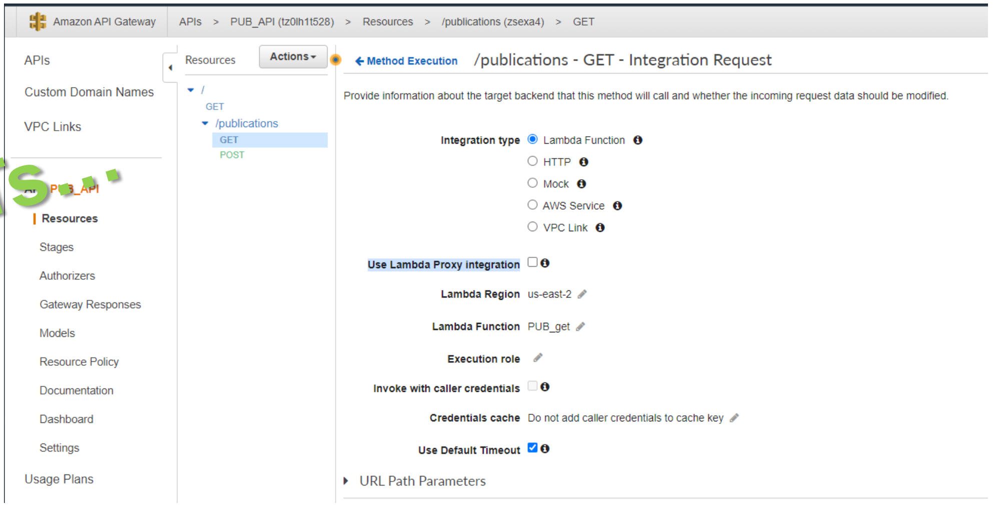
```
import json
import boto3

dynamodb = boto3.resource('dynamodb')
table = dynamodb.Table('PUBLICATIONS')

def lambda_handler(event, context):
    table.put_item(Item = event)
    message = 'Publication Added
                                 Successfully!'
    return {
        "statusCode": 200,
        "message": message
    }
```

Demo

Go REST with AWS Lambda



Some HINTS…


Amazon API Gateway | APIs > PUB_API (tz0lh1t528) > Resources > /publications (zsexa4) > GET

APIs
Custom Domain Names
VPC Links
API: PUB_API
Resources
Stages
Authorizers
Gateway Responses
Models
Resource Policy
Documentation
Dashboard
Settings
Usage Plans

Resources | Actions
- /
  - GET
  - /publications
    - GET
    - POST

← Method Execution /publications - GET - Integration Request

Provide information about the target backend that this method will call and whether the incoming request data should be modified.

Integration type	● Lambda Function ○ HTTP ○ Mock ○ AWS Service ○ VPC Link
Use Lambda Proxy integration	☐
Lambda Region	us-east-2
Lambda Function	PUB_get
Execution role	
Invoke with caller credentials	☐
Credentials cache	Do not add caller credentials to cache key
Use Default Timeout	☑

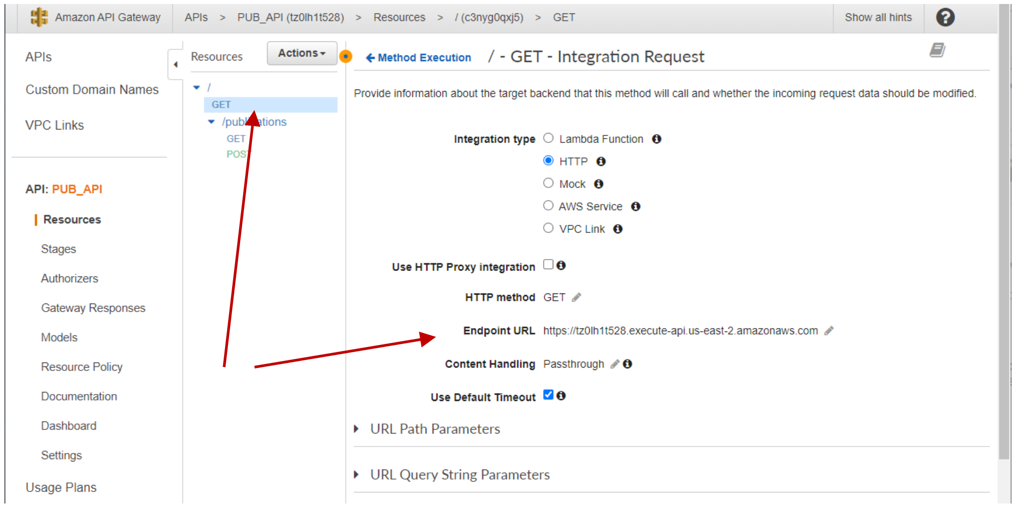
▸ URL Path Parameters

API Gateway

- *While creating the methods under resources, do not tick the "Use Lambda Proxy integration" option…*

However, you may still have an issue with access to the deployed API Gateway with {"message": "Missing Authentication Token"} error…

- *In addition to the recourse(s) added to the root endpoint, add GET method with "HTTP" integration type and set the "Endpoint URL" to the root url of your API Gateway (without the deployment stage).*


Amazon API Gateway | APIs > PUB_API (tz0lh1t528) > Resources > / (c3nyg0qxj5) > GET | Show all hints

APIs
Custom Domain Names
VPC Links
API: PUB_API
Resources
Stages
Authorizers
Gateway Responses
Models
Resource Policy
Documentation
Dashboard
Settings
Usage Plans

Resources | Actions
- /
  - GET
  - /publications
    - GET
    - POST

← Method Execution / - GET - Integration Request

Provide information about the target backend that this method will call and whether the incoming request data should be modified.

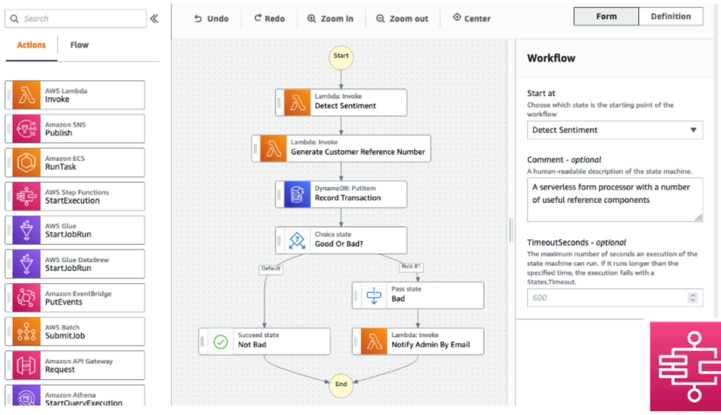
Integration type	○ Lambda Function ● HTTP ○ Mock ○ AWS Service ○ VPC Link
Use HTTP Proxy integration	☐
HTTP method	GET
Endpoint URL	https://tz0lh1t528.execute-api.us-east-2.amazonaws.com
Content Handling	Passthrough
Use Default Timeout	☑

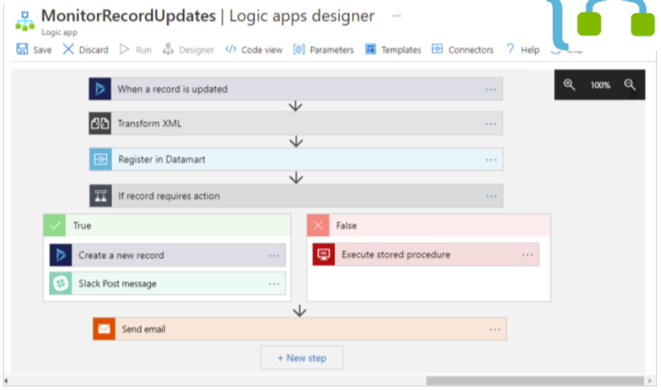
▸ URL Path Parameters

▸ URL Query String Parameters

Developers Experience

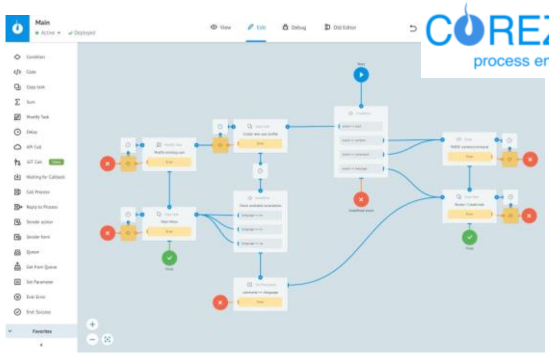
AWS Step Functions *is a visual workflow service that helps developers use AWS services to build distributed applications, automate processes, orchestrate microservices, and create data and machine learning (ML) pipelines.* *https://aws.amazon.com/step-functions*


Search
Actions
Flow
AWS Lambda Invoke
Amazon SNS Publish
Amazon ECS RunTask
AWS Step Functions StartExecution
AWS Glue StartJobRun
AWS Glue DataBrew StartJobRun
Amazon EventBridge PutEvents
AWS Batch SubmitJob
Amazon API Gateway Request
Amazon Athena StartQueryExecution
Undo
Redo
Zoom in
Zoom out
Center
Form
Definition
Start
Lambda: Invoke Detect Sentiment
Lambda: Invoke Generate Customer Reference Number
DynamoDB: PutItem Record Transaction
Choice state Good Or Bad?
Default
Rule #1
Pass state Bad
Succeed state Not Bad
Lambda: Invoke Notify Admin By Email
End
Workflow
Start at
Choose which state is the starting point of the workflow
Detect Sentiment
Comment - optional
A human-readable description of the state machine.
A serverless form processor with a number of useful reference components
TimeoutSeconds - optional
The maximum number of seconds an execution of the state machine can run. If it runs longer than the specified time, the execution fails with a States.Timeout.
600
AWS Step Functions



Azure Logic Apps *is a cloud platform where you can create and run automated workflows with little to no code. By using the visual designer and selecting from prebuilt operations, you can quickly build a workflow that integrates and manages your apps, data, services, and systems.* *https://azure.microsoft.com/en-us/products/logic-apps/#overview, https://learn.microsoft.com/en-us/azure/logic-apps/logic-apps-overview*

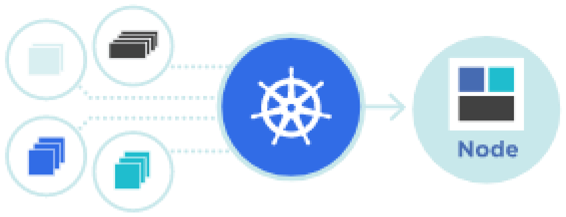
Corezoid Process Engine *provides a Platform-as-a-Service cloud operating system that enables companies to build agile business processes, helps companies build, manage, host and run their processes in the cloud without coding.* *https://corezoid.com, https://aws.amazon.com/marketplace/pp/prodview-dbnuv3b3g5jf2*



Serverless Functions on Containers

***Kubernetes**, also known as K8s, is an open-source system for automating deployment, scaling, and management of containerized applications. It groups containers that make up an application into logical units for easy management and discovery.* *https://kubernetes.io*

https://kubernetes.io/docs/concepts/overview/



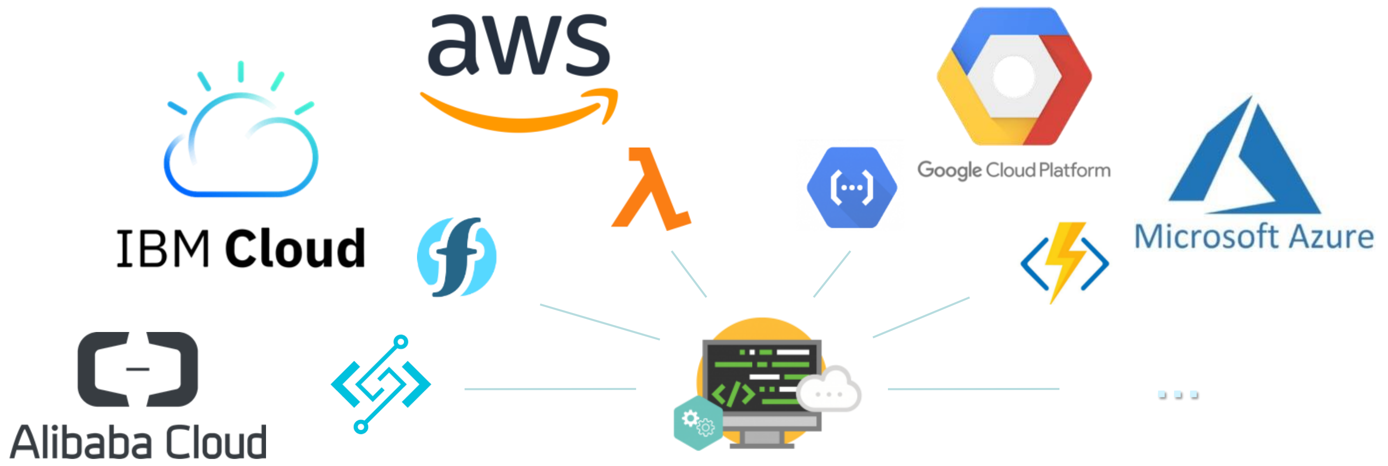
***Fission** is an open-source Serverless Framework for serverless functions on Kubernetes.* *https://fission.io*

- *Write short-lived functions in any language and map them to HTTP requests (or other event triggers).*
- *Deploy functions instantly with one command. There are no containers to build, and no Docker registries to manage.*

***Kubeless** is an open-source serverless computing framework run on top of Kubernetes and allows deploying code without having to worry about infrastructure using Kubernetes resources to provide auto-scaling, routing, monitoring, and troubleshooting.*
https://github.com/vmware-archive/kubeless, https://www.xenonstack.com/insights/kubeless

Kubeless

Serverless Frameworks



*Serverless platforms need infrastructures where they can be executed, **provider agnostic frameworks** provide a platform agnostic way to define and deploy Serverless code on various cloud platforms or commercial services. This is an option to avoid (or reduce) vendor lock-in without the necessity to operate an own infrastructure.*

- *Serverless Framework (Javascript, Python, Go) https://serverless.com/*
- *Zappa (Python) https://www.zappa.io/*
- *AWS Chalice (Python) https://aws.github.io/chalice/*
- *Apex (Javascript) http://apex.run/*
- *ClaudiaJS (Javascript) https://claudiajs.com/*
- *Flogo(Go, AWS) https://tibcosoftware.github.io/flogo/*
- *Sparta (Go) https://gosparta.io/*
- *Gordon (Javascript) https://github.com/jorgebastida/gordon/*
- *Up (Javascript, Python, Go, Crystal) https://github.com/apex/up*
- *…*

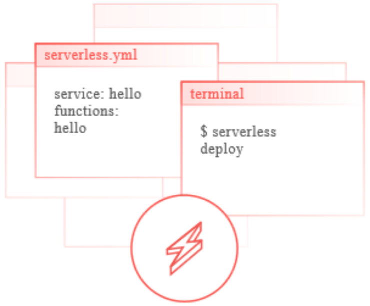
Links:

- *https://www.interviewbit.com/blog/serverless-frameworks/*

Serverless Frameworks

Serverless Platform *empowers developers to build & deploy serverless applications, on any provider (https://serverless.com/).*

Everything you need to operationalize serverless development…



Serverless Framework *is an open-source CLI for building and deploying serverless applications. The easy, open way to build serverless applications quickly on any vendor.*

Serverless Dashboard *allows to observe and monitor your functions in action:*

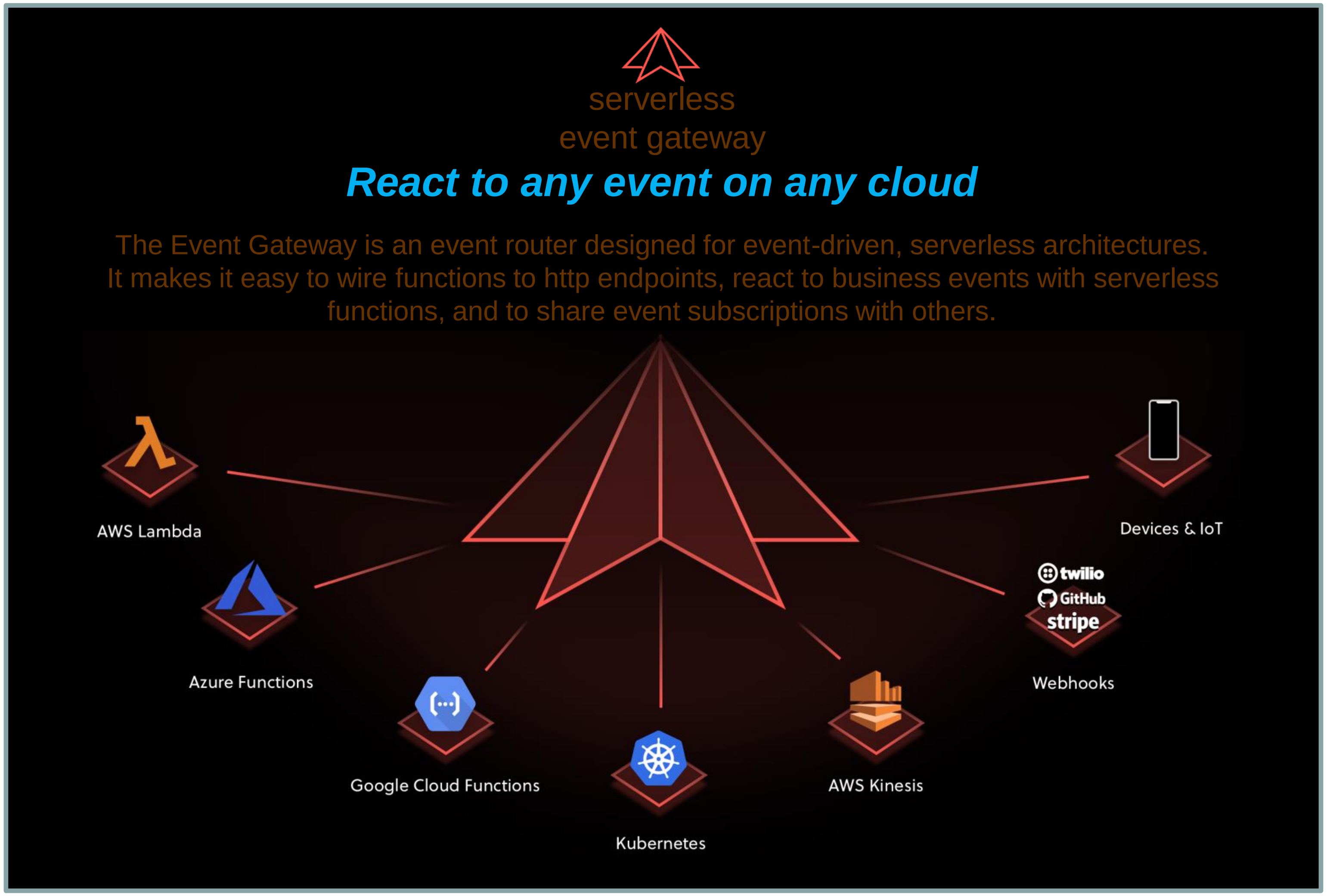
- *shared overview of the functions, events, and subscriptions*
- *application logs and metrics from within the UI to ease debugging and increase velocity*
- *control security and compliance needs by managing which team members have access to which cloud resources*

Event Gateway *is an event router designed for event-driven, serverless architectures:*

- *simplest developer experience for wiring Serverless functions to http endpoints*
- *expand Serverless use-cases by reacting to any custom event with Serverless functions*
- *it is open source and cloud agnostic, allowing for more choice and flexibility*

Links:
- *https://serverless.com/framework/docs/*

Serverless Event Gateway



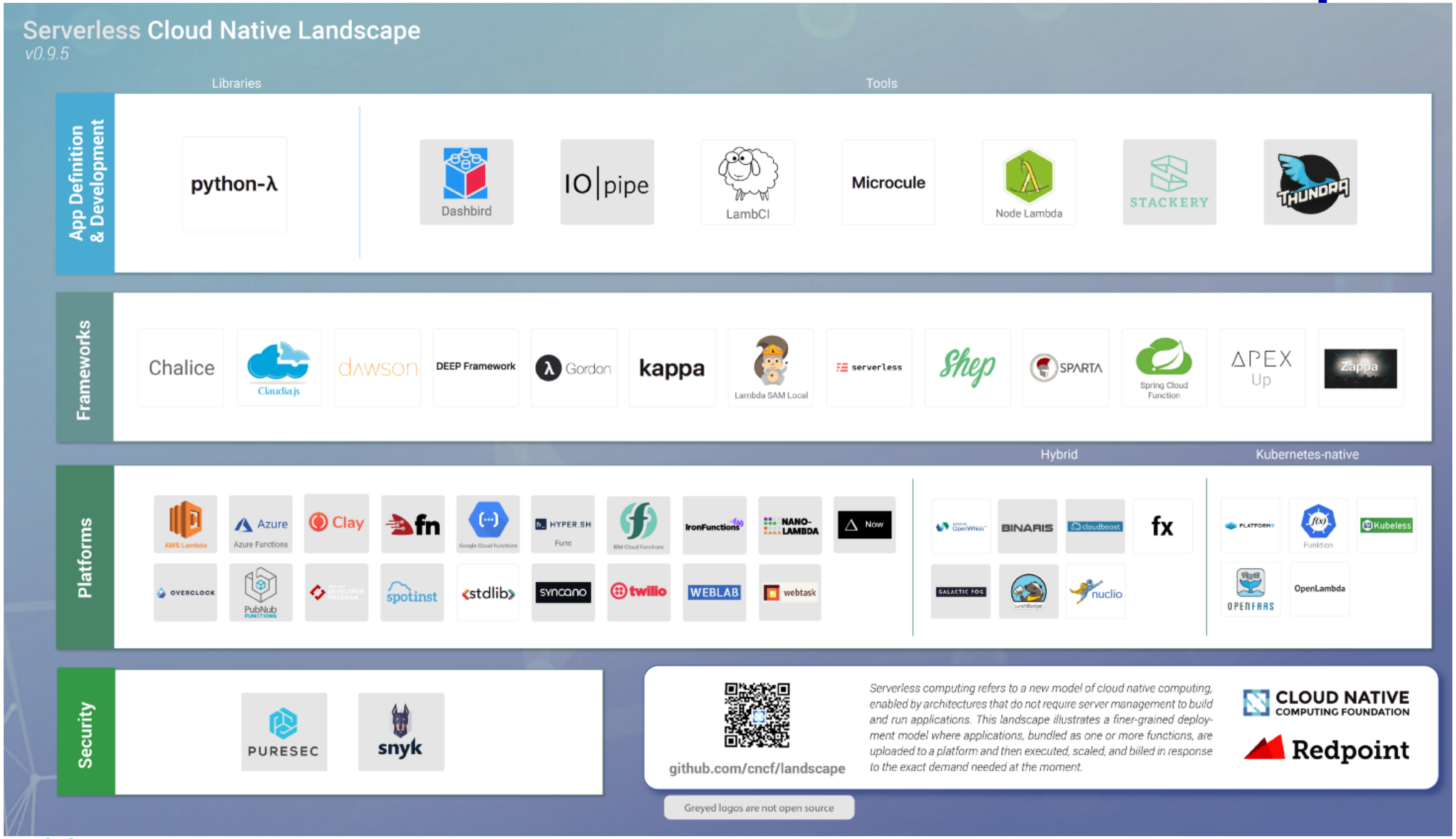
React to any event on any cloud

The Event Gateway is an event router designed for event-driven, serverless architectures. It makes it easy to wire functions to http endpoints, react to business events with serverless functions, and to share event subscriptions with others.

Links:

- *https://www.youtube.com/watch?v=Va6Ve9oKNyY*
- *https://github.com/serverless/event-gateway*

Serverless Landscape



Serverless computing refers to a new model of cloud native computing, enabled by architectures that do not require server management to build and run applications. This landscape illustrates a finer-grained deployment model where applications, bundled as one or more functions, are uploaded to a platform and then executed, scaled, and billed in response to the exact demand needed at the moment.

Links:

- *https://medium.com/memory-leak/serverless-cloud-native-landscape-new-from-redpoint-ventures-and-the-cloud-native-computing-181711d885f7*
- *https://www.youtube.com/watch?v=uMCtcZ46gns*
- *https://www.interviewbit.com/blog/serverless-frameworks/*

Task 6

Related courses

- **TIES4520** - *Semantic Technologies for Developers (7 ECTS)*

(http://users.jyu.fi/~olkhriye/ties4520)
Semantic Technologies are designed to standardize and support interoperability and integration of information content and capabilities (services) of Web-based systems and components at local and global scale. As a software technology semantic technology encodes meanings separately from data and from application code to enable machines to understand, share and reason with them at execution time. This course provides practical introduction on storing, querying, merging, matchmaking and reasoning with the metadata and ontologies for the semantic applications; as well as covers aspects of semantic programming and Linked Data. Same time, the course addresses the challenge related to the Web of Things where all the things are interconnected and interoperate with each other regardless of whether a thing is a real world object, a digital entity or human. Following the vision of Everything-as-a-Service-Consumer and Everything-as-a-Service-Provider, in addition to a traditional GUI (where a user of applications and services is assumed to be a human), the course concerns interfaces needed if a user of the application happens to be not a human but some other application, service, devise or anything else. Such Everything-to-Everything interfaces in addition to the traditional APIs have to enable understanding among interacting entities, which requires either sharing common ontology or the support for the ontology alignment process followed by semantically enhanced interaction. The course will review available techniques and tools for practical design of such interfaces.

- **TIES4911** - *TIES4911 Deep-Learning for Cognitive Computing for Developers (8-10 ECTS)*

(http://users.jyu.fi/~olkhriye/ties4911)
By any measure, the past few years have been landmark years for the discussion around Artificial Intelligence and its potential impact on business and society. Being based on Artificial Intelligence, Cognitive Computing Systems are "systems that learn at scale, reason with purpose and interact with humans naturally". Cognitive Computing solutions encompass Machine Learning, Reasoning, Natural Language Processing, Deep Learning, Speech and Vision, Human-Computer Interaction and more. The course aims to provide practical view to the domain of Cognitive Computing and Machine Intelligence. Students will be capable to design and build own services and apps using cloud-based Cognitive Services of such big competing player in this field as IBM, Google, Microsoft, Amazon, etc. At the same time, students will learn how to build Machine Intelligence based solutions using corresponding open-source software libraries (e.g. TensorFlow). Python programming language is used in practical tasks implementation.

Relevant links

- *https://www.youtube.com/watch?v=wWEID0d6wfo*
- *https://www.youtube.com/watch?v=qnVfWG8N7Fw*
- *https://www.youtube.com/watch?v=Y711Cbb-g5w*
- *https://www.gocd.org/2017/06/26/serverless-architecture-continuous-delivery/*
- *https://hackernoon.com/what-is-serverless-architecture-what-are-its-pros-and-cons-cc4b804022e9*
- *https://medium.com/@MarutiTech/what-is-serverless-architecture-what-are-its-criticisms-and-drawbacks-928659f9899a*
- *https://hackernoon.com/aws-lambda-serverless-framework-python-part-1-a-step-by-step-hello-world-4182202aba4a*
- *https://www.globallogic.com/gl_news/serverless-architecture-evolution-of-a-new-paradigm/*
- *https://medium.com/memory-leak/serverless-cloud-native-landscape-new-from-redpoint-ventures-and-the-cloud-native-computing-181711d885f7*
- *https://archbee.io/blog/why-serverless-is-not-there-yet/*
- *https://www.alibabacloud.com/blog/serverless-computing-with-alibaba-cloud-function-compute_593960*