CF-BBA-01

Despliegue de microservicios en la nube

David Victoria
(he/him)
Cloud Architect
Caylent

¿Quién soy?

Orgullosamente mexicano

Cloud Architect @ **Caylent**

Casado y ¡esperando nuestro primer hijo! *(Sep-24)*

8x AWS Certified, AWS Ambassador

Me encanta la comunidad de AWS, la cerveza artesanal y viajar

¿Mencioné que me encanta la cerveza artesanal? De hecho, **¡vamos por una después de clase!**

¿Qué soy?

Builder

Builder

[bil·dr]

A person *(with cool swag)* that creates or develops a particular thing.

Synonyms: Wizard, Creator, Mage

Agenda

01 Monolitos

02 Microservicios

03 Despliegue en la nube

04 ¿Dónde corren?

05 Consideraciones adicionales

06 Demo

El viejo conocido: Monolito

Un dependencia están interconectados en un solo componente. monolito es una arquitectura de software donde todas las funcionalidades de una aplicación **están empaquetadas y ejecutadas dentro de un solo proceso**. Esto significa que el código, la base de datos y cualquier otra

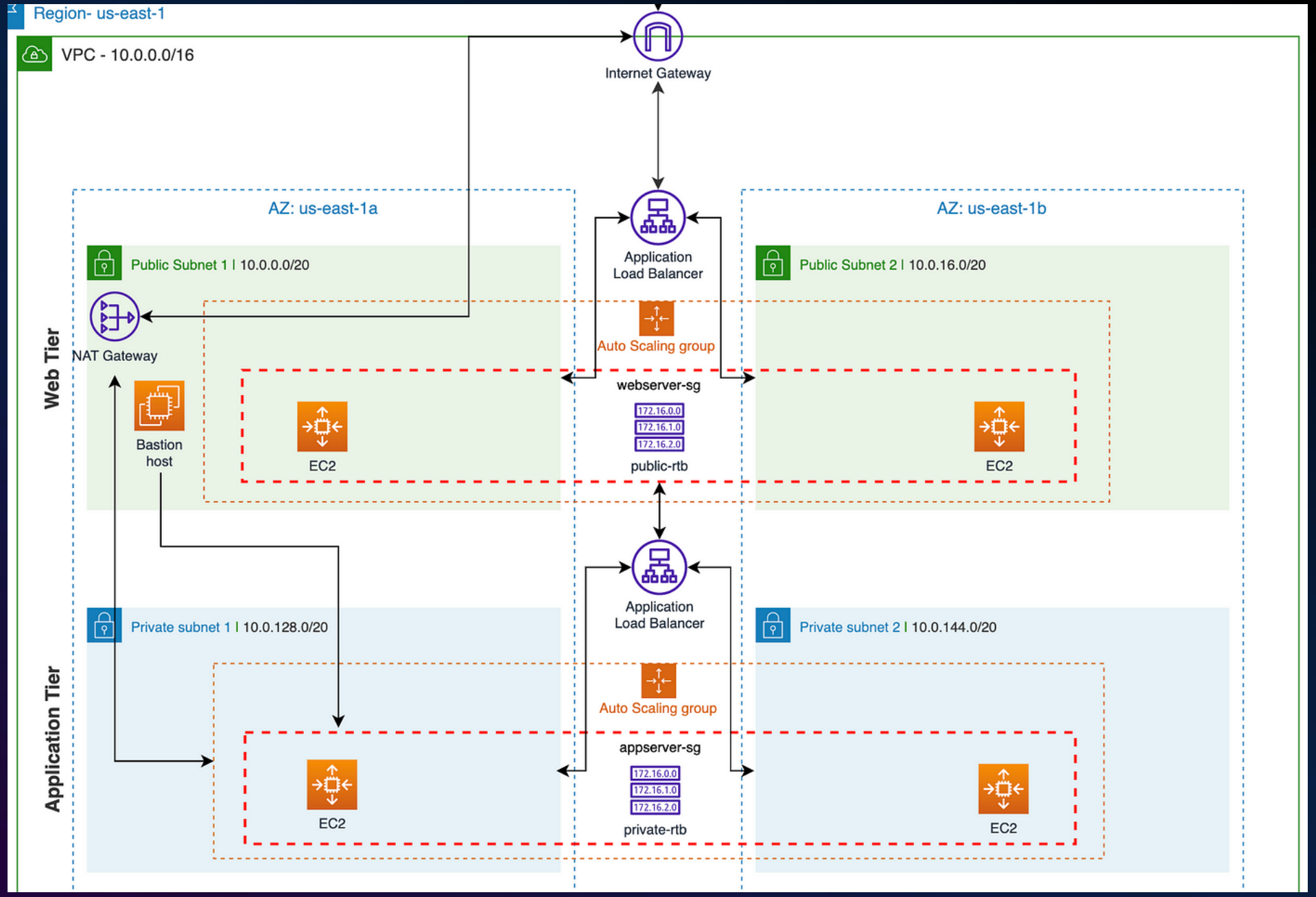
Ventajas de los monolitos

Simplicidad

Menor complejidad en despliegues

Facil depuración

Menos requerimientos de escalabilidad

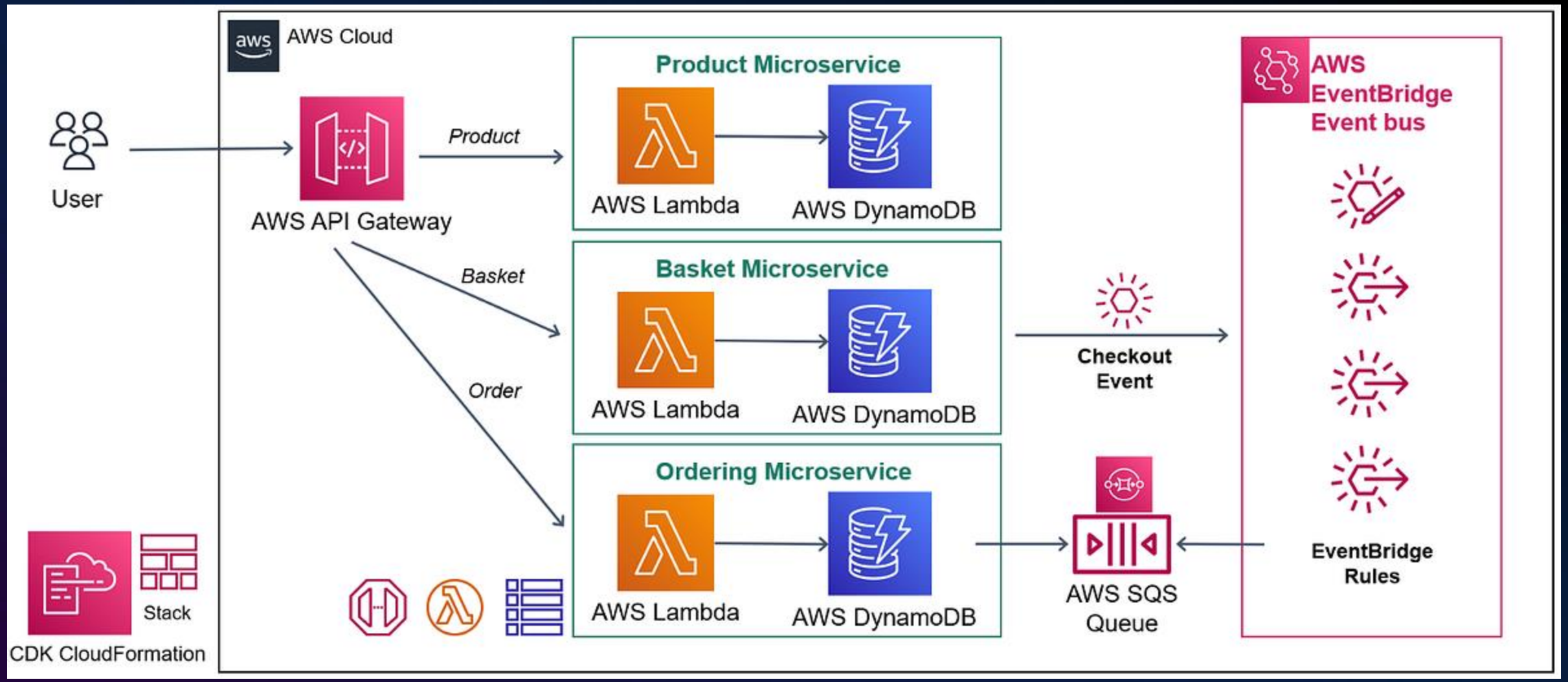

Region- us-east-1
VPC - 10.0.0.0/16
Internet Gateway
AZ: us-east-1a
AZ: us-east-1b
Application Load Balancer
Public Subnet 1 | 10.0.0.0/20
Public Subnet 2 | 10.0.16.0/20
NAT Gateway
Auto Scaling group
webserver-sg
172.16.0.0
172.16.1.0
172.16.2.0
public-rtb
Bastion host
EC2
EC2
Web Tier
Application Load Balancer
Private subnet 1 | 10.0.128.0/20
Private subnet 2 | 10.0.144.0/20
Auto Scaling group
appserver-sg
172.16.0.0
172.16.1.0
172.16.2.0
private-rtb
EC2
EC2
Application Tier

¿Microservicios?

Los microservicios son un enfoque arquitectónico y organizativo para el desarrollo de software donde el software está compuesto por pequeños servicios independientes que se comunican a través de API bien definidas. **Los propietarios de estos servicios son equipos pequeños independientes.**

Microservicios

- Desarrollo de software ágil
- Arquitecturas orientadas a servicios
- Diseño que prioriza las API
- Integración continua y entrega continua (CI/CD)


AWS Cloud
User
AWS API Gateway
Product
Basket
Order
Product Microservice
AWS Lambda
AWS DynamoDB
Basket Microservice
AWS Lambda
AWS DynamoDB
Ordering Microservice
AWS Lambda
AWS DynamoDB
Checkout Event
AWS SQS Queue
AWS EventBridge Event bus
EventBridge Rules
Stack
CDK CloudFormation

Mayores diferencias entre arquitecturas

Categoría	Arquitectura monolítica	Arquitectura de microservicios
Diseño	Código base único con múltiples funciones interdependientes.	Componentes de software independientes con funcionalidad autónoma que se comunican entre sí mediante API.
Desarrollo	Al principio requiere menos planificación, pero se hace cada vez más compleja de entender y mantener.	Al principio requiere más planificación e infraestructura, pero se hace más fácil de administrar y mantener con el tiempo.
Implementación	Toda la aplicación se implementó como una sola entidad.	Cada microservicio es una entidad de software independiente que requiere una implementación individual en contenedores.
Depuración	Hace un seguimiento de la ruta de acceso al código en el mismo entorno.	Requiere herramientas de depuración avanzadas para hacer un seguimiento del intercambio de datos entre varios microservicios.

Mayores diferencias entre arquitecturas

Categoría	Arquitectura monolítica	Arquitectura de microservicios
Modificación	Pequeños cambios introducen mayores riesgos, ya que afectan a todo el código base.	Puede modificar microservicios individuales sin afectar a toda la aplicación.
Escalado	Debe escalar toda la aplicación, incluso si solo ciertas áreas funcionales experimentan un aumento de la demanda.	Puede escalar microservicios independientes según sea necesario, lo que ahorra costos generales de escalado.
Inversión	Baja inversión inicial a costa de aumentar los esfuerzos continuos y de mantenimiento.	Inversión adicional de tiempo y costos para configurar la infraestructura requerida y desarrollar la competencia del equipo. Sin embargo, ahorro de costos a largo plazo, mantenimiento y adaptabilidad.

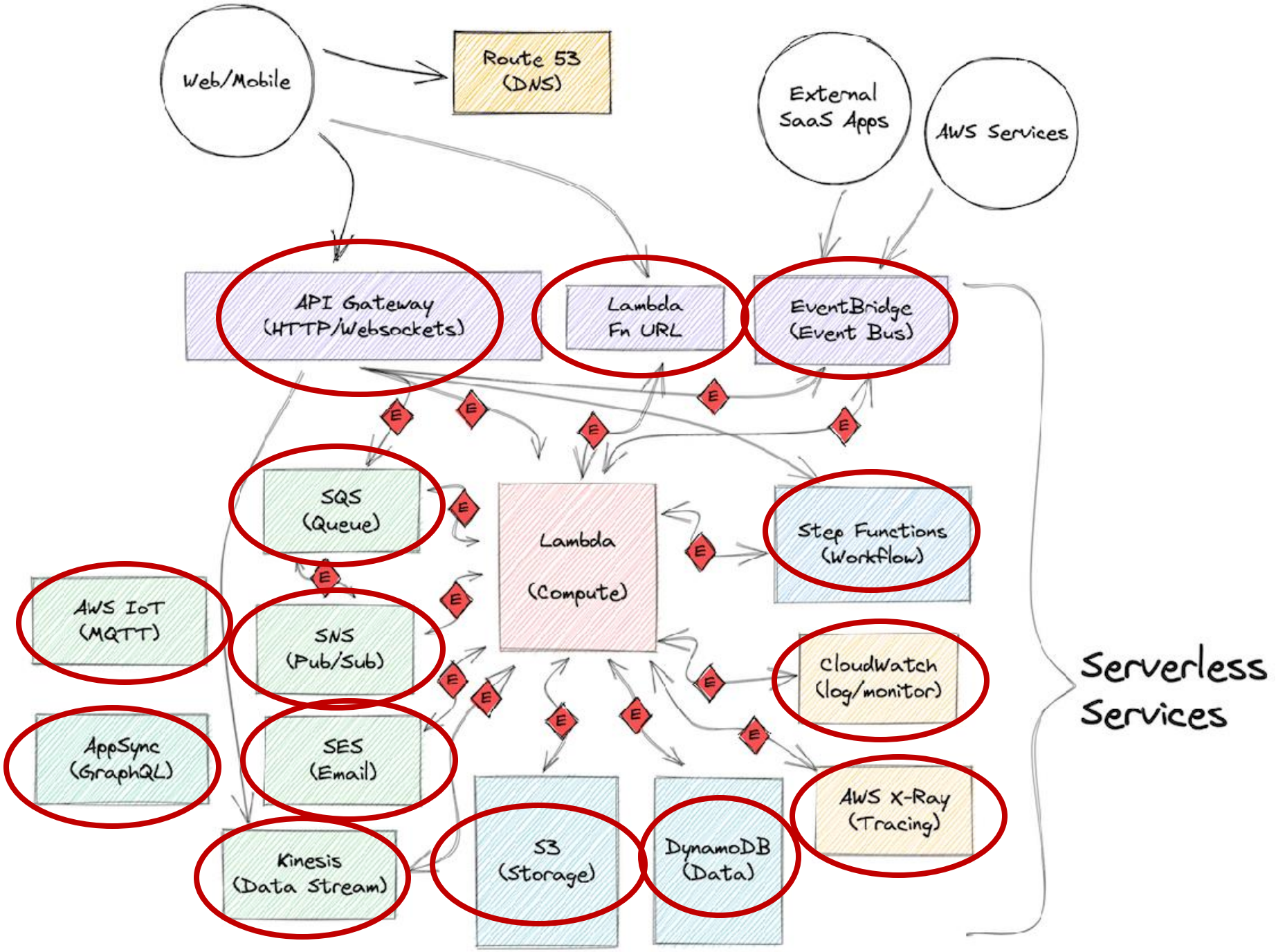
¿Microservicios == serverless? –Not really–

- Inexistencia de aprovisionamiento y administración de infraestructura
- Ajuste de escala automático por unidad de consumo
- Modelo de facturación de pago por valor
- Disponibilidad y tolerancia a errores integradas

Sin embargo… deshacerse de los servidores es una excelente manera de eliminar la complejidad operativa.

APIs

Las API son la puerta principal de los microservicios, lo que significa que las API funcionan como **el punto de entrada** para la lógica de aplicaciones detrás de un conjunto de interfaces de programación, normalmente una API de servicios web RESTful, sin embargo, no es el único medio para invocar un servicio.


Web/Mobile
Route 53
(DNS)
External
SaaS Apps
AWS Services
API Gateway
(HTTP/Websockets)
Lambda
Fn URL
EventBridge
(Event Bus)
SQS
(Queue)
Lambda
(Compute)
Step Functions
(Workflow)
AWS IoT
(MQTT)
SNS
(Pub/Sub)
CloudWatch
(log/monitor)
AppSync
(GraphQL)
SES
(Email)
AWS X-Ray
(Tracing)
Kinesis
(Data Stream)
S3
(Storage)
DynamoDB
(Data)
Serverless
Services

Despliegue en la nube

CI/CD

Integración continua: la práctica de fusionar todas las copias de trabajo del desarrollador en una línea principal compartida varias veces al día.

CI/CD

Entrega continua: un enfoque en el que los equipos producen software en ciclos cortos, garantizando que el software pueda lanzarse de manera confiable en cualquier momento, pero que aún depende de la intervención humana para determinar qué se pone en producción.

Implementación continua: lleva la entrega continua un paso más allá al implementar automáticamente todas las versiones en producción. Mientras que la entrega continua exige que el software se pueda lanzar de manera confiable en cualquier momento, no requiere que cada versión se lance a producción.

CI/CD

V1.1
CONTINUOUS INTEGRATION
CONTINUOUS DELIVERY
APPROVE DEPLOY
CONTINUOUS DEPLOYMENT
AUTOMATIC DEPLOY
AUTOMATED
AUTOMATED
SOURCE CONTROL
COMMIT CHANGES
BUILD
RUN BUILD AND UNIT TESTS
STAGING
DEPLOY TO TEST ENVIRONMENT
RUN INTEGRATION TESTS, LOAD TESTS, AND OTHER TESTS
PRODUCTION
DEPLOY TO PRODUCTION ENVIRONMENT

¿En qué servicios me apoyo?

AWS proporciona una serie de servicios que facilitan la creación de canalizaciones de CI/CD y la posibilidad de crear plantillas para estas canalizaciones para que puedan reutilizarse para múltiples servicios.

Los servicios relevantes de AWS incluyen:

- AWS CodeCommit
- AWS CodeBuild
- AWS CodeDeploy
- AWS CodePipeline
- AWS CloudFormation

¿Y en Azure?

- AWS CodeCommit = GitHub
- AWS CodeBuild = GitHub Actions/Azure Pipelines
- AWS CodeDeploy = Azure DevOps
- AWS CodePipeline = Azure DevOps
- AWS CloudFormation = Azure Resource Manager

https://learn.microsoft.com/es-es/azure/architecture/aws-professional/services

¿Y en GCP?

- AWS CodeCommit = Google Cloud Deploy
- AWS CodeBuild = Cloud Build
- AWS CodeDeploy = Google Cloud Deploy
- AWS CodePipeline = Cloud Build
- AWS CloudFormation = Cloud Deployment Manager

https://cloud.google.com/docs/get-started/aws-azure-gcp-service-comparison?hl=es-419

¿Dónde pueden correr?

Capa de cómputo

Amazon Elastic Container Service
Amazon Elastic Kubernetes Service

Managed or serverless services

Elastic Compute Cloud

Infrastructure-as-a-service

AWS Lambda

Serverless service

Registro de imágenes

Amazon ECR

¿Persistencia?

Amazon DynamoDB

Amazon ElastiCache

Amazon S3

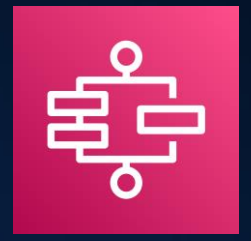
Amazon Aurora

Orquestación y coreografía

Amazon Simple Notification Service

Amazon Simple Queue Service

Amazon EventBridge

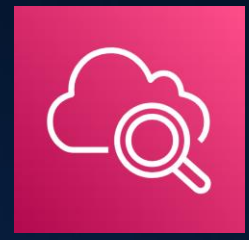


Amazon Step Functions

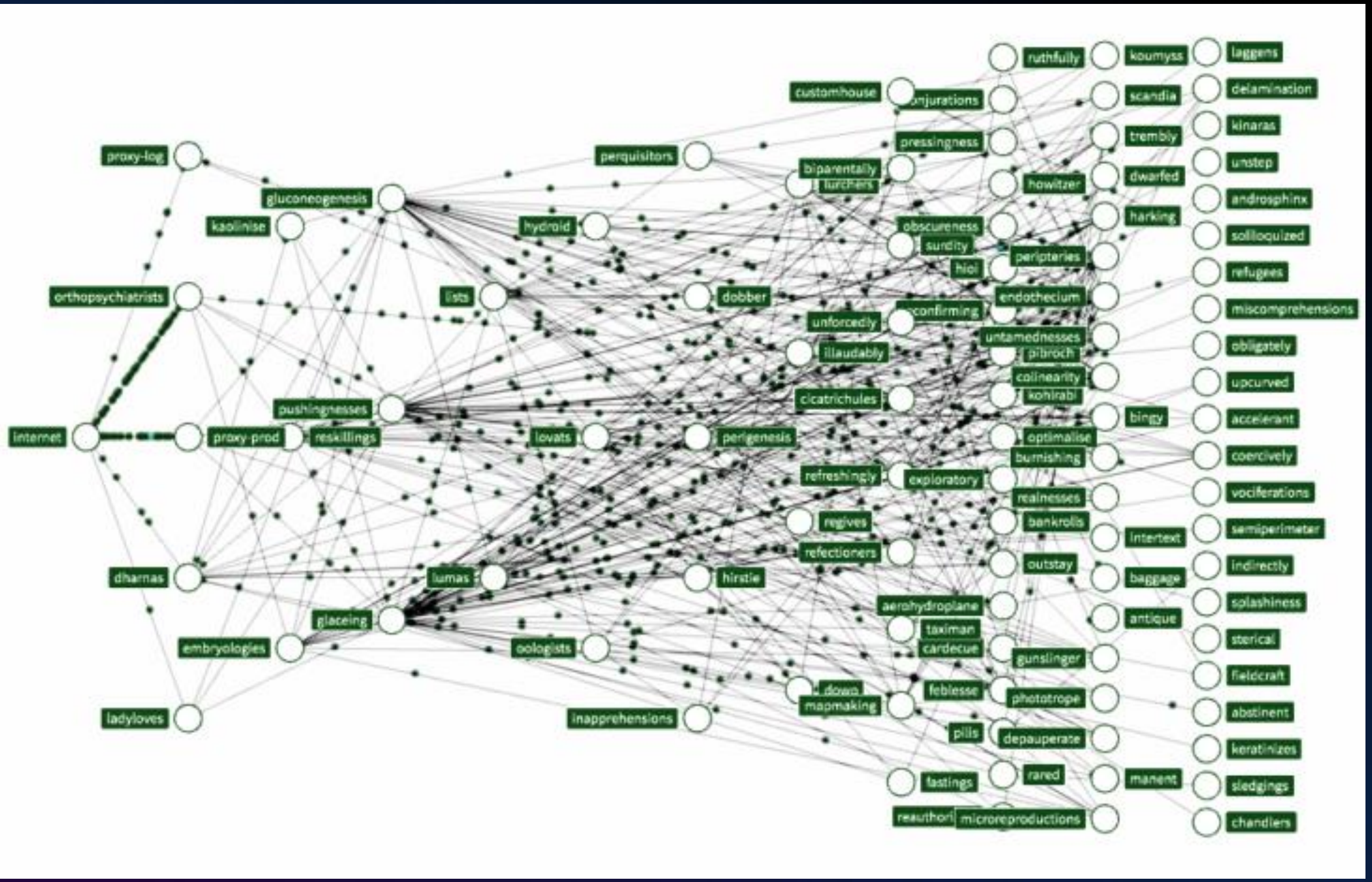
¡Monitoreo! ~~(y sus nuevos retos)~~

Amazon CloudWatch

AWS X-Ray

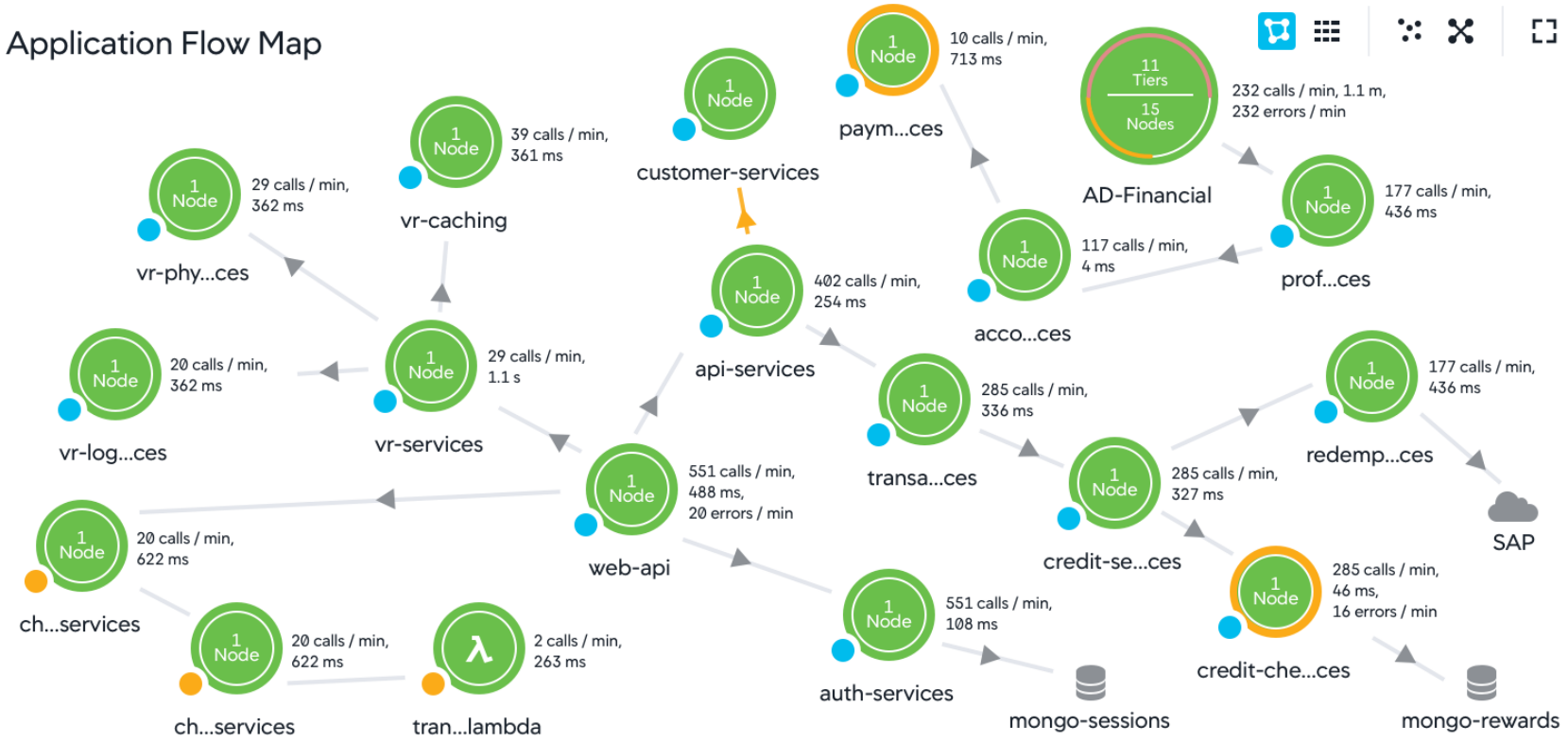


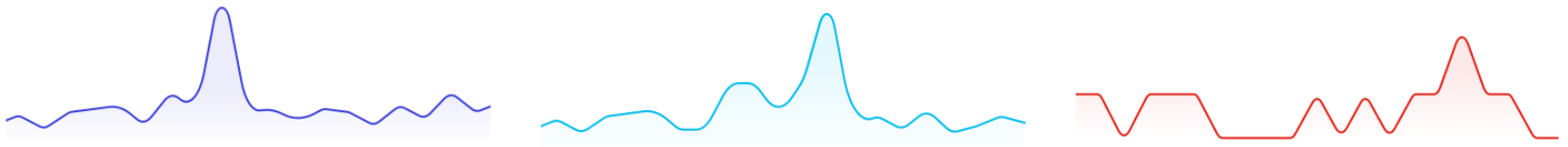
AWS CloudTrail


proxy-log
gluconeogenesis
kaolinise
orthopsychiatrists
internet
proxy prod
reskillings
pushingnesses
dharmas
embryologies
ladyloves
glaceing
perquisitors
hydroid
lists
lovats
lumas
oologists
inapprehensions
dobber
perigenesis
hirstie
biparentally
lurchers
unforcedly
illaudably
cicatrichules
refreshingly
regives
refectioners
customhouse
pressingness
obscureness
surdity
hiol
confirming
aerohydroplane
taximan
cardecue
feblesse
mapmaking
ruthfully
howitzer
peripteries
endothecium
untamednesses
pibroch
collinearity
kohlrabi
optimalise
burnishing
exploratory
realnesses
bankrolls
outstay
gunslinger
phototrope
depauperate
rared
microreproductions
koumyss
scandia
trembly
dwarfed
harking
bingy
intertext
baggage
antique
manent
laggens
delamination
kinaras
unstep
androsphinx
soliloquized
refugees
miscomprehensions
obligately
upcurved
accelerant
coercively
vociferations
semiperimeter
indirectly
splashiness
sterical
fieldcraft
abstinent
keratinizes
sledgings
chandlers

NextGen Financial

Application Flow Map





Business Transaction Health

2 critical, 0 warning, 36 normal

Node Health

0 critical, 2 warning, 18 normal

Server Health

Transaction Scorecard

	97.2%
	1.0%
	0.2%
	0.0%
	1.6%

Exceptions

	- total	- / min
	- total	- / min

Demo

microservices
serverless
Multi AZ
auto scaling
COCOA PEBBLES
YOUR APP
WITH 0 USERS
imgflip.com

¡Gracias!

David Victoria

@vikomex

linkedin.com/in/vikomex

hi@davidvictoria.com