



AWS  
**Architect &  
Developer**  
Series

# Microservices Architectures on AWS

Adam Lynch – Snn. Technical Account Manager

September 2016



## Overview



## Microservices Patterns on AWS

## Challenges of Microservices

## Demo

# Microservices

“Microservices is an approach to **application development** in which a **large application** is built as a suite of **modular services**. Each module supports a specific **business goal** and uses a **simple, well-defined** interface to communicate with other modules”

# Microservices?

## Related Concepts

- Service Oriented Architectures
- API First
- Agile Software Development
- Continuous Delivery
- **DevOps**

# Benefits of Microservices

## Speed

- Faster development and deployment

# Benefits of Microservices

## Speed

- Faster development and deployment

## Innovation

- Autonomy of teams, culture of change
- Ownership and DevOps culture

# Benefits of Microservices

## Speed

- Faster development and deployment

## Innovation

- Autonomy of teams, culture of change
- Ownership and DevOps culture

## Quality

- Composability and reusability
- More maintainable code
- Better scaling and optimisations
- Failure isolation and resiliency

# Infrastructure Concerns



**Scalability**



**Connectivity**



**Security**



**High Availability**



# Infrastructure Concerns



**Logging**



**Scalability**



**Connectivity**



**Security**



**Deployments**



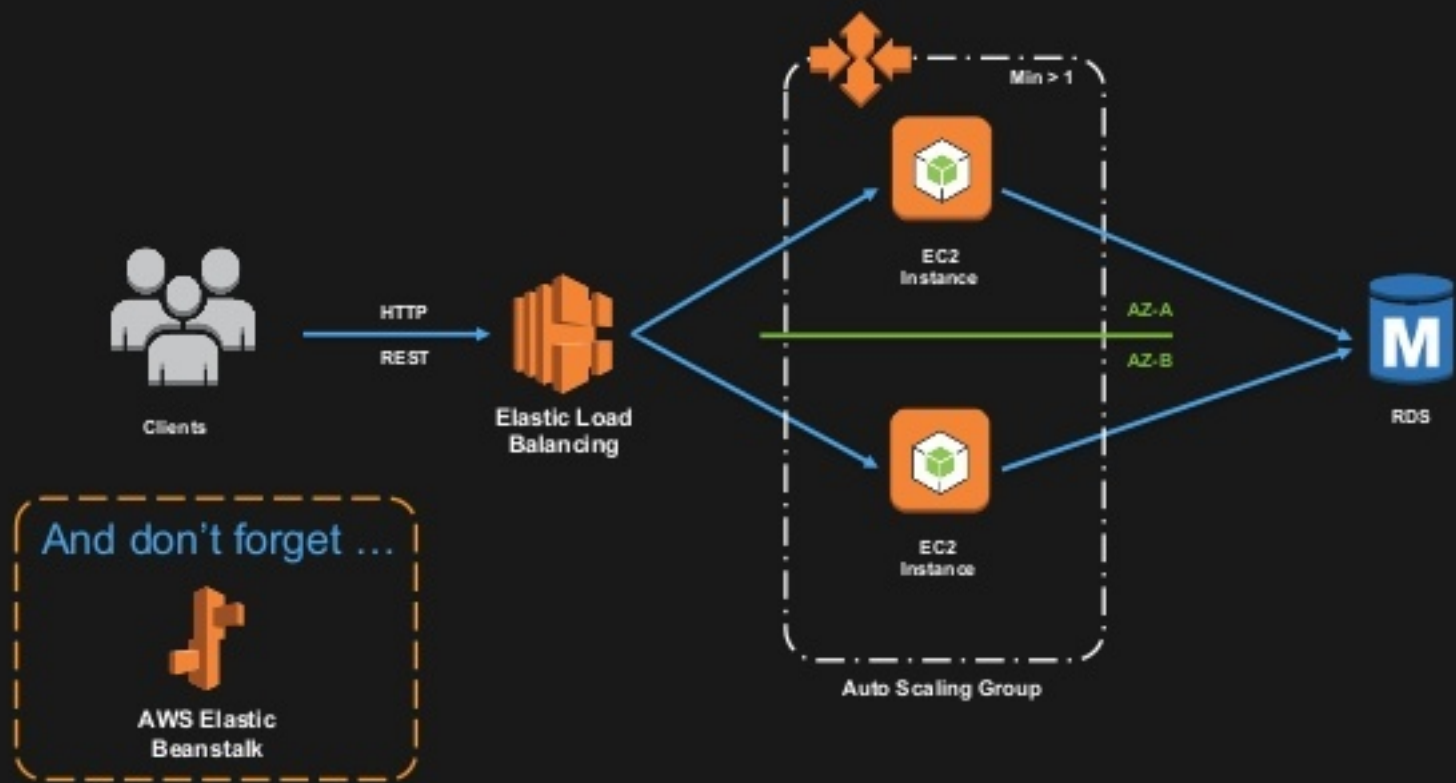
**Monitoring and  
Alerting**



**High Availability**

# Microservices Patterns on AWS

# The Traditional Microservice



# A Typical Microservice Architecture on AWS

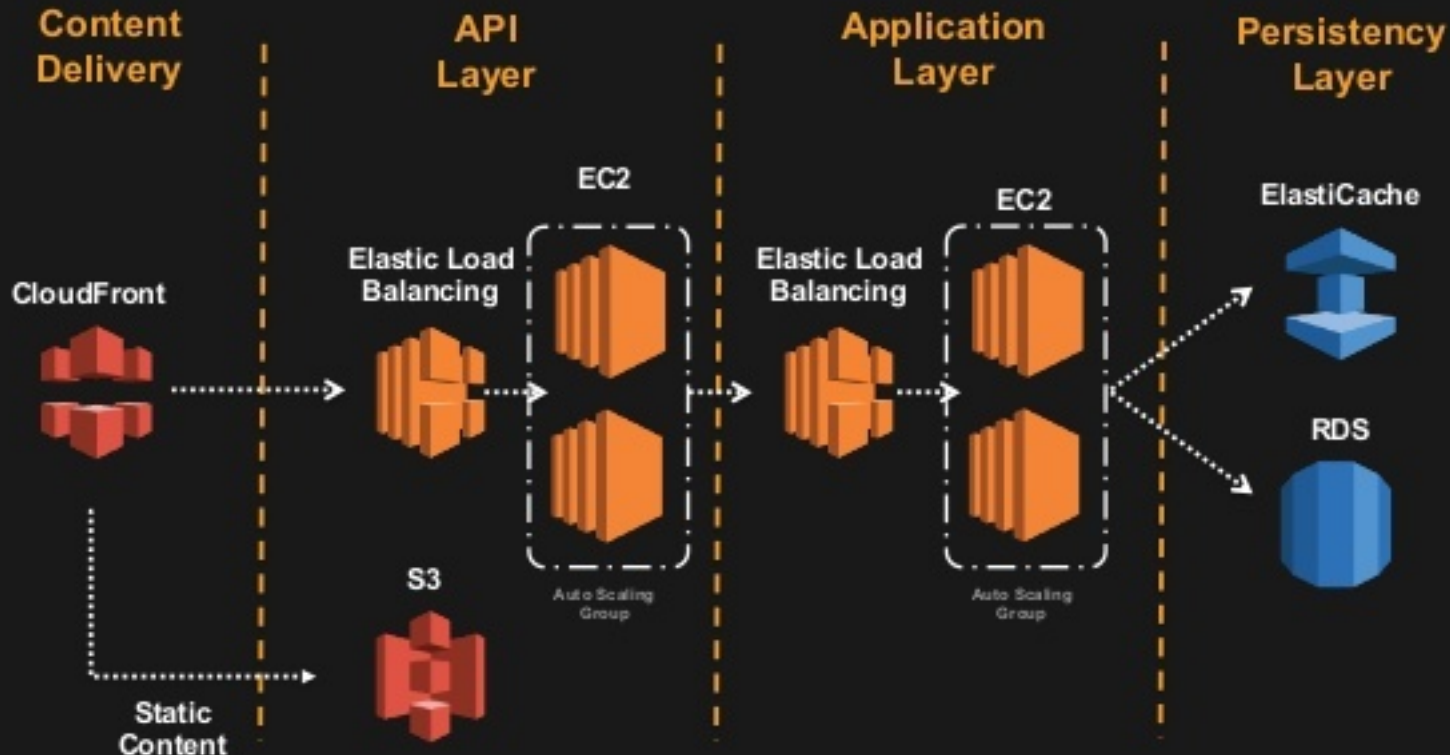
Content  
Delivery

API  
Layer

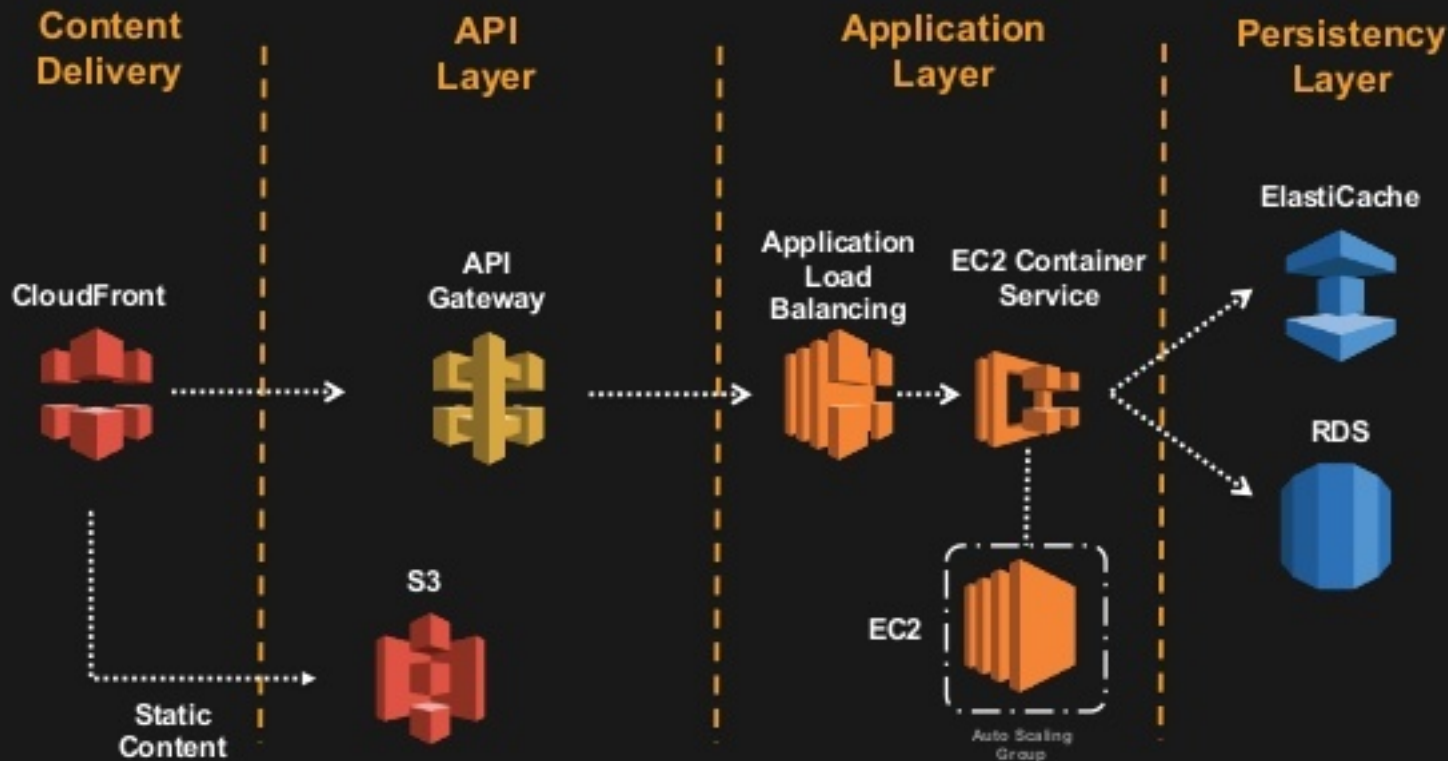
Application  
Layer

Persistency  
Layer

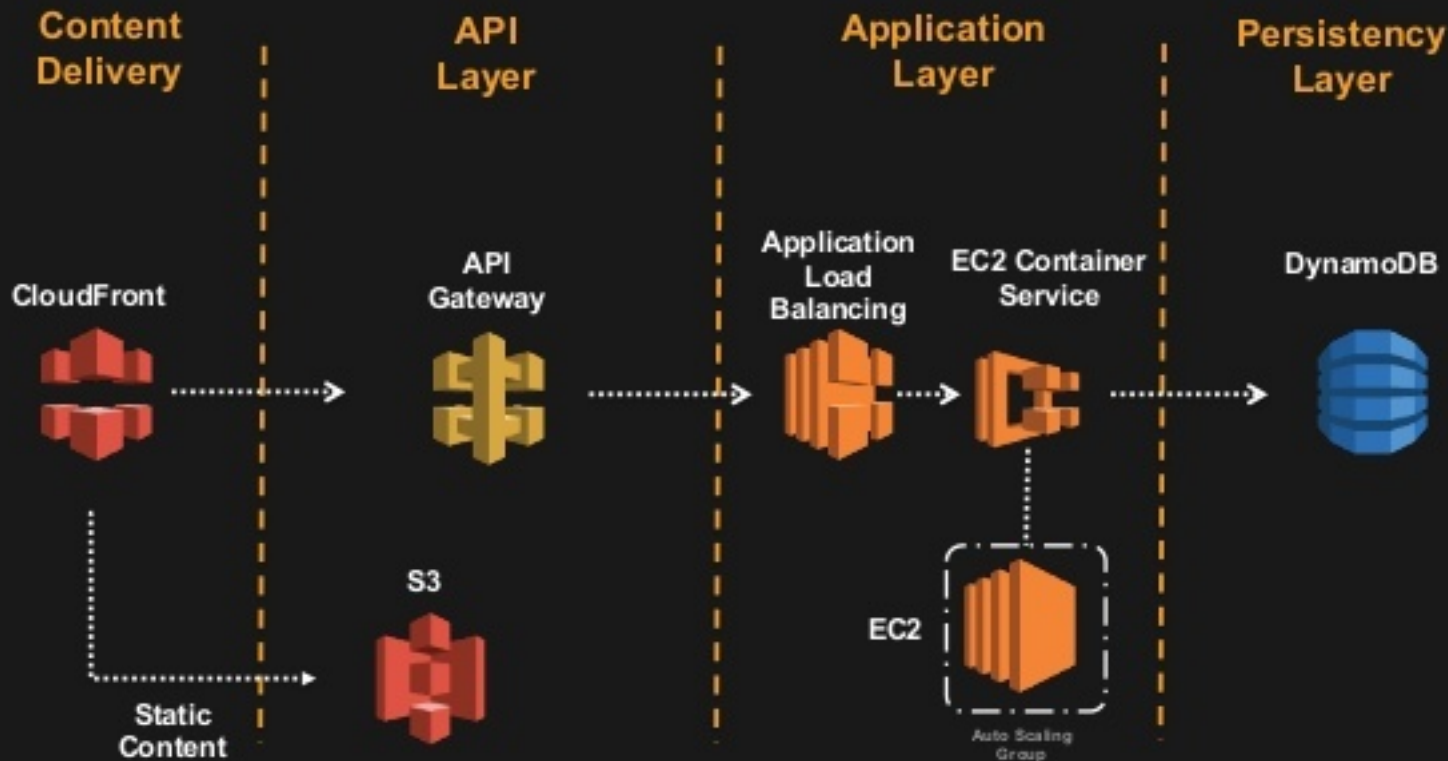
# A Typical Microservice Architecture on AWS

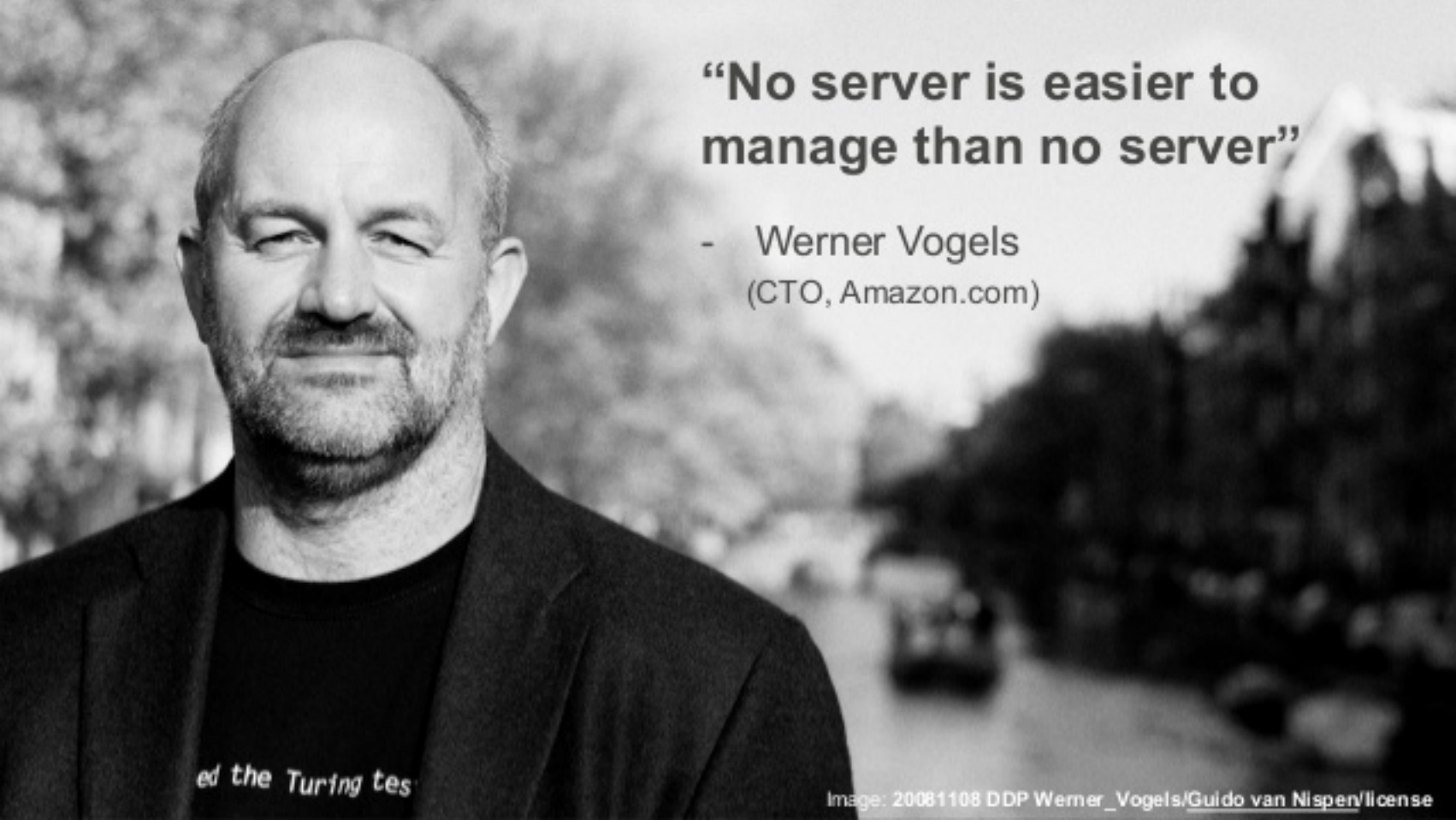


# A Typical Microservice Architecture on AWS



# A Typical Microservice Architecture on AWS



A black and white photograph of Werner Vogels, a middle-aged man with a beard and mustache, wearing a dark blazer over a dark t-shirt. He is looking directly at the camera with a slight smile. The background is a blurred outdoor scene with trees and a path.

**“No server is easier to  
manage than no server”**

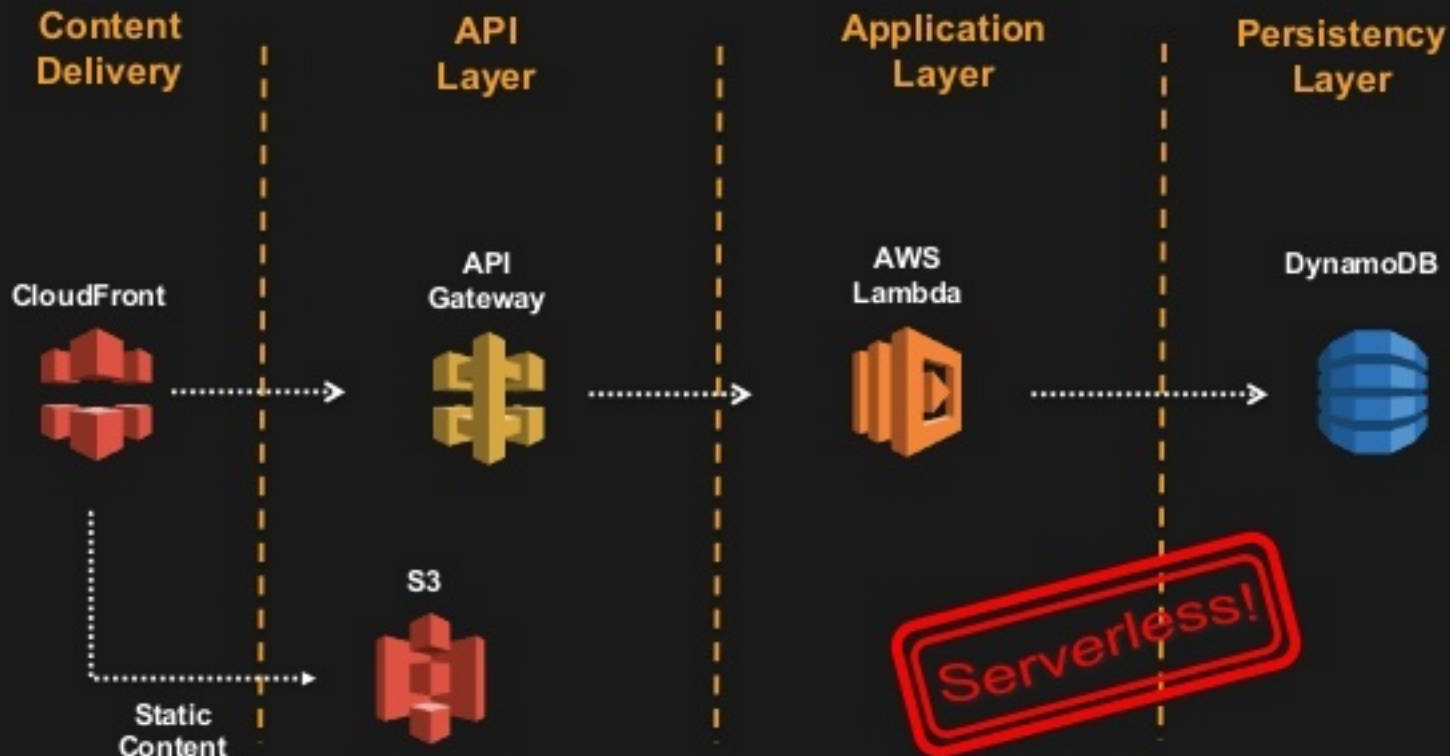
- Werner Vogels  
(CTO, Amazon.com)

*ed the Turing tes*

Image: 20081108 DDP Werner\_Vogels/[Guido van Nispen](#)/license

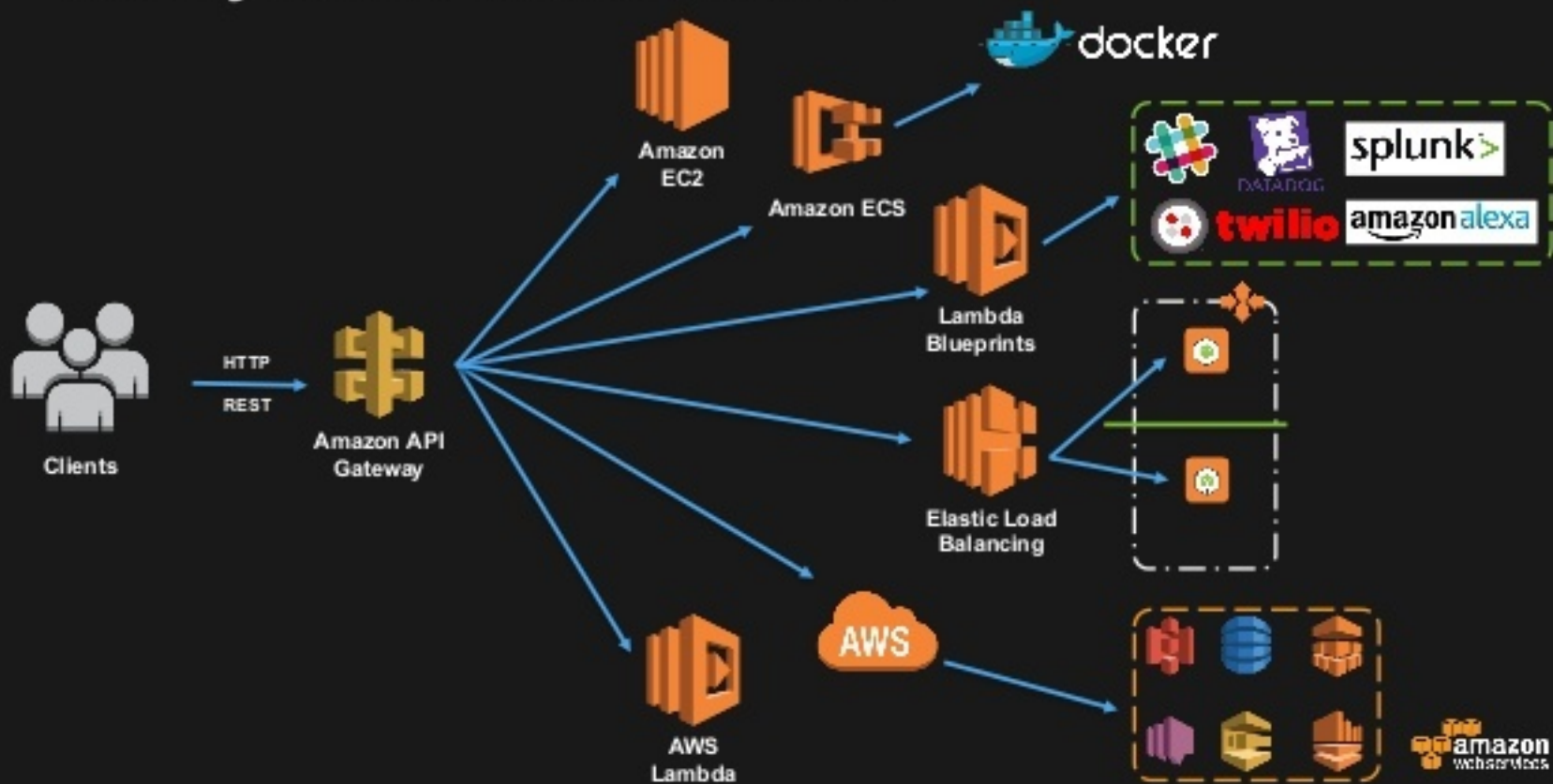


# The Goal – Serverless

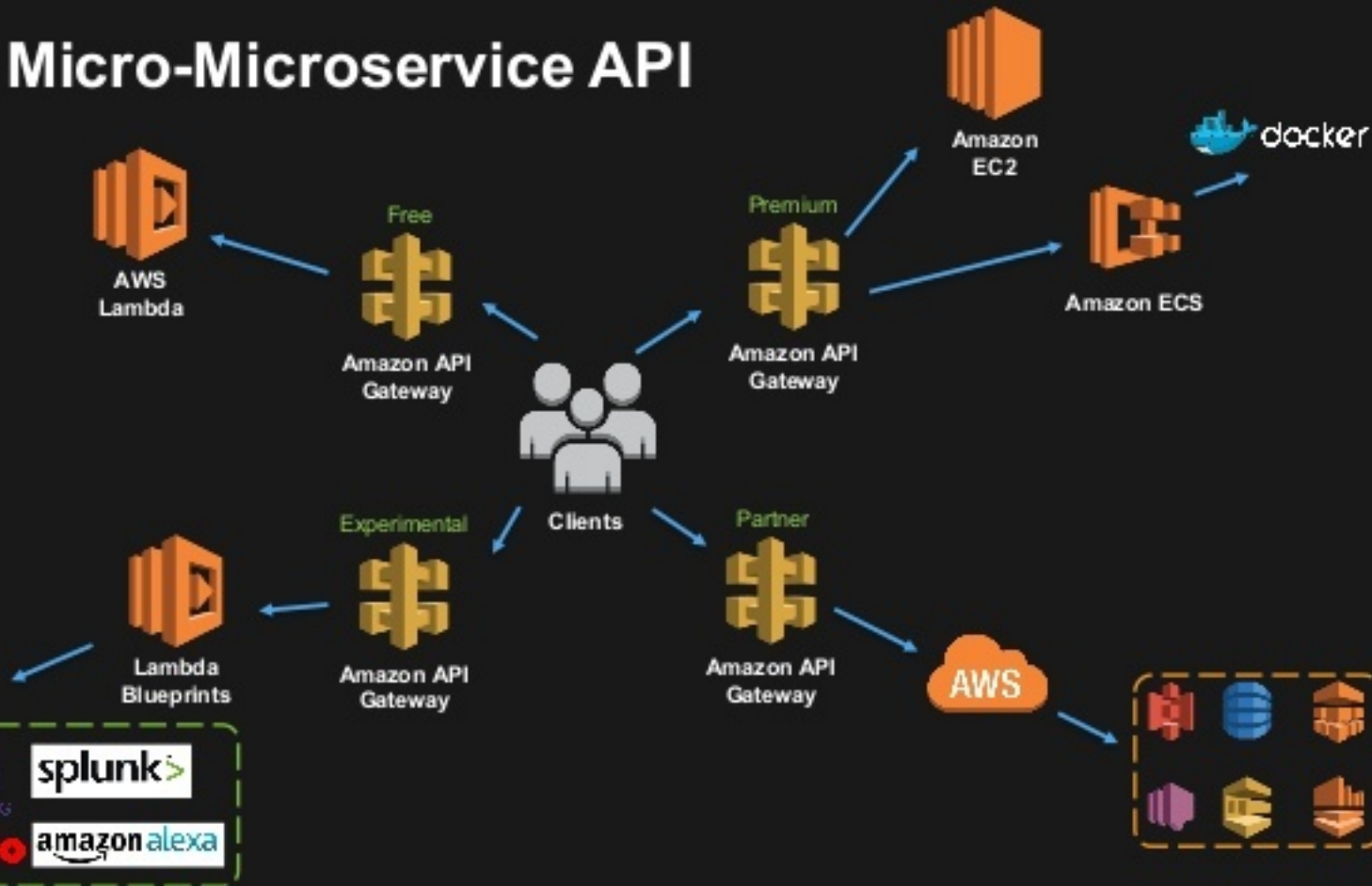


**Then You Put It All Together**

# The Hybrid Microservice API



# The Micro-Microservice API



# Don't Reinvent the Wheel

If you find yourself writing your own...

Notification system

E-Mail component

Search engine

Workflow engine

Queue

Transcoding system

Monitoring system



Amazon SNS



Amazon  
CloudSearch



Amazon SQS



Amazon SES



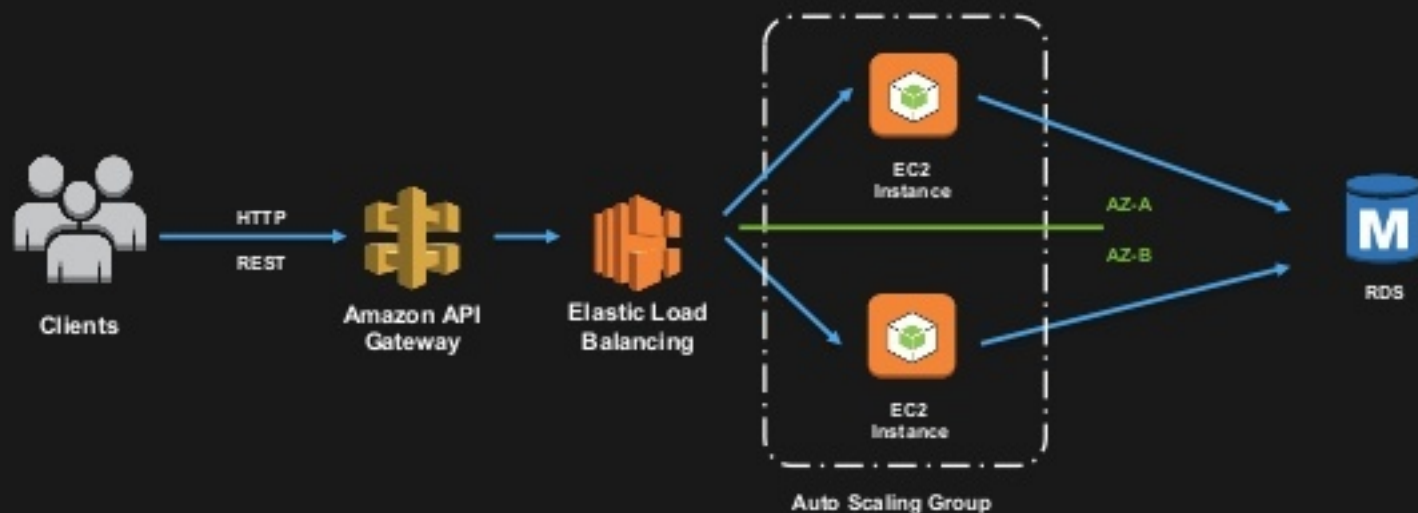
Amazon SWF



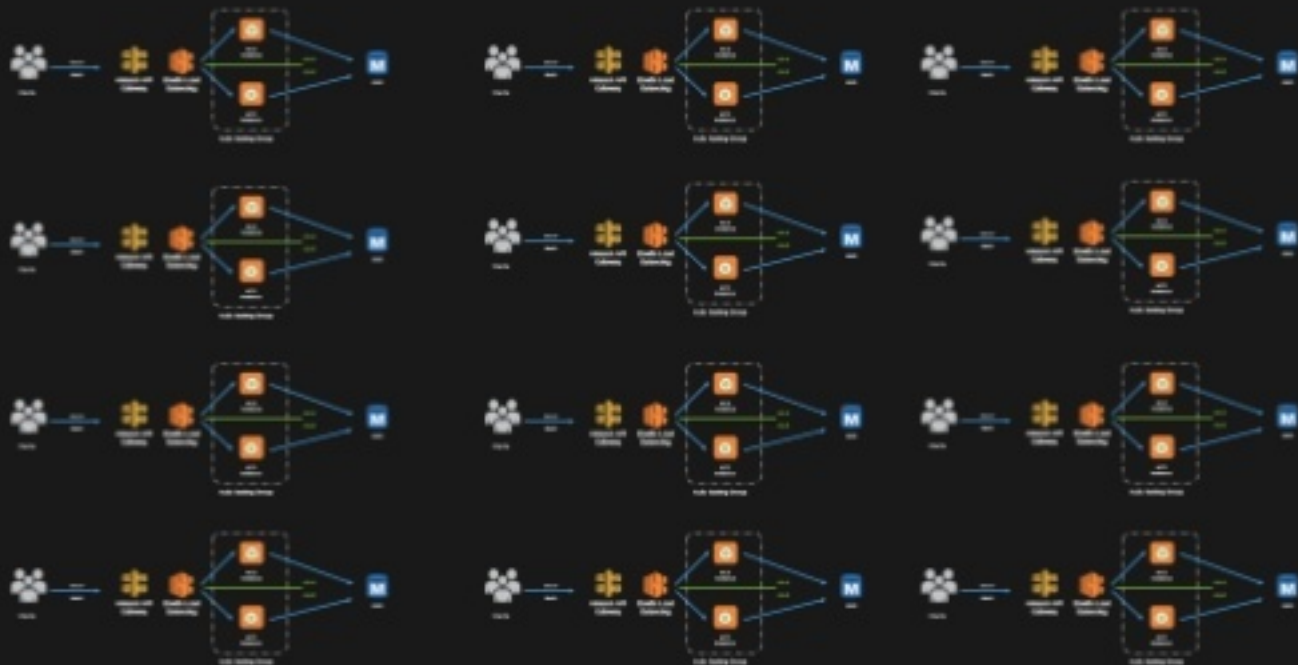
Amazon Elastic  
Transcoder

**...take a deep breath and stop it now!**

# Managing One Microservice is Straightforward



# Managing a Fleet can be a Challenge



# How Can AWS Help?



# Logging, Monitoring

- **Built-in Features**

- Monitoring via CloudWatch
- Logging: CloudWatch Logs



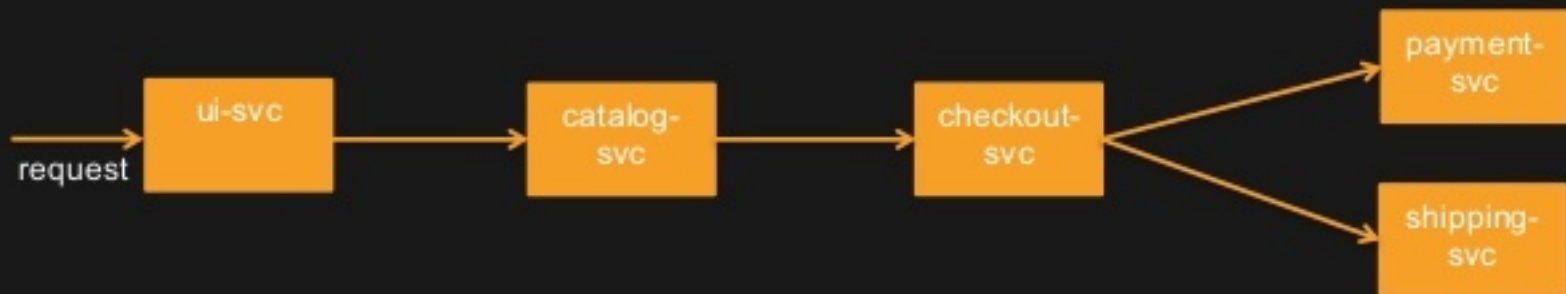
# Logging, Monitoring

- **Publish** externally relevant metrics
  - Latency
  - RPS
  - Error rate
- **Understand** internally relevant metrics
  - Basic – CloudWatch
  - OS
  - Application

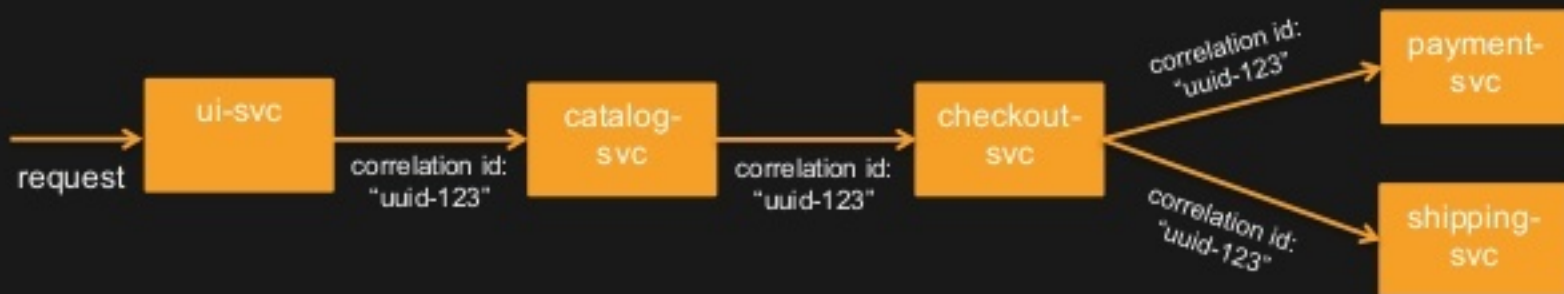
# Logging, Monitoring

- Pick a common log aggregation solution
- Agree on log entry formats
- Use naming conventions
- Agree on correlation strategy

# Challenge: Correlating Requests



# Use Correlation IDs



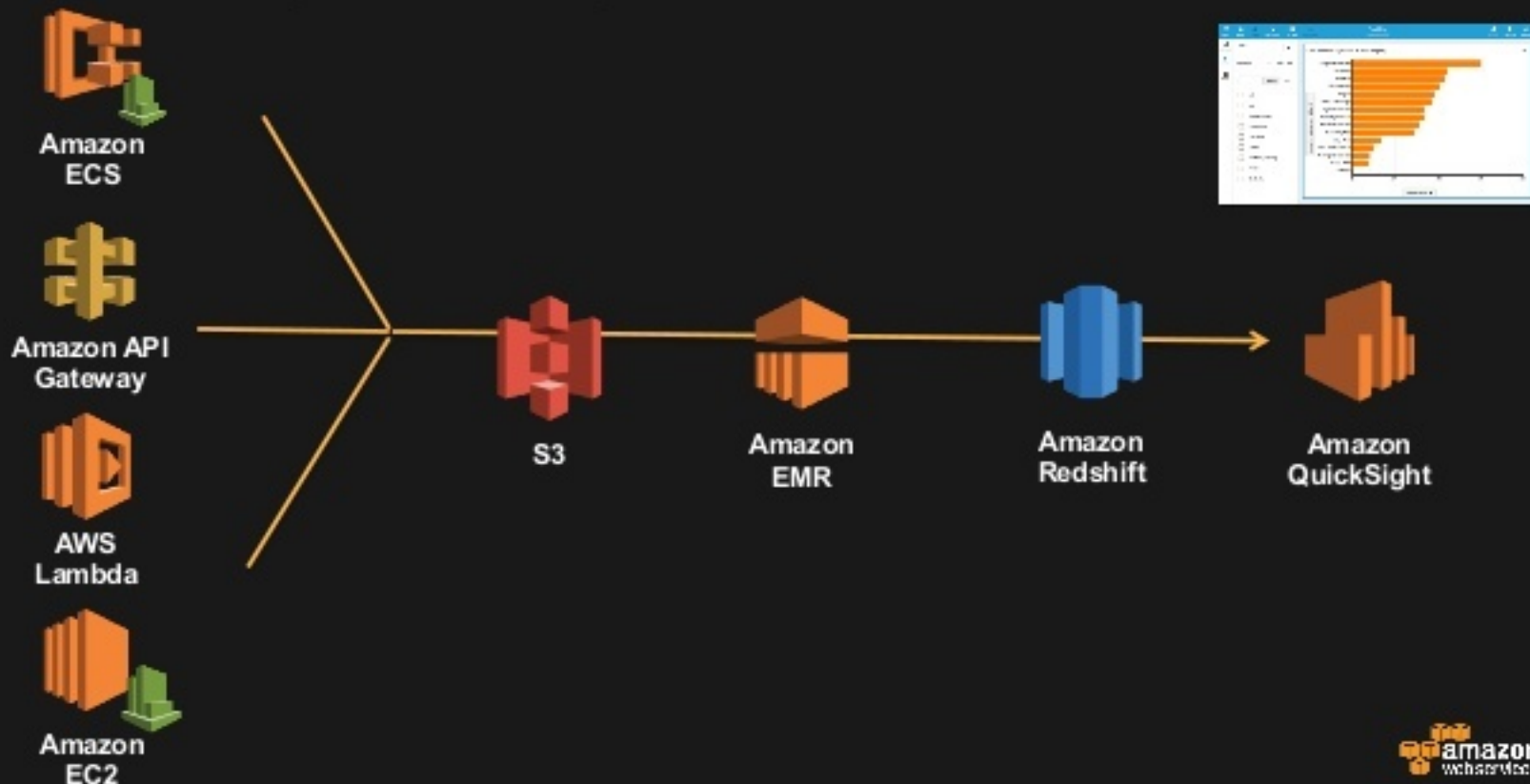
```
09-02-2015 15:03:24 ui-svc INFO [uuid-123] .....
09-02-2015 15:03:25 catalog-svc INFO [uuid-123] .....
09-02-2015 15:03:26 checkout-svc ERROR [uuid-123] .....
09-02-2015 15:03:27 payment-svc INFO [uuid-123] .....
09-02-2015 15:03:27 shipping-svc INFO [uuid-123] .....
```

# Real-time Dashboard



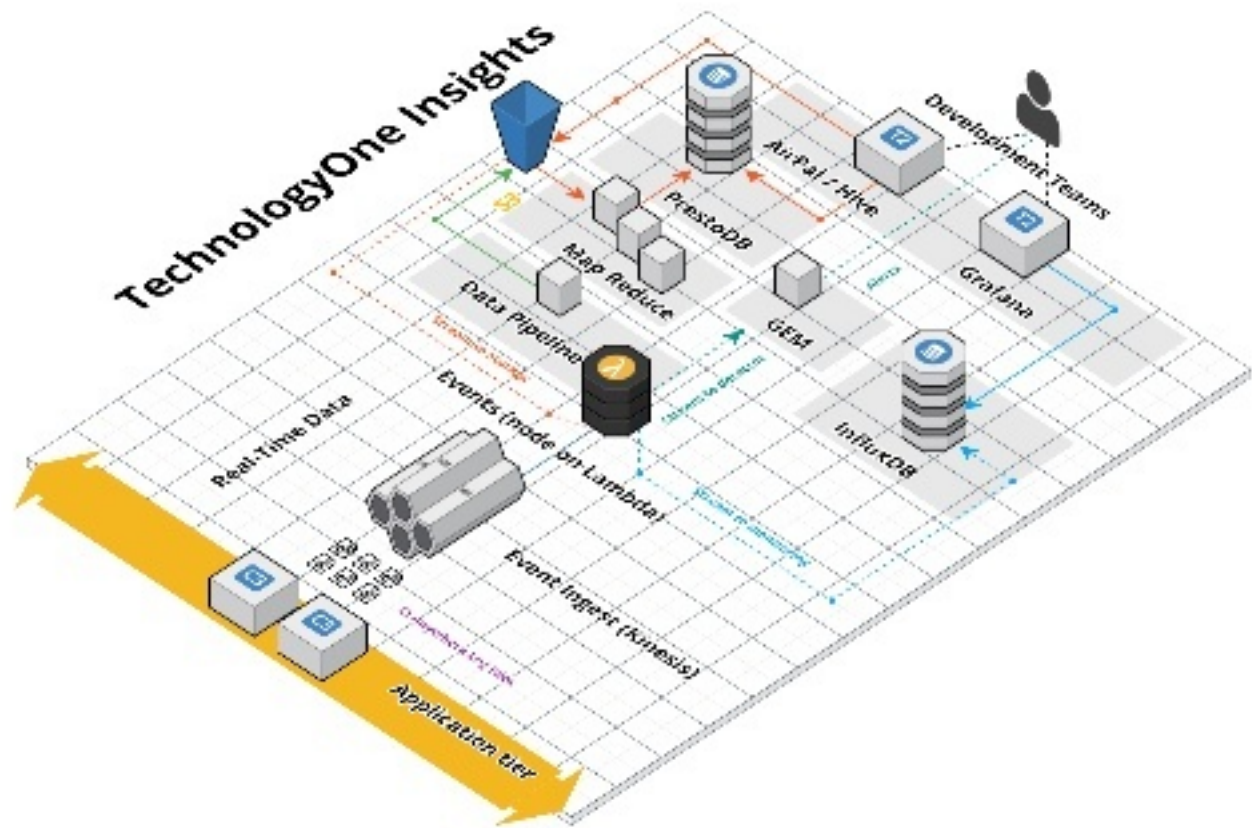
# Demo 1 – Real-time Dashboard

# Reporting and Analytics





# TechnologyOne Insights



User request rate

852.00

Same time yesterday

944

App errors

1

90th Percentile

1.062 s

90th Percentile

2.781 s

Rate of change (delta)

-0.229

DB Performance - Current release minus one



DB Anywhere Requests - Current release minus one



Instance Performance



Requests per Instance



Customer Performance



Requests per Customer



App Performance



Requests per App



+ add more

DP Job Rate / sec

Job rate past hour (per minute)

Rate of change (5m)

7

963

1.004



## Error Messages

Time -	Message	Value
2014-09-14 14:06:00	Dispatcher failed to find active engine job queue: \$ECMGOFELEBNOFM	10.00
2014-09-14 14:06:00	Job Runner for job 3263363 job service failed, T1 DistributedProcessorBusinessTierCloudOpDescription: Error: Workflow Item Auto-generated key: DEFAULT/SIN_PUPO_PCON/23 was changed by another user. The changes you have made can not be saved. Please Retrieve again and redo changes. Error: Workflow Item Auto-generated key: DEFAULT/SIN_PUPO_PCON/23 was changed by another user. The changes you have made can not be saved. Please Retrieve again and redo changes. Error: Workflow Item Auto-generated key: DEFAULT/SIN_PUPO_PCON/40 was changed by another user. The changes you have made can not be saved. Please Retrieve again and redo changes. at T1 DistributedProcessorJobRunnerJobRunner.RunJobService()	1.00
2014-09-14 14:06:00	Dispatcher set job 4153507 to enqueued because it died, stopped, or was killed.	1.00
2014-09-14 14:06:00	Dispatcher set job 3485190 to enqueued because it died, stopped, or was killed.	1.00
2014-09-14 16:06:00	Dispatcher set job 2897167 to enqueued because it died, stopped, or was killed.	1.00
2014-09-14 16:06:00	Dispatcher set job 3682072 to enqueued because it died, stopped, or was killed.	1.00



# Recap

- Devops is the key to success
- Set standards early
- Log everything and have tooling to gain insights
- Don't reinvent the wheel



AWS  
**Architect &  
Developer**  
Series

**Thank You!**