

AWS ReInvent

Serverless everything: Replatforming for speed and ownership

(sponsored by Datadog)

Companies all over the world are reaping the benefits of serverless and pay-for-what-you-use compute. Few, however, have adopted serverless to the extent Dunelm has. During a replatform of its online sales platform, Dunelm chose to build a completely serverless microservices application. Hear the story of how Dunelm moved to deploying changes over 200 times a month and saw an average speed improvement of 472 percent across its entire platform. In addition, hear how it has decided to organize teams around core business domains, and see how observability is a core component of moving fast with serverless. This presentation is brought to you by Datadog, an AWS Partner.

SVS381-S

SPONSORED BY DATADOG



Serverless everything: Replatforming for speed and ownership

Tom Hayman
Head of Platform Engineering
Dunelm

Kirk Kaiser
Technical Evangelism Team Lead
Datadog

AWS re:Invent

-03:32 © Amazon Web Services, Inc. or its affiliates. All rights reserved.

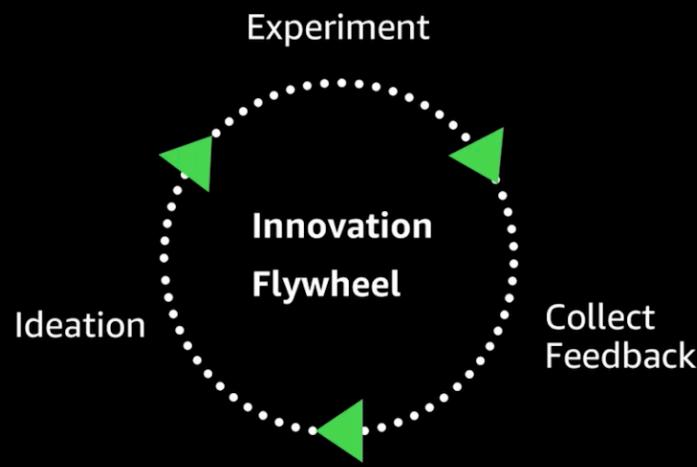
The Promise of Serverless

Focus exclusively on delivering business value



Companies all over the world are reaping the benefits of serverless and pay-for-what-you-use compute. Few, however, have adopted serverless to the extent Dunelm has. During a replatform of its online sales platform, Dunelm chose to build a completely serverless microservices application. Hear the story of how Dunelm moved to deploying changes over 200 times a month and saw an average speed improvement of 472 percent across its entire platform. In addition, hear how it has decided to organize teams around core business domains, and see how observability is a core component of moving fast with serverless. This presentation is brought to you by Datadog, an AWS Partner.

Building a Positive Feedback Loop for Developers with Minimal Cognitive Overhead

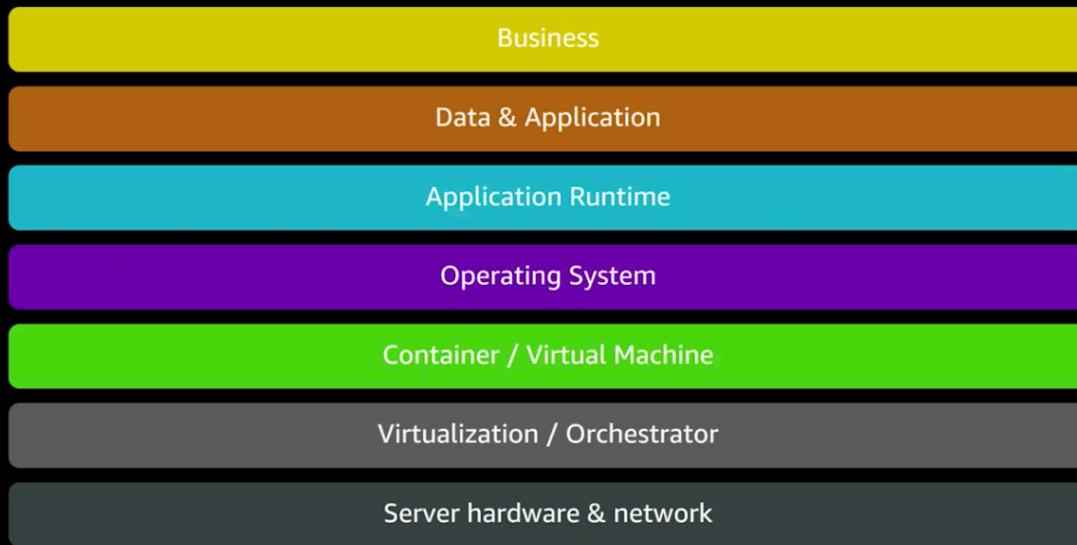


Smaller companies are adopting serverless more than larger companies because they are more nimble.

With Serverless, forced to reason about code in smaller pieces, think less about infrastructure underneath it and more about the direct value of the code you are writing.

Getting out of loop of existing infrastructure and how we deploy code.

Traditional application layers



Traditional web app: Have to make decisions about all of these layers. Instead of thinking about what we should be building, we think about each of these layers.

Serverless application layers

Business

Data & Application

Serverless Platform

Write smaller units of code that are written independently and deployed on their own.
Individual empowerment: don't have to coordinate releases as much, more autonomy.

Serverless application layers

Business

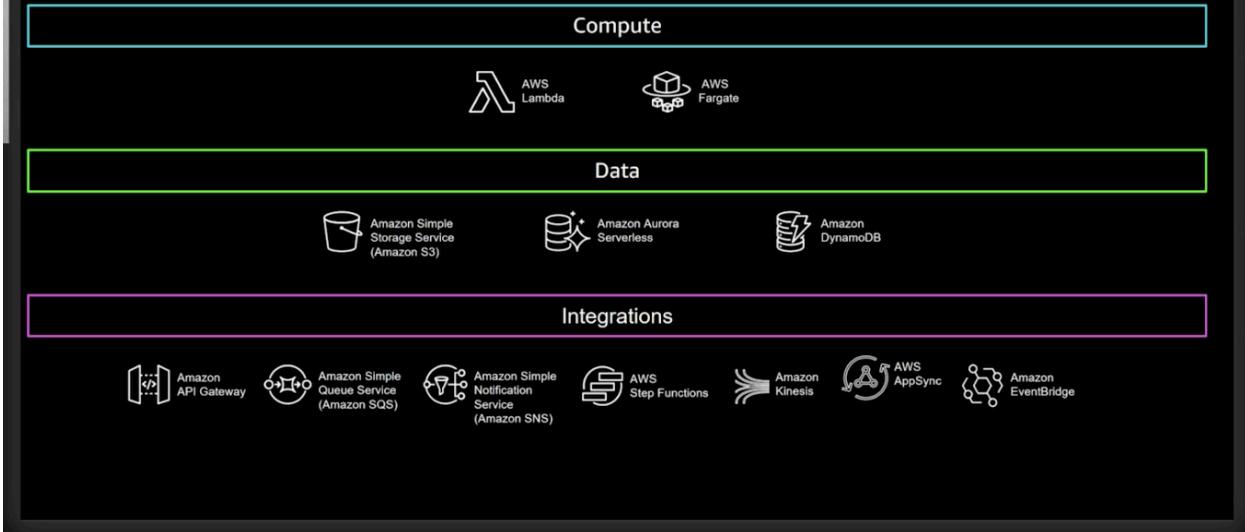
Data & Application

Serverless Platform

AWS Ecosystem

Your code never exists in isolation. Serverless fits into the entire AWS ecosystem. Don't have to think about underlying way they run and are built.

Rather than thinking in terms of layers of abstractions, thinking in terms of building blocks



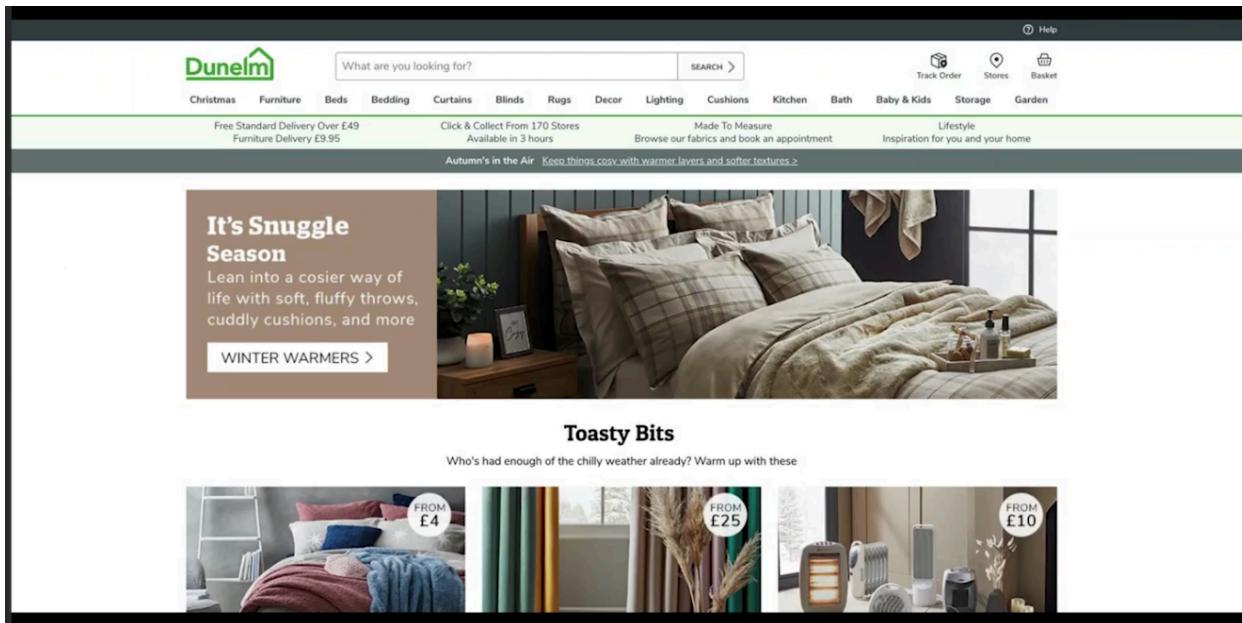
Write code to APIs instead of writing code to run in a specific place. Building blocks to fit together rather than abstractions.

- AWS Lambda (application platform as an API) => write, upload code, run on demand.
- AWS Fargate => containers on demand

Can call code via SQS, which is powerful!

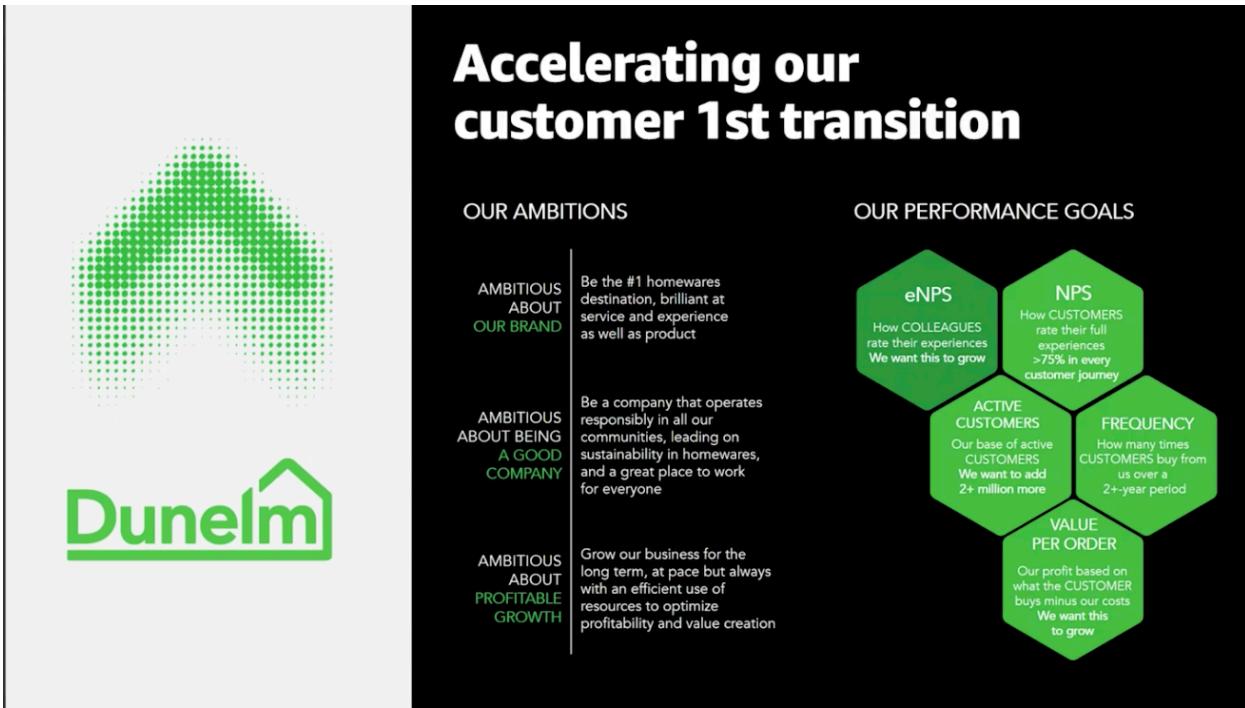


Case study: Dunelm



Accelerating our customer 1st transition





Challenges with our existing platform



- Tightly coupled **monolithic architecture**
- **Not scalable**
- **Deployment frequency limited** and cumbersome
- Not resilient to failure; one part failed, **everything failed**
- **Third-party** reliance

Long feedback loop, monthly deployments.

Serverless first

From monolith to serverless microservices



AWS
re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Decided to move to Serverless.

Key digital technology enablers for our future strategy

Dunelm



Microservices

- Developed and operated by teams **focused on core business domain capability**
- **Independently deployable** using cloud automation to facilitate a **faster pace of change**
- **Allow teams to be scaled horizontally** as new requirements are identified
- Adopt **new technologies** and innovation with shorter refresh cycles



Cloud

- Focus on **automation** ensures consistency across environments and **ensures security & IT practices are consistently applied**
- **Adopted serverless**, which enabled **high availability** and **automatic scaling**
- Serverless means **less overhead** of managing servers and infrastructure
- Provides teams with **self-service capability** while still ensuring compliance with IT processes



API

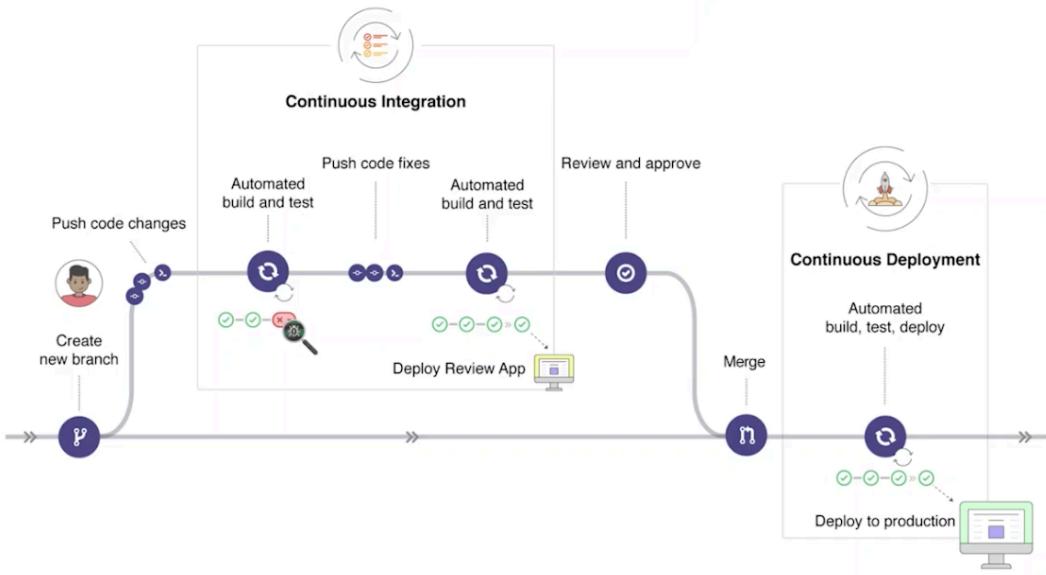
- **Loosely coupled** API-first design allows for services to be worked on without affecting the others or overall operation and without interfering with the user experience
- **Extensible** – With APIs built into every layer of the application, making it **easier to add new functionality** and connect with new features, **accelerating the launch of new and innovative value propositions**
- **Drives collaboration** and **reuse** – with an easy-to-use API, services or products are opened up to the creativity of the people who want to use them across the business and our future IT partners

Scale faster, de-couple systems and services => autonomous.

Moving from Component Teams to Outcome Teams

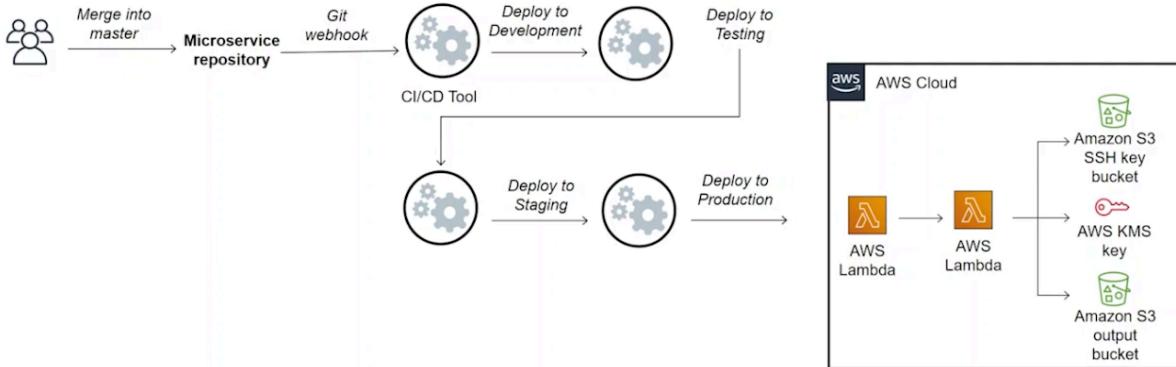
- Started with Component Teams who owned standalone services i.e. Search, Basket, and Payment. Got the services dev underway quickly, ensured standards and service ownership
- Does however create dependencies between the teams. We managed this but it did create an overhead on coordination and e2e testing
- Now well into our journey shaping Outcome Teams, giving them autonomy and ownership to drive the strategic outcomes
 - Less about giving the work to do, more about understanding the problem and letting the teams respond through their own innovation and creativity
 - Equipping our teams with the expertise they need to make changes to all services required for their outcome
 - Needs a cultural mind shift , teams need to learn other systems and push beyond their comfort zone
 - Outcome Examples - "Reduce basket abandonment by 5%" or "Reduce deployment lead times"

How Dunelm Creates New Microservices



This gives teams more autonomy.

How Dunelm Deploys to Production



Deployment flow.

Monitoring serverless with Datadog

Strategies for instrumentation

AWS re:Invent

© 2020 AWS Services, Inc. or its affiliates. All rights reserved. 07:56

Datadog Pitch:

Tradeoffs of switching to Serverless: Where do you begin when debugging? What if multiple services are involved? No servers to look at anymore. If many people deploying at the same time, how to determine who pushed the breaking change?

Start with the AWS integration

The screenshot shows the 'Amazon Web Services Integration' configuration page. At the top left is the AWS logo. Below it, a green button says 'INSTALLED'. The main content area has tabs for 'Overview' (which is selected), 'Configuration', 'Metrics', and 'Collect Logs'. A sub-section titled 'Connect to Amazon Web Services (AWS) in order to:' lists several benefits:

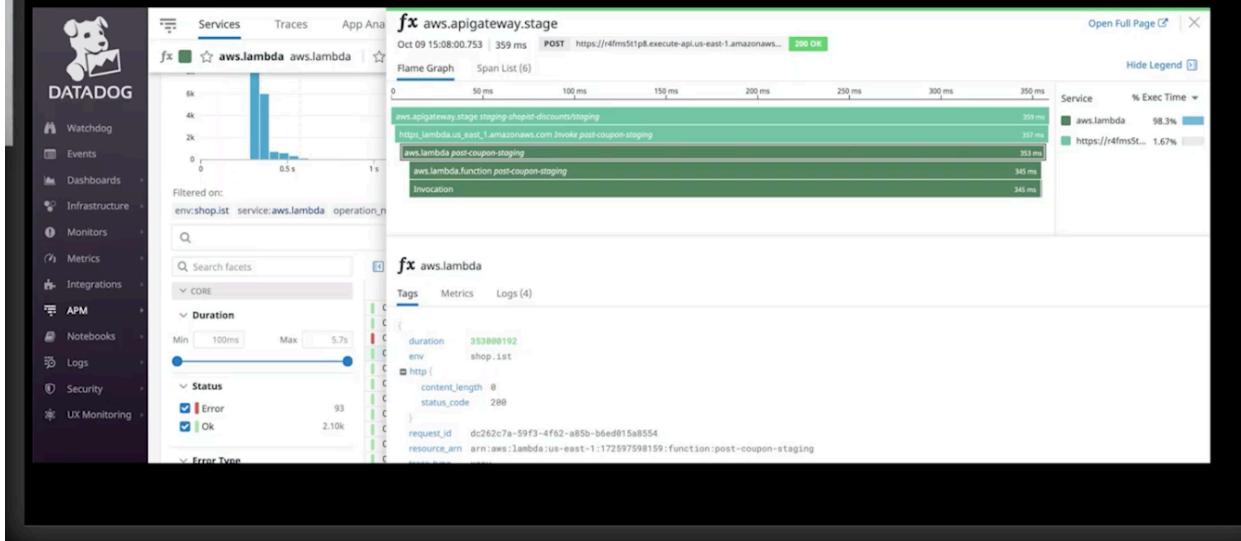
- See automatic AWS status updates in your stream
- Get Cloudwatch metrics for EC2 hosts without installing the Agent
- Tag your EC2 hosts with EC2-specific information (e.g. availability zone)
- Get Cloudwatch metrics for other services: ELB, RDS, EBS, AutoScaling, DynamoDB, ElastiCache, CloudFront, CloudSearch, Kinesis, Lambda, OpsWorks, Redshift, Route53, SQS, and SNS
- See EC2 scheduled maintenance events in your stream

The screenshot shows the 'Serverless View' dashboard in Datadog. On the left is a sidebar with navigation links like 'New Stuff!', 'Watchdog', 'Events', 'Dashboards', 'Infrastructure', 'Monitors', 'Metrics', 'Integrations', 'APM', 'Notebooks', 'Logs', 'Security', 'UX Monitoring', 'Live Support', 'Help', and 'Team'. The main area has three charts: 'Invocations' (purple bars), 'Errors' (red bars), and 'Duration (P95)' (yellow lines). Below the charts is a table showing 59 serverless functions. The table includes columns for FUNCTION NAME, INVOCATIONS, DURATION (AVERAGE), ERRORS, THROTTLES, EST COST, MEMORY USED, LAST START, and % MEMORY USED.

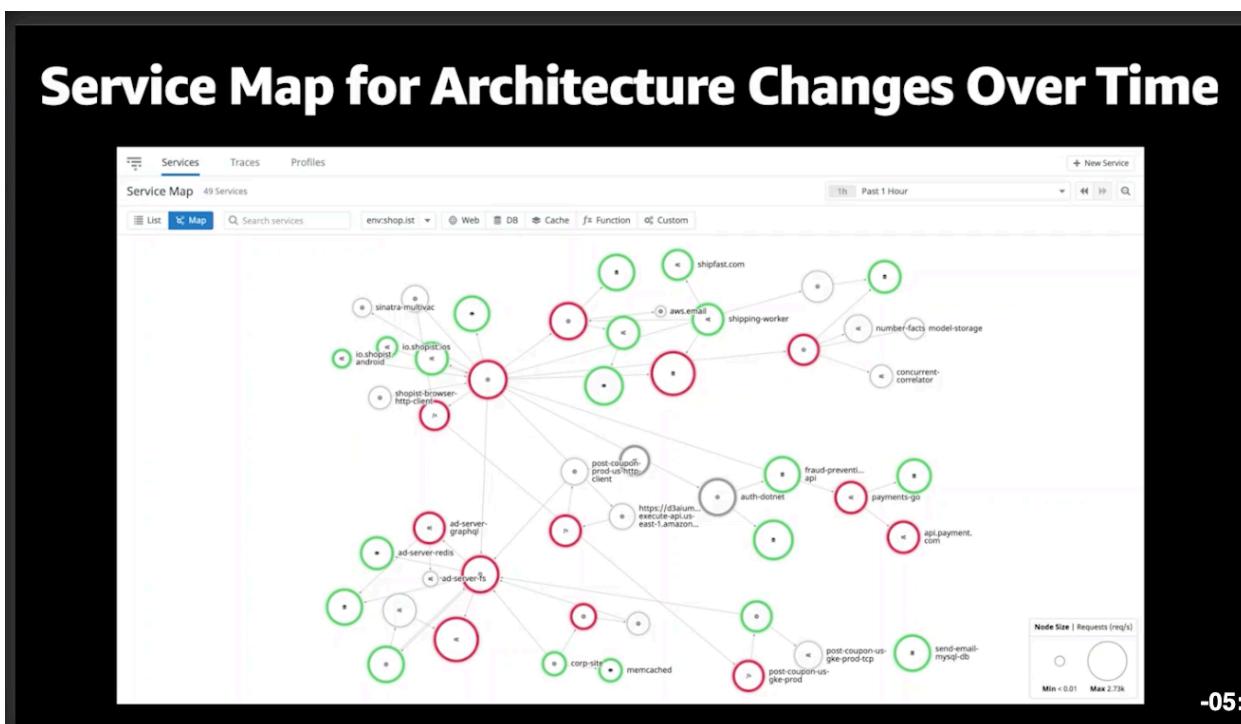
FUNCTION NAME	INVOCATIONS	DURATION (AVERAGE)	ERRORS	THROTTLES	EST COST	MEMORY USED	LAST START	% MEMORY USED
post-coupon-us-gke-prod	128k	1.0 s	5,33k	1,26k	\$0.46	135 MB	< 1 min ago	71 %
post-coupon-prod-us	103k	1.1 s	4,35k	927	\$0.29	123 MB	< 1 min ago	95 %
datadog-forwarder-prod-11287	33.5k	131 ms	0	0	—	—	—	—
dsgdatadogforwarder3	1.43k	337 ms	0	0	—	—	—	—
requestunicorn	224	364 ms	112	0	\$0.00	97 MB	< 1 min ago	75 %
lambda-layer-step-functions-dev-step-four	166	5.0 ms	111	1	\$0.00	128 MB	< 1 min ago	12 %
lambda-layer-step-functions-dev-step-two	56	4.0 ms	0	0	\$0.00	118 MB	< 1 min ago	12 %
lambda-layer-step-functions-dev-step-five	55	7.0 ms	0	0	\$0.00	121 MB	< 1 min ago	12 %
lambda-layer-step-functions-dev-step-one	55	6.0 ms	0	0	\$0.00	122 MB	< 1 min ago	12 %
lambda-layer-step-functions-dev-step-three	55	7.0 ms	0	0	\$0.00	122 MB	< 1 min ago	12 %
lambda-log-test-dev-ruby-timeout	34	30.0 s	34	0	\$0.00	13 MB	< 1 min ago	10 %
lambda-log-test-dev-java-rakeserror	34	92.0 ms	34	1	\$0.00	91 MB	< 1 min ago	71 %

No need to communicate to team that you are about to deploy: everyone has shared view.

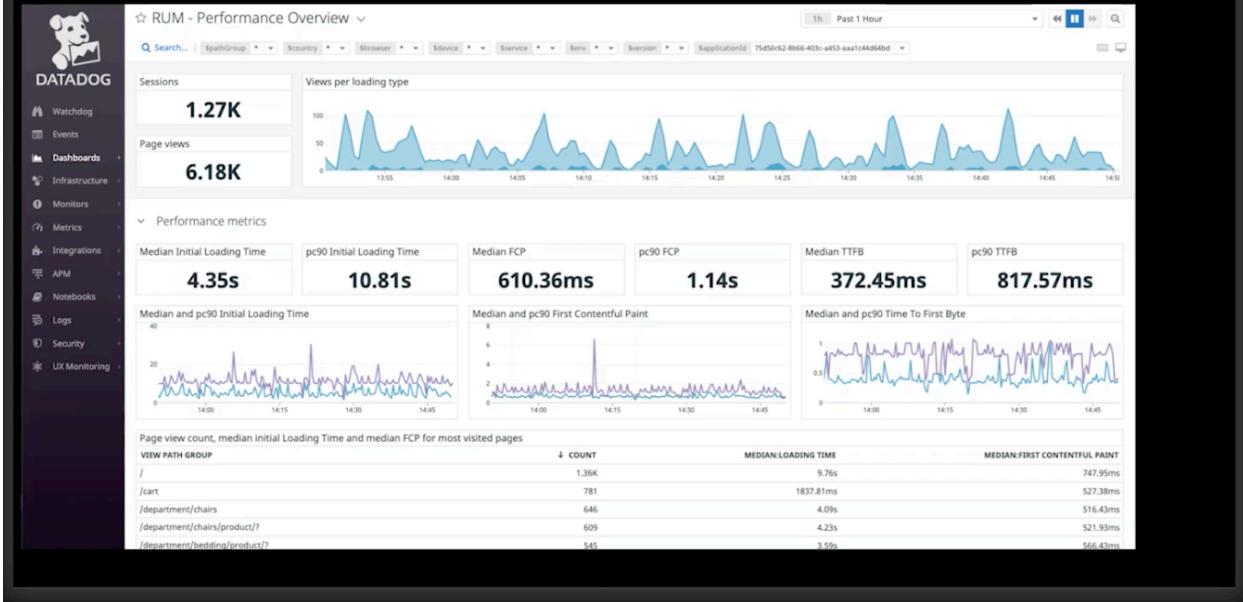
Add APM and logs for application-level insights with Datadog Forwarder



Service Map for Architecture Changes Over Time



Real User Monitoring for End User Experience



Deployment tracking

Make sure your systems are in sync

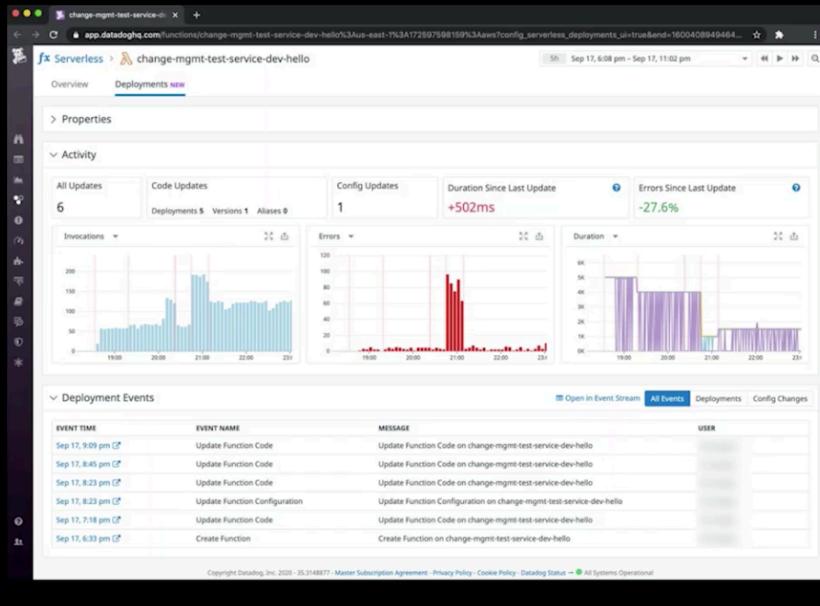


re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Another challenge with Serverless: How to track deployments over time? Track AWS Lambda versions over time.

Track AWS Lambda versions over time



Zero Instrumentation Serverless

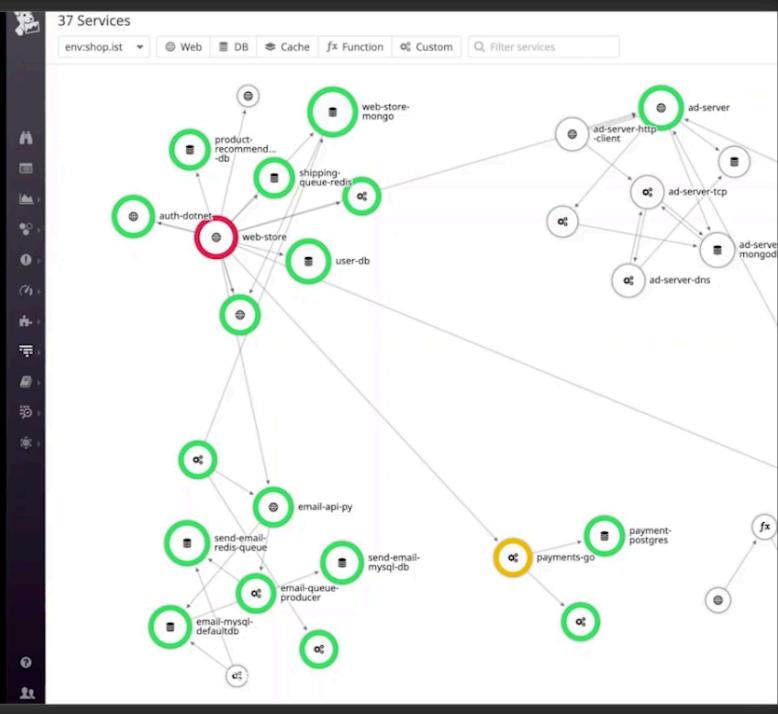
With AWS SAM and CDK Integrations



AWS
re:invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Zero instrumentation serverless observability with Datadog's AWS SAM Macro



Alerts as Code with AWS SAM

```
Resources:  
[...]  
MyTestMonitor:  
  Type: Datadog::Monitors::Monitor  
  Properties:  
    Type: metric alert  
    Query: 'avg(last_5m):max:aws.lambda.enhanced.duration{*} by {functionname} /  
(avg:aws.lambda.timeout{*} by {functionname} / 80 ) > 0.9'  
    Name: "Function duration is 90% of limit, increase timeout in AWS Console."  
    Multi: true  
  Options:  
    Thresholds:  
      Critical: 0.9  
      Warning: 0.7  
DatadogCredentials:  
  ApiURL: https://api.datadoghq.com  
  ApiKey: <DATADOG_API_KEY>  
  ApplicationKey: <DATADOG_APP_KEY>
```