AWS re:Invent

NOV. 28 – DEC. 2, 2022 | LAS VEGAS, NV

SVS203-R

Getting started with serverless

Mark Richman

Senior Solutions Architect
Amazon Web Services

Agenda

Step 0: Overview

Step 1: Build a serverless backend: AWS Lambda and AWS SAM

Step 2: Configure API authorization: Amazon API Gateway

Step 3: Build and deploy a web application: AWS Amplify

Step 4: Test the application

Step 5: Configure image metadata extraction: Amazon Rekognition

Step 6: Terminate the resources

Workshop link

s12d.com/svs203

Step 0: Overview

What do organizations need to drive success?

Get to market faster

Lower total cost of ownership

High performance and scalability

Security and isolation by design

How are organizations adopting AWS?

REDUCE

the amount of DIY

Retire + SaaS

MIGRATE

to AWS

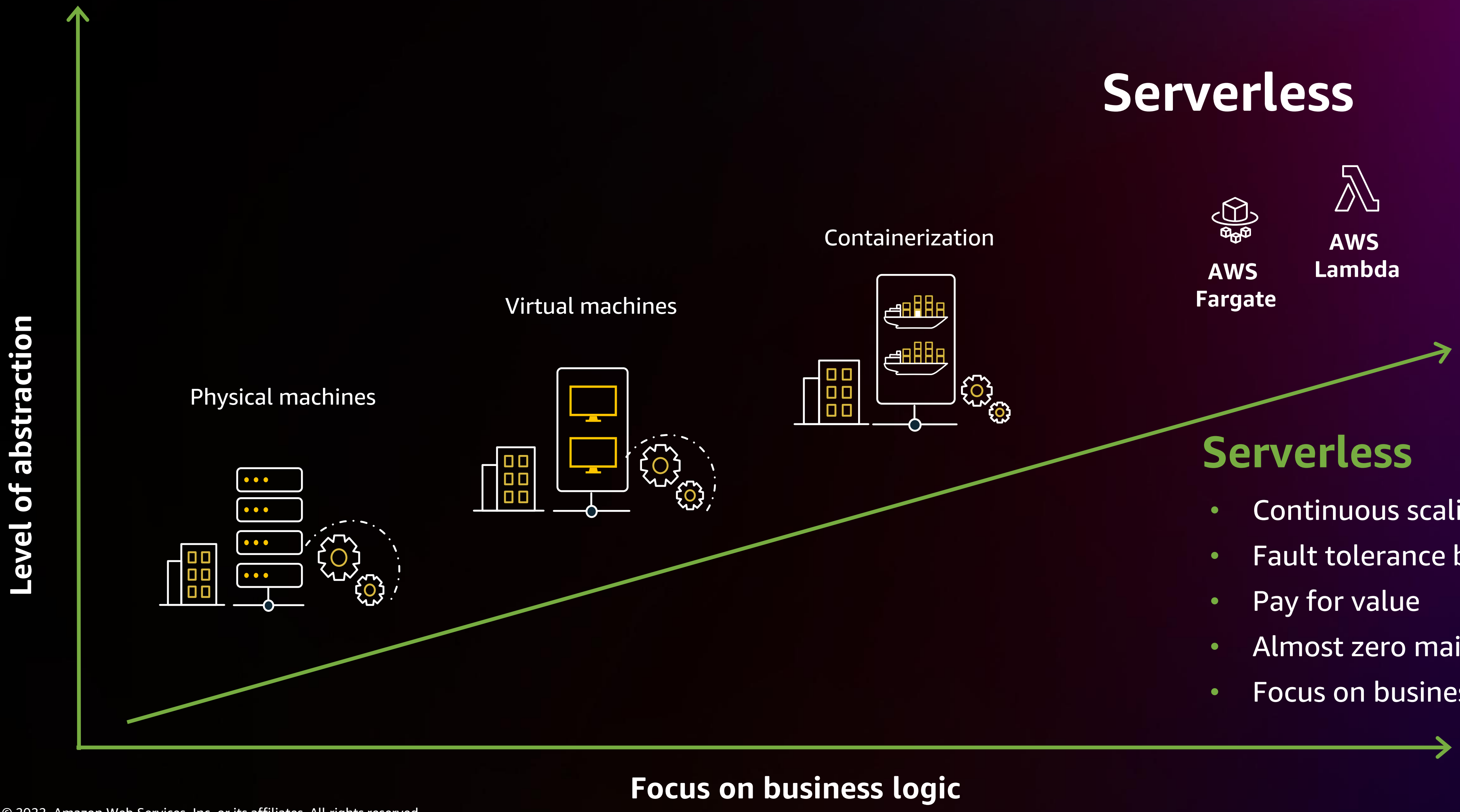
Lift and shift

MODERNIZE

on AWS

Replatform + Refactor

There's a paradigm shift happening



Serverless

- Continuous scaling
- Fault tolerance built in
- Pay for value
- Almost zero maintenance
- Focus on business value

What is serverless?

No infrastructure provisioning, no management

Automatic scaling

Pay for use

Highly available and secure

Serverless spans many different categories

Compute

- AWS Lambda
- AWS Fargate

Data stores

- Amazon Simple Storage Service (Amazon S3)
- Amazon Aurora Serverless
- Amazon DynamoDB

Integration

- Amazon API Gateway
- Amazon Simple Queue Service (Amazon SQS)
- Amazon Simple Notification Service (Amazon SNS)
- AWS Step Functions
- AWS AppSync

Serverless in the operational landscape

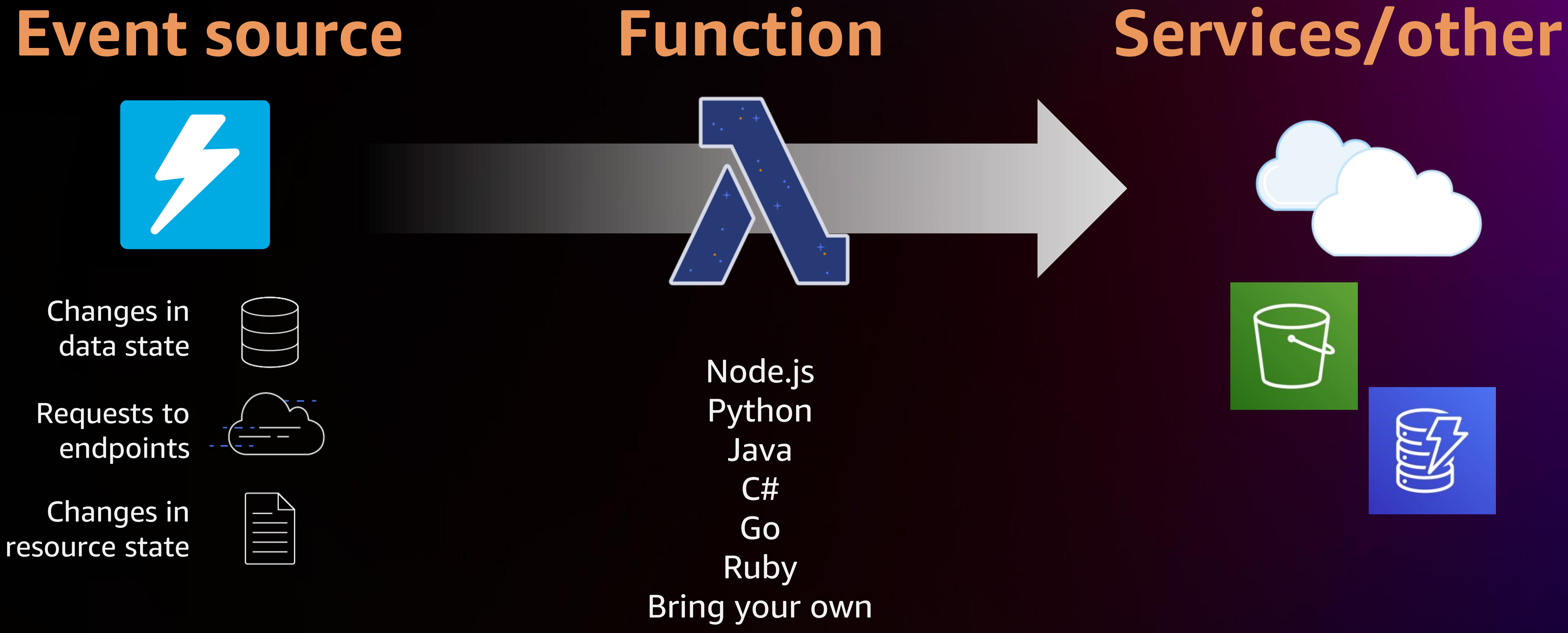
What you manage		AWS manages	Customer manages	
Least	**AWS Lambda** Serverless functions	Data source integrations Physical hardware, software, networking, and facilities Provisioning	Application code	
	AWS Fargate Serverless containers	Container orchestration, provisioning Cluster scaling Physical hardware, host OS/kernel, networking, and facilities	Application code Data source integrations	Security configuration and updates Network configuration Management tasks
	Amazon ECS/ Amazon EKS Container management as a service	Container orchestration control plane Physical hardware, software, networking, and facilities	Application code Data source integrations Work clusters	Security configuration and updates, network configuration, firewall, management tasks
Most	**Amazon EC2** Infrastructure as a service	Physical hardware, software, networking, and facilities	Application code Data source integrations Scaling Security configuration and updates	Network configuration Management tasks Provisioning, managing, scaling, and patching of servers

What does the future look like?

All the code you ever write is business logic

Step 1: Build a serverless backend

Serverless architecture



Event source

Changes in data state

Requests to endpoints

Changes in resource state

Function

Node.js
Python
Java
C#
Go
Ruby
Bring your own

Services/other

Anatomy of a function

Handler function

- Function executed on invocation
- Processes incoming event

Event

- Invocation data sent to function
- Shape differs by event source

Context

- Additional information from Lambda service
- Examples: request ID, time remaining

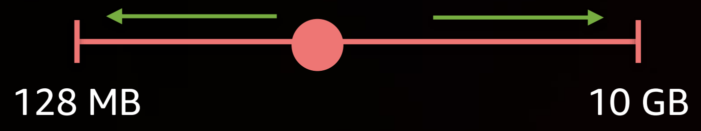
app.py

```
def handler(event, context):
  msg = ‘Hello {}’.format(
    event[‘name’]
  )
  return { ‘message’: msg }
```

Function configuration

Power rating

- Select between 128 MB and 10 GB
- CPU and network allocated proportionally
- Power tune to balance cost and speed



Permissions model

- Execution Role grants function access to resources via AWS Identity and Access Management (IAM)
- Function Policy controls invocation

Function configuration

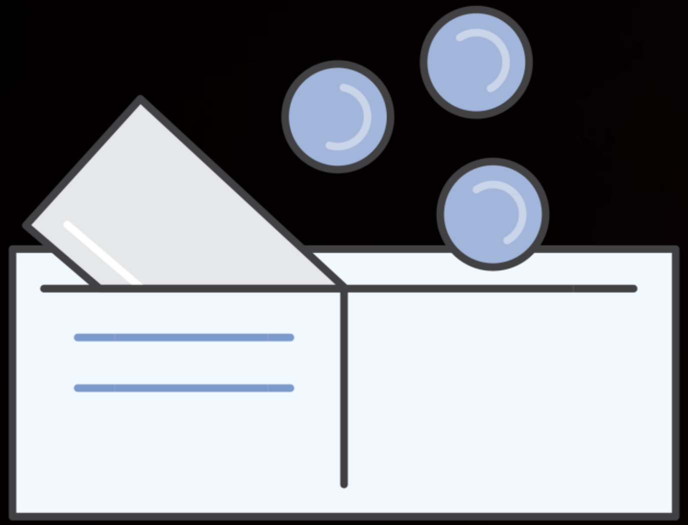
Timeout

- Up to 15 minutes
- Synchronous vs. asynchronous
- API Gateway timeout = 30 seconds

Network access

- Configure access to Amazon VPC
- Security group rules apply
- Amazon VPC does **not** enhance security of function

Fine-grained pricing



AWS Free Tier
1M requests and 400,000 GBs of compute
Every month, every customer

Pay for value

- Priced by power rating
- Charged in **1 ms** increments
- Low per-request charge

No minimum

Never pay for idle

AWS Serverless Application Model (AWS SAM)

- AWS CloudFormation extension **optimized for serverless**
- Shorthand syntax to express functions, APIs, databases, and event source mappings
- Simplifies IAM policy and event trigger management
- Model with YAML, deploy using AWS CloudFormation
- Open source!

https://aws.amazon.com/serverless/sam/
https://github.com/awslabs/serverless-application-model

AWS SAM template example

```
AWSTemplateFormatVersion: '2010-09-09’
Transform: AWS::Serverless-2016-10-31
Resources:
  GetHtmlFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: s3://sam-demo-bucket/todo_list.zip
      Handler: index.handler
      Runtime: nodejs14.x
      Policies: DynamoDBReadPolicy
      Events:
        GetToDo:
          Type: Api
          Properties:
            Path: /todo/{id}
            Method: GET
  ListTable:
    Type: AWS::Serverless::SimpleTable
```

SAM template transform

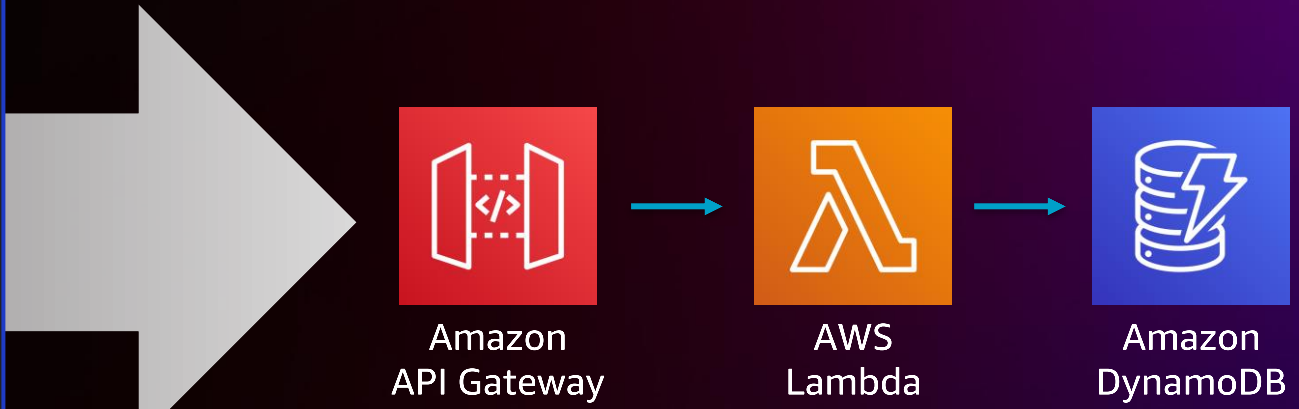
Creates Lambda function
- Runtime
- Execution policy
- Code
- Handler

Creates API Gateway
- API endpoint
- Permissions

Create DynamoDB table with sane defaults

AWS SAM template example

```
AWSTemplateFormatVersion: '2010-09-09’
Transform: AWS::Serverless-2016-10-31
Resources:
  GetHtmlFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: s3://sam-demo-bucket/todo_list.zip
      Handler: index.handler
      Runtime: nodejs12.x
      Policies: DynamoDBReadPolicy
      Events:
        GetToDo:
          Type: Api
          Properties:
            Path: /todo/{id}
            Method: GET

  ListTable:
    Type: AWS::Serverless::SimpleTable
```



/todo/{id} - GET

AWS SAM CLI

- CLI tool for local building, validating, and testing of serverless applications
- Works with Lambda functions and proxy-style APIs
- Response object and function logs available on your local machine
- Mimic Lambda execution environment with Docker images
- Emulates timeout, memory limits, runtimes

https://github.com/aws/aws-sam-cli

Getting started with SAM CLI

sam init

Generates a preconfigured AWS SAM template and example application code in the language that you choose

sam package

Bundles your application code and dependencies into a "deployment package"

sam build

Prepares it for subsequent steps like deploy or local testing

sam deploy

Deploys your serverless application to the AWS Cloud

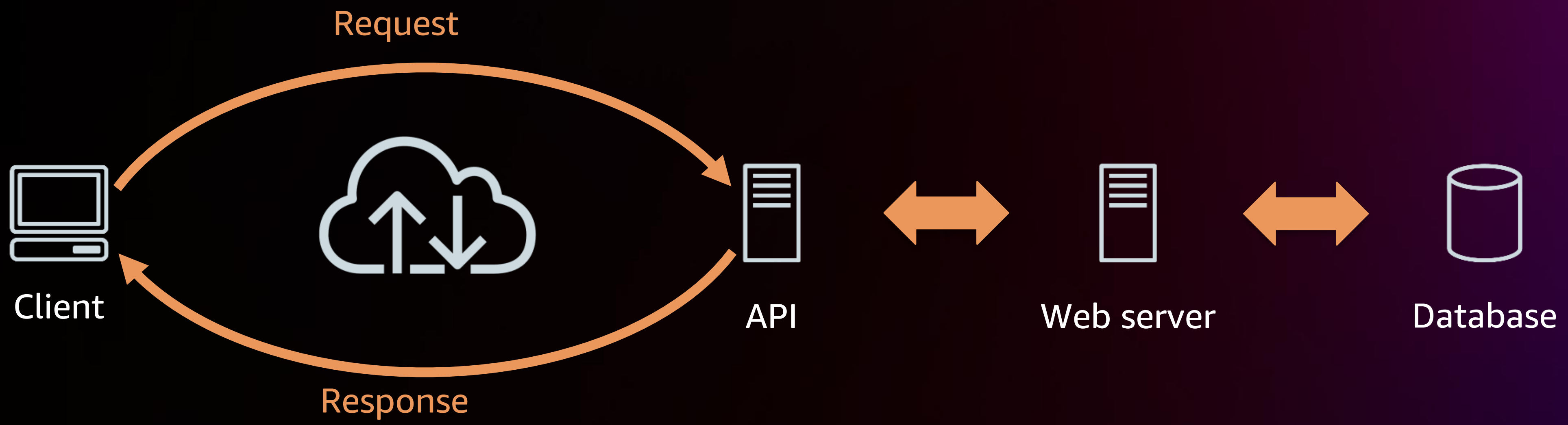
sam local

Test your application code locally

Step 2: Configure API authorization

What’s an API?

> “In building applications, an API simplifies programming by abstracting the underlying implementation and only exposing objects or actions the developer needs.”



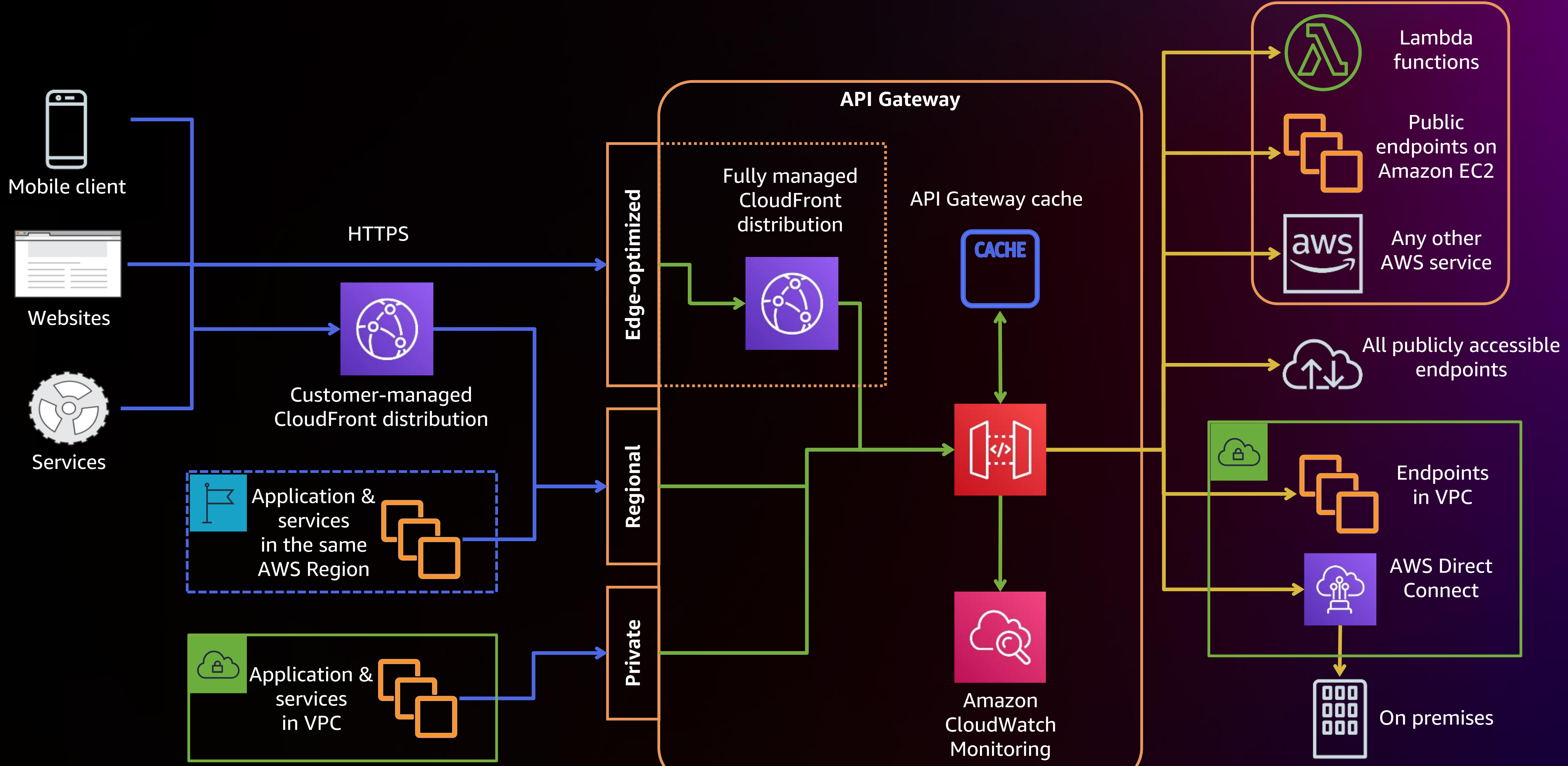
Web-based companies and services offer APIs for developers to use, such as:

- Social networks – Facebook, Twitter, etc.
- Payment processing – Amazon Pay, PayPal, etc.

Amazon API Gateway

Amazon API Gateway is a fully managed (serverless) service that makes it easy for developers to create, publish, maintain, monitor, and secure APIs at any scale

API architecture


Mobile client
Websites
Services
HTTPS
Customer-managed CloudFront distribution
Application & services in the same AWS Region
Application & services in VPC
API Gateway
Edge-optimized
Regional
Private
Fully managed CloudFront distribution
API Gateway cache
CACHE
Amazon CloudWatch Monitoring
Lambda functions
Public endpoints on Amazon EC2
aws
Any other AWS service
All publicly accessible endpoints
Endpoints in VPC
AWS Direct Connect
On premises

Endpoint types

Edge-optimized

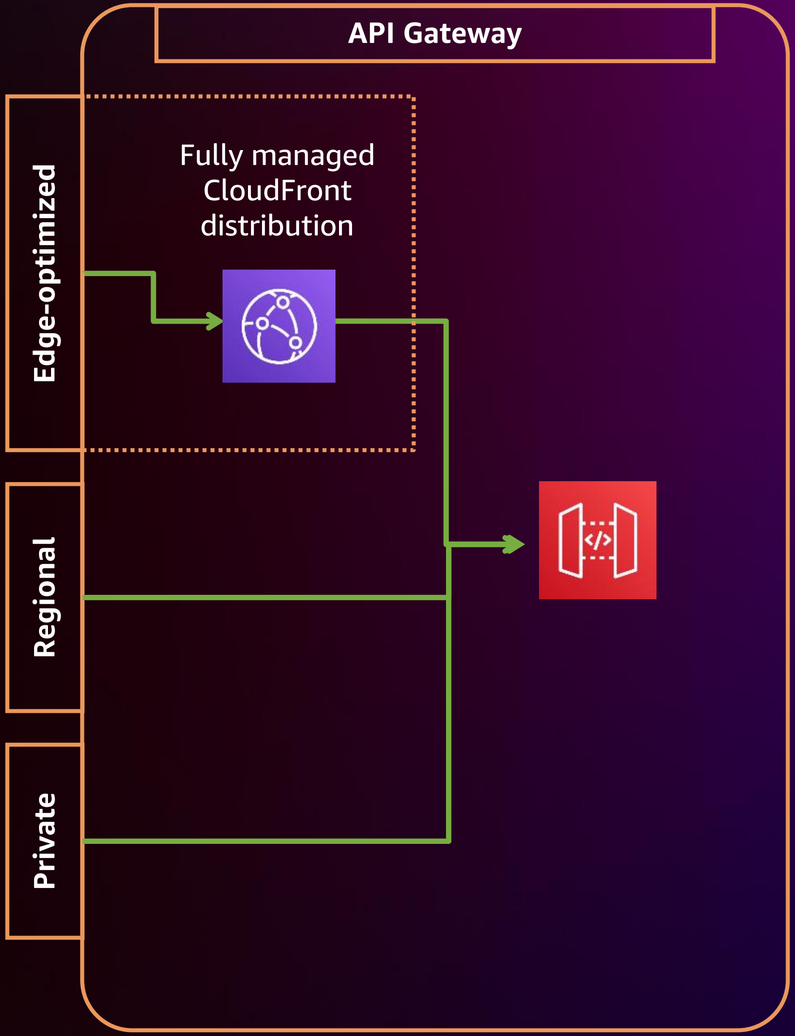
- Uses CloudFront to reduce TLS connection overhead (reduces round-trip time)
- Designed for a globally distributed set of clients

Regional

- Recommended API type for general use cases
- Designed for building APIs for clients in the same region

Private

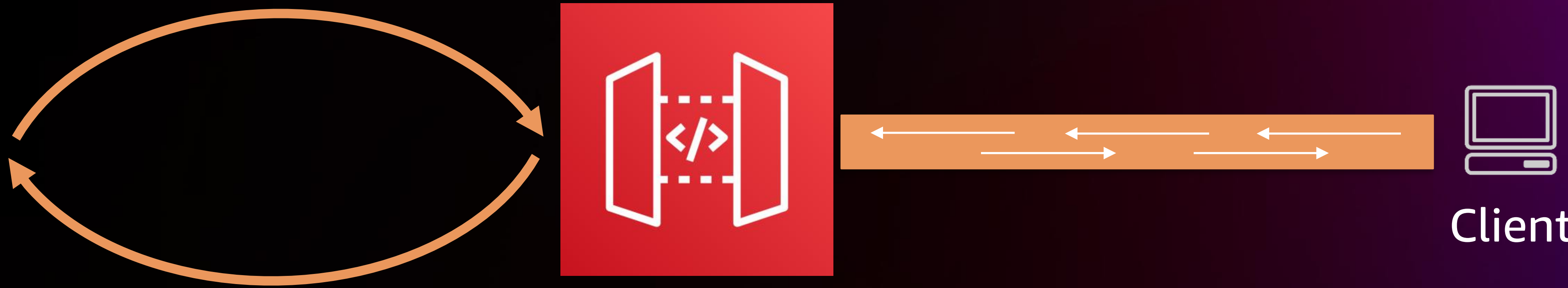
- Only accessible from within VPC (and networks connected to VPC)
- Designed for building APIs used internally or by private microservices



Supported protocols

RESTful APIs

WebSocket APIs



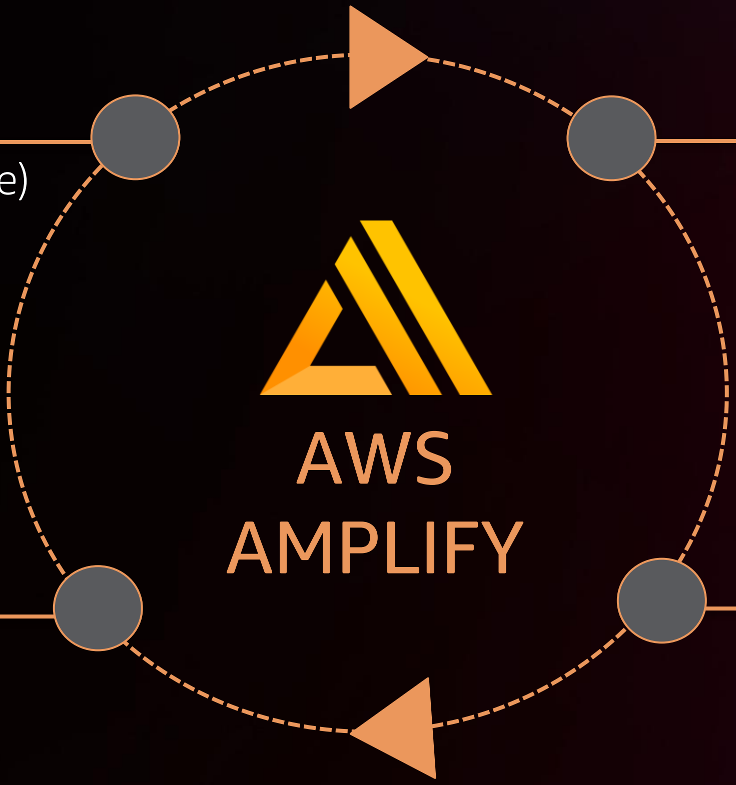
- Request/response
- HTTP methods like GET, POST, etc.
- Short-lived communication
- Stateless

- Serverless WebSocket
- 2-way communication channel
- Long-lived communication
- Stateful

Step 3: Build and deploy a web application

AWS Amplify

DESIGNED FOR THE DEVELOPMENT LIFECYCLE



AWS AMPLIFY

ENGAGE/MEASURE

- Multi-channel (push/SMS/email/voice)
- Behavior-based and personalized audience segments
- Engagement performance measurements
- Real-time customer data for immediate optimization

DEPLOY/HOST

- AWS infrastructure
- Ease of deployment with CLI or hosting
- CI/CD capability
- Fully managed global hosting

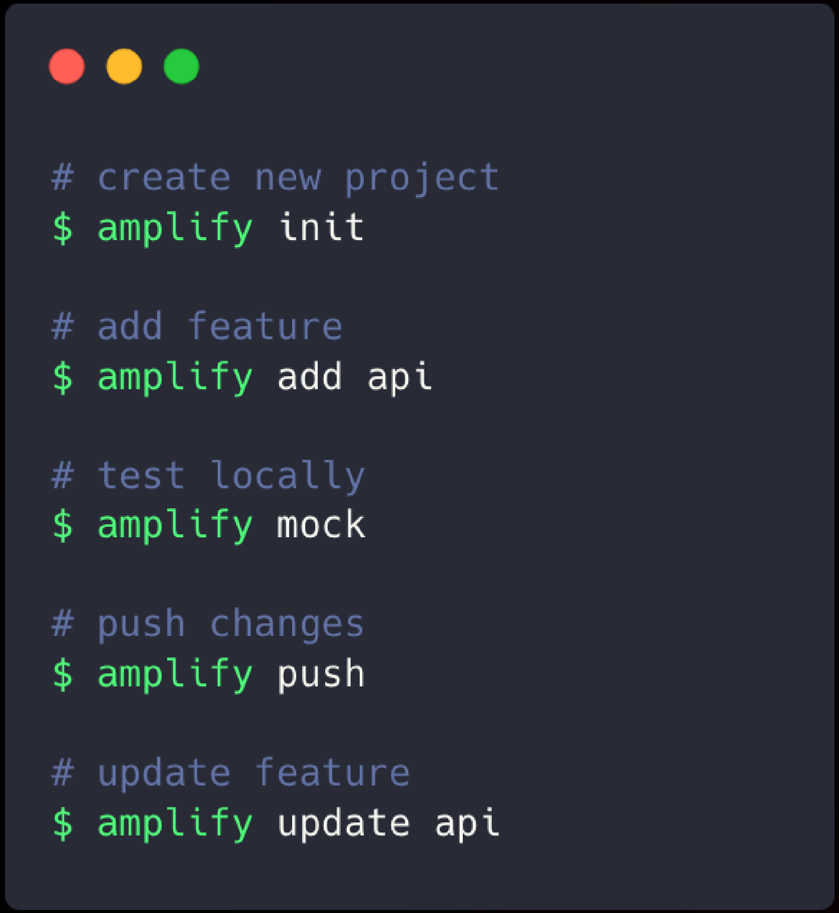
DEVELOP

- Open-source libraries
- iOS-/Android-native
- JavaScript/React/Vue/Angular
- UI components
- Escape hatches

TEST

- AWS Device Farm
- Test on real devices
- Test on real browsers
- Integrate testing with CI/CD

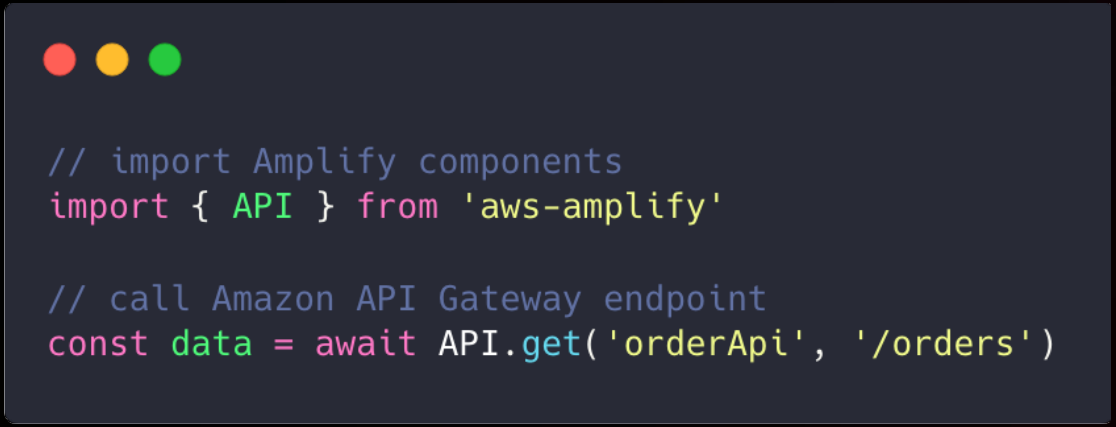
AWS Amplify CLI



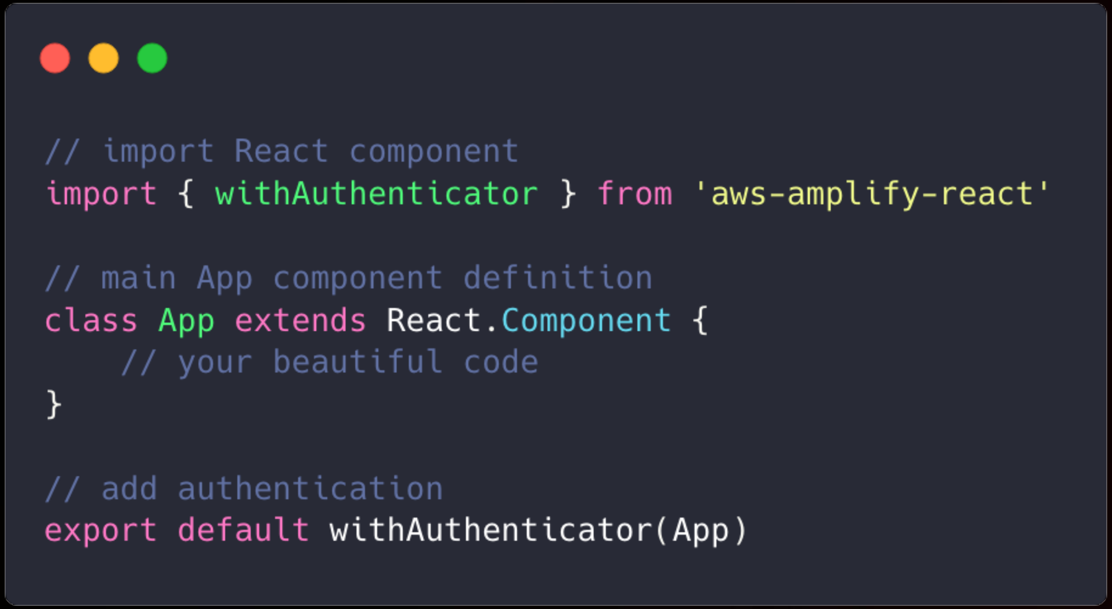
```
# create new project
$ amplify init

# add feature
$ amplify add api

# test locally
$ amplify mock

# push changes
$ amplify push

# update feature
$ amplify update api
```

- Create, update, and delete cloud services
- Manage multiple environments
- GraphQL Transform
- GraphQL codegen

AWS Amplify Client



```
// import Amplify components
import { API } from 'aws-amplify'

// call Amazon API Gateway endpoint
const data = await API.get('orderApi', '/orders')
```



```
// import React component
import { withAuthenticator } from 'aws-amplify-react'

// main App component definition
class App extends React.Component {
    // your beautiful code
}

// add authentication
export default withAuthenticator(App)
```

Android

iOS

React Native

Ionic

JavaScript

React

Angular

Vue

AWS Amplify libraries

Authentication

Authentication APIs with prebuilt UI components for your application

DataStore

On-device persistent storage that automatically synchronizes data between your apps and the cloud

API

HTTP requests using REST and GraphQL with support for real-time data

Analytics

Track user sessions, custom user attributes, and in-app metrics

PubSub

Connect your app to message-oriented middleware on the cloud

Predictions

Add AI/ML capabilities to your app powered by cloud services

Interactions

Conversational bots powered by deep learning technologies

Notifications

Push notifications with campaign analytics and targeting

Storage

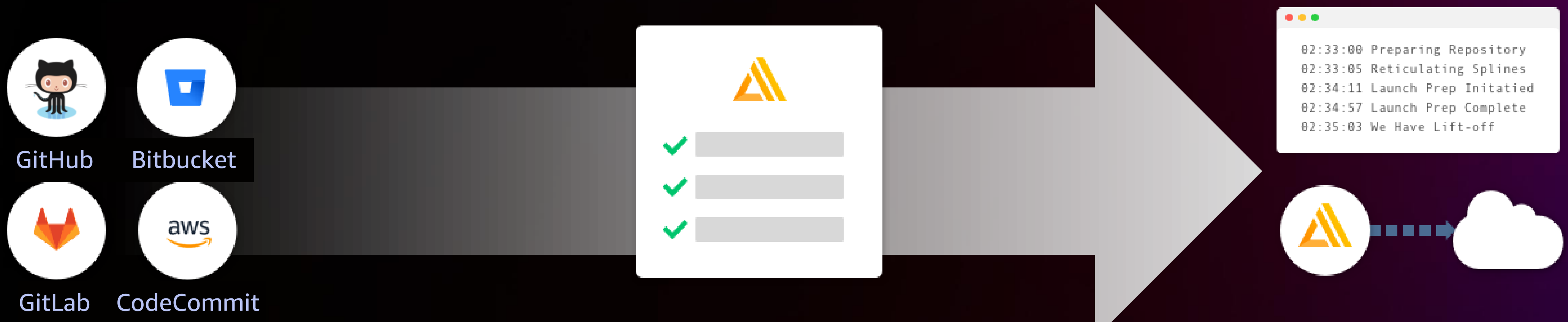
Manage user content securely in public, protected, and private storage

XR

Work with augmented reality and virtual reality content in your apps

AWS Amplify Console

BUILD, DEPLOY, AND HOST CLOUD-POWERED MODERN WEB APPS



```
02:33:00 Preparing Repository
02:33:05 Reticulating Splines
02:34:11 Launch Prep Initatied
02:34:57 Launch Prep Complete
02:35:03 We Have Lift-off
```

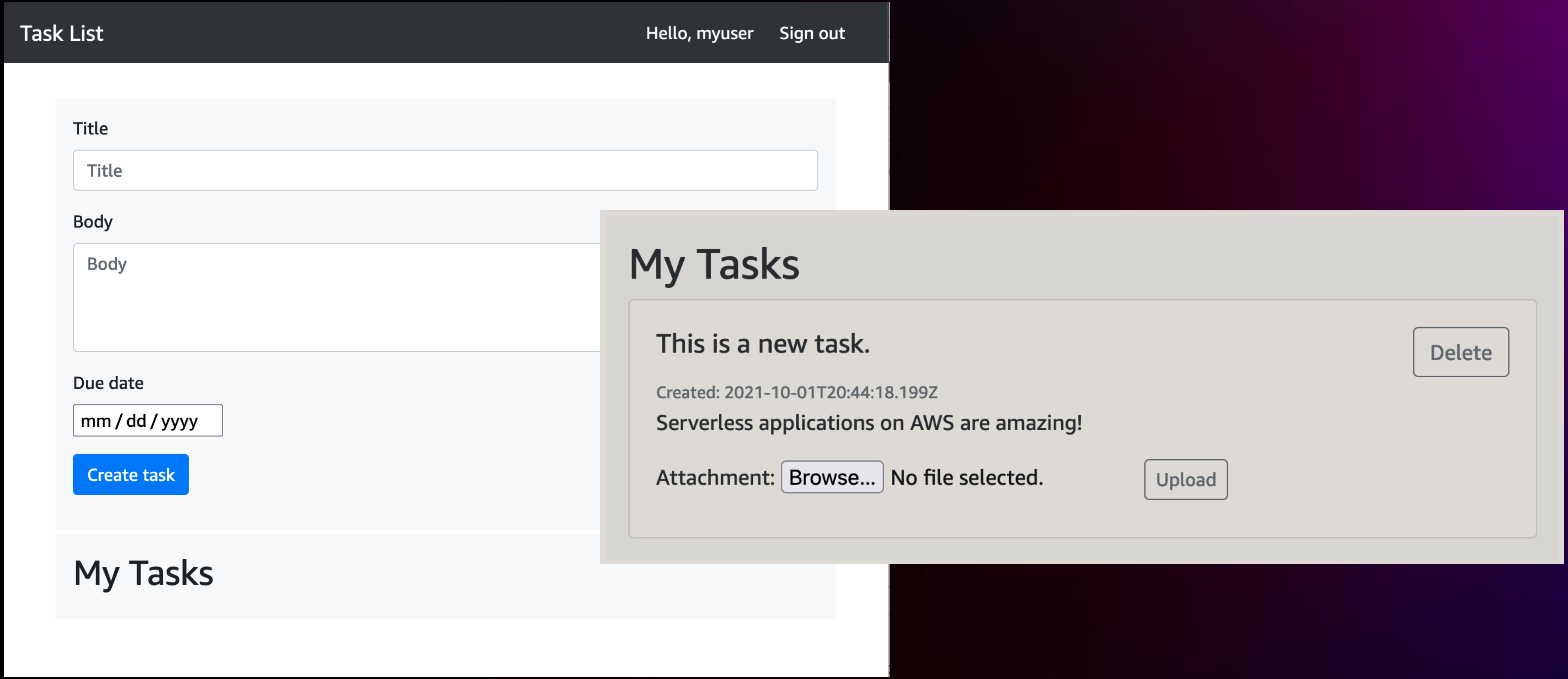
Connect your repository

Configure build settings

Deploy your application

Step 4: Test the application

Create a task



Task List

Hello, myuser

Sign out

Title

Title

Body

Body

Due date

mm / dd / yyyy

Create task

My Tasks

My Tasks

This is a new task.

Delete

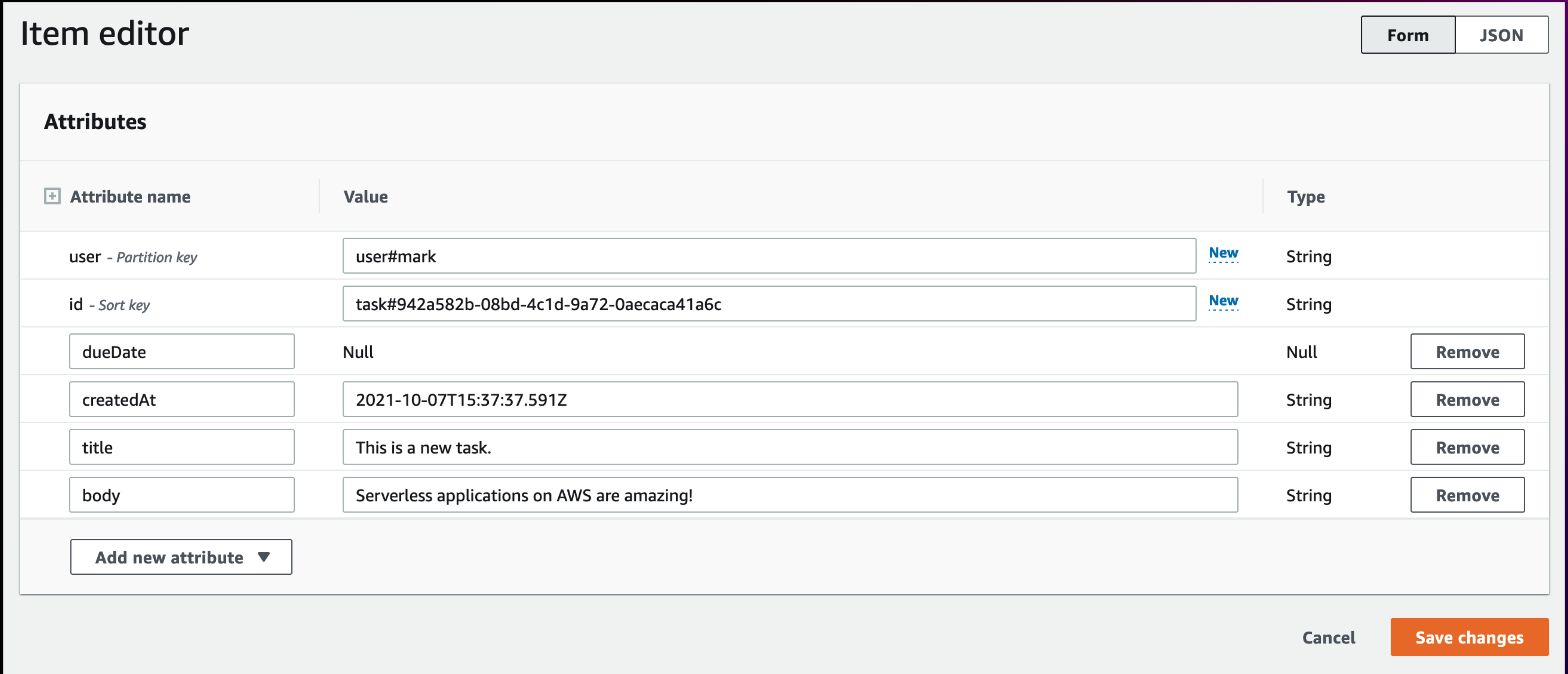
Created: 2021-10-01T20:44:18.199Z

Serverless applications on AWS are amazing!

Attachment: Browse... No file selected.

Upload

View data in DynamoDB

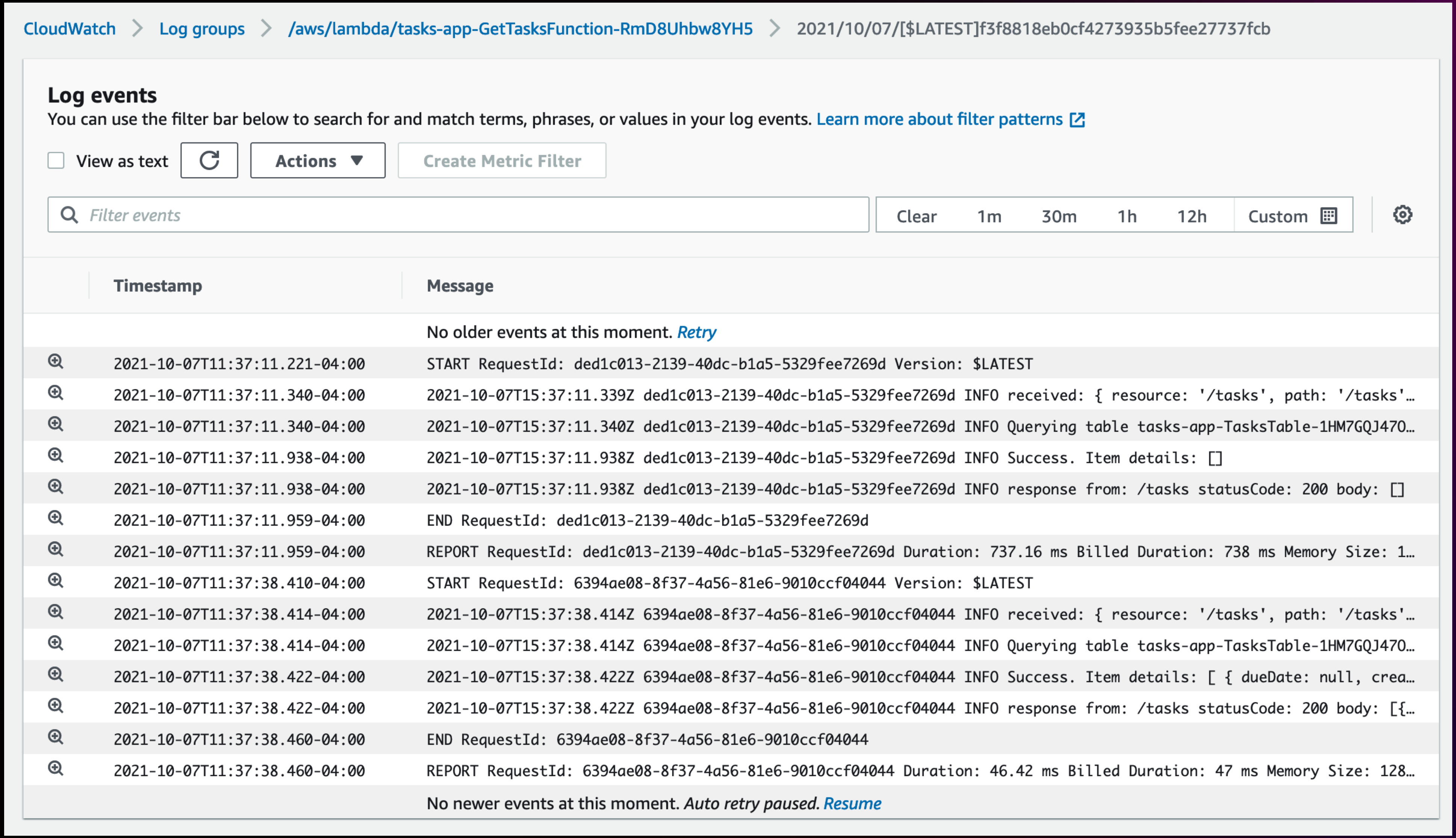


Item editor

Attributes

Attribute name	Value	Type
user - *Partition key*	user#mark	String
id - *Sort key*	task#942a582b-08bd-4c1d-9a72-0aecaca41a6c	String
dueDate	Null	Null
createdAt	2021-10-07T15:37:37.591Z	String
title	This is a new task.	String
body	Serverless applications on AWS are amazing!	String

Logging and monitoring



CloudWatch > Log groups > /aws/lambda/tasks-app-GetTasksFunction-RmD8Uhbw8YH5 > 2021/10/07/[$LATEST]f3f8818eb0cf4273935b5fee27737fcb

Log events

You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

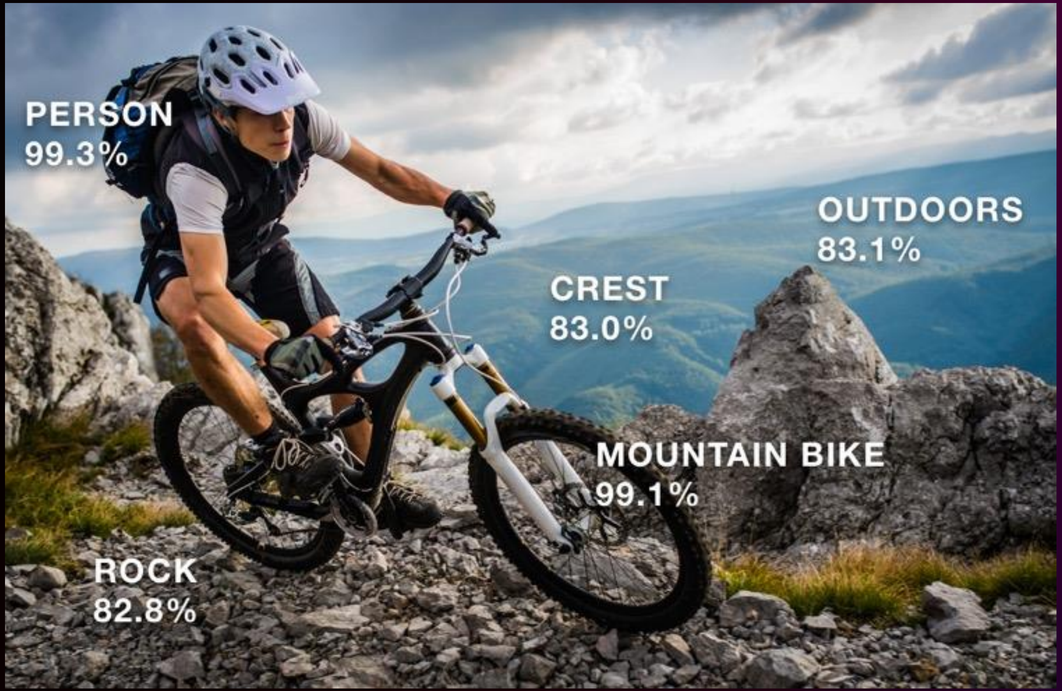
□ View as text
Actions ▼
Create Metric Filter
Filter events
Clear
1m
30m
1h
12h
Custom

```
Timestamp                         Message
                                  No older events at this moment. Retry
2021-10-07T11:37:11.221-04:00     START RequestId: ded1c013-2139-40dc-b1a5-5329fee7269d Version: $LATEST
2021-10-07T11:37:11.340-04:00     2021-10-07T15:37:11.339Z ded1c013-2139-40dc-b1a5-5329fee7269d INFO received: { resource: '/tasks', path: '/tasks'…
2021-10-07T11:37:11.340-04:00     2021-10-07T15:37:11.340Z ded1c013-2139-40dc-b1a5-5329fee7269d INFO Querying table tasks-app-TasksTable-1HM7GQJ470…
2021-10-07T11:37:11.938-04:00     2021-10-07T15:37:11.938Z ded1c013-2139-40dc-b1a5-5329fee7269d INFO Success. Item details: []
2021-10-07T11:37:11.938-04:00     2021-10-07T15:37:11.938Z ded1c013-2139-40dc-b1a5-5329fee7269d INFO response from: /tasks statusCode: 200 body: []
2021-10-07T11:37:11.959-04:00     END RequestId: ded1c013-2139-40dc-b1a5-5329fee7269d
2021-10-07T11:37:11.959-04:00     REPORT RequestId: ded1c013-2139-40dc-b1a5-5329fee7269d Duration: 737.16 ms Billed Duration: 738 ms Memory Size: 1…
2021-10-07T11:37:38.410-04:00     START RequestId: 6394ae08-8f37-4a56-81e6-9010ccf04044 Version: $LATEST
2021-10-07T11:37:38.414-04:00     2021-10-07T15:37:38.414Z 6394ae08-8f37-4a56-81e6-9010ccf04044 INFO received: { resource: '/tasks', path: '/tasks'…
2021-10-07T11:37:38.414-04:00     2021-10-07T15:37:38.414Z 6394ae08-8f37-4a56-81e6-9010ccf04044 INFO Querying table tasks-app-TasksTable-1HM7GQJ470…
2021-10-07T11:37:38.422-04:00     2021-10-07T15:37:38.422Z 6394ae08-8f37-4a56-81e6-9010ccf04044 INFO Success. Item details: [ { dueDate: null, crea…
2021-10-07T11:37:38.422-04:00     2021-10-07T15:37:38.422Z 6394ae08-8f37-4a56-81e6-9010ccf04044 INFO response from: /tasks statusCode: 200 body: [{…
2021-10-07T11:37:38.460-04:00     END RequestId: 6394ae08-8f37-4a56-81e6-9010ccf04044
2021-10-07T11:37:38.460-04:00     REPORT RequestId: 6394ae08-8f37-4a56-81e6-9010ccf04044 Duration: 46.42 ms Billed Duration: 47 ms Memory Size: 128…
                                  No newer events at this moment. Auto retry paused. Resume
```

Step 5: Configure image metadata extraction

Amazon Rekognition

DEEP LEARNING-BASED IMAGE AND VIDEO ANALYSIS



Amazon Rekognition is a service that can identify objects, people, text, scenes, and activities in images and videos

Facial analysis

Demographic data

Age range:	29–45
Appears to be male:	96.5%

Facial landmarks

EyeLeft, EyeRight, Nose, RightPupil, LeftPupil, MouthRight, LeftEyeBrowUp Bounding Box…

Image quality

Brightness	23.6%
Sharpness	99.9%

Emotion expressed

Happy	83.8%
Surprised	0.65%

General attributes

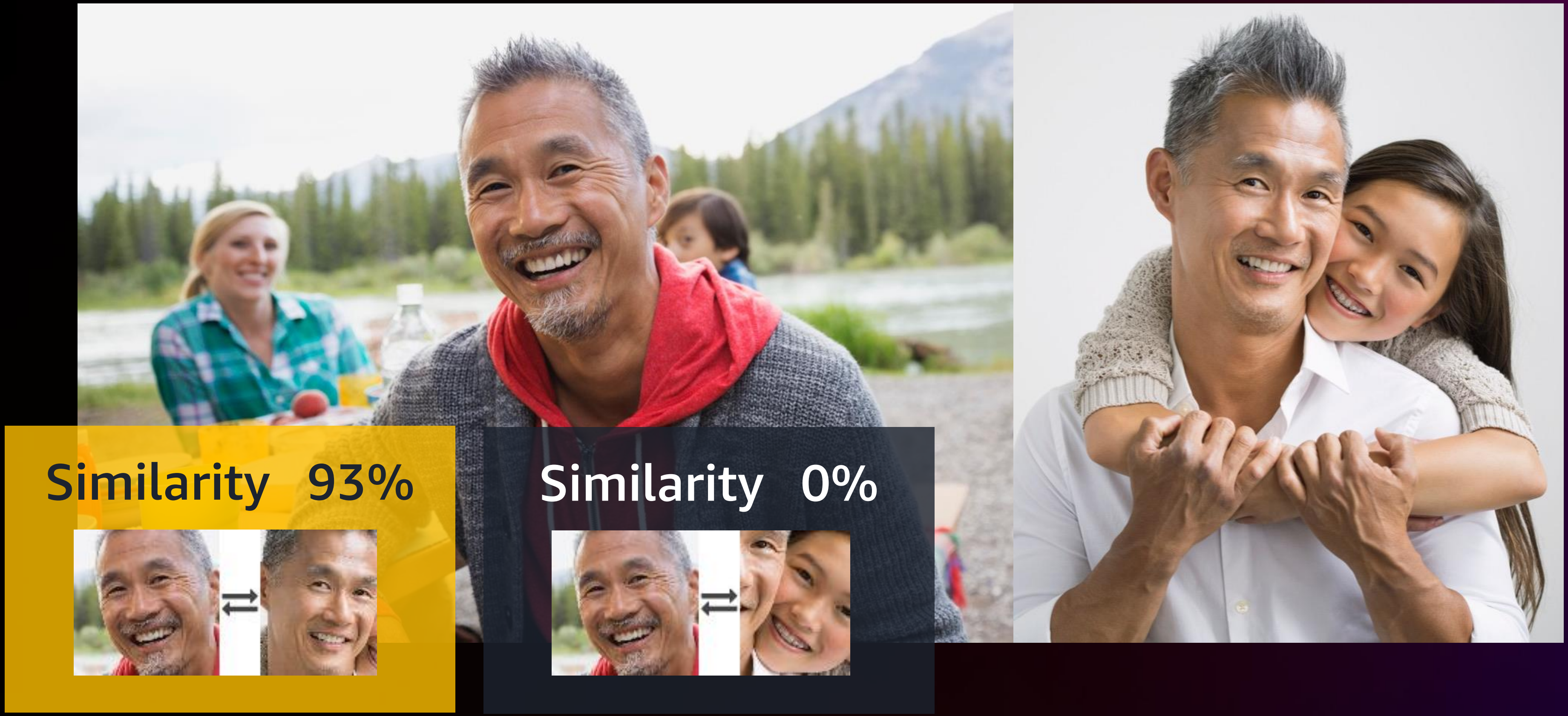
Smile: True	23.6%
Eyes Open: True	99.8%

Facial pose

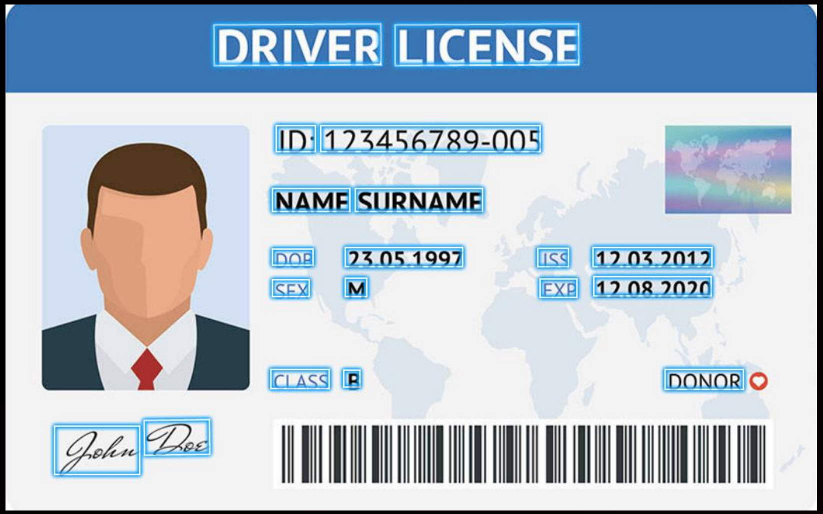
Pitch	1.446
Roll	5.725
Yaw	4.383

Face comparison

MEASURE THE LIKELIHOOD THAT FACES ARE OF THE SAME PERSON



Text detection



Extract text content from real-world images and videos in various layouts, fonts, and styles

Step 6: Terminate the resources

Terminating resources

- Not necessary in an AWS hosted event
- In your own account:
  - `sam delete`
  - `amplify delete`
  - Delete Amplify IAM user
  - Delete S3 bucket content

Workshop link

s12d.com/svs203

Thank you!

Mark Richman

mrkrchm@amazon.com

in mrichman

Please complete the session survey in the mobile app