



C - 4

# Best practices for building interactive applications with AWS Lambda

Benjamin Smith

Principal Developer Advocate, AWS  
Serverless

# About Me



- Ben Smith
  - Email: [benjasl@amazon.com](mailto:benjasl@amazon.com)
  - Twitter: [@benjamin\\_l\\_s](https://twitter.com/benjamin_l_s)
- Most interested in
  - Serverless developer workflow
  - Serverless and PHP
- Random fact
  - I once taught David Hasselhoff to ski

# Agenda

Interactive Lambda applications

Synchronous and asynchronous

Accessing business logic

Optimizing Lambda

# Today's focus

# AWS Lambda based applications



# Today's focus

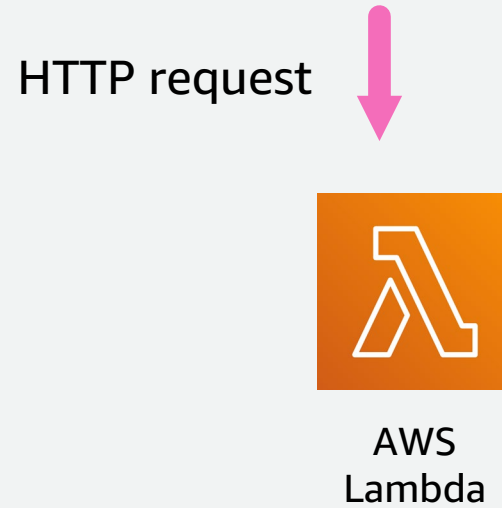
Interactive



AWS Lambda based applications

# Today's focus

## Interactive



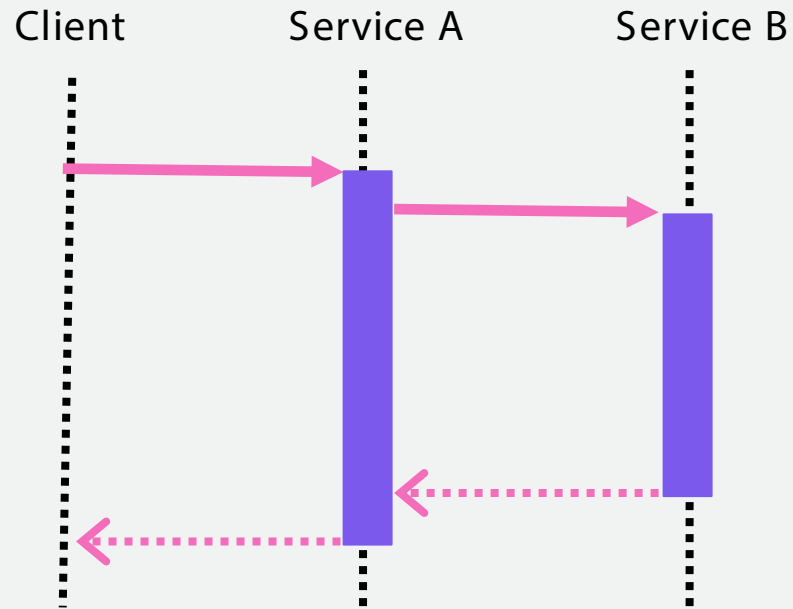
# Today's focus

## Interactive



AWS  
Lambda

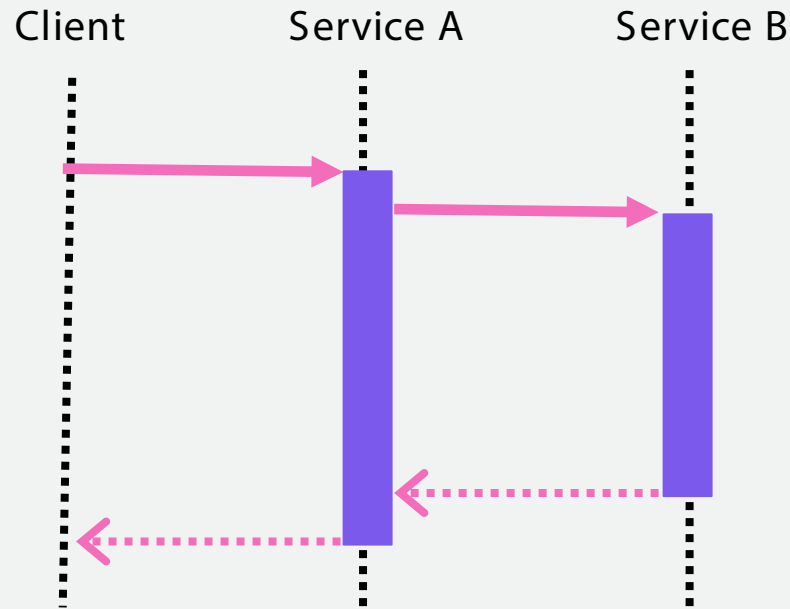
# Synchronous and asynchronous



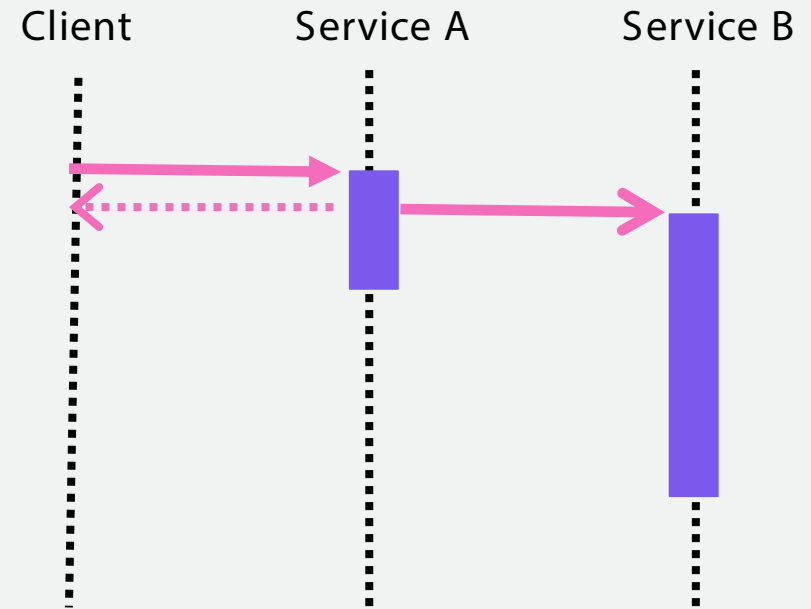
Synchronous  
commands



# Synchronous and asynchronous



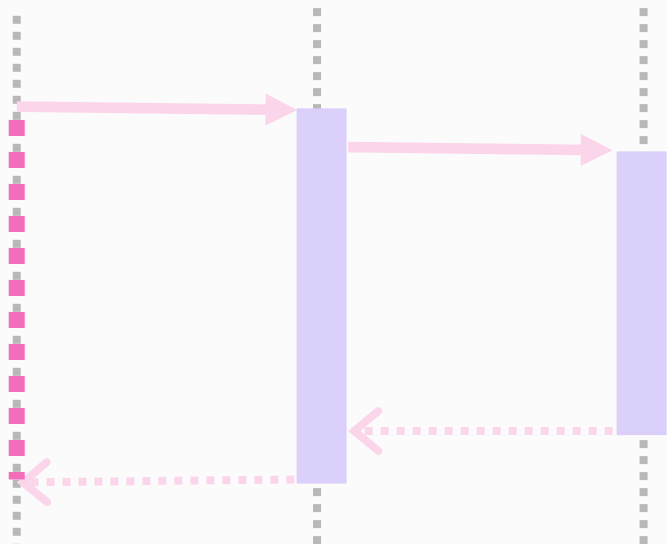
Synchronous  
commands



Asynchronous  
events

# Synchronous and asynchronous

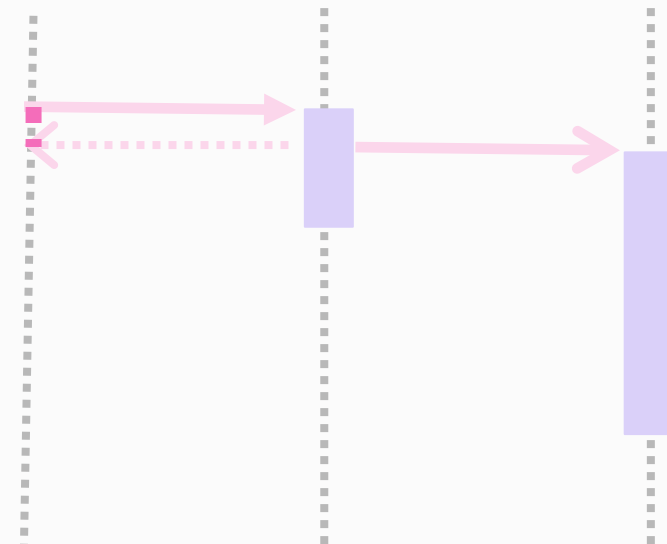
Client      Service A      Service B



---

Synchronous  
commands

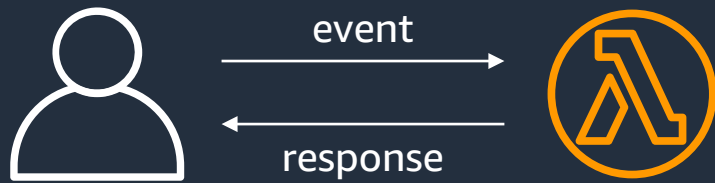
Client      Service A      Service B



---

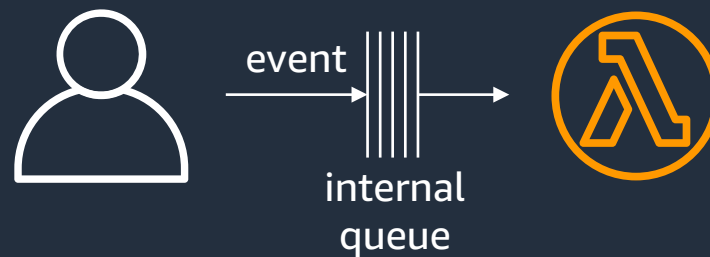
Asynchronous  
events

# Lambda invocation modes



## Synchronous

When the caller expects a response from the function



## Asynchronous

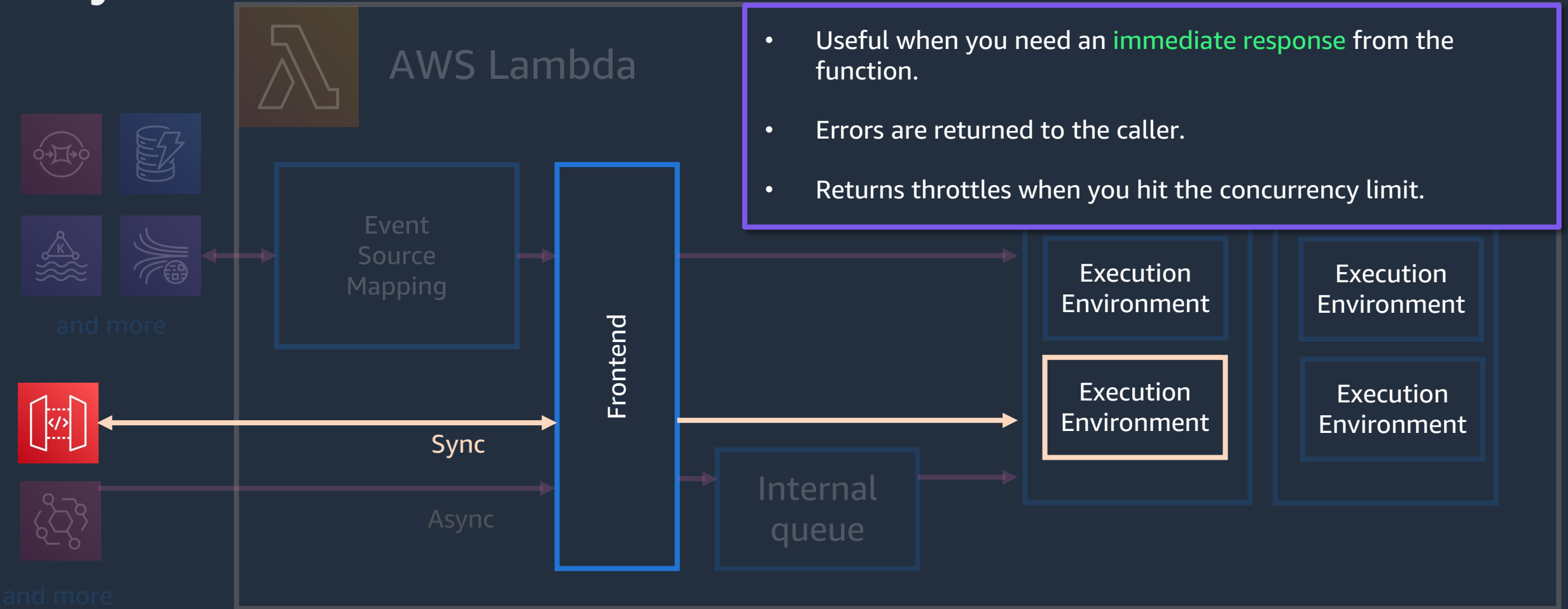
When the caller doesn't expect a response from the function



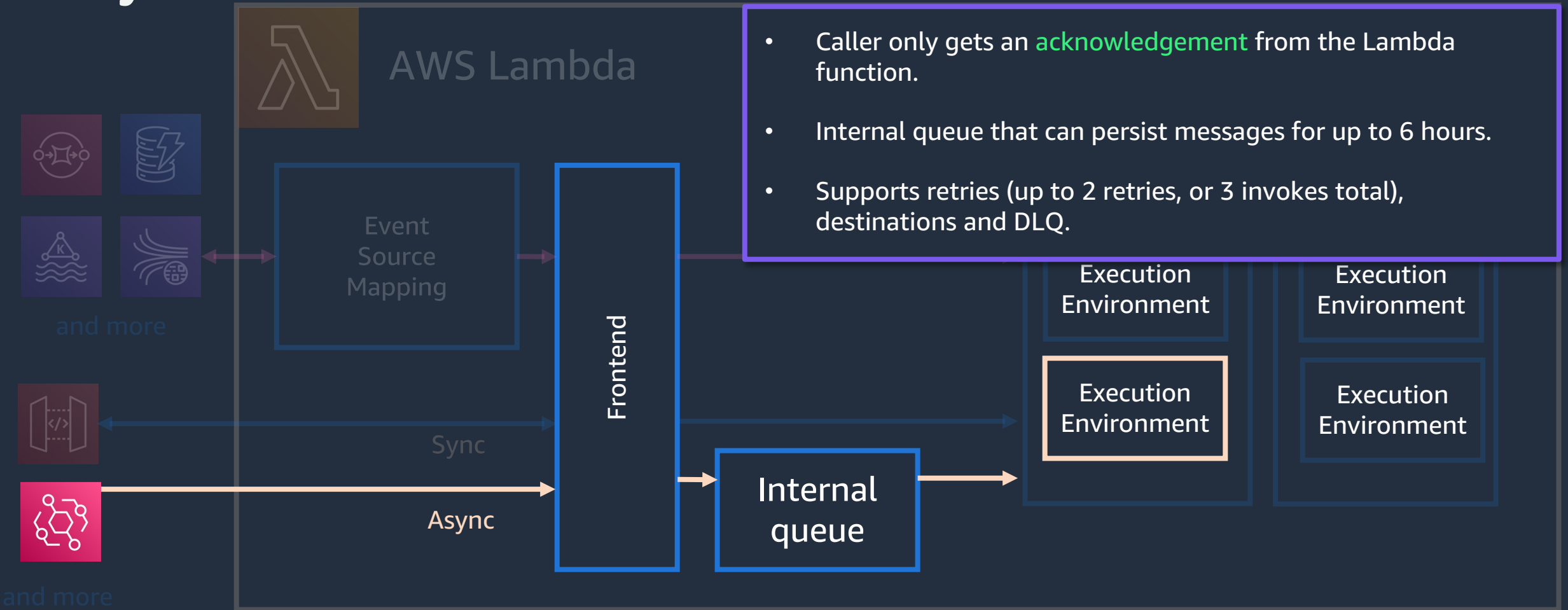
## Event Source Mapping

Integration with specific event sources  
Synchronous under the hood

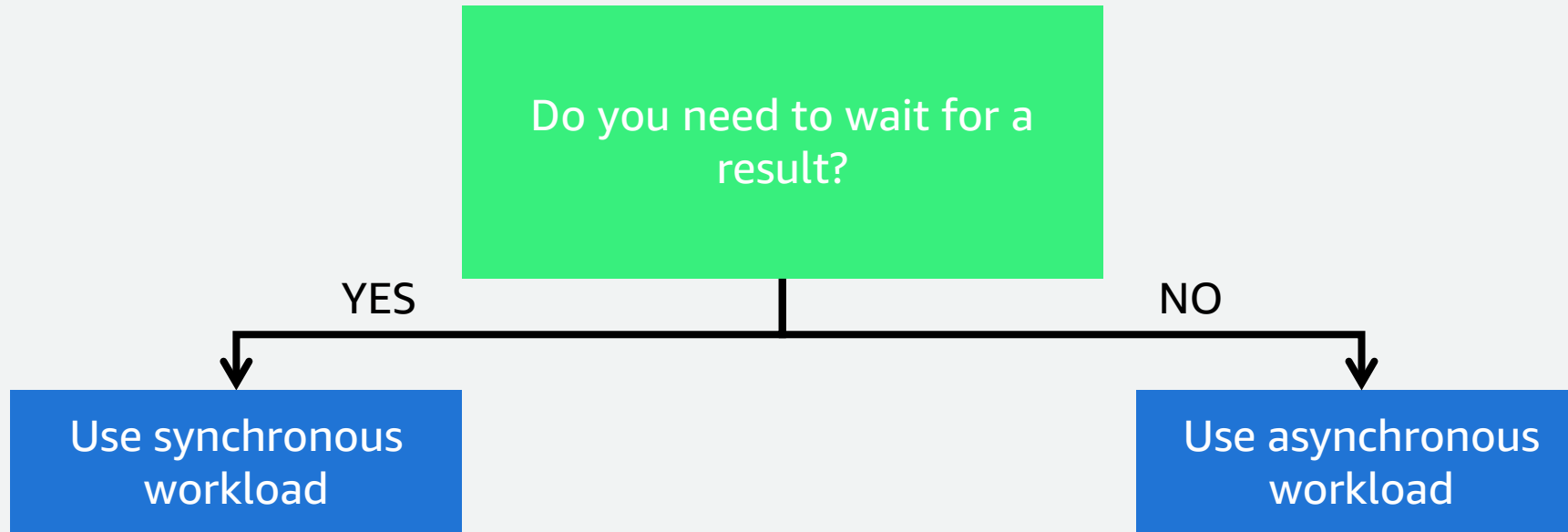
# Synchronous invocation mode



# Asynchronous invocation mode



# Choosing synchronous or asynchronous





# Interactive AWS Lambda applications

Connecting applications

Extending applications

Asynchronous workloads

ML inferencing

Web applications

Backend applications

Data processing

Chatbots

IT automation





# Interactive AWS Lambda applications

Connecting applications

Extending applications

Asynchronous workloads

ML inferencing

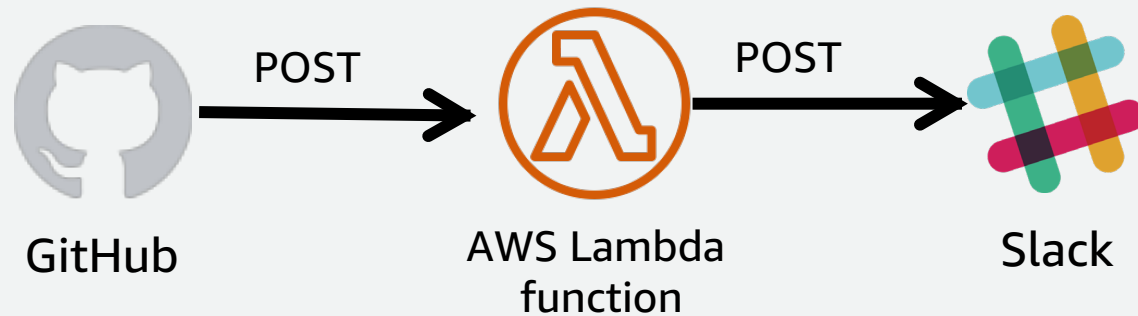
Web applications

Backend applications

Data processing

Chatbots

IT automation



Learn more

<http://s12d.com/getting-started>

# Interactive AWS Lambda applications

Connecting applications

Extending applications

Asynchronous workloads

ML inferencing

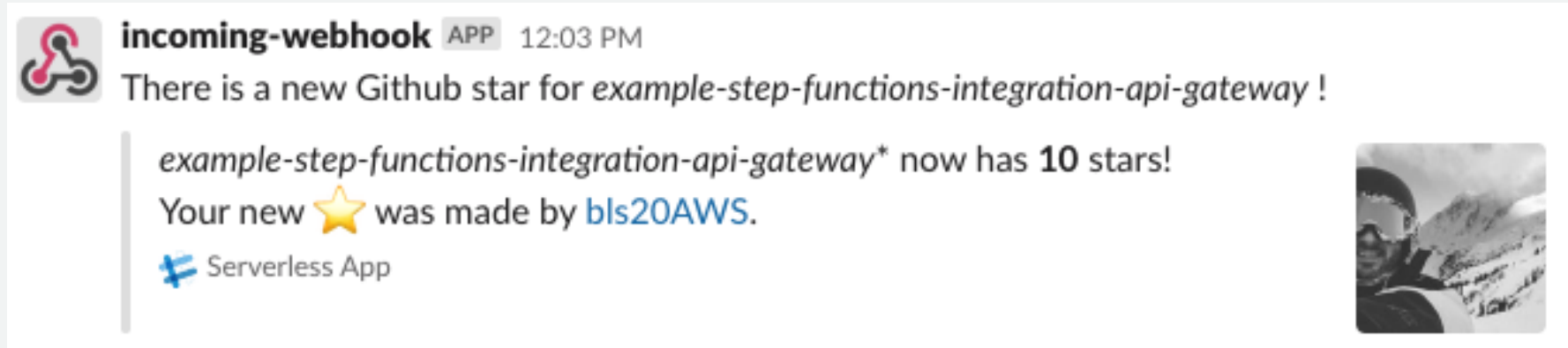
Web applications

Backend applications

Data processing

Chatbots

IT automation



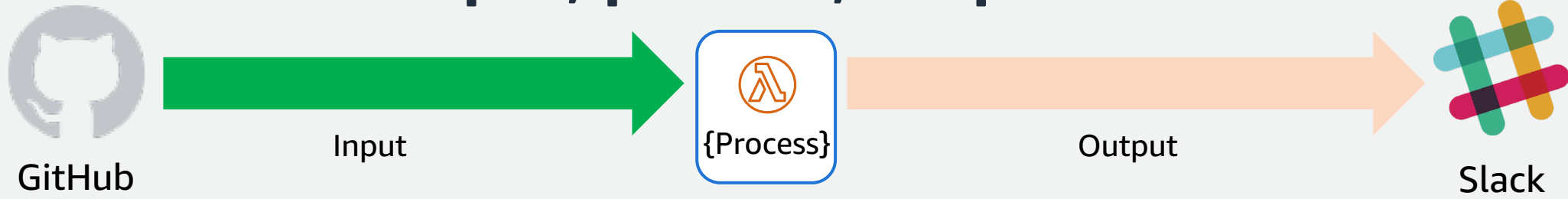
Learn more  
<http://s12d.com/getting-started>



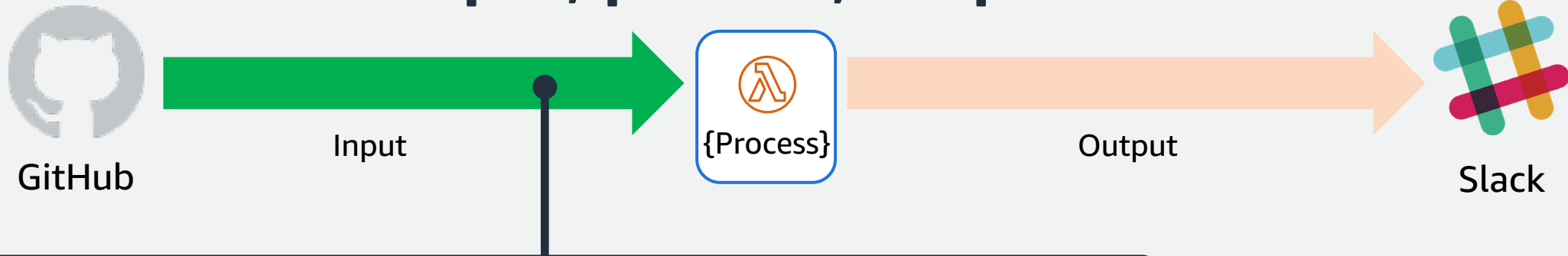
# The business logic

*Do “something” to some data  
to produce valuable output.*

# AWS Lambda - Input, process, output



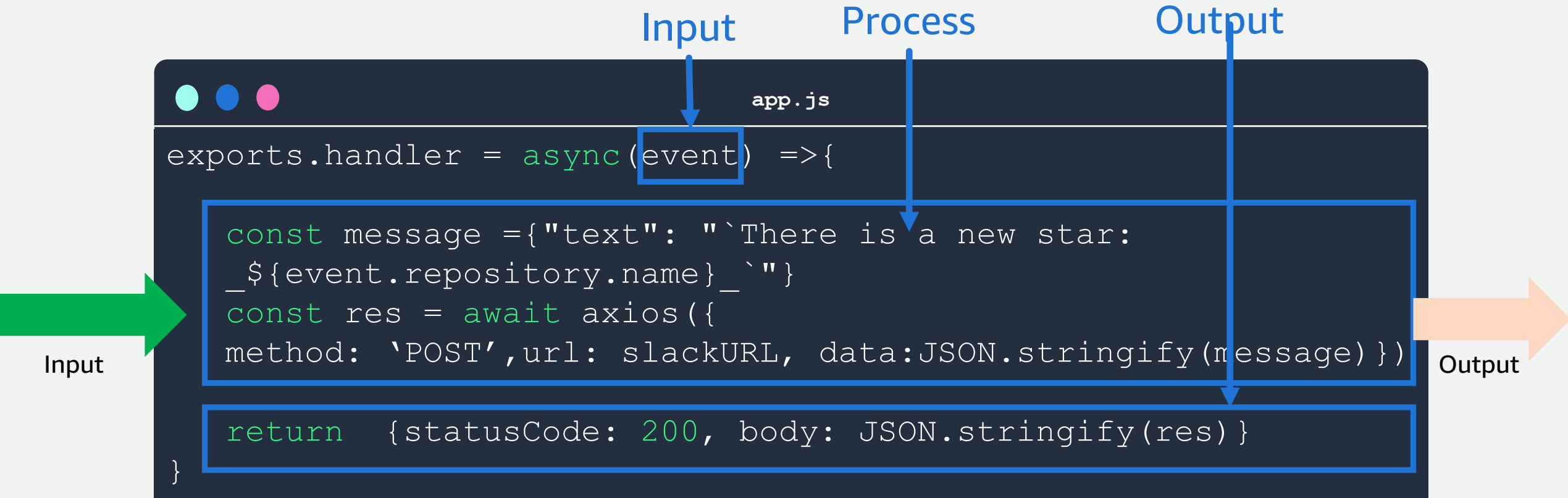
# AWS Lambda - Input, process, output



```
Event Payload
{
  "repository": {
    "name": "getting-started-with-serverless",
    "stargazers_count": 38
  },
  "sender": {
    "login": "bls20AWS"
  }
}
```

# AWS Lambda function

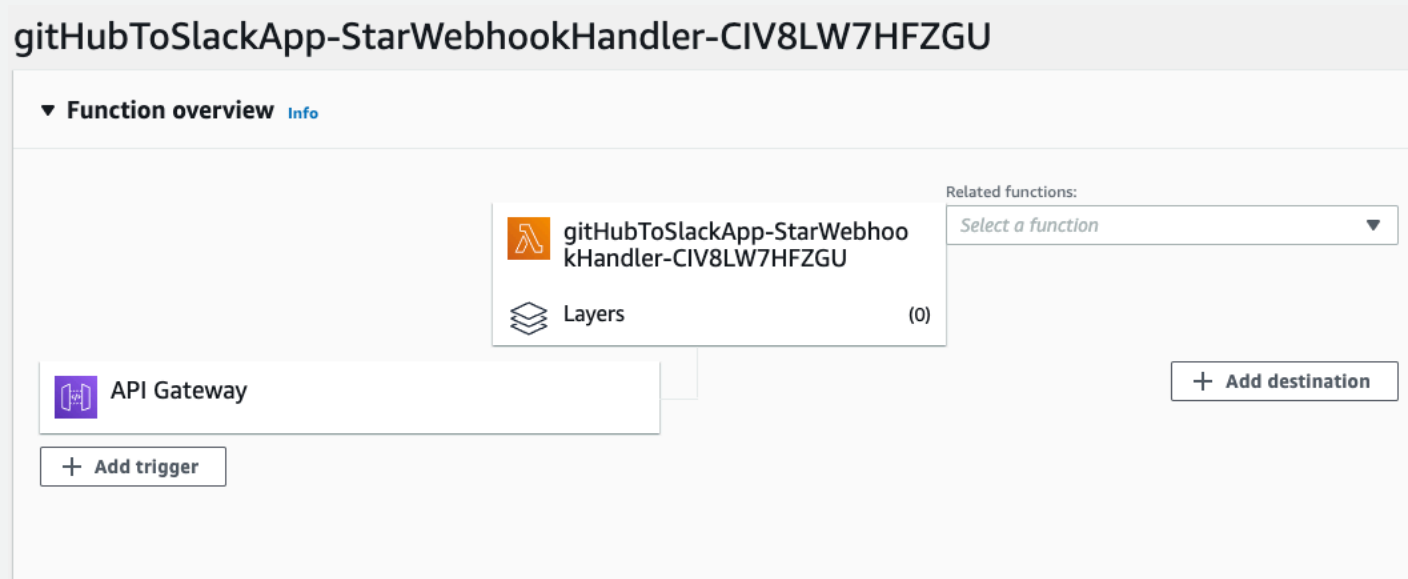
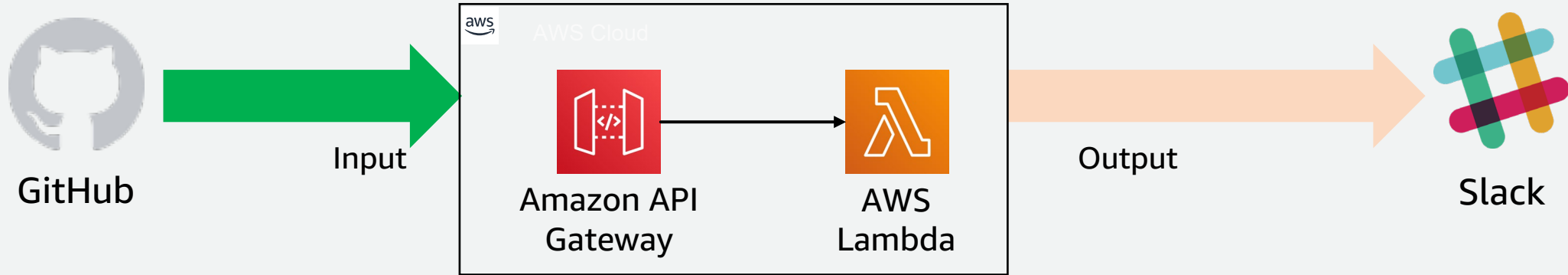
RUN BUSINESS LOGIC IN RESPONSE TO EVENTS.



# How did the event payload get from GitHub to AWS Lambda?



# AWS Lambda and Amazon API Gateway



**Customers were looking for new ways to simplify the number of steps it takes to access business logic in AWS Lambda functions**

# Introducing AWS Lambda function URLs

A simpler way to invoke AWS Lambda functions synchronously  
with an HTTP request



# AWS Lambda function URL

`https://<url-id>.lambda-url.<region>.on.aws/`



Learn more

<http://s12d.com/furls>

# AWS Lambda function URLs

A NEW WAY TO INVOKE AWS LAMBDA FUNCTIONS WITH AN HTTP REQUEST

- Part of the AWS Lambda service
- A separate resource in AWS CloudFormation
- Generates a unique URL to invoke an AWS Lambda function
- Can point to \$LATEST or a user-defined alias

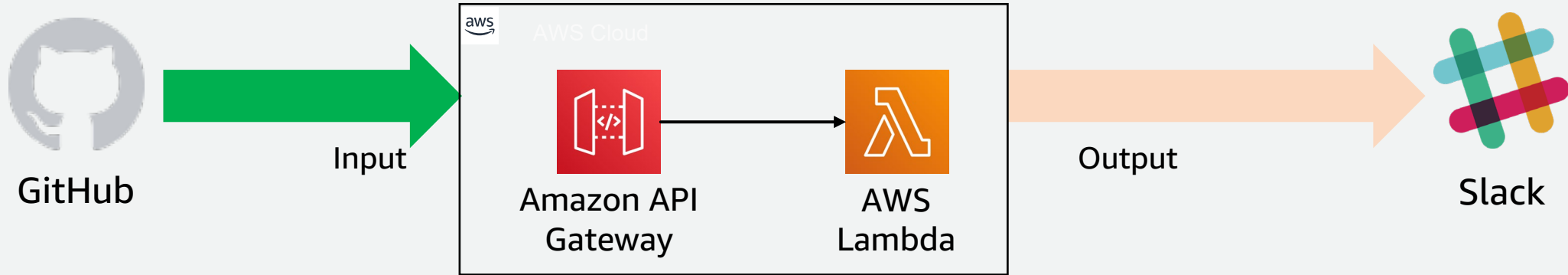
`https://<url-id>.lambda-url.<region>.on.aws/`



Learn more

<http://s12d.com/furls>

# AWS Lambda and Amazon API Gateway



gitHubToSlackApp-StarWebhookHandler-CIV8LW7HFZGU

▼ **Function overview** [Info](#)

gitHubToSlackApp-StarWebhookHandler-CIV8LW7HFZGU

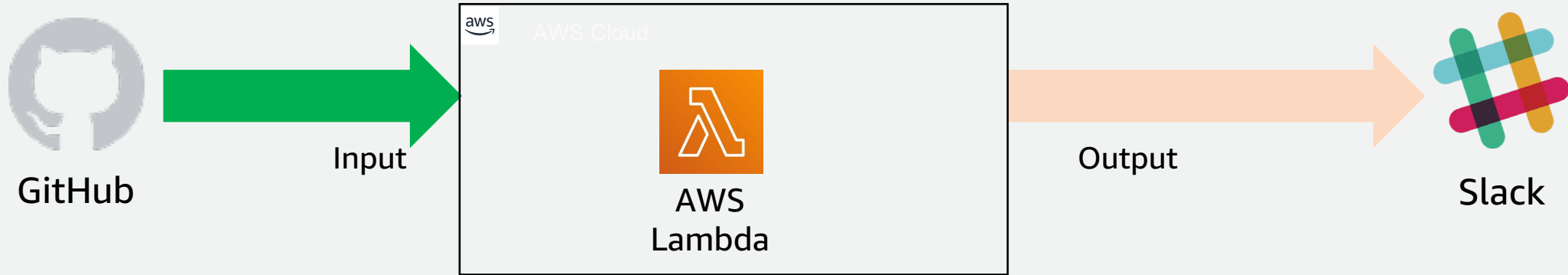
Layers (0)

Related functions:

API Gateway + Add destination


+ Add trigger


# AWS Lambda and function URLs




gitHubToSlackApp-StarWebhookHandler-CIV8LW7HFZGU

▼ **Function overview** [Info](#)

 gitHubToSlackApp-StarWebhookHandler-CIV8LW7HFZGU

 Layers (0)

Related functions:

 API Gateway + Add destination

+ Add trigger

# AWS Lambda function URLs

A SIMPLER WAY TO INVOKE LAMBDA FUNCTIONS WITH AN HTTP REQUEST

Create function URLs via:

- AWS Management Console.
- AWS CLI.
- AWS Serverless Application Model (AWS SAM).
- AWS CloudFormation.
- AWS CDK.



# AWS Lambda function URLs - Invoking

Browser

```
https://cxkeqw3klaw.lambda-urls.us-west-2.amazonaws.com
```

Curl

```
bash
curl https://cxkeqw3klaw.lambda-urls.us-west-2.on.aws/
```

Postman

```
GET curl https://cxkeqw3klaw.lambda-urls.us-west-2.amazonaws.com/
```

Programmatically


```
Get.js
axios.get(https://cxkeqw3klaw.lambda-urls.us-west-2.on.aws/, config)
```



# AWS Lambda function URLs - Authentication

- IAM authentication by default
- Optionally disable authentication
- Roll your own with code
- Add cross-origin resource sharing (CORS) configuration

## Authorization type

Choose the authorization type for your function URL. [Learn more](#) 


**IAM**

Use IAM to authorize requests to your function URL.

**None**

Lambda will not perform any authorization checks on requests to your function URL. The URL endpoint will be public unless you implement your own authorization logic in your function.

**Configure cross-origin resource sharing (CORS)**

Use CORS to allow access to your function URL from any origin. You can also use CORS to control access for specific HTTP headers and methods in requests to your function URL. By default, all origins are allowed. You can edit this after creating the function. [Learn more](#) 

# AWS Lambda function URLs - Authentication

Available Actions:

Action	Description	Access level
lambda:CreateFunctionUrlConfig	Create a function URL and set its AuthType.	Write
lambda:UpdateFunctionUrlConfig	Update a function URL configuration and its AuthType.	Write
lambda>DeleteFunctionUrlConfig	Grants permission to delete an AWS Lambda function url	Write
lambda:GetFunctionUrlConfig	View the details of a function URL.	Read
lambda:ListFunctionUrlConfigs	List function URL configurations.	Read
lambda:InvokeFunctionUrl	Grants permission to invoke a AWS Lambda function using the url	Write

# AWS Lambda function URLs - Authentication

ENSURE THAT IAM USER OR ROLE HAS PERMISSION TO INVOKE VIA FUNCTION URL

## IAM user permissions

```
{
  "SID": "InvokeFunctions",
  "Effect": "Allow",
  "Action": [
    "lambda:InvokeFunctionUrl"
  ],
  "Resource": "arn:aws:lambda:*:*:url:*",
  "Condition": {
    "StringLike": {
      "lambda:FunctionArn":
        "arn:aws:lambda:*:123456789012:function:dev-*"
    }
  }
}
```

# AWS Lambda function URLs - CORS

A BROWSER SECURITY FEATURE TO ENABLE ACCESS TO RESOURCES LOCATED OUTSIDE OF A DOMAIN.

**ERROR:** Access to fetch from origin has been blocked by **CORS** policy: Response to preflight request doesn't pass access control check:

No '**Access-Control-Allow-Origin**' header is present on the requested resource.



“ If CORS had a face, I’d punch it in the nose ,”

**Eric Johnson**

Principal Developer Advocate, AWS Serverless

# AWS Lambda function URLs - CORS

## Configure with Function URLs

Configure CORS headers on a Lambda function URL resource.

## Roll your own

Add CORS headers manually to each response in your Lambda function code.

# AWS Lambda function URLs – **input** Payload

## PATHS:

`https://<url-id>.lambda-url.<region>.on.aws/customer`

## HTTP methods:

PUT, GET, POST, HEAD, OPTIONS, PATCH, DELETE

## Query string parameters:

`https://<url-id>.lambda-url.<region>.on.aws/order?id=1&name= ben`





# AWS Lambda function URLs – **input** Payload

```
Input payload
{
  "version": "2.0",
  "routeKey": "$default",
  "rawPath": "/my/path",
  "rawQueryString": "parameter1=value1&meter1=value2&meter2=value",
  "cookies": [
    "cookie1",
    "cookie2"
  ],
  "headers": {
    "header1": "value1",
    "header2": "value1,value2"
  },
  "queryStringParameters": {
    "parameter1": "value1,value2",
    "parameter2": "value"
  },
  "requestContext": {
    "accountId": "123456789012",
    "apiId": "",
    "authentication": null,
    "authorizer": {
      "iam": {
        "accessKey": "AKIA...",
        "accountId": "111122223333",
        "callerId": "AIDA...",
        "cognitoIdentity": null,
        "principalOrgId": null,
```



# AWS Lambda function URLs – **input** Payload

```
Input payload
{
  "version": "2.0",
  "routeKey": "$default",
  "rawPath": "/mountain/japan/hokkaido",
  "rawQueryString": "parameter1=value1&meter1=value2&meter2=value"
```

```
app.js
exports.handler = async (event) => {

  console.log(event.rawPath) // mountain/japan/hokkaido
}
```

```
},
"requestContext": {
  "accountId": "123456789012",
  "apiId": "",
```

<https://<url-id>.lambda-url.<region>.on.aws/mountain/japan/hokkaido>



```
"accessKey": "AKIA...",
"accountId": "111122223333",
"callerId": "AIDA...",
"cognitoIdentity": null,
"principalOrgId": null
```

# AWS Lambda function URLs – request Payload

```
Input payload
{
  "version": "2.0",
  "routeKey": "$default",
  "rawPath": "/mountain/japan/hokkaido",
  "rawQueryString": "parameter1=value1&meter1=value2&meter2=value",
  .....
  "http": {
    "method": "GET",
```

```
app.js
exports.handler = async (event) => {

  console.log(event.rawPath) // mountain/japan/Hokkaido
  console.log(event.http.method) // GET

}
```

```
pathParameters : null,
"isBase64Encoded": false,
```

<https://<url-id>.lambda-url.<region>.on.aws/mountain/japan/hokkaido>



```
"cognitoIdentity": null,
"principalOrgId": null,
"userArn": "arn:aws:iam::111122223333:user/example-user",
"identity": "IAM",
```

# AWS Lambda function URLs – request Payload

```
Input payload
{
  "version": "2.0",
  "routeKey": "$default",
  "rawPath": "/mountain/japan/hokkaido",
  "rawQueryString": "Mountain=mtyotei",
  "queryStringParameters": {
    "mountain": "mtyotei"
  },
}
```

```
app.js
exports.handler = async (event) => {
  console.log(event.queryStringParameters.mountain) // mtyotei
}
```

```
"userArn": "arn:aws:iam::111122223333:user/example-user",
"userId": "AIDA..."
```

<https://<url-id>.lambda-url.<region>.on.aws/?mountain=mtyotei>



```
"domainPrefix": "",
"http": {
  "method": "POST",
  "path": "/my/path",
}
```

# AWS Lambda function URLs

## Response Payload

AWS Lambda infers the response format.

If your function returns valid JSON and no status code, the following is inferred:

- statusCode is **200**
- content-type is **application/json**
- The function's response is **body**
- isBase64Encoded is **false**

# AWS Lambda function URLs **response** payload

Examples of inferred responses, string-only response:

Lambda function output

```
app.js
exports.handler = async (event) => {
  return "Hello world"
}
```

Interpreted output

```
response.js
{
  "statusCode": 200,
  "body": "Hello world",
  "headers": {
    "content-type": "application/json"
  }
  "isBase64Encoded": false,
}
```

# AWS Lambda function URLs **response** payload

Examples of inferred responses, custom response:

Lambda function output

```
app.js
exports.handler = async (event) => {
  return {
    "statusCode": 201,
    "body": {
      "message": "Hello, world!"
    },
    "headers": {
      "custom-header": "custom-val"
    }
  }
}
```

Interpreted output

```
response.js
{
  "statusCode": 201,
  "body": {
    "message": "Hello world!"
  },
  "headers": {
    "custom-header": "custom-val"
  }
}
```

# AWS Lambda function URLs – Monitoring

Function URLs sends aggregate metrics to Amazon CloudWatch:

Metric name	Description	Statistics
UrlRequestCount	The number of requests processed by the URL	Sum
Url4xxError	The number of 4XX responses returned by the URL	Sum
Url5xxError	The number of 5XX responses returned by the URL	Sum
UrlLatency	Latency between request received and response returned.	Avg, Max



# AWS Lambda function URLs – **Limits**

Shares the same limits as AWS Lambda functions

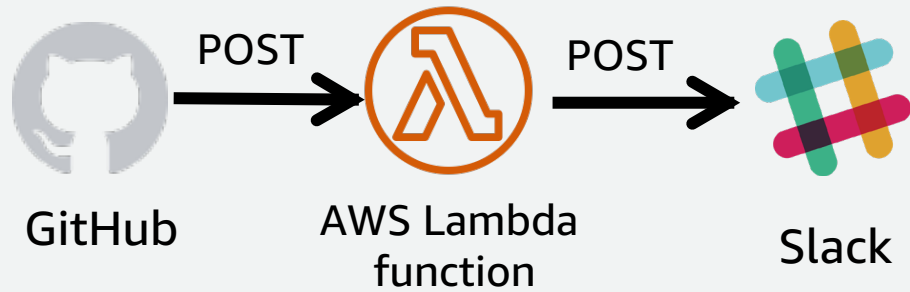
- 6 MB Response and request payload size
- Default 1,000 up to tens of thousands TPS
- 15-minute timeout

## AWS Lambda function URLs – Limits

No additional cost,  
No additional timeout limits,  
No additional payload limits

# AWS Lambda Function URLs – Use cases

## WEBHOOKS



**incoming-webhook** APP 12:03 PM

There is a new Github star for *example-step-functions-integration-api-gateway* !

*example-step-functions-integration-api-gateway*\* now has **10** stars!

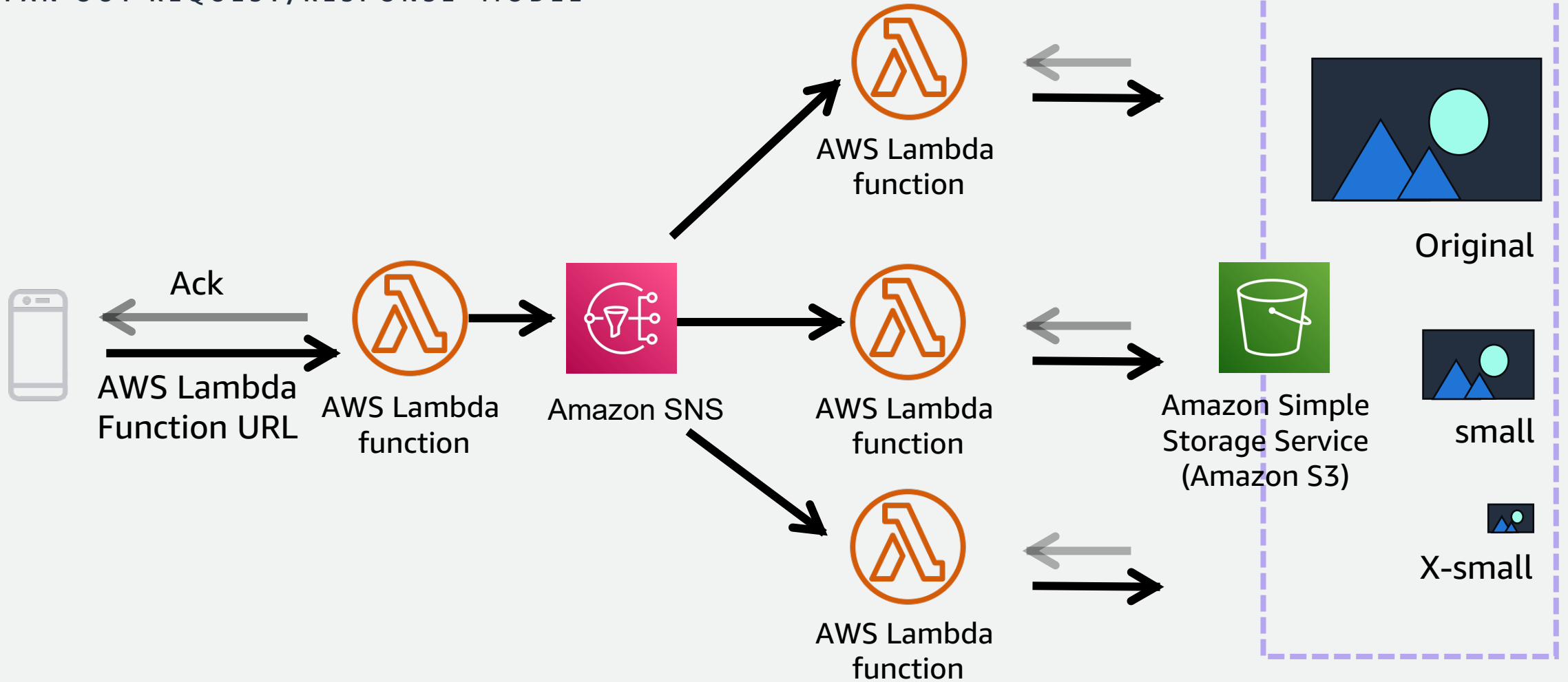
Your new  was made by [bls20AWS](#).

 Serverless App



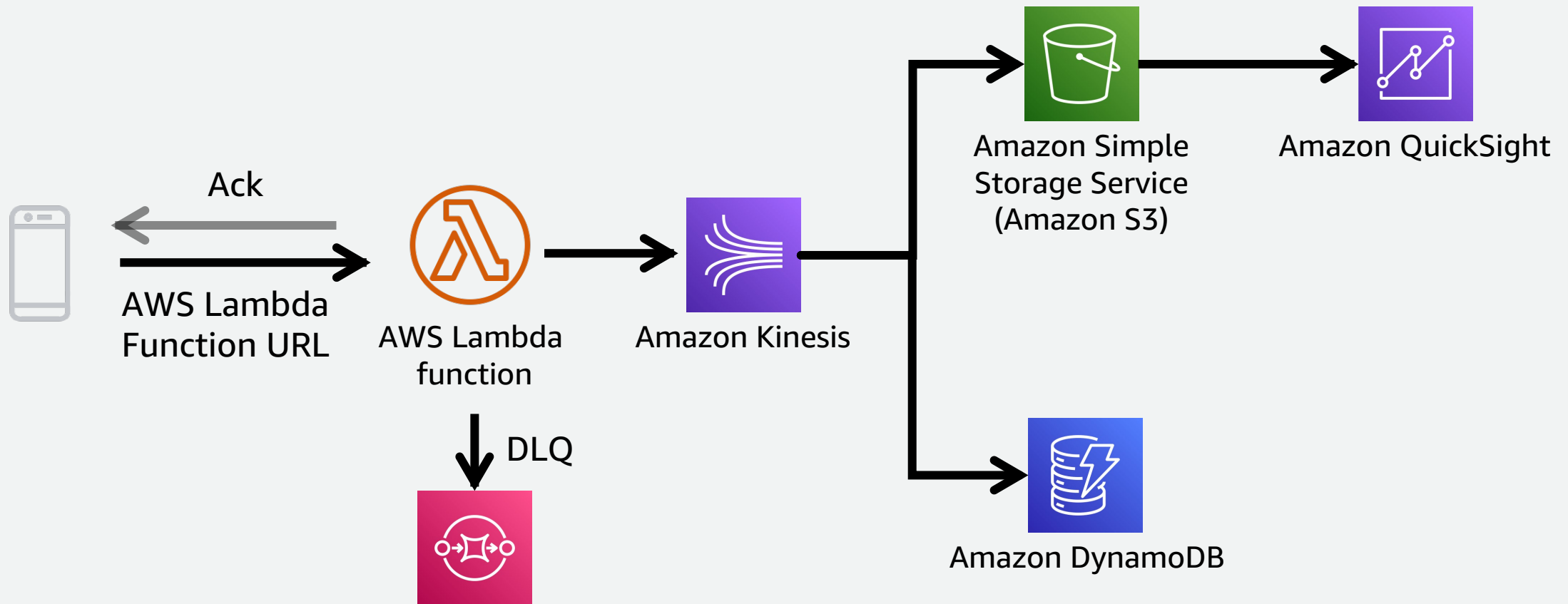
# AWS Lambda Function URLs – Use cases

FAN OUT REQUEST/RESPONSE MODEL



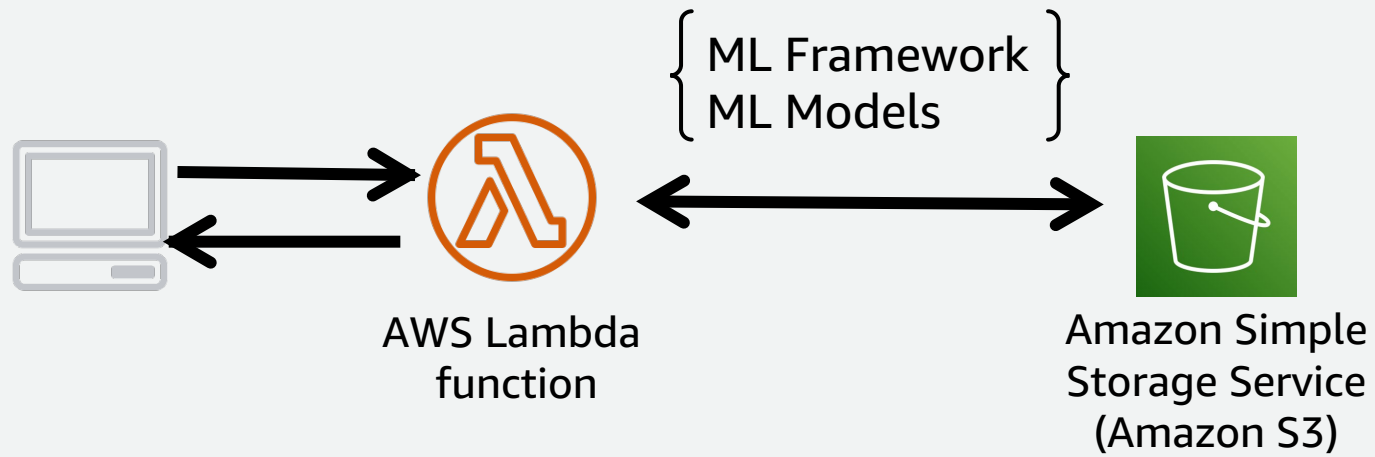
# AWS Lambda Function URLs – Use cases

## HIGH VOLUME DATA INGESTION



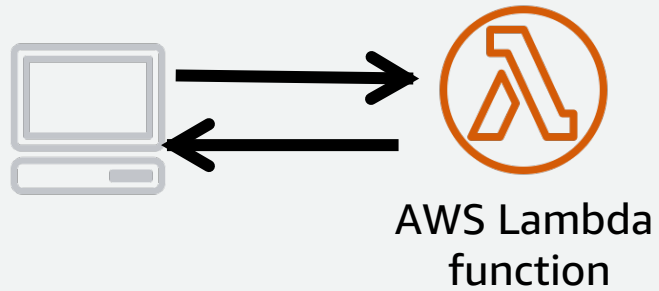
# AWS Lambda Function URLs – Use cases

SERVERLESS MACHINE LEARNING INFERENCE



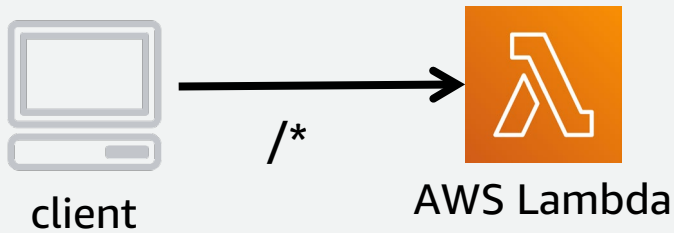
# AWS Lambda Function URLs – Use cases

WEB APPLICATIONS AND WEB FRAMEWORKS



# Function URLs to catch all requests

## CATCH ALL ROUTING

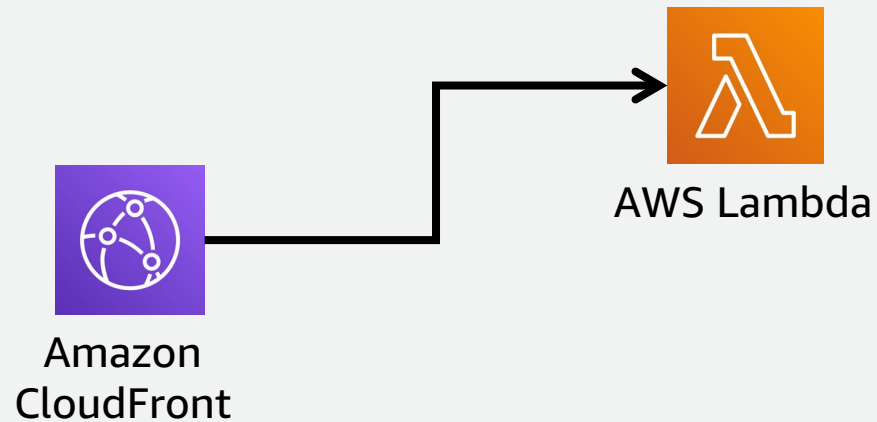


Lambda function URL acts as the HTTP router.

The AWS Lambda function *handler* is invoked when an HTTP request is received- Similar to the traditional `index.php` (PHP) or `app.js` (Node Express)



# Add custom domains with Amazon CloudFront

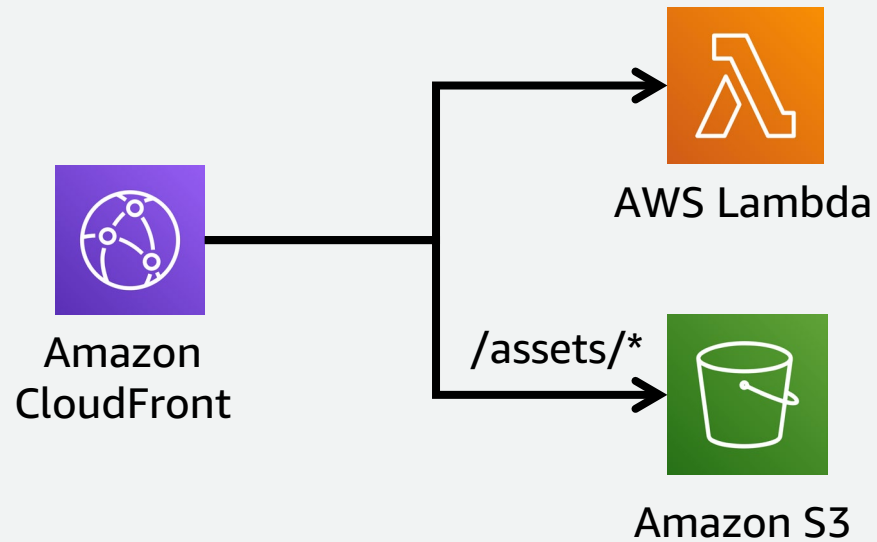


Amazon CloudFront delivers content through a worldwide network of data centers

Bring your data closer to your users

Use Amazon CloudFront CDN to customize the Lambda URL domain

# Add caching and Amazon S3 for static assets

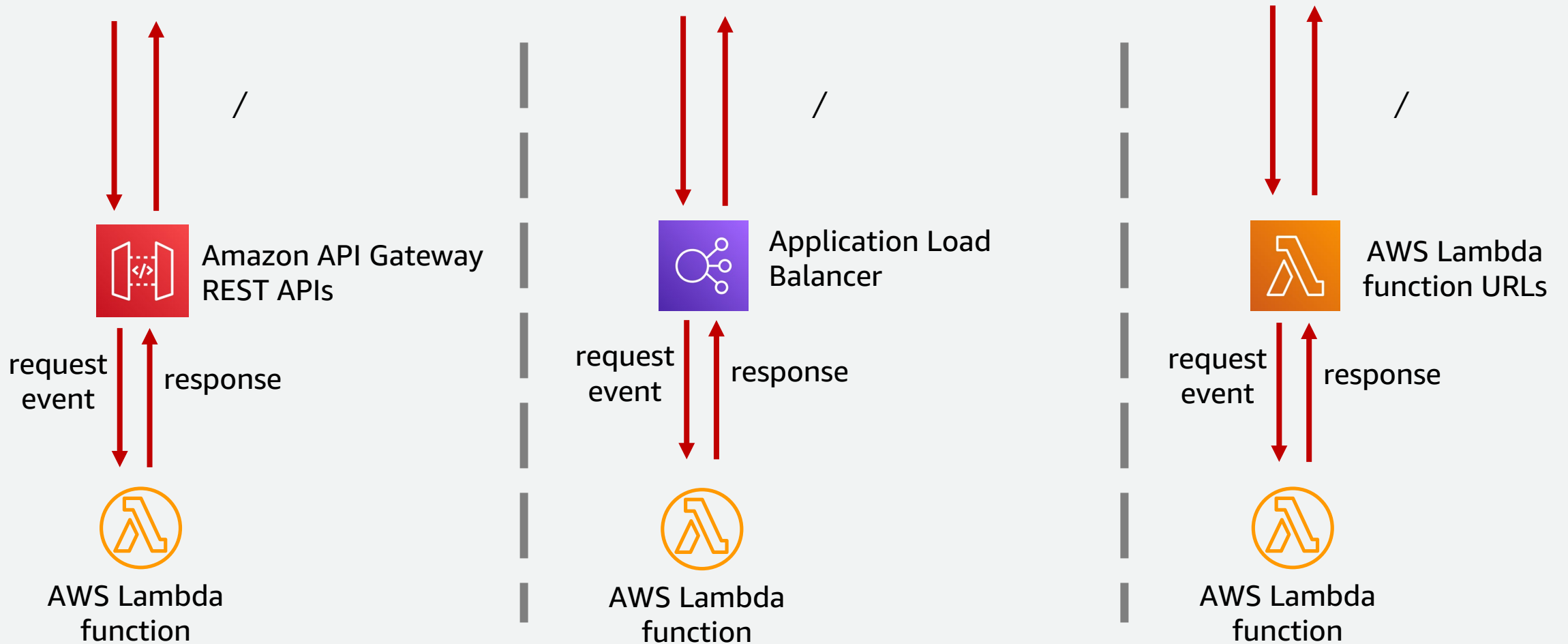


Amazon CloudFront forwards HTTP requests to "Origins" (AWS Lambda, Amazon S3, etc.) based on "Cache behaviors".

# More synchronous AWS Lambda invocations

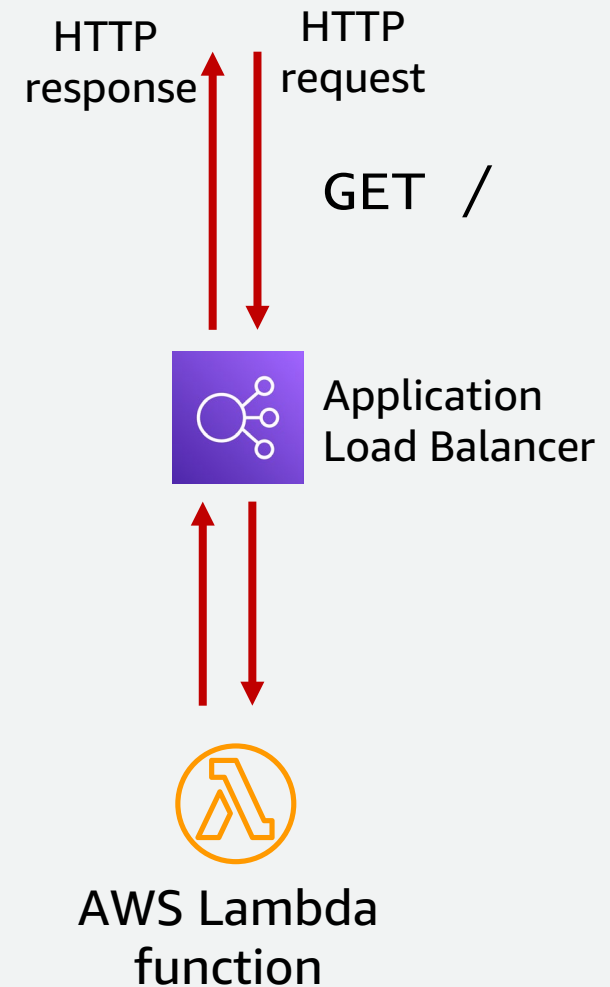


# Synchronous AWS Lambda invocation models

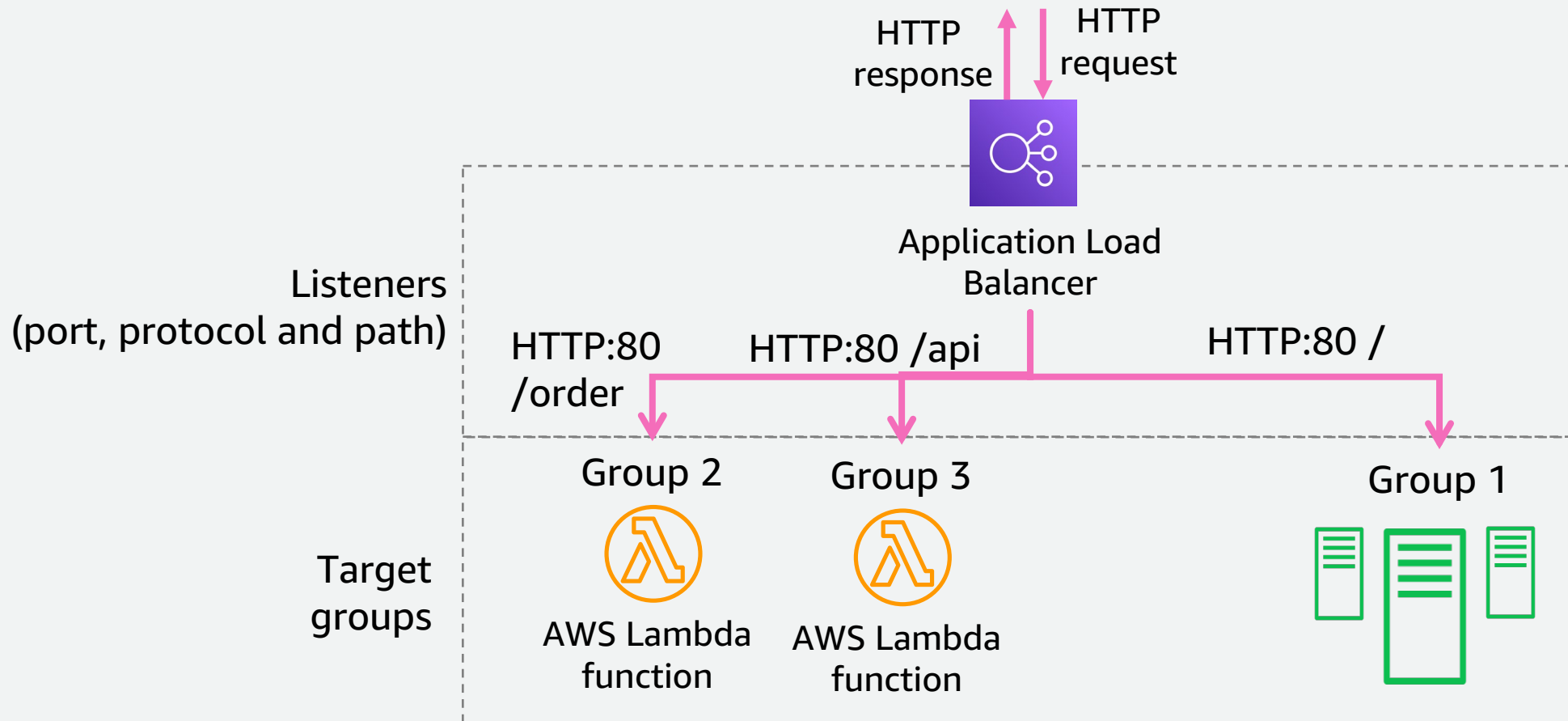


# AWS Lambda functions and Application Load Balancers (ALBs)

- ALB routes HTTP/S requests to target groups
- Groups are a collection of AWS resources (Amazon EC2 instances) or an AWS Lambda function
- Listeners define which groups are mapped to each request
- Health checks

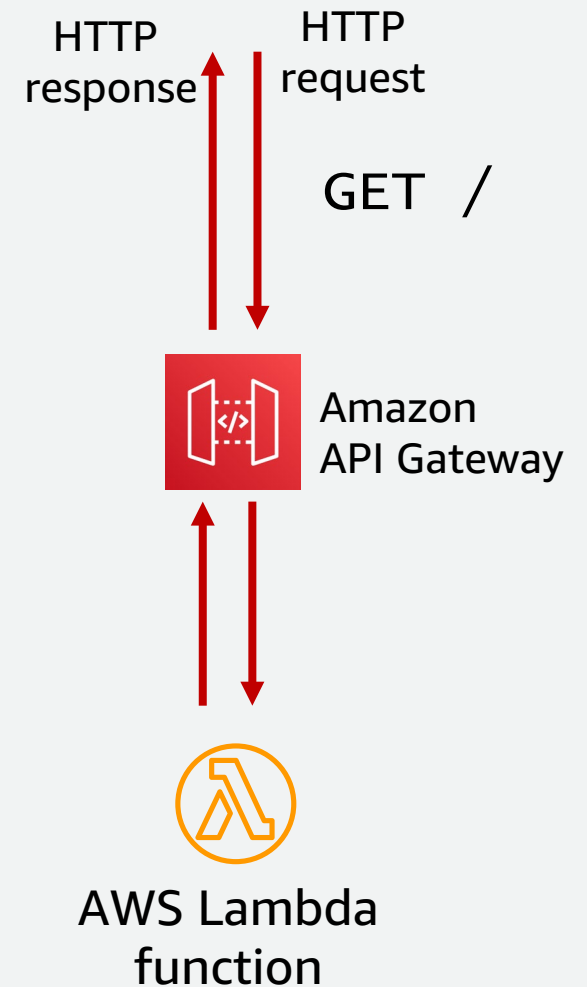


# AWS Lambda functions and Application Load Balancers (ALBs)

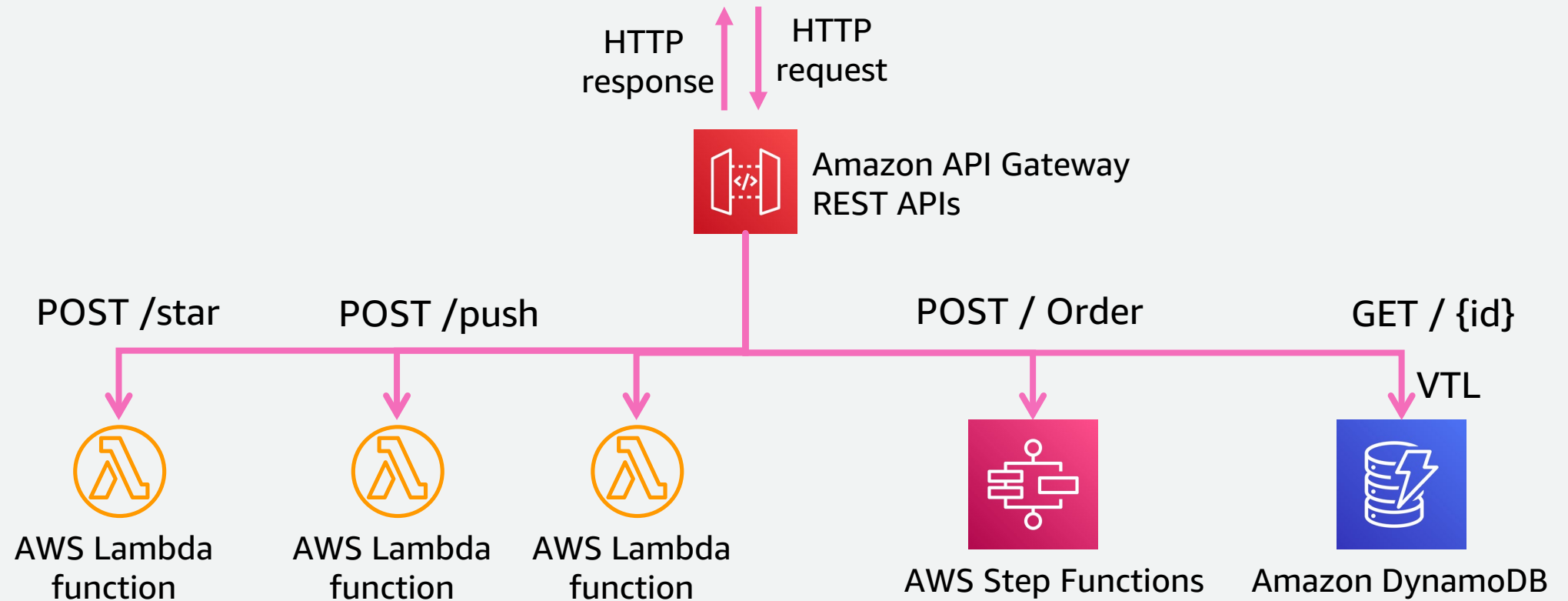


# AWS Lambda functions and Amazon API Gateway

- Create and manage REST and Websocket APIs
- Supports template mapping with VTL (Velocity Template Language)
- Supports parameter and payload validation
- Multiple Authorization options
- Natively integrate with multiple services



# AWS Lambda functions and Amazon API Gateway





# AWS Lambda Synchronous invocation comparison

	Amazon API Gateway REST APIs	Application Load Balancer	AWS Lambda Function URLs
HTTP	No	Yes	No
HTTPS	Yes	Yes	Yes
AuthN/Z	Yes	No	Yes (AWS IAM)
Payload limit	6MB (with AWS Lambda)	6MB (with AWS Lambda)	6MB
Max Timeout	29 seconds	Up to 15 minutes	Up to 15 minutes
SDK generation	Yes	No	No
Cost	from \$3.50/million for first 333 million	Hourly + capacity units	Free
AWS WAF	Yes	Yes	No

# AWS Lambda Synchronous invocation comparison

Amazon API Gateway REST APIs	Application Load Balancer	AWS Lambda Function URLs
<p>Ideal for:</p> <ul style="list-style-type: none"><li>• Service integrations</li><li>• Reducing code</li><li>• REST APIS</li><li>• Added security</li></ul>	<p>Ideal for:</p> <ul style="list-style-type: none"><li>• Hybrid applications</li><li>• Migrating to serverless</li></ul>	<p>Ideal for:</p> <ul style="list-style-type: none"><li>• Getting started</li><li>• Simple invocations</li><li>• Web applications</li><li>• Longer workloads</li></ul>

# Recap

## AWS Lambda Function URLs

A new simpler way to synchronously invoke Lambda functions with an HTTP request.

Use **asynchronous** if you don't need a response

**Initialize resources outside** of the function handler

**Minimize** deployment package size to its runtime necessities



# Thank you!

Benjamin Smith  
@benjamin\_L\_S