

Extending Your Toolkit with AWS and SiteFarm

Carson Black
Lead Web Developer
University of California, Davis
IET Professional Services
Web Development

Topics

- AWS Lambda Functions
- AWS State Machine / Step Functions
- AWS CloudWatch Event Rules
- AWS CloudWatch Logs
- AWS S3 Events
- AWS API Gateway
- Serverless Framework
- SiteFarm (Drupal 8) JSON API
- SiteFarm Subrequests
- Will touch on some other random AWS stuff along the way...

What & Why?



AWS Lambda Functions

AWS Lambda is a compute service that lets you run code without provisioning or managing servers.

- A *single* function that runs contained to itself.
- You only pay for what you use.
- You're only responsible for your code.
- Runs in response to events: API Gateway endpoint, Changes to S3 bucket or Dynamo DB table, SNS Message or CloudWatch Event.
- Node.js (Python, Ruby, Java, Go, .Net also available)
- Can be set up via Console.



cpe-course-refresh-production-IndividualCourseSMTrigger

Throttle

Qualifiers

Actions

Select a test event

Test

Save

Configuration

Monitoring

▼ Designer

[Go back to application cpe-course-refresh-production](#)



cpe-course-refresh-production-IndividualCourseSM Trigger



Layers (0)

Related functions:

Select a function



S3



+ Add trigger



AWS Step Functions



Amazon CloudWatch Logs




Amazon EventBridge




Amazon S3

Resources that the function's role has access to appear here


 S3 ✕

+ Add trigger

 AWS Step Functions

 Amazon CloudWatch Logs

 Amazon EventBridge

 Amazon S3

Resources that the function's role has access to appear here

S3

[cpe-individual-course](#)

arn:aws:s3:::cpe-individual-course

Enabled

Delete

▼ Details

Event type: **ObjectCreated, ObjectRemoved**

Notification name: **exS3-v2--b27b7718cf0246c0dcdbeb644fa9c1db**

Prefix: **processing/**

Suffix: **.json**

AWS State Machine / Step Functions

AWS State Machine / Step Functions help you orchestrate several Lambda Functions into a process.

- Step Functions: Orchestrate which Lambda Functions happen in what order.
- State Machine: Helps to maintain state data in the event passed from one Lambda to the next.
- Has various types of Steps/"States":
 - Task
 - Choice
 - Wait
 - Succeed
 - Fail
 - Parallel
 - Pass

Edit IndividualCourseSM

[Start execution](#)[Save](#)

Changes will overwrite previous values. Running executions will continue to use the definition they were started with.

Definition

Generate code snippet

[Learn more](#)

```
4  "States": {
5  "CreateOrDelete": {
6    "Type": "Choice",
7    "Choices": [
8      {
9        "Variable": "$.deleted",
10       "BooleanEquals": true,
11       "Next": "GetDrupalCourseToDelete"
12     },
13     {
14       "Variable": "$.deleted",
15       "BooleanEquals": false,
16       "Next": "CheckCourseList"
17     }
18   ],
19 },
20 "CheckCourseList": {
21   "Type": "Task",
22   "Resource": "arn:aws:lambda:us-east-1:456835951551:function:cpe-course-refresh-production-CheckCourseList",
23   "ResultPath": "$.exists",
24   "Next": "CourseExists"
25 },
26 "CourseExists": {
```



EXECUTION STATUS

❌ Failed

Execution ARN

arn:aws:states:us-east-1:456835951551:execution:IndividualSubjectArea:b6a5feaa-cb1d-4cfa-beba-2caa14ce221f

▶ Input

Started

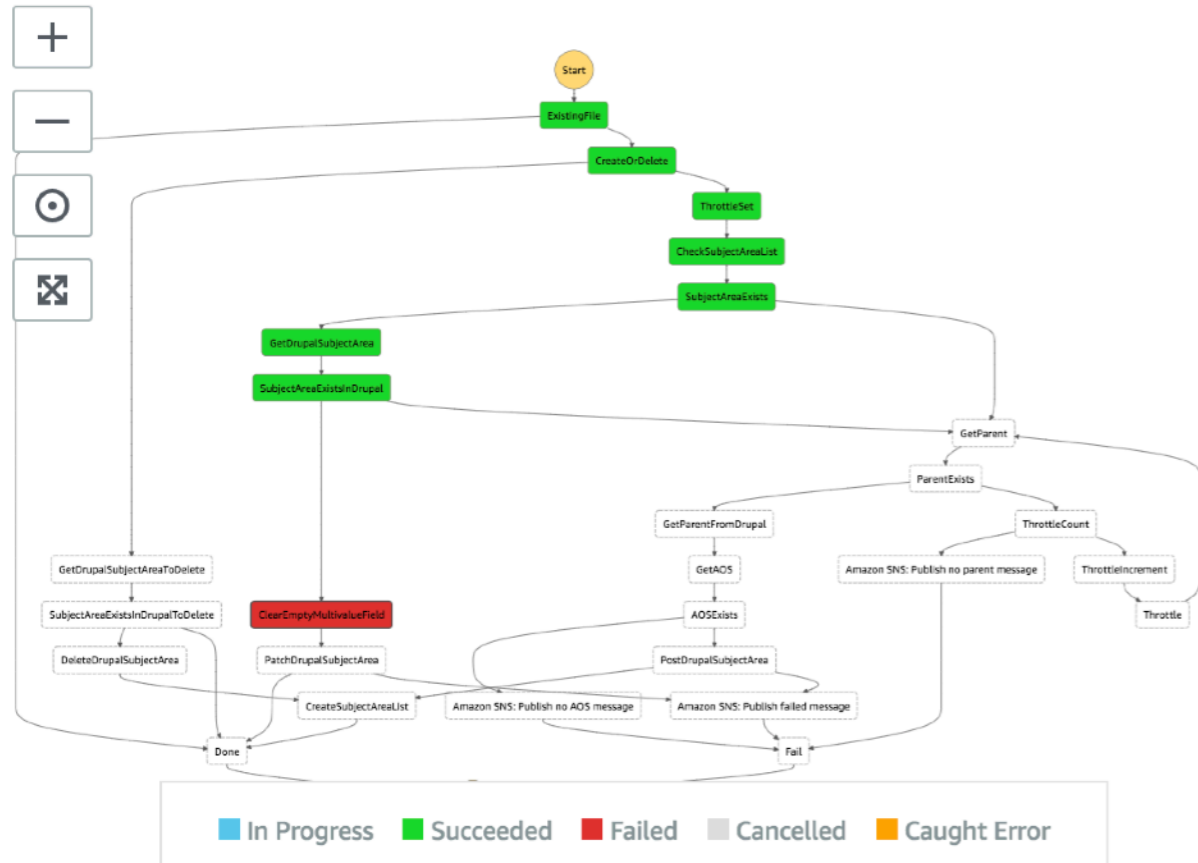
Sep 7, 2019 12:00:34.040 AM

End Time

Sep 7, 2019 12:00:35.875 AM

▶ Output

Visual workflow



Code

Step details

Name

Type

ClearEmptyMultivalueField

Task

Status

❌ Failed

Resource

arn:aws:lambda:us-east-1:456835951551:function:cpe-program-streams-production-ClearEmptyMultivalueField | CloudWatch logs

▶ Input

▶ Output

▼ Exception

Error

CloudWatch Logs

CloudWatch > Log Groups > Streams for /aws/lambda/cpe-certificate...

Search Log Group

Create Log Stream

Delete Log Stream

Filter: Log Stream Name Prefix x

<input type="checkbox"/> Log Streams	Last Event Time
<input type="checkbox"/> 2019/09/07/[\$LATEST]6edfcb4058af4e48a8e94b19fa1b4a84	2019-09-07 07:00 UTC-7
<input type="checkbox"/> 2019/08/31/[\$LATEST]9dfc89fb8a4a46c3b2bcb4b1b65c7393	2019-08-31 07:00 UTC-7
<input type="checkbox"/> 2019/08/29/[\$LATEST]a298e1522c434483a02b8a4e6097a96f	2019-08-29 07:01 UTC-7
<input type="checkbox"/> 2019/08/24/[\$LATEST]99cf928a290e4a2180f2a648dac499aa	2019-08-24 07:00 UTC-7
<input type="checkbox"/> 2019/08/24/[\$LATEST]e8eea197f9bc4e6ebf8f67705a9443eb	2019-08-24 07:00 UTC-7
<input type="checkbox"/> 2019/08/20/[\$LATEST]8137b93a0a9941539dd1d886467781ec	2019-08-20 07:01 UTC-7
<input type="checkbox"/> 2019/08/09/[\$LATEST]c099c47285384ab98102cf811df648db	2019-08-09 07:00 UTC-7
<input type="checkbox"/> 2019/08/06/[\$LATEST]f59aeb0ed98c493ebd700ec95b80b427	2019-08-06 07:01 UTC-7
<input type="checkbox"/> 2019/08/06/[\$LATEST]d3a0e17ab61847a09e2290f18daaaf2f	2019-08-06 07:01 UTC-7
<input type="checkbox"/> 2019/08/06/[\$LATEST]3f38ebb7ca5948ddb5a90f92a9de337c	2019-08-06 07:01 UTC-7

CloudWatch Logs

Filter events		all 2019-09-06 (14:00:4
Time (UTC +00:00)	Message	
2019-09-07 14:00:37	START RequestId: 3f602b1a-0d8d-4a17-a471-4c234dd41cab Version: 2.1.0	
▼ 14:00:37	2019-09-07T14:00:37.827Z 3f602b1a-0d8d-4a17-a471-4c234dd41cab { Records: [{ eventVersion: '2.1', eventSource: 'aws:s3', awsRegion: 'us-east-1', eventTime: '2019-09-07T14:00:35.378Z', eventName: 'ObjectCreated:Put', userIdentity: [Object], requestParameters: [Object], responseElements: [Object], s3: [Object] }], sectionKey: 'processing/cpe.ucdavis.edu/cpe-certificates-1.json', sectionETag: 'fe45bd73ad213c056644e1dfe2ed4518', bucketName: 'cpe-certificates-processing', site: 'cpe.ucdavis.edu', count: 17 }	
▶ 14:00:38	2019-09-07T14:00:38.126Z 3f602b1a-0d8d-4a17-a471-4c234dd41cab 16: [object Object]	
▶ 14:00:38	END RequestId: 3f602b1a-0d8d-4a17-a471-4c234dd41cab	
▶ 14:00:38	REPORT RequestId: 3f602b1a-0d8d-4a17-a471-4c234dd41cab Duration: 301.99 ms Billed Duration: 400 ms Memory Size: 512 MB Max Memory Used: 80 MB Init Duration: 233.25 ms XRAY	

Triggers - How to start things

- CloudWatch Events
- API Gateway
- S3 Bucket Changes

AWS CloudWatch Events

AWS CloudWatch Events are a stream of system events that describe changes in AWS resources.

- Events: A change in an AWS resource. *Can have scheduled events.
- Rules: matches an event and routes it to a target.
- Target: A target process like AWS Lambda or AWS Step Functions State Machine

- CloudWatch
- Dashboards
- Alarms
 - ALARM 0
 - INSUFFICIENT 1
 - OK 1
- Billing
- Events
- Rules**
- Event Buses
- Logs
- Insights
- Metrics
- Settings
- Favorites
- + Add a dashboard

Rules > CoursesInternational

Actions ▾

Summary

ARN ⓘ `arn:aws:events:us-east-1:456835951551:rule/CoursesInternational`

Schedule Cron expression `40 07 * * ? *`

- Next 10 Trigger Date(s)**
1. Wed, 11 Sep 2019 07:40:00 GMT
 2. Thu, 12 Sep 2019 07:40:00 GMT
 3. Fri, 13 Sep 2019 07:40:00 GMT
 4. Sat, 14 Sep 2019 07:40:00 GMT
 5. Sun, 15 Sep 2019 07:40:00 GMT
 6. Mon, 16 Sep 2019 07:40:00 GMT
 7. Tue, 17 Sep 2019 07:40:00 GMT
 8. Wed, 18 Sep 2019 07:40:00 GMT
 9. Thu, 19 Sep 2019 07:40:00 GMT
 10. Fri, 20 Sep 2019 07:40:00 GMT

Status Enabled

Description Starts the nightly INTL Course process.

Monitoring [Show metrics for the rule](#)

Targets

Filter:

« < Viewing 1 to 1 of 1 Targets > »

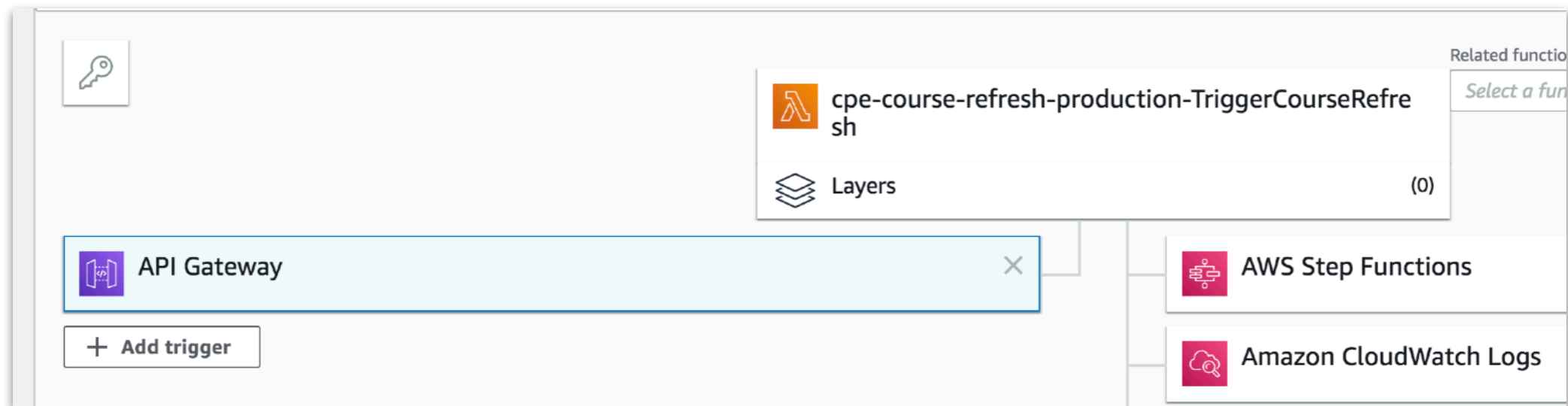
Type	Name	Input	Role	Additional parameters
Step Functions state machine	GetDestinyCourses	Constant: {"site": "cie.ucdavis.edu", "programArea": "PA0002", "throttle": 5}	AWS_Events_Invoke_Step_Function_Perm	

AWS API Gateway

AWS API Gateway is a service to help create, publish, maintain, and secure APIs.

- Basically, it gives your functionality in AWS a REST endpoint.

https://0vpnmpn72.execute-api.us-east-1.amazonaws.com/production/course/refresh/{site}/{program_area}



AWS S3 Event Notification

- Event that gets sent any time there is a change to an S3 bucket.
- For example, if a file is added or deleted.
- Configurable, for type of change (s3:ObjectCreated:*, s3:ObjectRemoved:*).
- Destinations:
 - Amazon Simple Notification Service (SNS) topic.
 - Amazon Simple Queue Service (SQS) queue.
 - *AWS Lambda Function

Cloudwatch Event Rule set using cron expression to run everyday at 8:30GMT



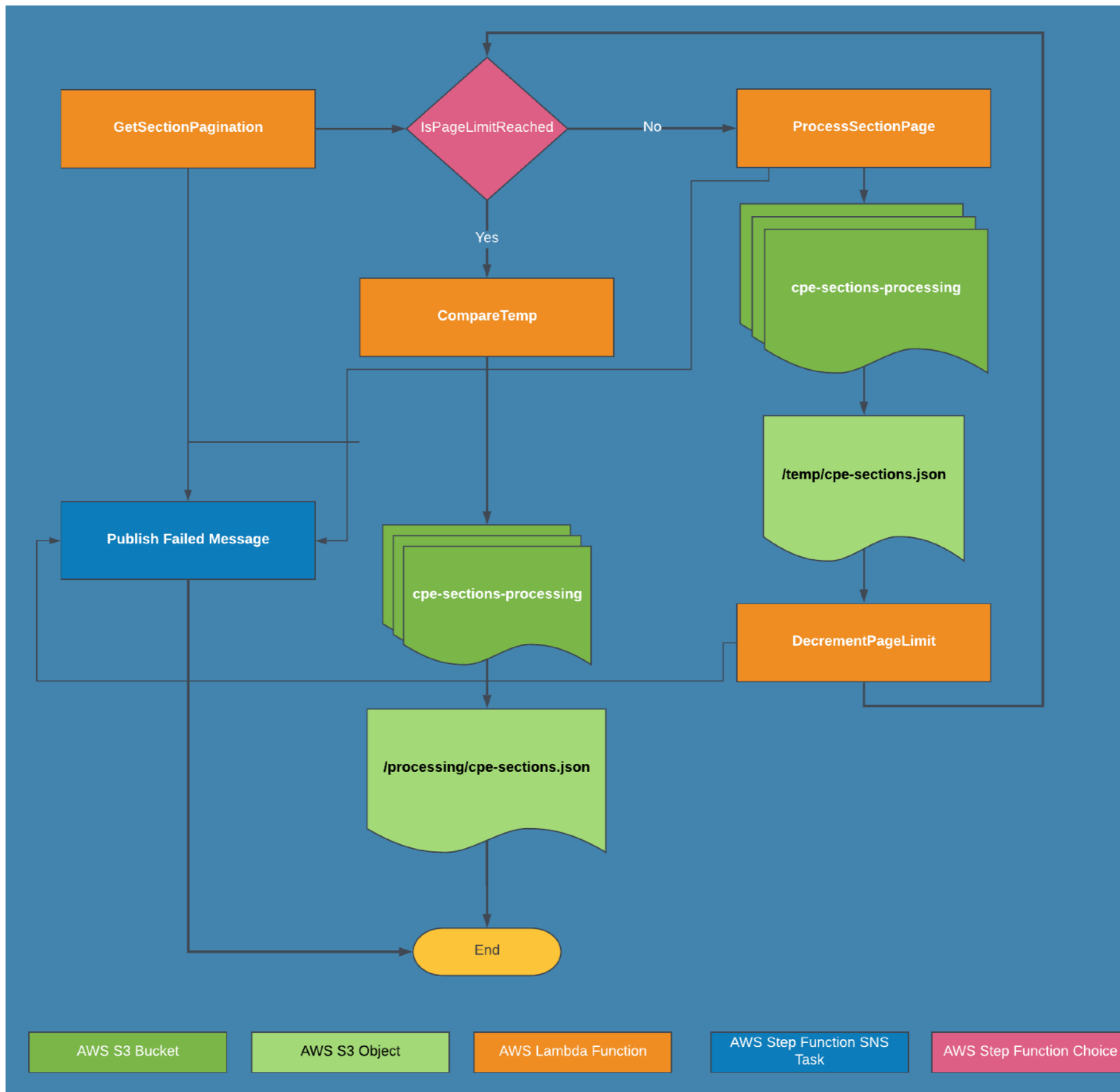
AWS S3 Bucket

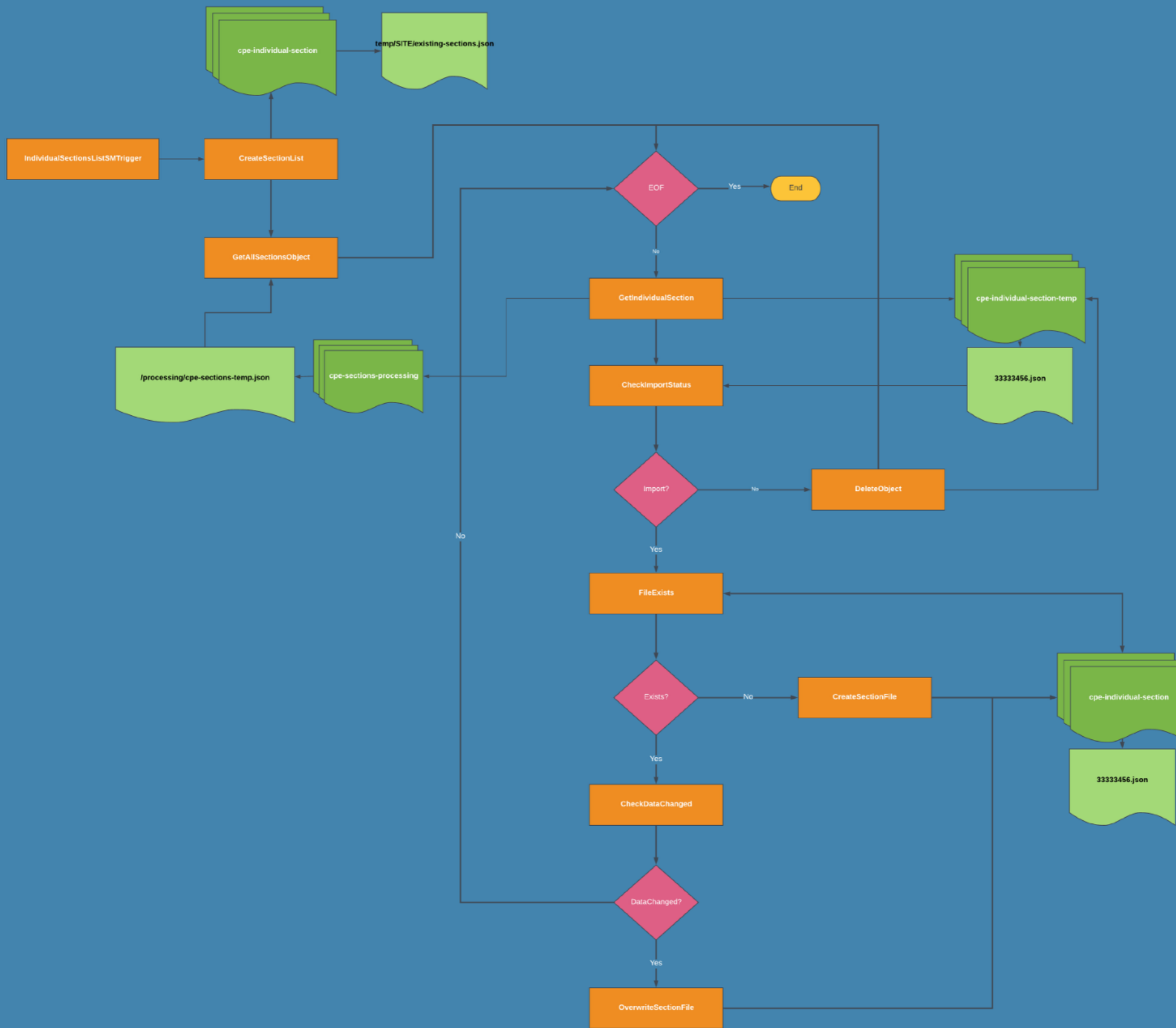
AWS S3 Object

AWS Step Function Process

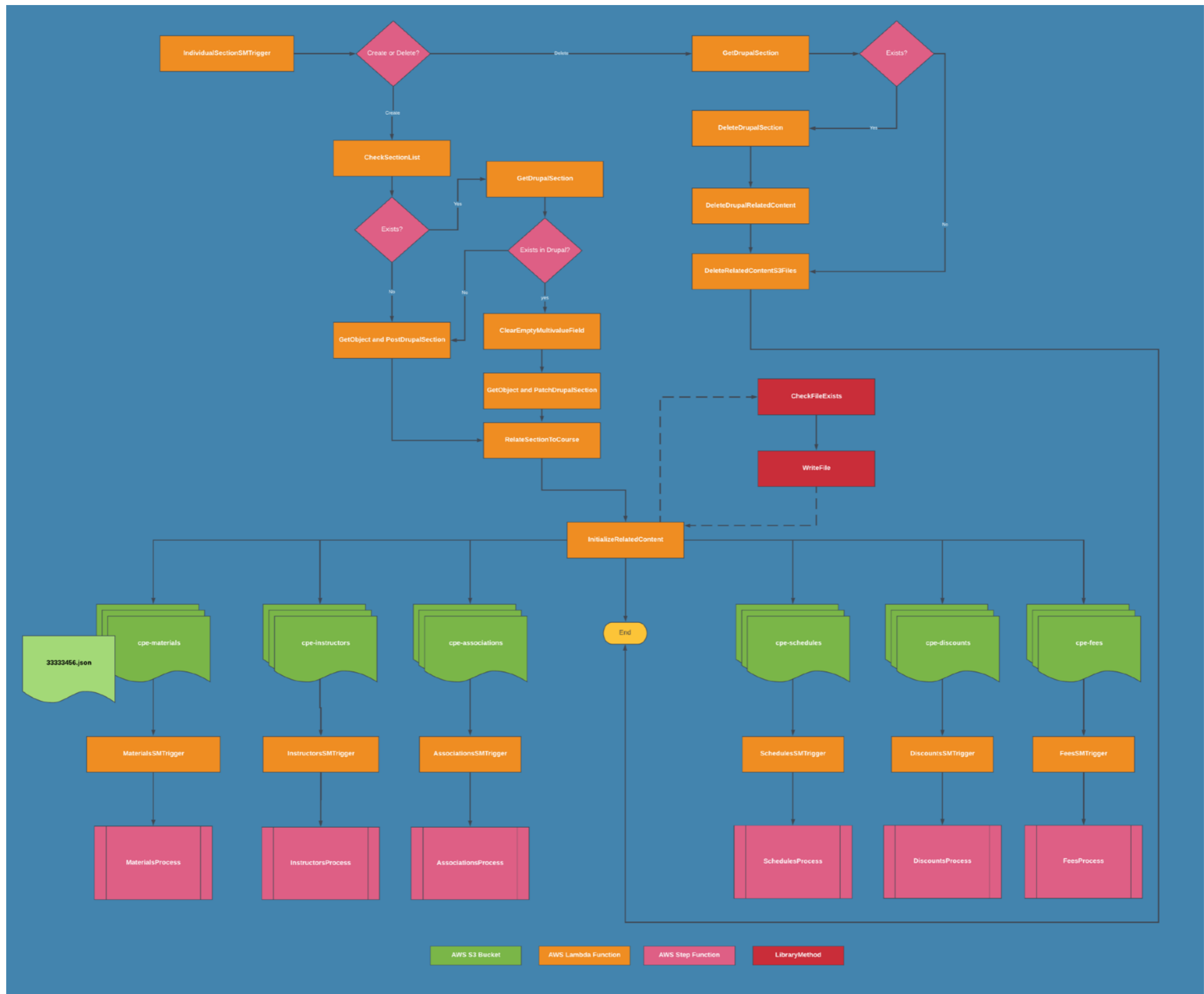
AWS Lambda Function

AWS Cloudwatch Event





AWS S3 Bucket
AWS S3 Object
AWS Lambda Function
AWS Step Function Choice



SiteFarm (Drupal)

- SiteBuilders only! And even then...
- JSON API - SiteFarm Decoupled module
- Subrequests Module

JSON API

- Exposes all Drupal Entities (Content Types, Blocks, etc.) via A REST API server that implements the JSON:API specification.
- jsonapi.org
- No configuration needed except setting up a user and role.
- Specific Role is important as it's only Basic authentication right now.

Subrequests

- Tell the system to execute several requests in a single Drupal bootstrap, then return all the things!

```
[
  {
    "requestId": "req-1",
    "uri": "/jsonapi/taxonomy_term/tags",
    "action": "create",
    "body": "{\"data\":{\"type\":\"taxonomy_term--tags\",\"attributes\":...}}",
    "headers": {
      "Accept": "application/vnd.api+json",
      "Content-Type": "application/vnd.api+json",
      "Authorization": "Basic YWRtaW46YWRtaW4="
    }
  },
  {
    "requestId": "req-2",
    "waitFor": ["req-1"],
    "uri": "/jsonapi/node/article",
    "action": "create",
    "body": "{\"data\":{\"type\":\"node--article\",\"attributes\":{\"langcode\":\"...\"}}",
    "headers": {
      "Accept": "application/vnd.api+json",
      "Content-Type": "application/vnd.api+json",
      "Authorization": "Basic YWRtaW46YWRtaW4="
    }
  }
]
```


Refresh Section Data ☆

Process New Section

Object ID *

NOTE! Make sure you are entering a valid Section Object ID. The following screen will not alert you if you have provided incorrect data.

[Submit](#)

[Home](#) » [Administration](#)

[Refresh Section Availability Data](#)

Object ID	UUID	Section Number
<input type="text"/>	<input type="text"/>	<input type="text"/>

[Apply](#)

Action

Delete content ▼

[Apply to selected items](#)

	TITLE	ID	OBJECT ID	UUID	ENROLL BUTTON TEXT	INSTRUCTION METHOD	COURSE OBJECT ID	CHANGED	OPERATIONS LINKS	REFRESH
<input type="checkbox"/>	Accounting Principles Fast Track Certificate Program	198376	25202936	8ed38778-9ce9-43c1-8624-45e12acd3191	Enroll Now	Online class	25200782	Thursday, July 25, 2019 - 10:33am	Edit ▼	Refresh
<input type="checkbox"/>	Managing Project Risk and Integration	201216	30626498	a2da5d77-6143-4b8c-84f3-2814d23f7a42	Enroll Now	Online class	21757302	Thursday, August 22, 2019 - 10:56am	Edit ▼	Refresh
<input type="checkbox"/>	Managing Project Risk and Integration	201231	30626498	d6e51f40-af75-4b09-b565-47e4987d3aca	Enroll Now	Online class		Friday, August 30, 2019 - 7:18am	Edit ▼	Refresh

Configuration in Code

- Don't do all of this in the AWS Console it will take forever to wire all of this up!
- Use Serverless Framework (or AWS SAM)

Serverless Framework

- <https://serverless.com/framework>
- Software you run on your machine that provides a unified experience to develop, deploy, test, secure and monitor your Serverless applications.
- Basically, this writes AWS Cloudformation code for you.

```
# Step 1. Install serverless globally
$ npm install serverless -g

# Step 2. Create a service
$ serverless

# Step 3. deploy to cloud provider
$ serverless deploy

# Your function is deployed!
$ http://xyz.amazonaws.com/hello-world
```

Serverless.yml

- Configuration in code!
- Service
- Plugins
- Provider (AWS, Google, Azure..)
- IAM (roles, resources, actions)
- Resources (S3 Buckets, Log Retention, Cloudwatch Event Rules)
- Lambda Functions (handlers, events)
- Step Functions

```
! serverless.yml ×
services > individual-section > ! serverless.yml
 1  # Individual Section checking and pushing to site.
 2
 3  service: cpe-individual-section
 4
 5  # Use the serverless-webpack plugin bundle npm packages
 6  plugins:
 7    - serverless-webpack
 8    - serverless-step-functions
 9    - serverless-pseudo-parameters
10    - serverless-plugin-existing-s3
11
12  custom:
13    # Location of the webpack configuration file.
14    webpack:
15      webpackConfig: ../../webpack.config.js
16
17  provider:
18    name: aws
19    runtime: nodejs8.10
20    memorySize: 512
21    stage: production
22    profile: cpe
23    tags:
24      Customer: CPE
25    timeout: 30
26    environment:
```

```

# Set up resources.
resources:
  Resources:
    # Create the AreasOfStudyHS Cloudwatch Event Rule.
    HSAreasOfStudyStart:
      Type: AWS::Events::Rule
      Properties:
        Description: 'Starts Areas of Study Lambda func
        Name: AreasOfStudyHS
        ScheduleExpression: cron(45 06 * * ? *)
        State: ENABLED
        Targets:
          - Arn: arn:aws:lambda:#{AWS::Region}:#{AWS::A
            Input: '{"site": "hs.sf.ucdavis.edu", "prog
            Id: IdHSAS4506

```

```

ForceImportSectionSMTrigger:
  handler: ForceImportSectionSMTrigger.process
  events:
    - http:
        path: section/import/{site}/{objectId}
        method: get
        request:
          parameters:
            paths:
              site: true
              objectId: true
GetDestinySection:
  handler: GetDestinySection.process
DeleteSectionS3File:

```

```

# Step Functions
stepFunctions:
  stateMachines:
    ForceRefreshIndividualSection:
      name: ForceRefreshIndividualSection
      definition: ${file(ForceRefreshIndividualSection.json)}
    ForceImportIndividualSection:
      name: ForceImportIndividualSection
      definition: ${file(ForceImportIndividualSection.json)}

```

```

resources:
  Resources:
    IndividualSectionS3:
      Type: AWS::S3::Bucket
      Properties:
        BucketName: cpe-individual-section
        VersioningConfiguration:
          Status: Enabled
      Tags:
        - Key: Customer
          Value: CPE
      LifecycleConfiguration:
        Rules:
          - Id: RemoveTwoWeeksOld

```

Serverless command line

\$ serverless info

\$ serverless deploy

\$ serverless remove

\$ serverless deploy function -f nameOfFunction

Tips & Tricks

- Organize your code: “Mono-repo” with various services organized into their own directories and shared libraries to improve code reuse.
- Use a configuration framework like Serverless from the start.
- Stay away from SNS or even SQS for triggering unless necessary. S3, API Gateway and CloudWatch events are much more reliable.
- Keep your Lambda Function JavaScript handler files **focused on doing one single function** and orchestrate with State Machine Step Functions.
- Use tags. We have a Customer tag. Can be added to most services/resources in serverless.yml config.

Tips & Tricks

- Webform has a Remote Post Submission Handler. Send data to endpoint.
- Not just POST. GET, PATCH and DELETE are available via JSON API on SiteFarm.

Gotchas and Pitfalls

- Learning Curve
- Out of date tutorials, documentation
- Unreliable services like SNS
- Refactoring time - Work on something small
- **AWS overcomes limitations of one service by creating a new service.**
- End to end test are going to be difficult, but unit tests are possible for libraries, etc.
- State Machine Step Functions have a 25,000 event limit in the execution history for each instance. It will just Cancel execution without any reason given.