

What's possible with AWS Step Functions

Anton Aleksandrov

Principal Solutions Architect
AWS, Serverless

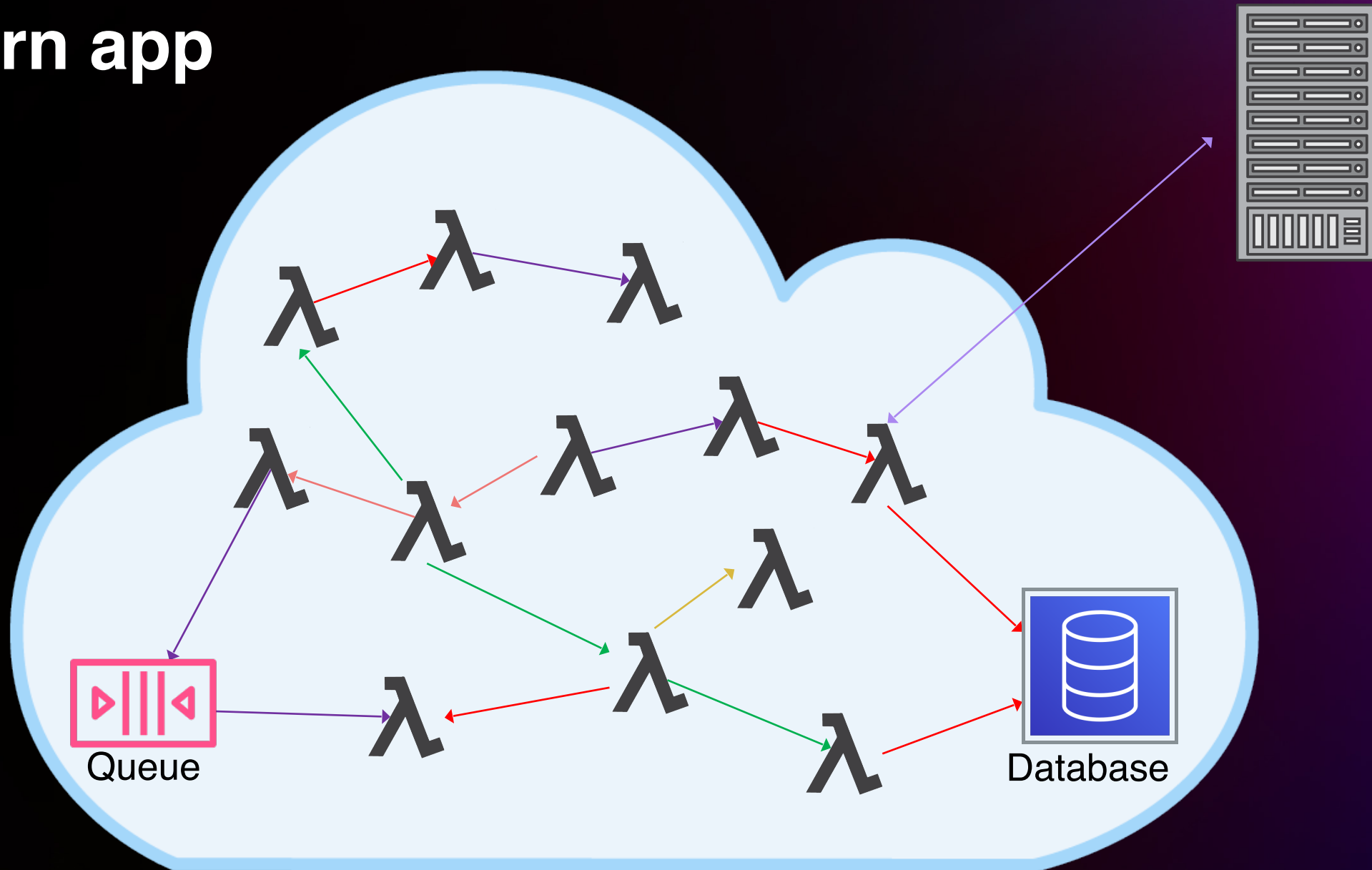
Dhiraj Mahapatro

Principal Solutions Architect
AWS, Serverless

The Problem



Modern app



Workflow orchestration

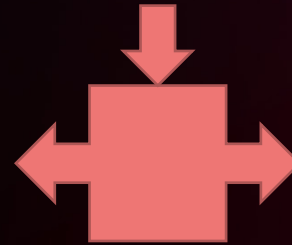
“I want to sequence tasks”



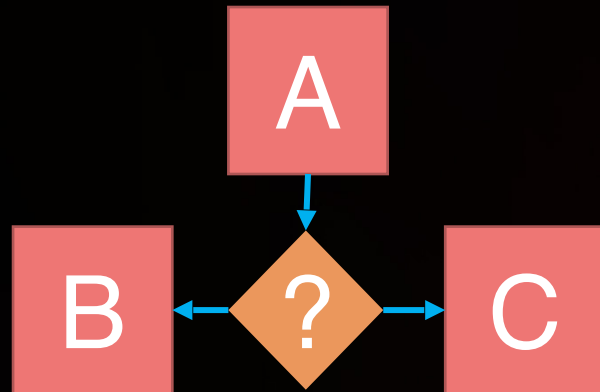
“I want to retry failed tasks”



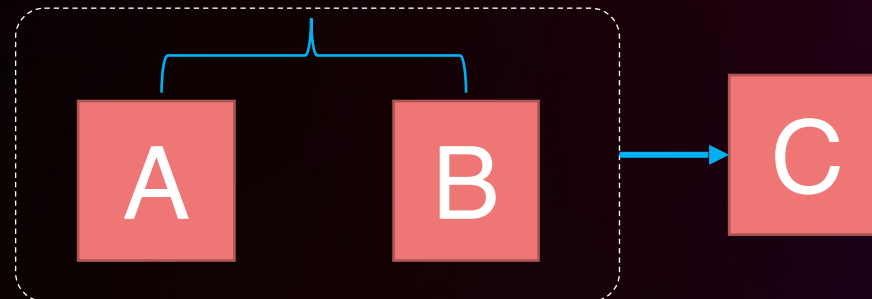
“I want try/catch/finally”



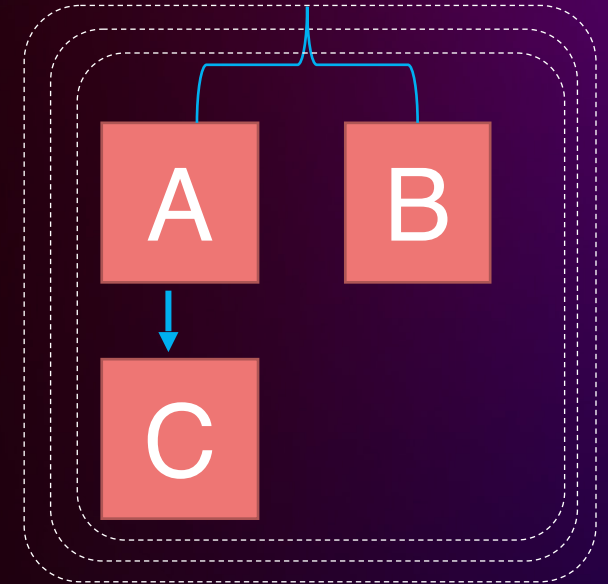
“I want to select tasks based on data”



“I want to run tasks in parallel”

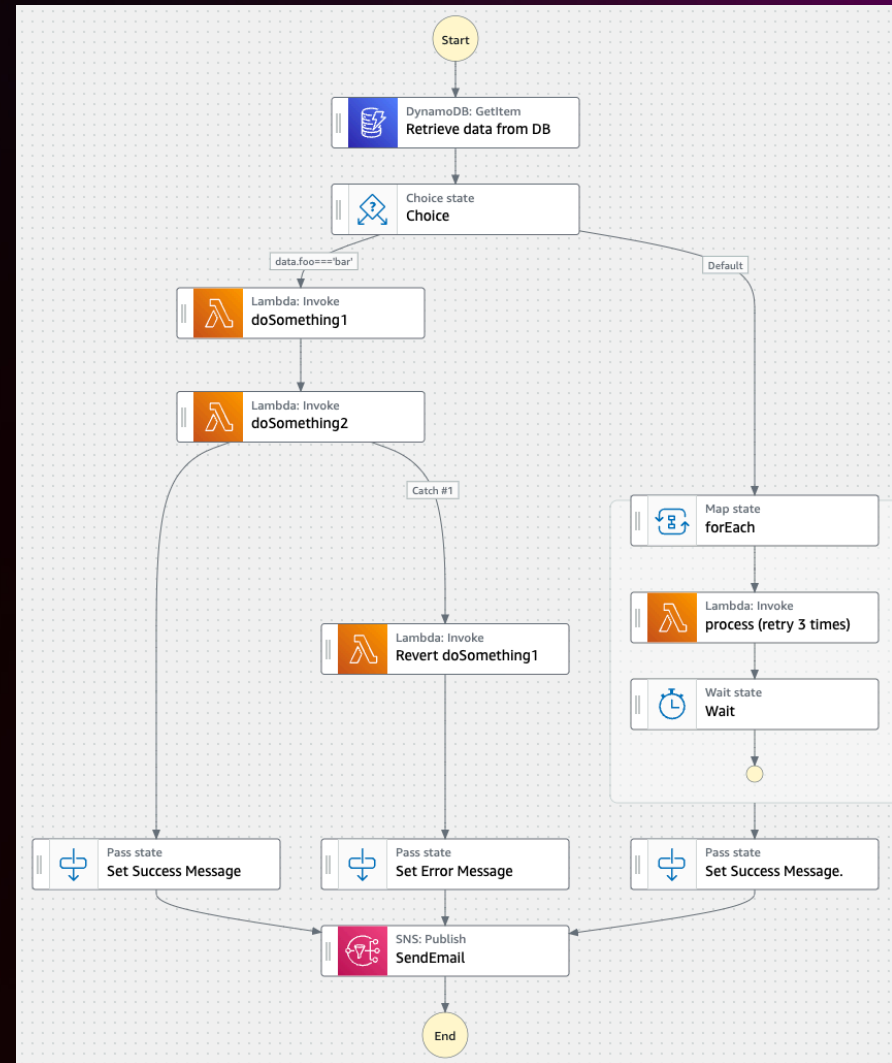


“I want concurrent and iterative tasks”



Workflow orchestration

```
1 function entryPoint(parameters){
2   const data = database.get();
3
4   if (data.foo==='bar'){
5     doSomething1(parameters);
6     try {
7       doSomething2(parameters);
8     } catch(e){
9       revertSomething1();
10      sendErrorEmail();
11      return;
12    }
13   } else if (data.foo==='baz'){
14     data.array.forEach((el)=>{
15       let retries = 0;
16       while (retries<=3){
17         const success = process(el);
18         if (!success) {
19           retries++;
20           sleep(2000);
21         } else {
22           storeToDatabase(el);
23         }
24       }
25     });
26   }
27   sendSuccessEmail();
28 }
29 }
```



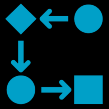
The Solution



Workflow orchestration

- Isolate tasks in **logical components**
- Centralize the **orchestration workflow**

AWS Step Functions



The **workflows** you build with Step Functions are called **state machines**, and **each step** of your workflow is called a **state**.



When you execute your state machine, each move from one state to the next is called a state transition.

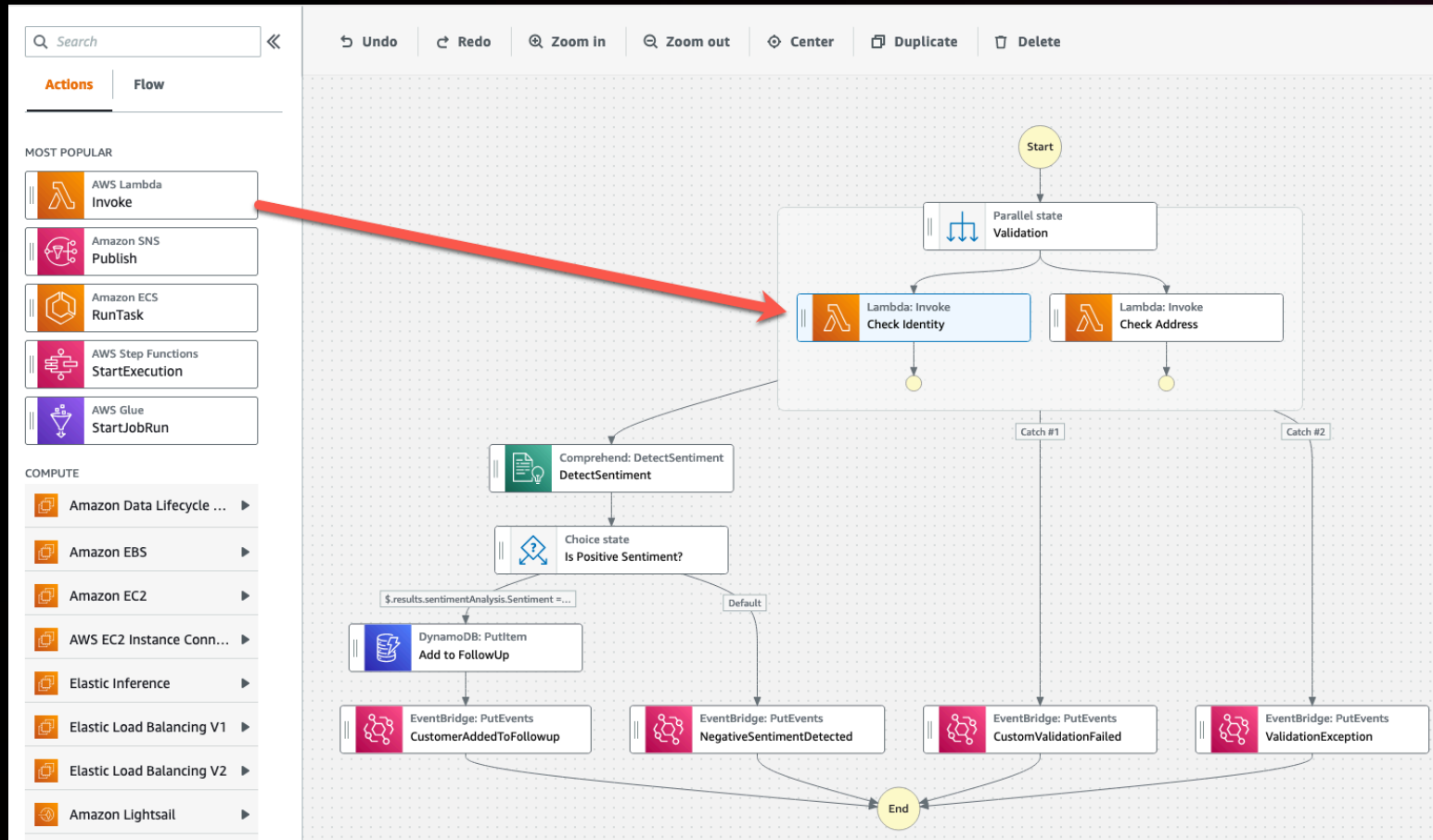


You can **reuse components**, easily edit the sequence of steps or swap out the code called by task states as your needs change.

The screenshot displays the AWS Step Functions workflow designer interface. The top navigation bar includes the AWS logo, Services, Resource Groups, and a feedback link. The main title is 'Step Functions workflow designer (Preview)'. Below the title is a search bar and a toolbar with Undo, Redo, Zoom in, Zoom out, and Center buttons. The left sidebar, titled 'Service APIs', lists various AWS services that can be used as tasks in a workflow, including AWS Lambda Invoke, Amazon SNS Publish, Amazon ECS RunTask, AWS Glue StartJobRun, AWS Step Function StartExecution, AWS Batch SubmitJob, AWS API Gateway Invoke, Amazon Athena GetQueryExecution, Amazon Athena StartQueryExecution, Amazon Athena StopQueryExecution, Amazon Athena GetQueryResults, and AWS Codebuild StartBuild. The central canvas shows a workflow diagram with a 'Start' node, two 'Lambda: Invoke' tasks (Invoke-Function-1 and Invoke-Function-2), and an 'End' node. The right sidebar, titled 'Invoke-Function-2', shows the configuration for the selected task, including tabs for Configuration, Input, Output, and Error handling. The Error handling section includes options for retrying on errors (Retrier #1 and Retrier #2), catching errors (Catch errors - optional), timeout (Timeout - optional), and heartbeat (Heartbeat - optional).

AWS STEP FUNCTIONS WORKFLOW STUDIO

Workflow Studio: Low-code visual designer



Workflow Studio is a drag-and-drop visual builder.

It reduces the time to build a first workflow for new developers.

Experienced developers can use it to build and share prototypes with stakeholders faster.

Improved workflow visualization using AWS service icons, automatic workflow layouts, and visual cues from workflow definition.

AWS Step Functions

- Resilient workflow automation
- Built-in error handling
- Powerful AWS service integrations
- Integrate with your own services
- Auditable history, visual monitoring
- Build workflows visually, save as JSON in source control system (e.g. git)

Visual workflow

+

-

⌂

🔍

Code

Step details

NameAutomated Checks ChoiceTypeChoice

Status✔ Succeeded

Resource-

▶ Input

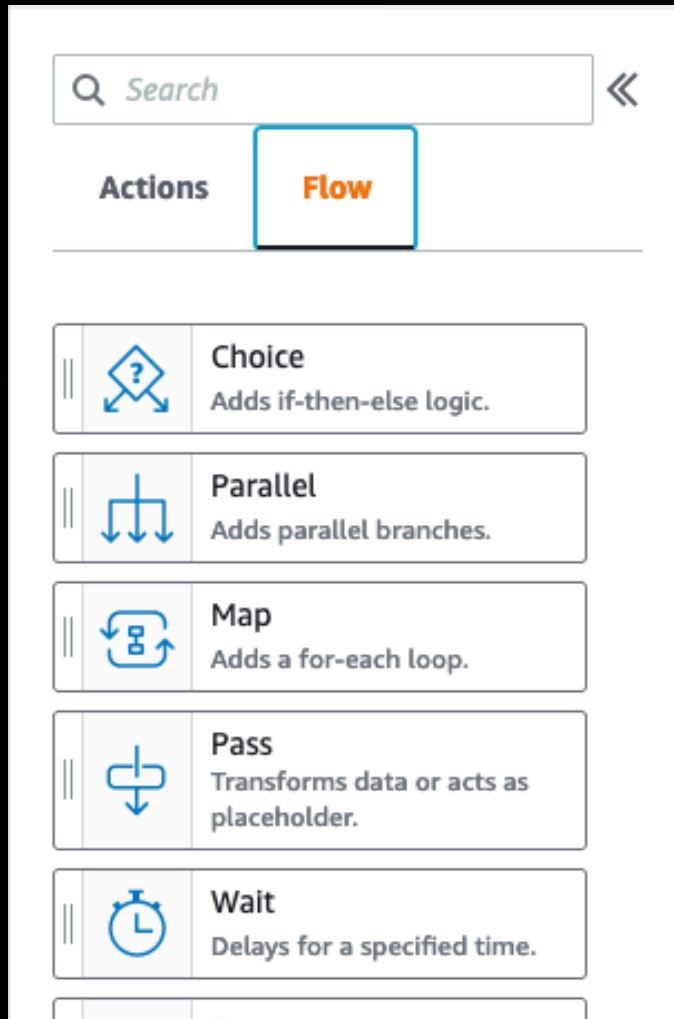
▶ Output

▶ Exception

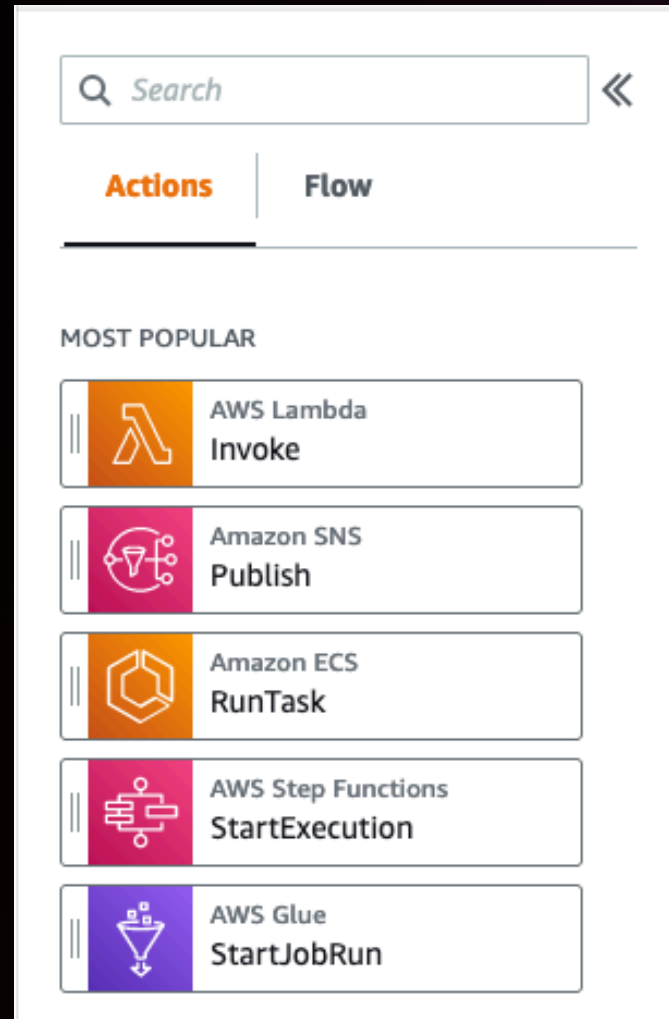
Execution event history

| ID | Type | Step | Resource | Elapsed Time (ms) | Timestamp |
|-----|-------------------------|--------------------------|--|-------------------|------------------------------|
| ▶ 1 | ExecutionStarted | | - | 0 | Sep 17, 2019 11:14:14.027 AM |
| ▶ 2 | ParallelStateEntered | Perform Automated Checks | - | 41 | Sep 17, 2019 11:14:14.068 AM |
| ▶ 3 | ParallelStateStarted | Perform Automated Checks | - | 41 | Sep 17, 2019 11:14:14.068 AM |
| ▶ 4 | TaskStateEntered | Check Identity | - | 144 | Sep 17, 2019 11:14:14.171 AM |
| ▶ 5 | LambdaFunctionScheduled | Check Identity | Lambda CloudWatch logs | 144 | Sep 17, 2019 11:14:14.171 AM |
| ▶ 6 | PassStateEntered | Check Fraud Model | - | 157 | Sep 17, 2019 11:14:14.184 AM |

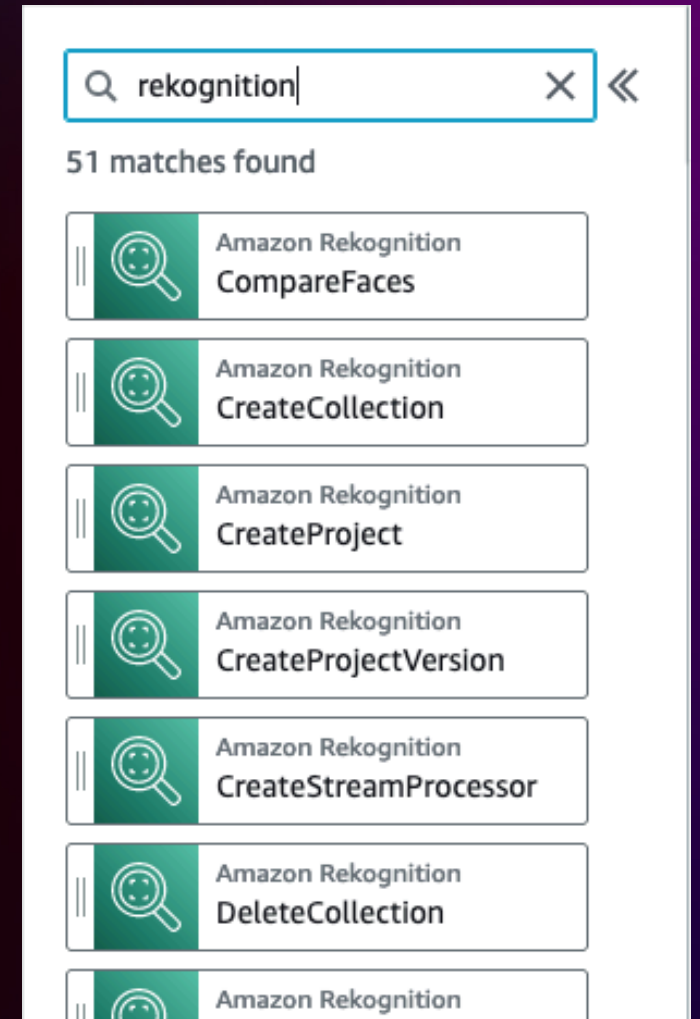
Workflow Studio: Integrations Sidebar



Flow States



Optimized Integrations



AWS SDK Integrations (9000+)

Workflow Studio: Form Configuration

Configuration

Input

Output

Error handling

State name

Check Identity

API

Lambda: Invoke

Integration type [Info](#)

The type of service integration to use. [Learn more](#)

Optimized

API Parameters

Function name

The Lambda function to invoke

Enter function name

arn:aws:lambda:us-east-1:068896461592:function:step-functions-lo

Must be a valid function name.

View function

Payload

The JSON that you want to provide to your Lambda function.

Use state input as payload

Edit as JSON

Task Configuration

Configuration

Input

Output

Error handling

During workflow execution, a Task state's input comes from the previous state's output. [Info](#)

☒ Filter input with InputPath - optional [Info](#)

Use the InputPath filter to select a portion of the state input to use.

\$.data.identity

Must use valid JSONPath syntax, and point to an existing key-value pair in the state input.

Configuration

Input

Output

Error handling

During execution, the Task state calls an API and the response goes into the task result. The result can be manipulated with filters before it is passed as output to the next state. [Info](#)

Lambda:Invoke task result example

A read-only example of the kind of task result to expect from this API:

```
{  "ExecutedVersion": "$LATEST",  "Payload": {    "foo": "bar",    "colors": [      "red",      "blue",      "green"    ],    "car": {      "year": 2000    }  }}
```

Input / Output Processing

Configuration

Input

Output

Error handling

Retry on errors

Retry the task when errors occur. You can specify one or more retry rules, called "retriers".

Retrier #1

Close

Comment - optional

Enter comment

Errors

Specify one or more error(s) that will trigger this retrier. You can select a built-in Amazon States error or enter a custom error name.

Enter an error name

Lambda.ServiceException

Lambda.AWSLambdaException

Lambda.SdkClientException

CustomValidationError

Interval - optional

The number of seconds before the first retry attempt.

1

seconds

Must be greater than zero.

Max attempts - optional

The maximum number of retry attempts.

3

Error Handling

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Use cases

- **IT/Security Automation**
- **Microservice orchestration**
- **$\{\text{your use case}\}$**
- **Additional use cases**

IT Automation

As an **IT engineer** I want to **automate** the process of engineers **requesting** new cloud resources, getting manager's **approval**, **provisioning** resources, **auditing** the process.

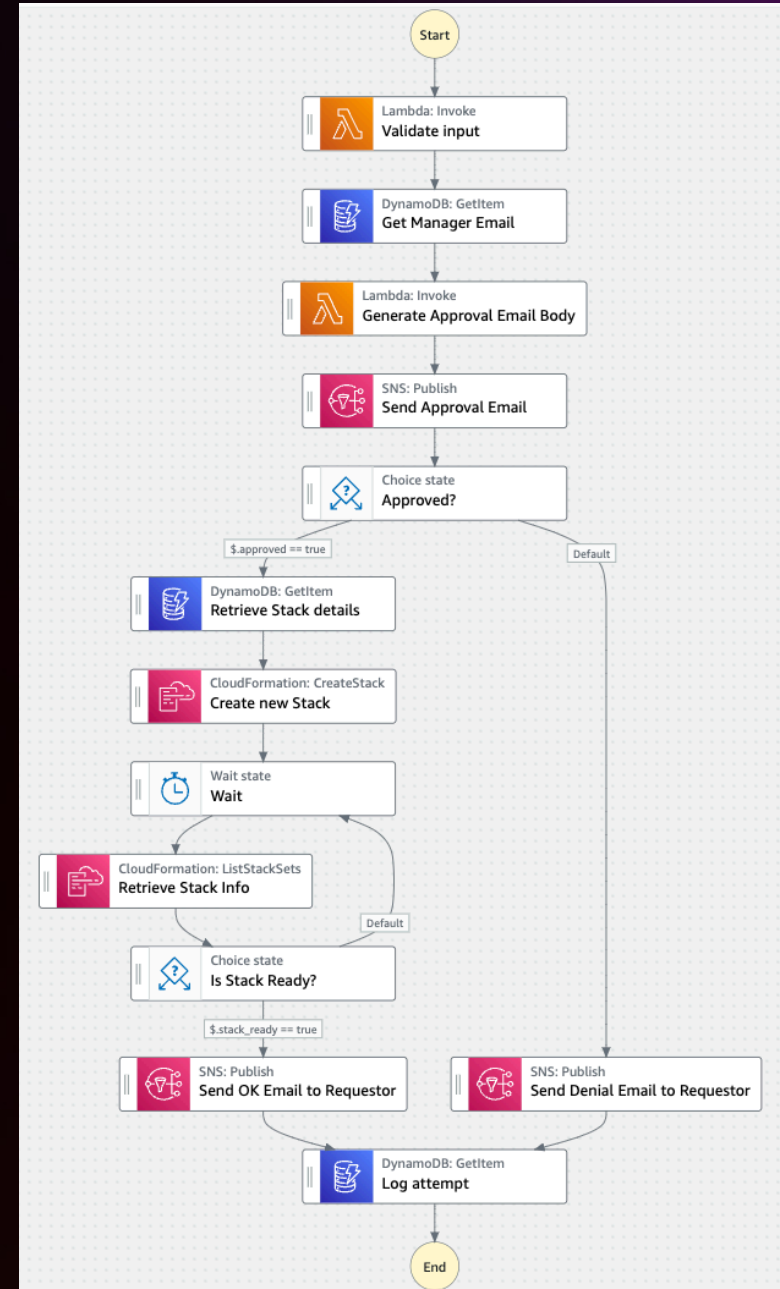
IT Automation

Requesting

Approval

Provisioning

Auditing



Microservices

As a **frontend user**, I want to **invoke** an endpoint that **orchestrates** the process of **accepting, validating, approving/rejecting, emitting an event** during an Account application process.

Microservices

Validation in parallel

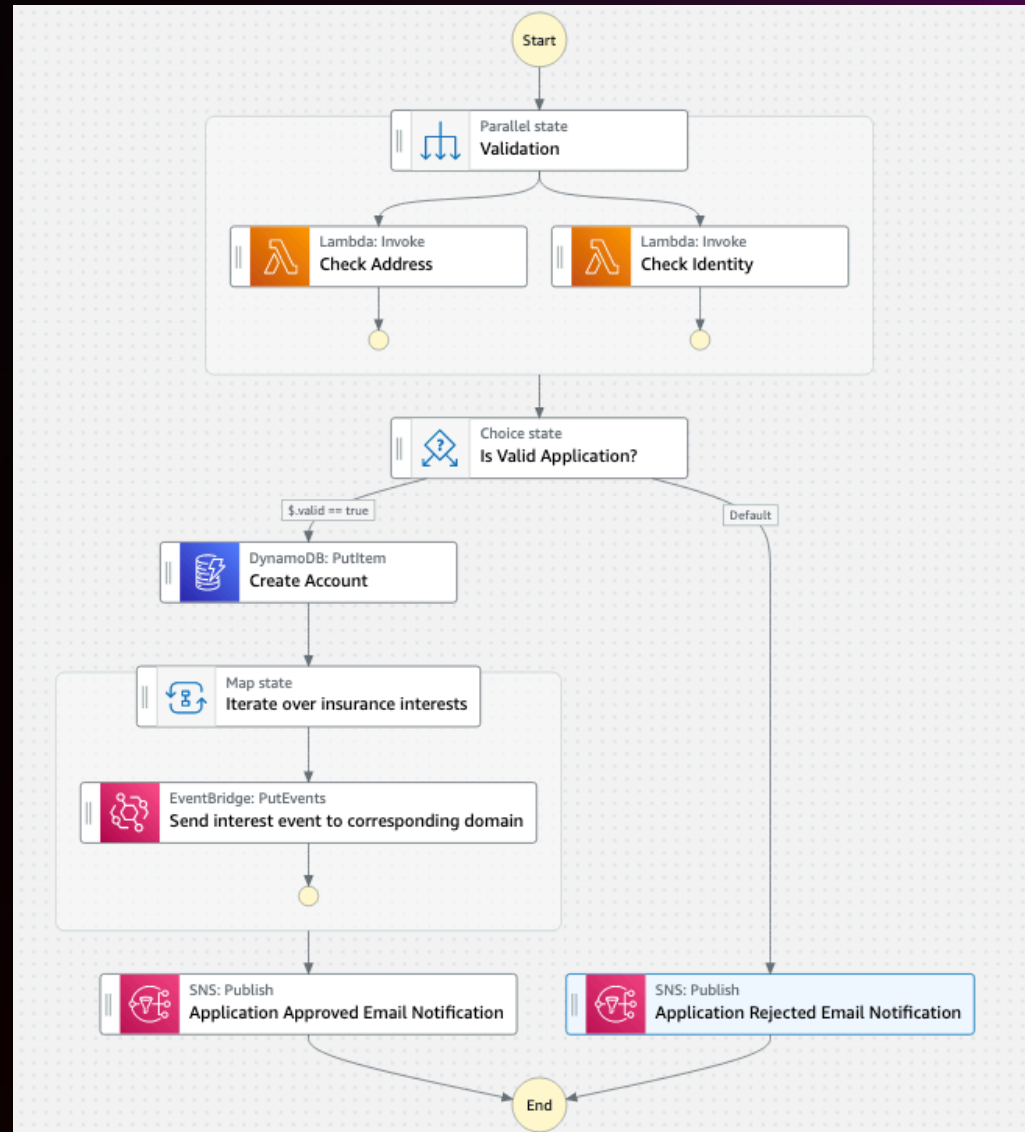
Validation Check

Create Account

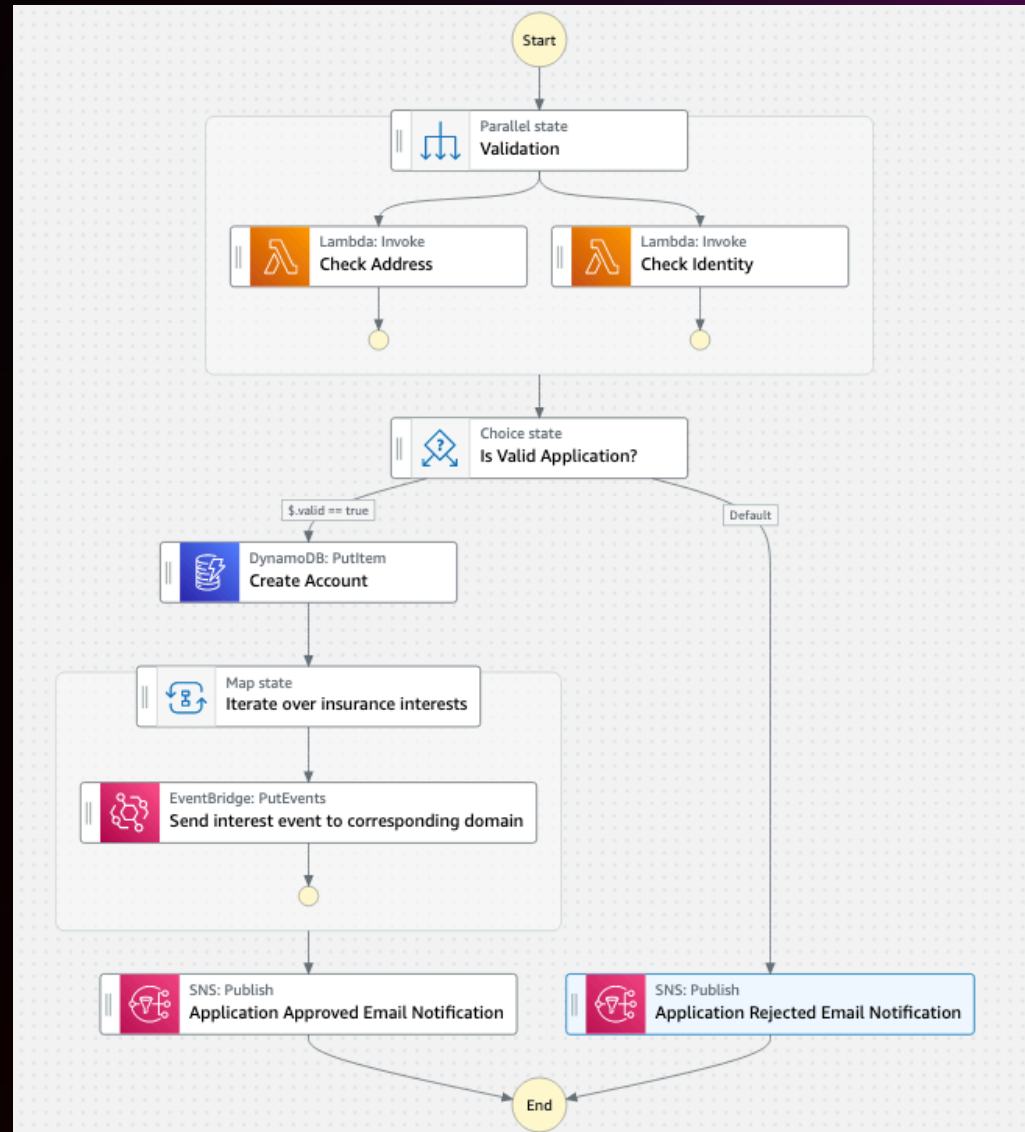
Iterate on interests

Put Events

Notify on approval
or rejection



Microservices



Your Use Cases



Train ML Models

As a **Rekognition user**, I want to **provide training and testing data in S3 bucket**, create **Rekognition Dataset**, **label** images, **train** the model, and finally **start** the model.

Train ML Models

Provide S3 bucket info with prefixed images as input

Check Bucket Permission

Create Rekognition Custom Label Project

Create Dataset from S3 objects

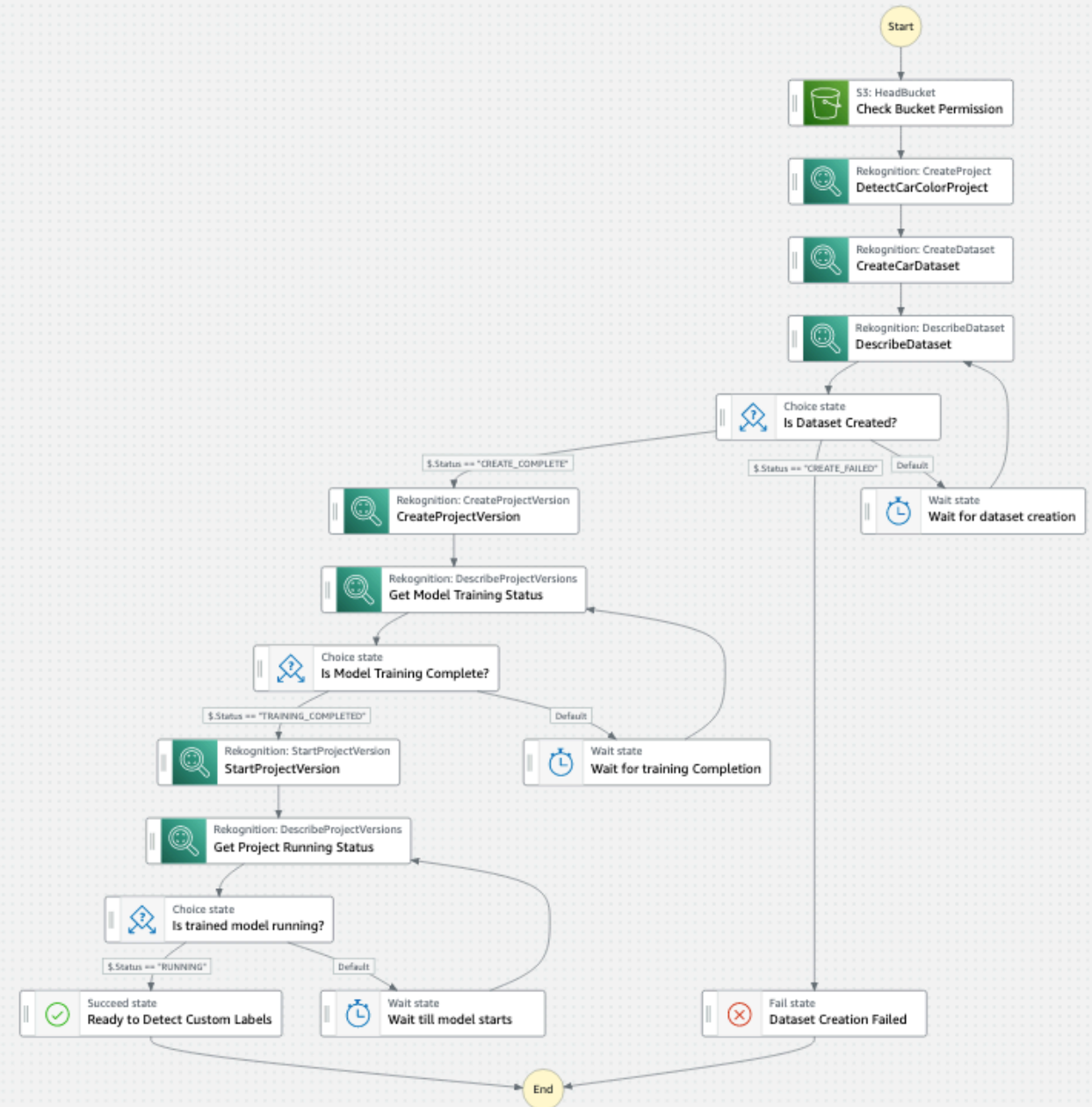
Wait until Dataset is created

Create Project Version (starts training)

Wait until training is complete

Start Project Version (model ready)

Wait until model is running



Document Processing

As **a user**, I want to **process unstructured documents** including PNG, JPG, PDF and **store analyzed data** in the database.

Document Processing

Provide S3 bucket and object info as input

Extract the ID of customer from S3 prefix

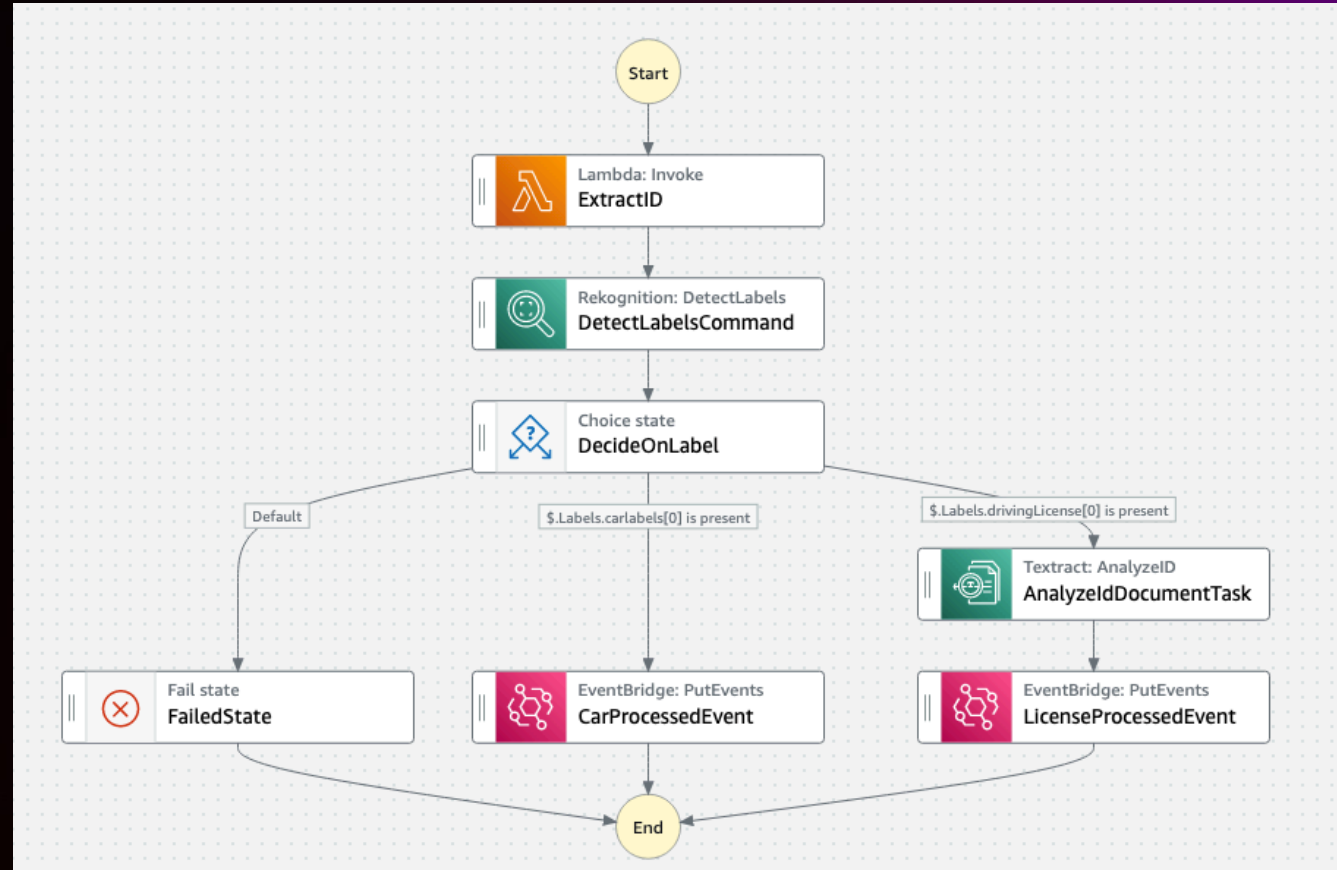
Rekognition DetectLabels to figure out whether image is Drivers License or a Car Image

If Drivers License, AnalyzeID (Textract) to extract unstructured data

Submit EventBridge event as LicenseProcessedEvent

Submit CarProcessedEvent otherwise

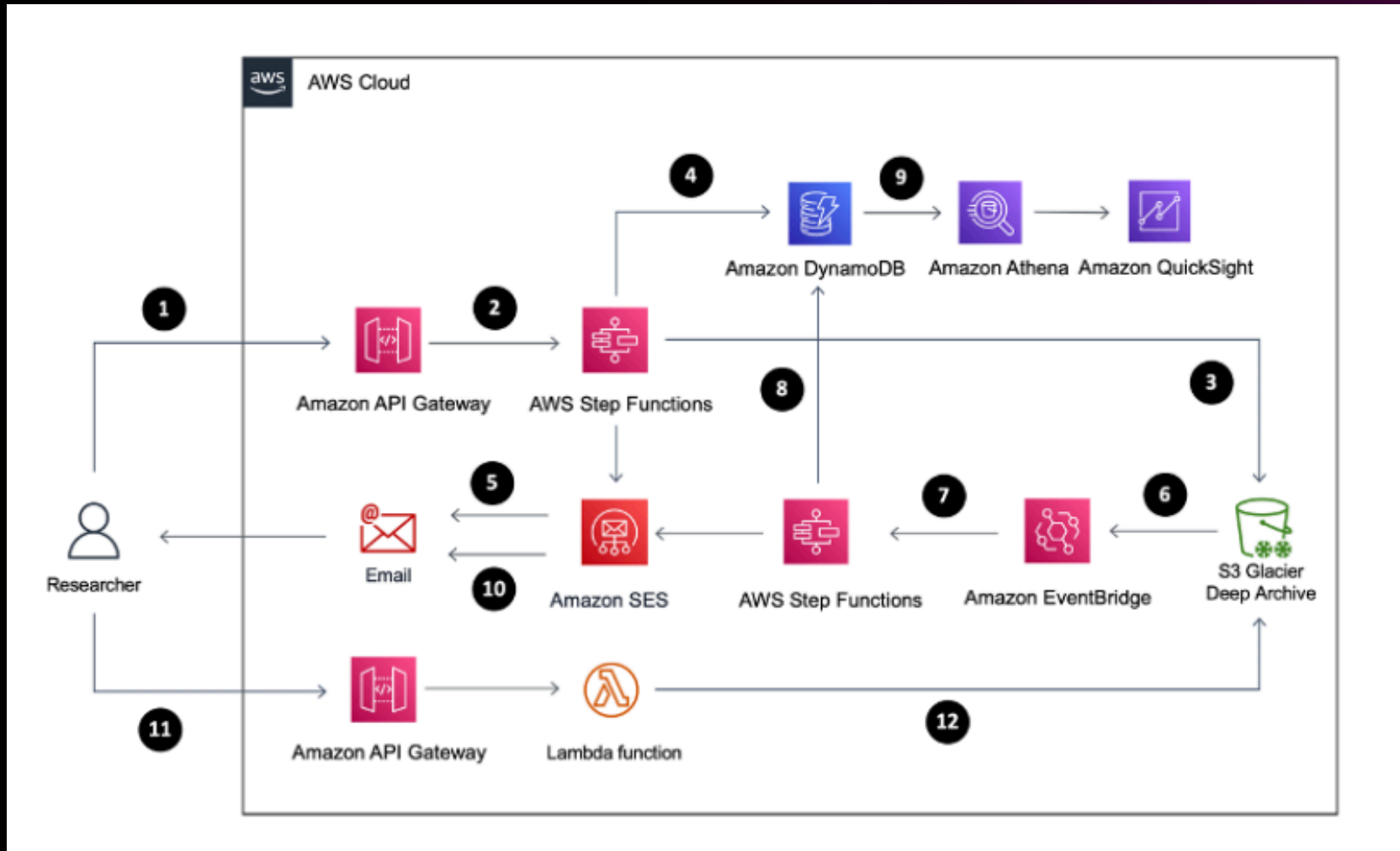
FailedState if no labels present



Restore objects from Amazon S3 Glacier Deep Archive

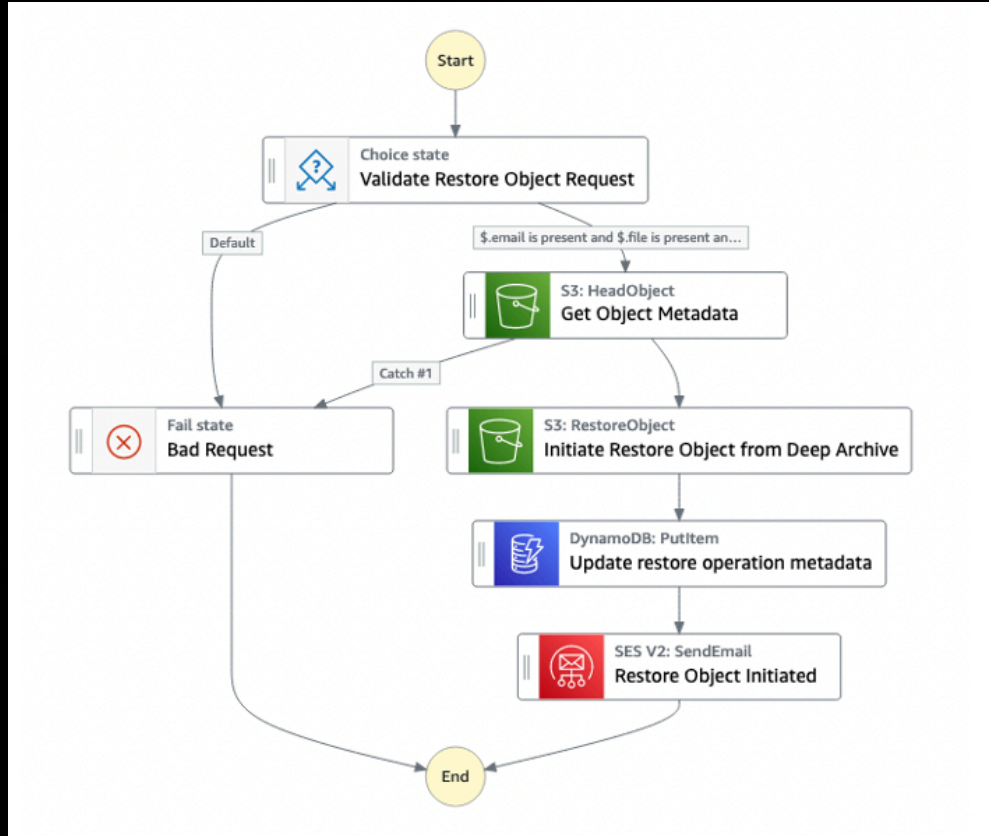
As **a user**, I want to request for **downloading an object from Amazon S3 Deep Archive** and be **notified** when the download is complete.

Restore objects from Amazon S3 Glacier Deep Archive

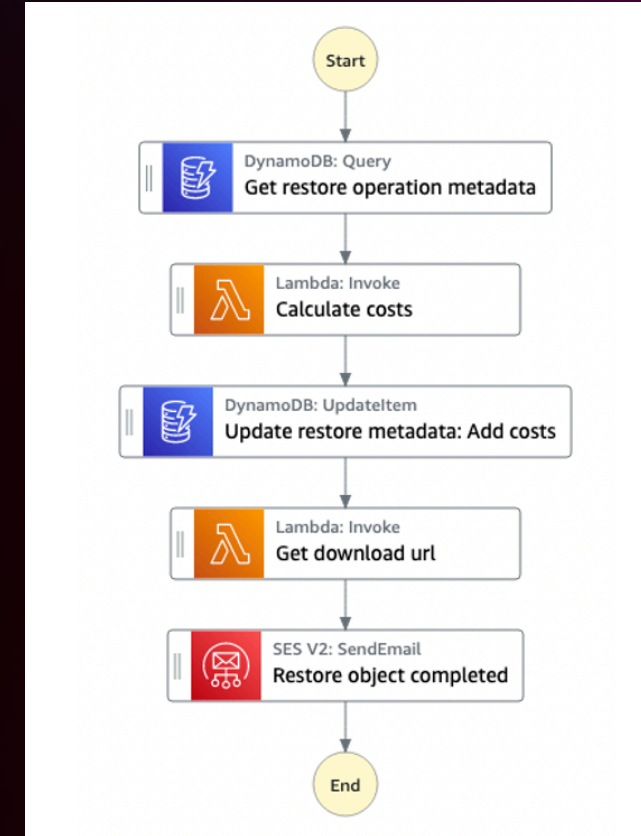


[Blog](#)

Restore objects from Amazon S3 Glacier Deep Archive



Initiate Object Restore Workflow



Object Restore post-processing Workflow

[Blog](#)

Additional Resources



[The AWS Step Functions Workshop](#)

API201 - Tuesday, November 29 11:00 AM - 1:00 PM, Level 3, Premier 312, MGM Grand



[Serverlessland Workflows](#)

Sample workflow patterns

Check out other sessions...

API311: Building next-gen applications with event-driven architectures

Breakout Session: Monday (Nov 28) @ 10:00am – Level 1, Lafite 5, Wynn

API204: Model your business process with AWS Step Functions

Builders' Session: Monday (Nov 28) @ 11:30am – Level 1, Academy 417, Caesars Forum

API301: Deploying a complete machine learning fraud detection solution

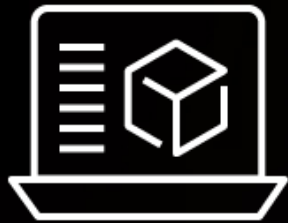
Chalk Talk: Tuesday (Nov 29) @ 1:15pm – Level 3, 354, MGM Grand

SVS306: Serverlesspresso: Building an event-driven app from the ground up

Workshop: Wednesday (Nov 30) @ 2:30pm – Upper Level, Cristal 3, Wynn

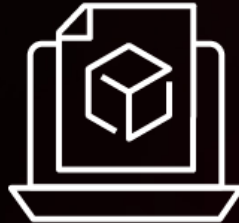
Continue your AWS Serverless learning

Learn at your
own pace



Expand your Serverless
skills with our Learning Plan
on **AWS Skill Builder**

Increase your
knowledge



Use our **Ramp-Up Guides**
to build your Serverless
knowledge

Earn AWS
Serverless badge




Demonstrate your
Knowledge by achieving
digital badges



<https://s12d.com/serverless-learning>

Thank you!

Anton Aleksandrov
antonaws@amazon.com
antonal80 (LinkedIn)

Dhiraj Mahapatro
mahadhir@amazon.com
 @dhirajmahapatro
dmahapatro (LinkedIn)

