



Build and orchestrate serverless generative AI applications

Mohammed Fazalullah Qudrath

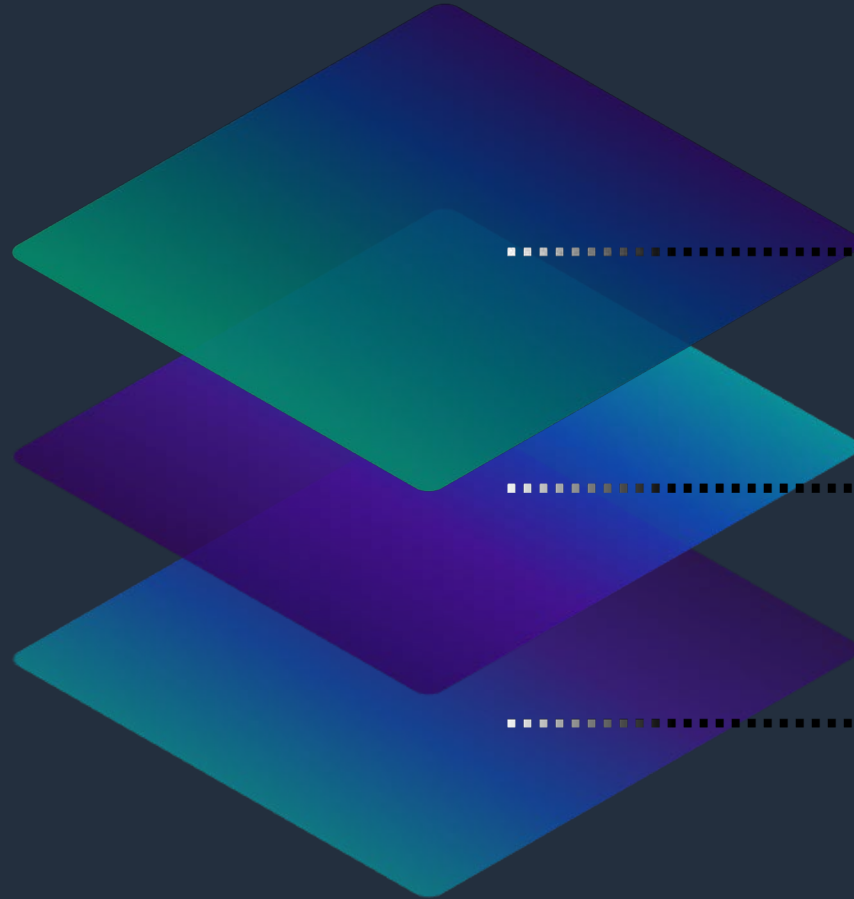
Sr Developer Advocate

AWS



Create product listings

GENERATIVE AI STACK



APPLICATIONS THAT
LEVERAGE FMs

TOOLS TO BUILD WITH
LLMs & OTHER FMs

INFRASTRUCTURE FOR FM
TRAINING & INFERENCE


Generative AI Stack












APPLICATIONS THAT LEVERAGE FMs

-  Amazon Q
-  Amazon Q in Amazon QuickSight
-  Amazon Q in Amazon Connect
-  Amazon CodeWhisperer

TOOLS TO BUILD WITH FMs AND LLMs

-  Amazon Bedrock
- Guardrails
- Agents
- Customization capabilities

INFRASTRUCTURE FOR FM TRAINING & INFERENCE

-  GPUs
-  Trainium
-  Inferentia
-  SageMaker
-  UltraClusters
-  EFA
-  EC2 Capacity Blocks
-  Nitro
-  Neuron



Amazon Bedrock

The easiest way to build and scale generative AI applications with LLMs and other foundation models

Broad choice of foundation models

Customize foundation models using your organization's data

Enterprise-grade security and privacy

Amazon Bedrock

Broad choice of models

AI21 labs

amazon

ANTHROPIC

cohere

Meta

Mistral AI

stability.ai

JURASSIC-2

AMAZON TITAN

CLAUDE

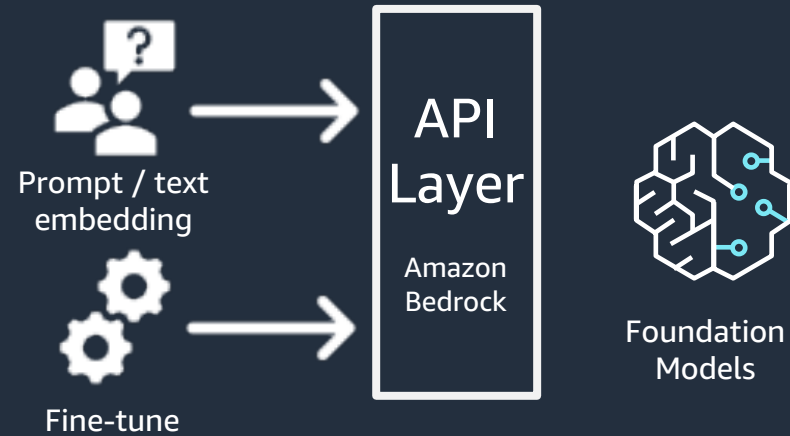
COMMAND + EMBED

LLAMA 2

Mistral 7B
Mixtral 8x7B

STABLE DIFFUSION XL

How do I access foundation models?



Amazon Bedrock

Integrating with Amazon Bedrock

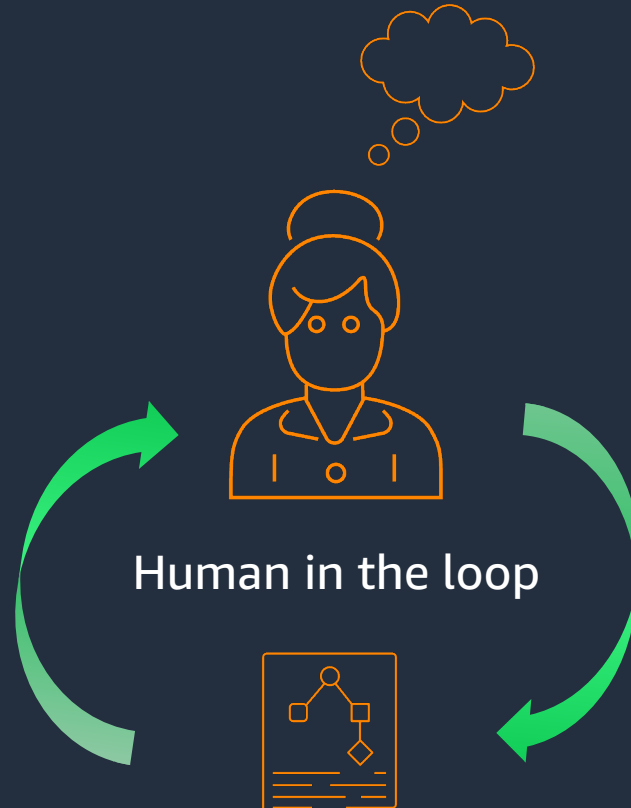
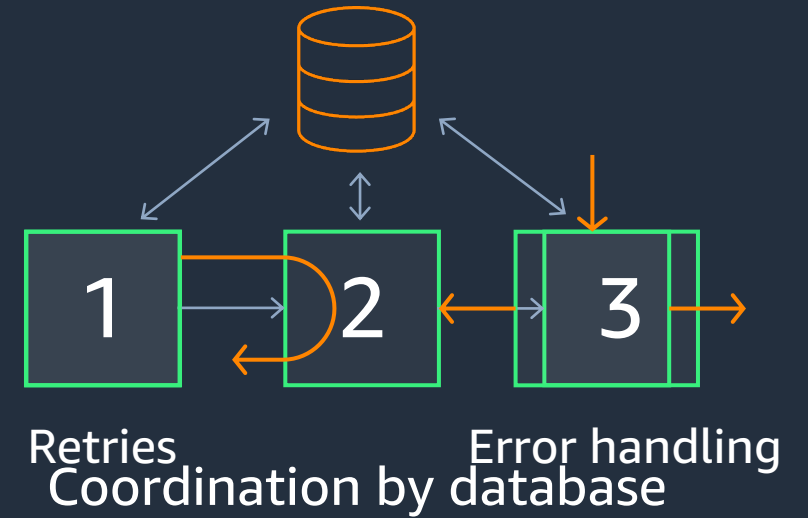
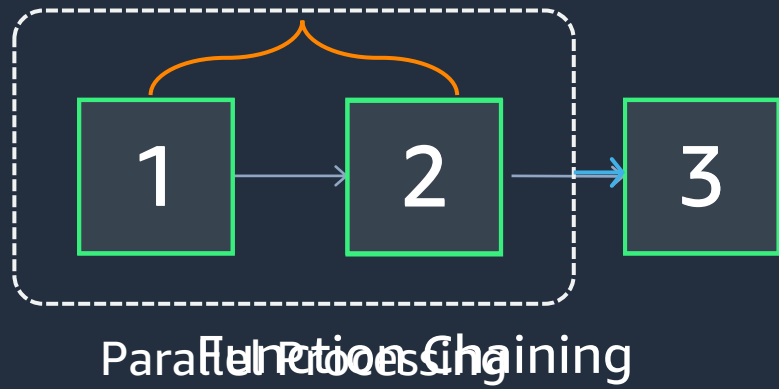
```
app.js
const AWS = require('aws-sdk');

prc
request = json.dumps({
  'prompt': f'Human:{prompt_data}\n\nAssistant:',
  'max_tokens_to_sample': 1028,
  'temperature': 1,
  'top_k': 250,
  'top_p': 0.999,
  'stop_sequences': ['\n\nHuman:']
})
client = boto3.client('bedrock-runtime')
response = bedrock_client.invoke_model(
  modelId=event["ModelId"],
  body=json.dumps(event["Body"]),
)
body = json.loads(response["body"].read().decode("utf-8"))
response["body"] = body

} catch (err) {
  return { error: err }
}
}
```



Sequencing is hard



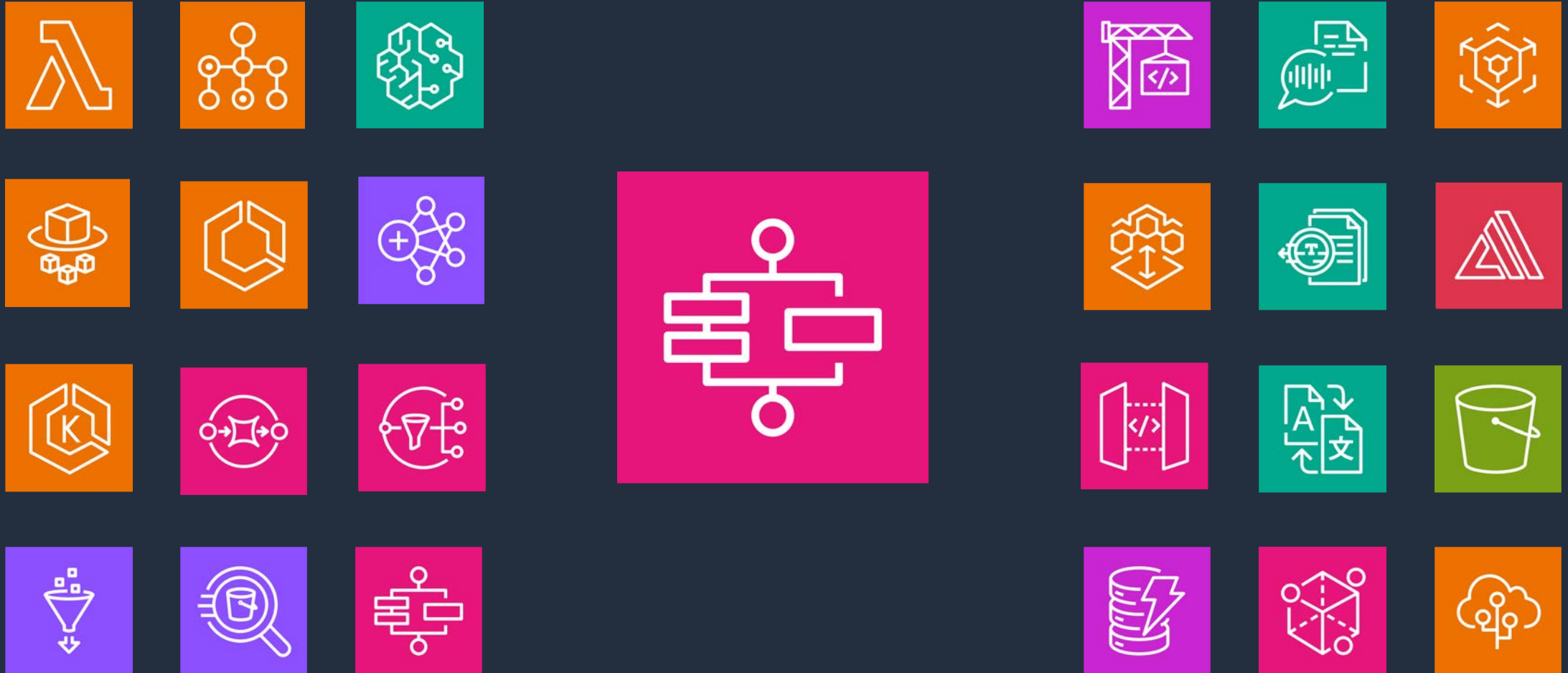
AWS Step Functions

SERVERLESS VISUAL WORKFLOW SERVICE

- Pay per use
- Scales automatically
- Fully managed
- Drag and drop or ASL
- Built-in error handling
- Integrates with over 220 AWS services

The screenshot displays the AWS Step Functions console interface. On the left, there is a sidebar with a search bar and two tabs: 'Actions' and 'Flow'. Under the 'Actions' tab, a list of actions is shown, including AWS Lambda Invoke, Amazon SNS Publish, Amazon ECS RunTask, AWS Step Functions StartExecution, AWS Glue StartJobRun, AWS Glue DataBrew StartJobRun, Amazon EventBridge PutEvents, and AWS Batch SubmitJob. The main workspace shows a workflow diagram with a 'Start' node, a state box labeled 'Drag first state here' (with a dashed border and a mouse cursor), and an 'End' node. The top of the workspace has a toolbar with 'Undo', 'Redo', 'Zoom in', 'Zoom out', and 'Center' buttons. On the right, there are two tabs: 'Form' and 'Definition'. The 'Form' tab is active, showing a 'Workflow' section with a 'Comment - optional' field containing 'My state machine' and a 'TimeoutSeconds - optional' field containing '600'.

Directly compose applications from over 220 AWS services and 10,000 API actions



Control service integrations with call patterns



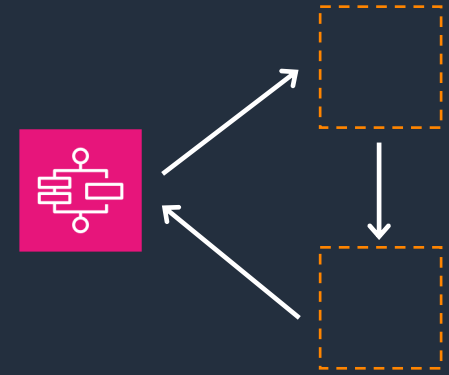
REQUEST-RESPONSE

Wait for an HTTP *response* then progress



JOB-RUN (.sync)

Wait for the request to *complete* then progress



CALLBACK (.waitForTaskToken)

Pause until a *task token is returned* then progress

Why direct integration is powerful

```
app.js
const AWS = require('aws-sdk');
const docClient = new AWS.DynamoDB.DocumentClient();

var params = {
  "TableName": "reinvent2023!",
  "Key": {
    "PK": {"S": "Wardrobe"},
    "SK": {"S": "shoes"}
  }
}

async function queryItems(){
  try {
    const data = await docClient.getItem(params).promise()
    return data
  } catch (err) {
    return err
  }
}

exports.handler = async (event, context) => {
  try {
    const data = await queryItems()
    return { body: JSON.stringify(data) }
  } catch (err) {
    return { error: err }
  }
}
```

A Lambda function that queries Amazon DynamoDB has multiple lines of code



Why direct integration is powerful

```

    app.js

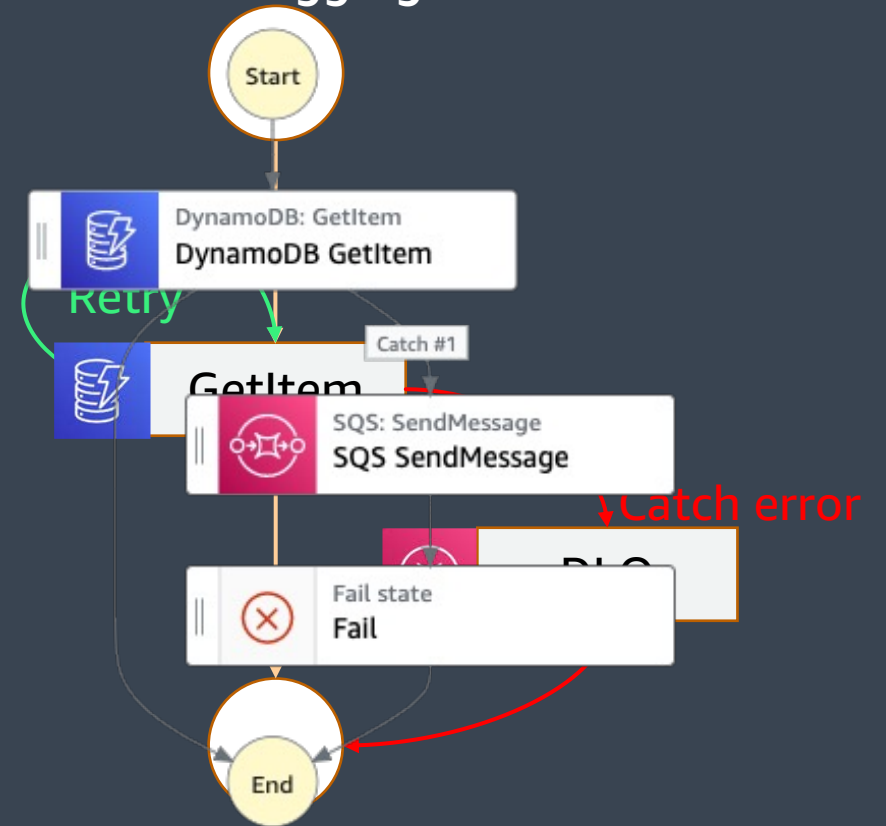
    const AWS = require('aws-sdk');
    const docClient = new AWS.DynamoDB.DocumentClient();

    var params = {
      "TableName": "reinvent2023",
      "Key": {
        "PK": {"S": "Wardrobe"},
        "SK": {"S": "shoes"}
      }
    }

    async function queryItems(){
      try {
        const data = await docClient.getItem(params).promise()
        return data
      } catch (err) {
        return err
      }
    }

    exports.handler = async (event, context) => {
      try {
        const data = await queryItems()
        return { body: JSON.stringify(data) }
      } catch (err) {
        return { error: err }
      }
    }
  
```

Even a single task “workflow” adds value with built-in error handling, catch, retry, observability, reduction of custom code, and centralized logging of each workload



Why direct integration is powerful

Drill down to trace execution path of every workload request

Executions (10)

View details Stop execution Start execution

Search for executions Filter by status < 1 >

Name	Status	Started	End Time
74a40347-bdb5-6655-d925-7608442c8d88	Failed	Sep 21, 2022 04:37:22.146 PM	Sep 21, 2022 04:37:22.647 PM
76180c66-a9d9-e986-3f78-0d192d284df3	Failed	Sep 21, 2022 04:37:11.989 PM	Sep 21, 2022 04:37:12.059 PM
b3195b78-3127-4ada-c1cd-4b1d5324f9b1	Failed		
b1a392c2-cd0f-c5a2-a0e2-a63310bbc735	Failed		
8e0fdc23-6074-be1f-5afa-6a3633b33750	Success		
813c077a-5bb5-7a98-71e7-59f76770c4ee	Success		
d8c43467-a5f1-c658-ccf0-ae4f08b045ca	Success		
478a9a01-d77d-f8fc-66d6-bcde36f932e0	Success		
d9f66115-8528-43fe-a0b5-5bee5abe101e	Success		
e679eed1-44d7-4a18-99f0-ca80bc571adb	Success		

Graph view Data flow simulator Export Layout

```
graph TD; Start((Start)) --> DynamoDB[DynamoDB GetItem]; DynamoDB --> SQS[SQS SendMessage]; SQS --> Fail[Fail]; Fail --> End((End));
```

Input & Output Details Definition Events

Advanced view

Input Learn more

```
1 {
2   "item": "shoes"
3 }
```

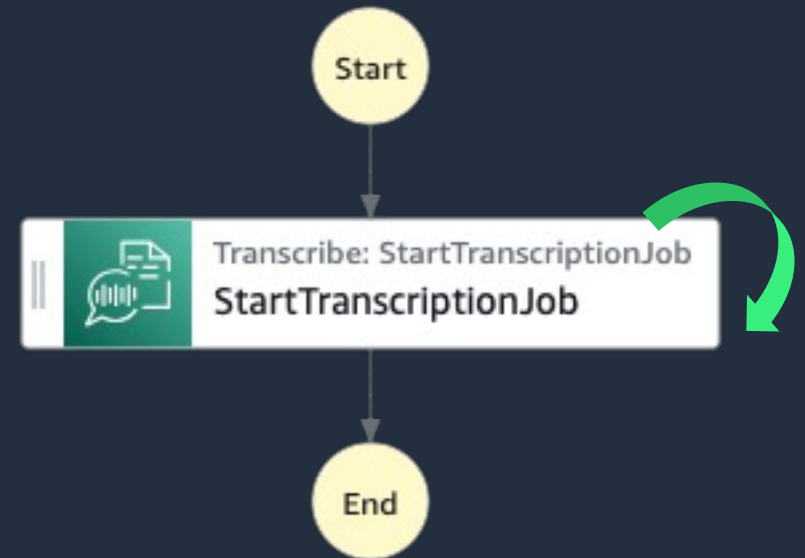
Output Learn more

```
1 {
2   "Error": "DynamoDB.AmazonDynamoDBException",
3   "Cause": "1 validation error detected: Value 'reinvent2022!' at 'tableName' fail
satisfy regular expression pattern: [a-zA-Z0-9_.-]+ (Service: AmazonDynamoDBv2; St
ValidationException; Request ID: 8Q80L4JJQSQ5A38S9G66GJ5NPNVV4KQNSO5AEMVJF66Q9ASUA
4 }
```

In progress Failed Caught error Canceled Succeeded

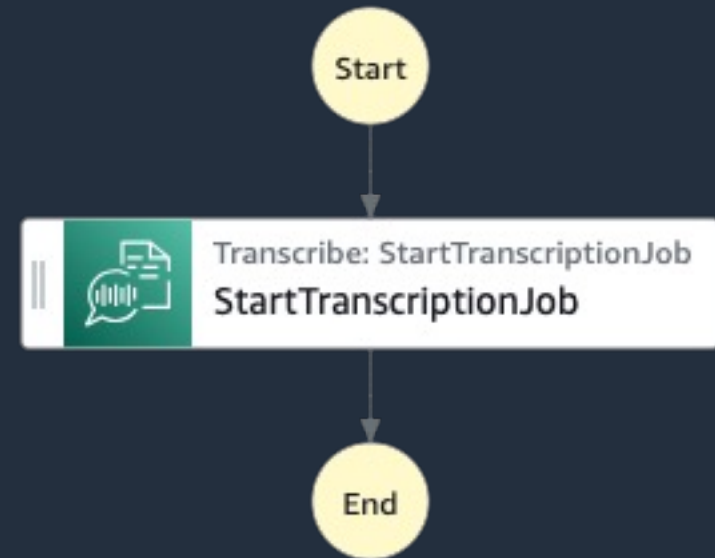
Machine learning and generative AI

Integrate directly with Amazon Transcribe API to convert video to text



Building a generative AI application

- Create multiple titles and descriptions for video
- Ask human to provide feedback
- Create an avatar for the video



Optimized integration for Amazon Bedrock



Two new optimized integrations to simplify building and scaling serverless generative AI applications

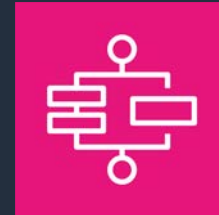
Prompt: "Write me a title..."



S3 bucket



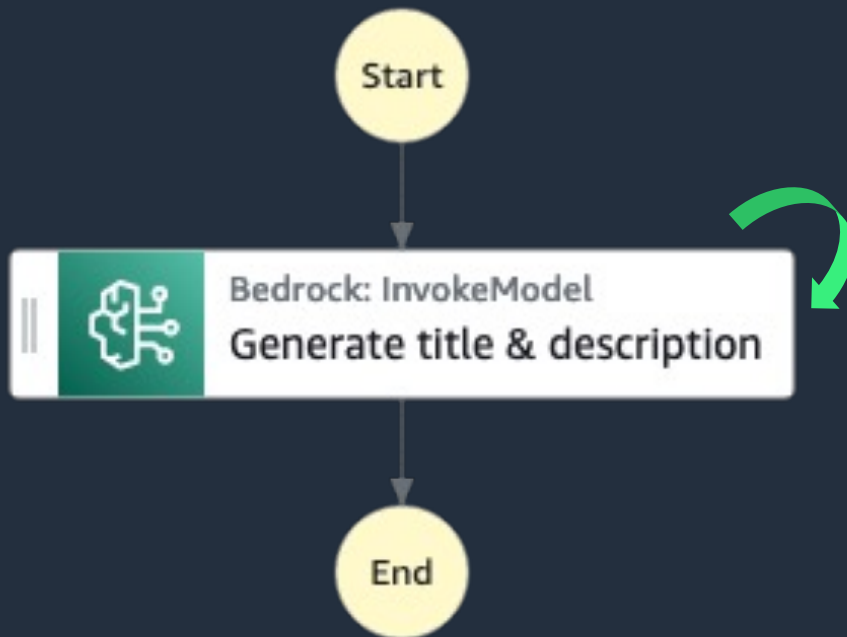
Invoke model



Create model customization job
.sync



Requirement 1: Generate title and description



Generate title & description Definition Test state >

Configuration | Input | Output | Error handling

State name
Generate title & description

API
Bedrock: InvokeModel

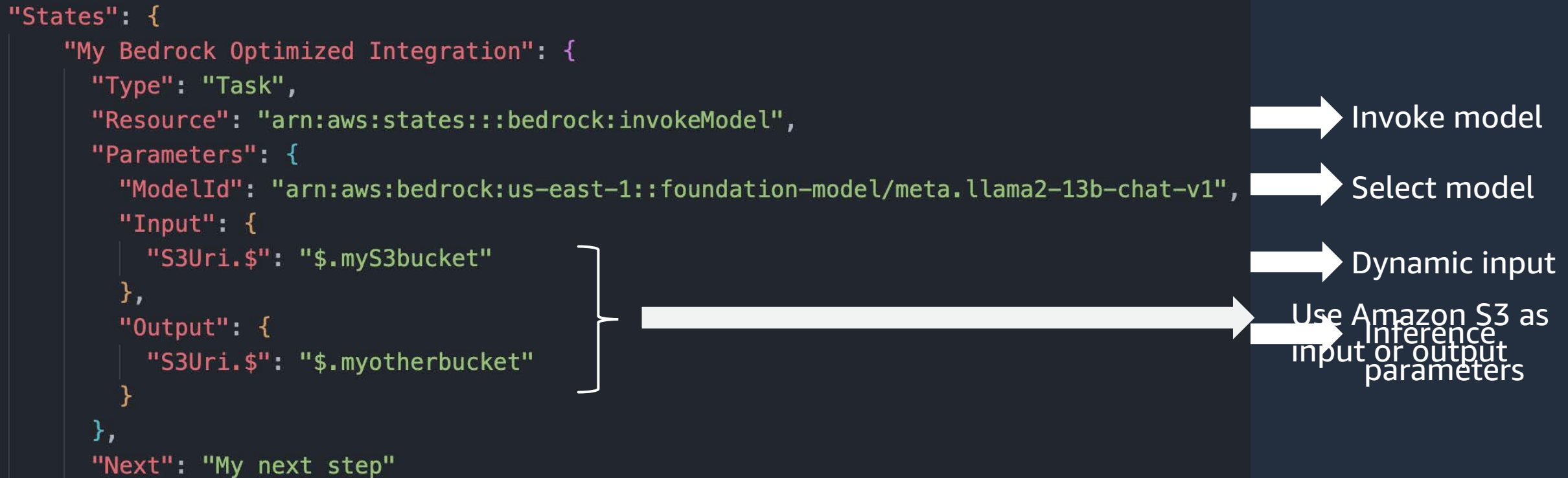
API Parameters

Foundation Models Provisioned Models

Bedrock model identifier
Determines which foundation model will be invoked
Enter model
arn:aws:bedrock:us-east-1::foundation-model/amazon.titan-tg1-large
Must be a valid model.

Bedrock Model Parameters
JSON object containing inference parameters for the selected Bedrock model. Contains sample values. Update the JSON with your own parameter values.
 Enter model inference parameters Load model inference parameters from S3

Amazon States Language (ASL) for Amazon Bedrock

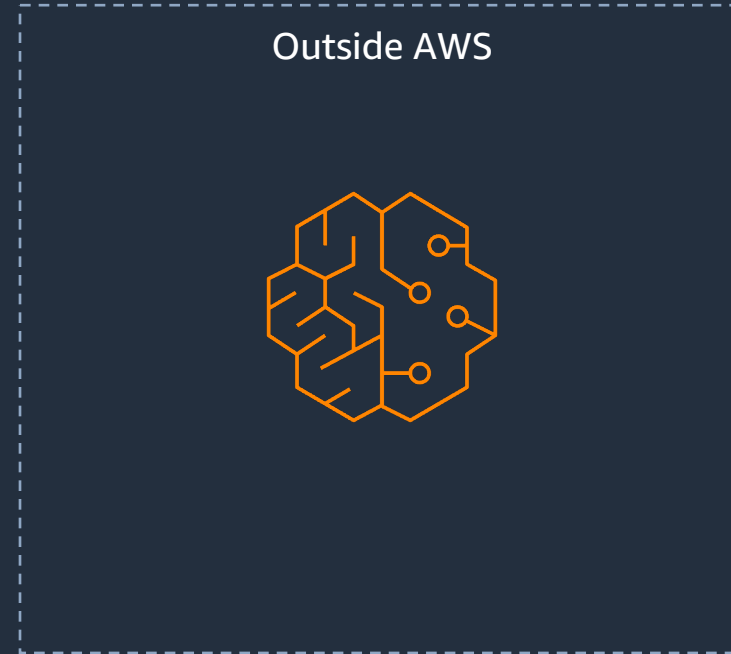
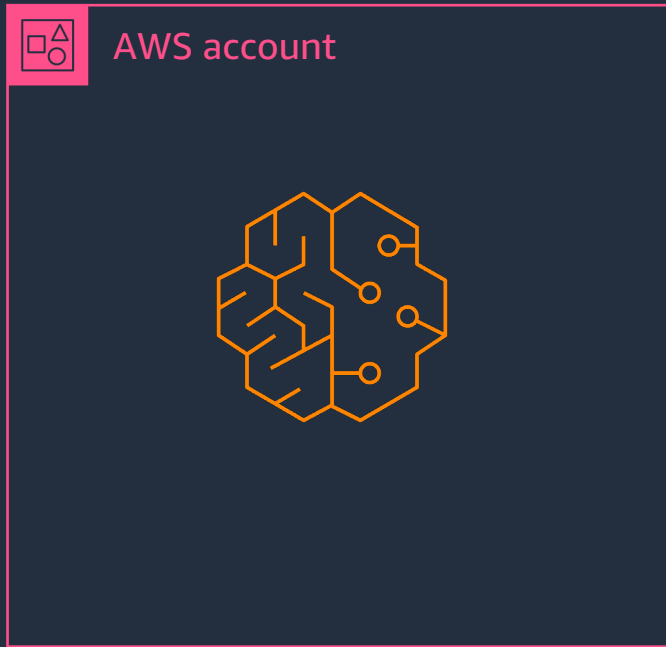


Building a generative AI application

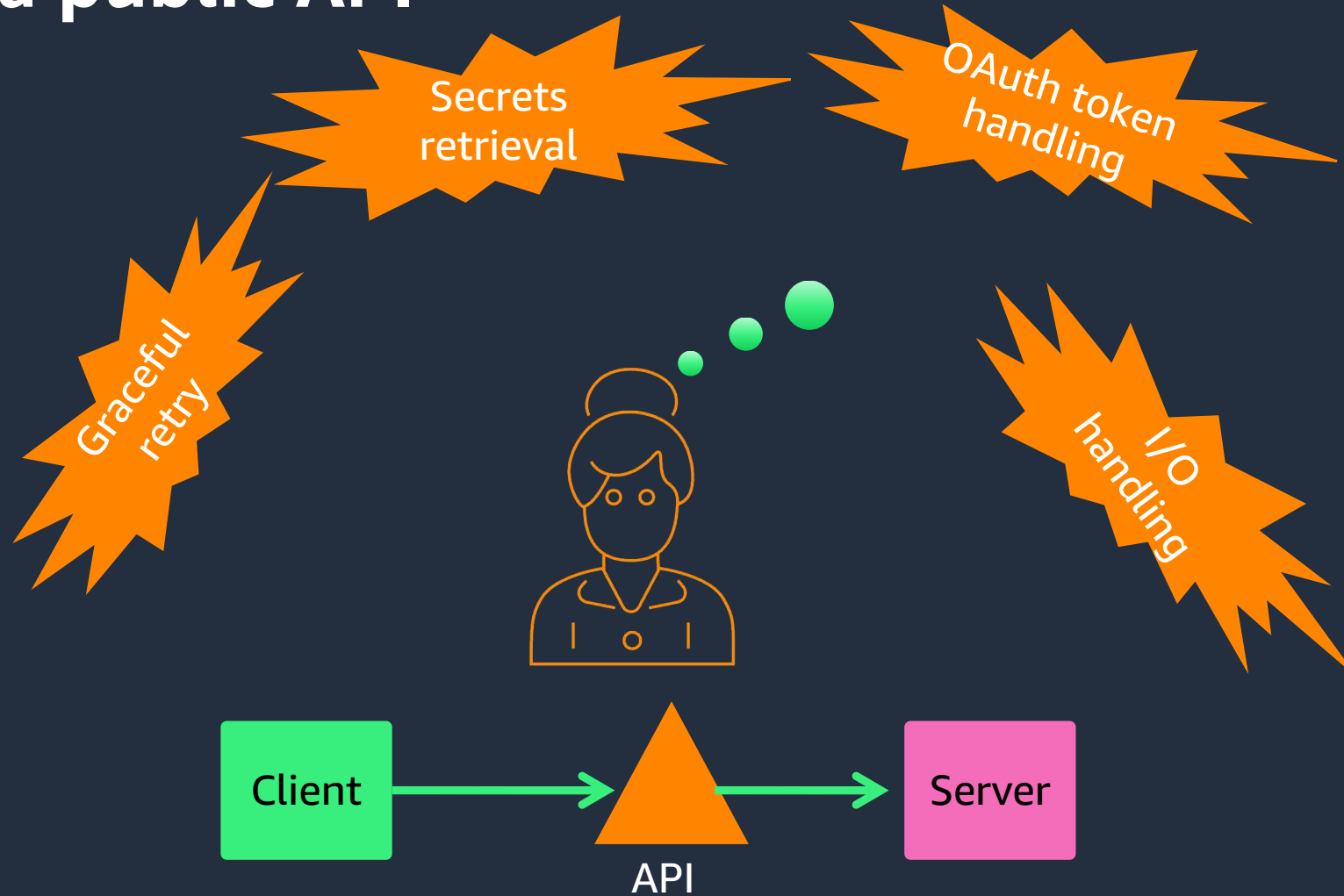
- Create **multiple** titles and descriptions for video
- Ask human to provide feedback
- Create an avatar for the video



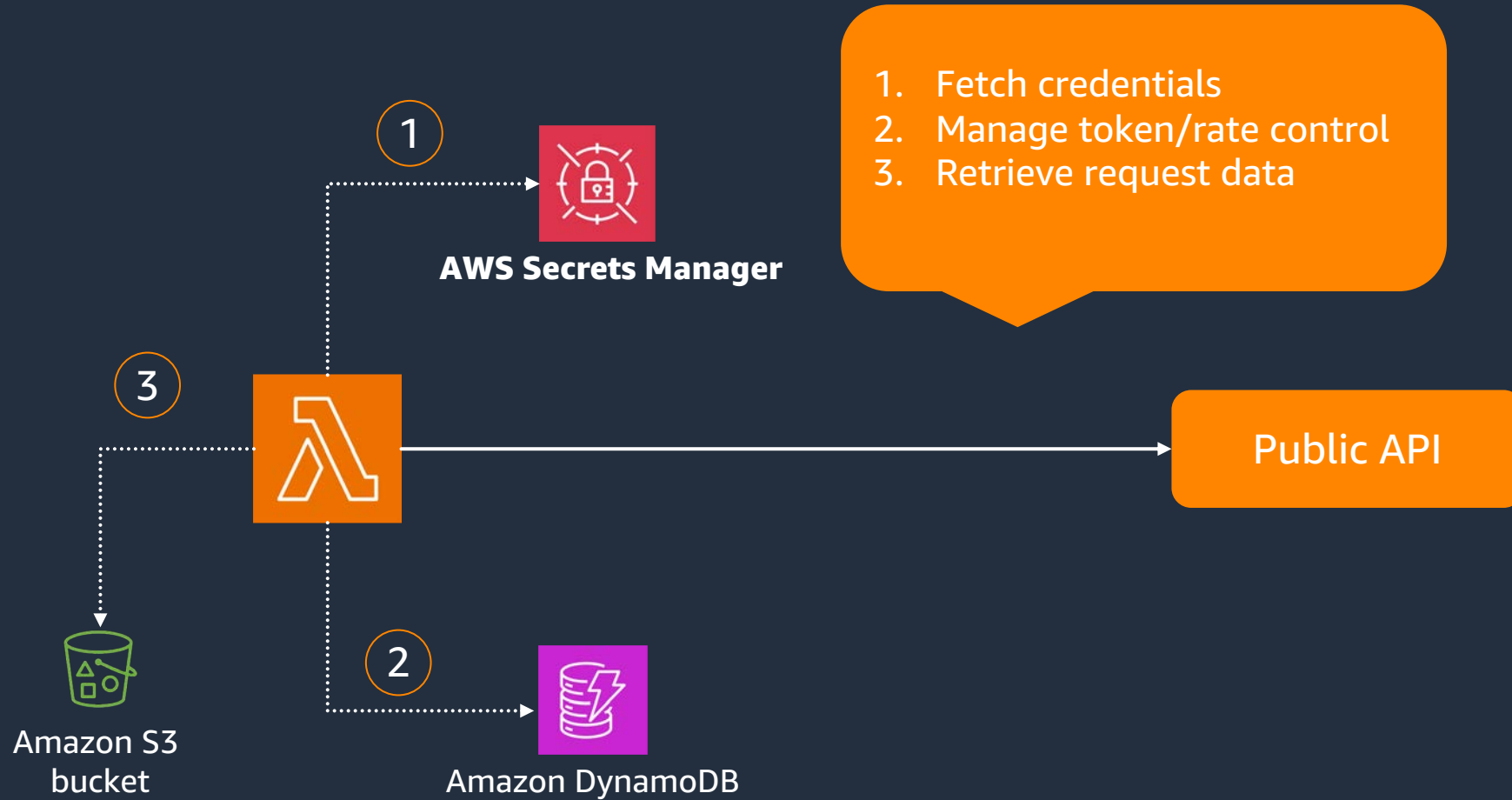
How do I access foundation models outside AWS?



Accessing a public API



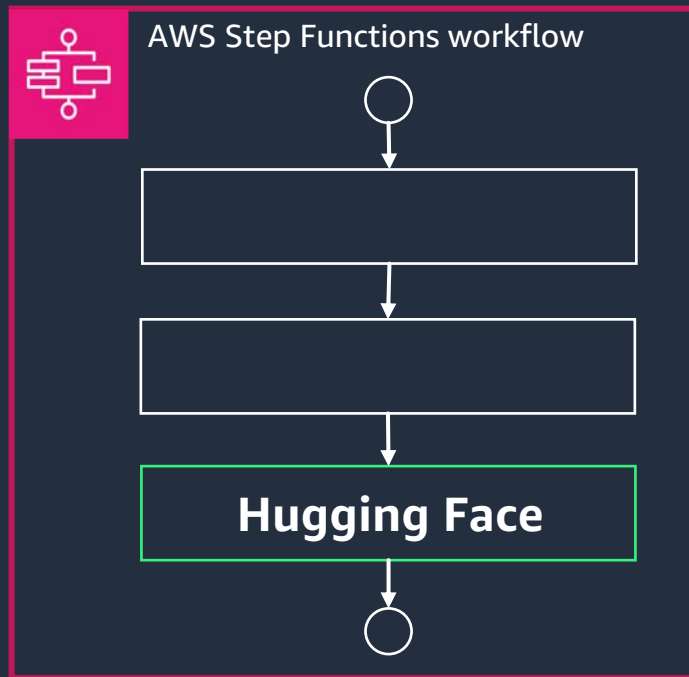
Accessing a public API



Public HTTPS API integration



Easily integrate with public HTTPS APIs



The screenshot shows the AWS Step Functions console interface for a state machine named "MyStateMachine-j9pem2g56". The interface includes a top navigation bar with "Design", "Code", and "Config" tabs, and a right-hand side with "Cancel", "Actions", and "Create" buttons. Below the navigation bar are utility buttons for "Undo", "Redo", "Zoom in", "Zoom out", "Center", "Duplicate", and "Delete". A search bar is present above the "Actions" tab. The main workspace displays a workflow diagram with a "Start" node, a dashed box labeled "Drag first state here", and an "End" node. On the left, a "MOST POPULAR" list includes "AWS Lambda Invoke", "Amazon SNS Publish", "Amazon ECS RunTask", "AWS Step Functions StartExecution", and "AWS Glue StartJobRun". Below that is a "THIRD-PARTY API" section with "HTTP Endpoint Call third-party API". The "COMPUTE" section lists "Amazon Data Lifecycle Manager", "Amazon EBS", "Amazon EC2", and "AWS EC2 Instance Connect". On the right, the "Workflow" panel shows "Definition" as the active view, with a description field containing "A description of my state machine" and a "TimeoutSeconds" field set to "600".

Easily integrate with public HTTPS APIs



Handle errors

Retry and catch using
http status code



Authorization

OAuth, Basic, and
API key



Transform data

Specify URL-encoding
for request body



Test state

Test step independently
to validate business logic

Amazon States Language (ASL) for HTTPS endpoint



Requirement 1: Generate **multiple** titles

The screenshot displays the AWS Step Functions console interface. At the top, there are tabs for 'Design', 'Code', and 'Config'. Below these are navigation and utility buttons like 'Undo', 'Redo', 'Zoom In', 'Zoom out', 'Center', 'Duplicate', and 'Delete'. A search bar is located on the left side of the design canvas.

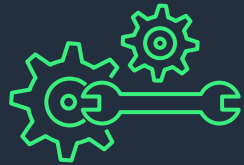
The central design canvas shows a workflow diagram starting with a 'Start' node, leading to a 'Parallel state' box labeled 'Parallel'. This state branches into two parallel actions: 'Bedrock: InvokeModel' with the task 'Generate title & description', and 'HTTP Endpoint' with the task 'Hugging Face'. Both actions lead to a final 'End' node.

On the right side, the configuration panel for the 'Hugging Face' action is open. It includes tabs for 'Configuration', 'Input', 'Output', and 'Error handling'. The 'Configuration' tab is active, showing fields for 'State name' (Hugging Face), 'State type' (Task: HTTP endpoint), and 'API Parameters' (API endpoint: https://api-inference.huggingface.co/models/google/tapas-base-finetun). The 'Method' is set to 'GET'.

What if something goes wrong...



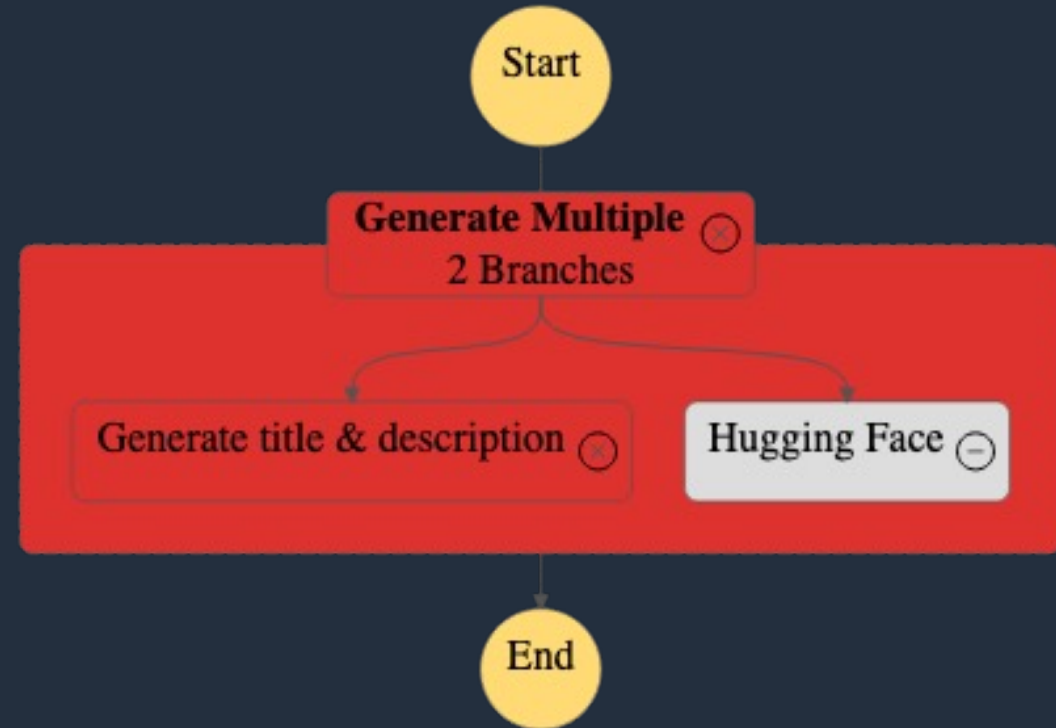
Issue arises



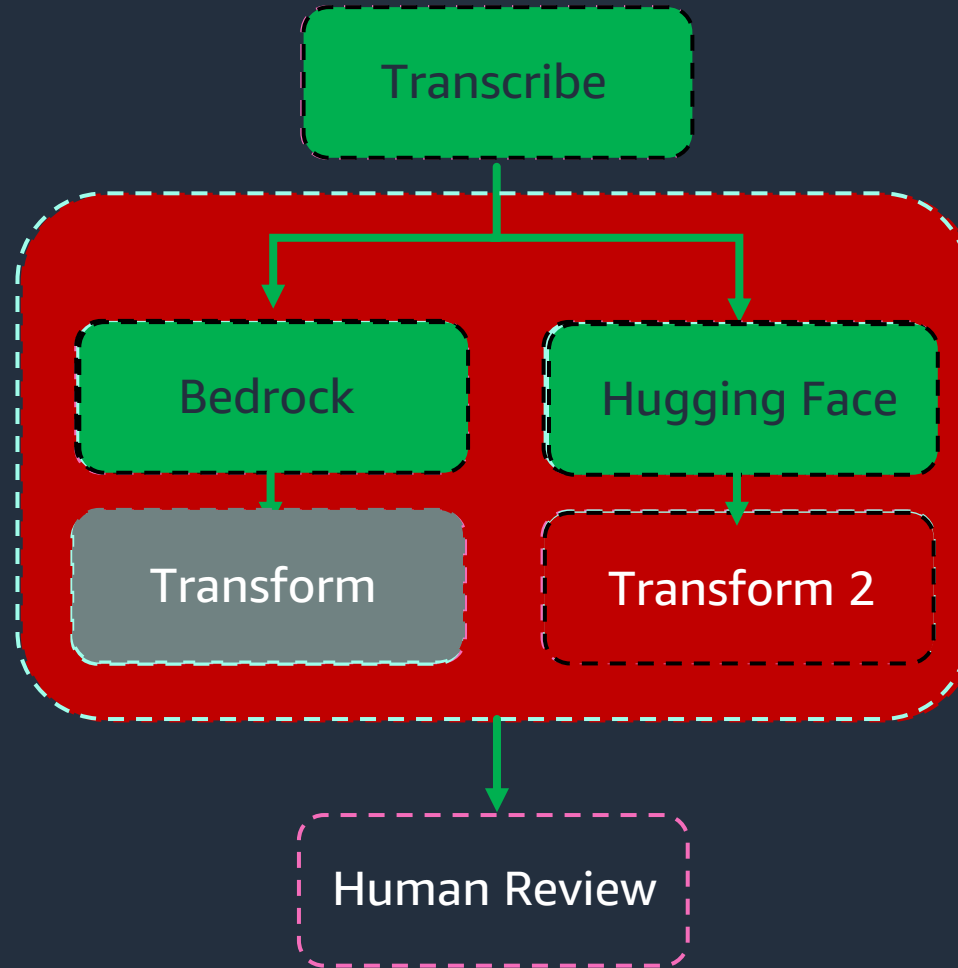
Fix the issue



Recover quickly



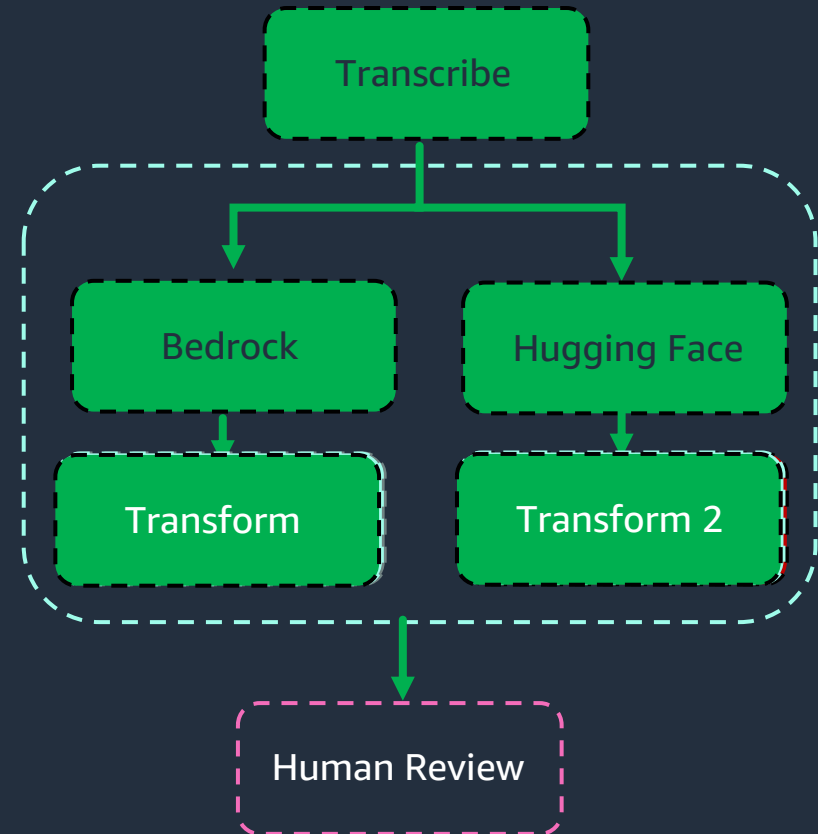
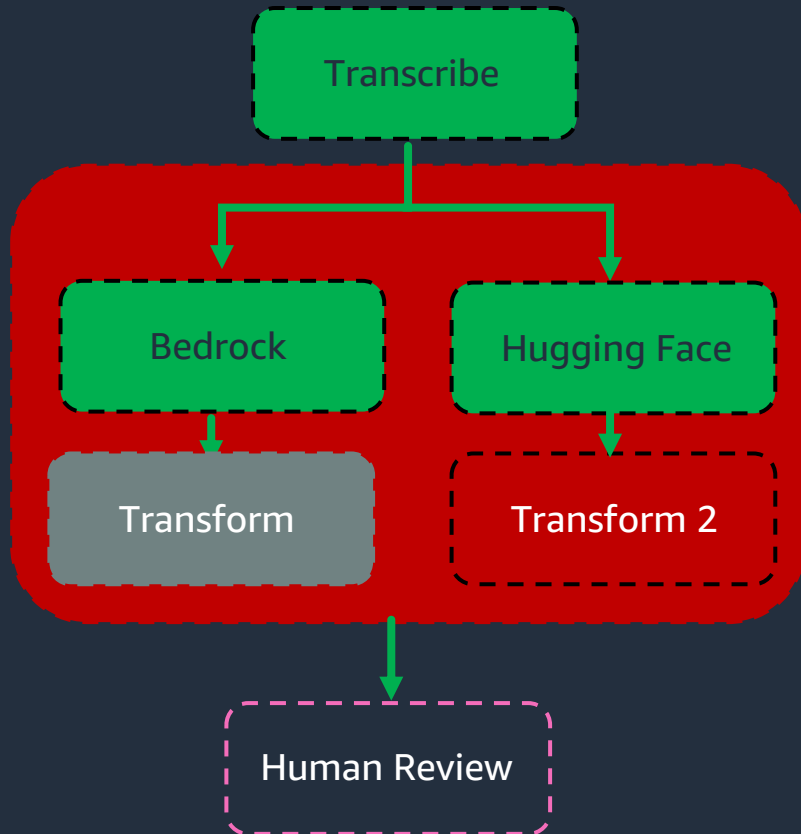
How do you handle hard failures?



Redrive



Recover from failure faster with Redrive



Execution event history for easy observability

EASILY TRACK EVERY STEP OF YOUR WORKFLOW EXECUTION

Filter down to what you need

CloudWatch logs to dive deeper

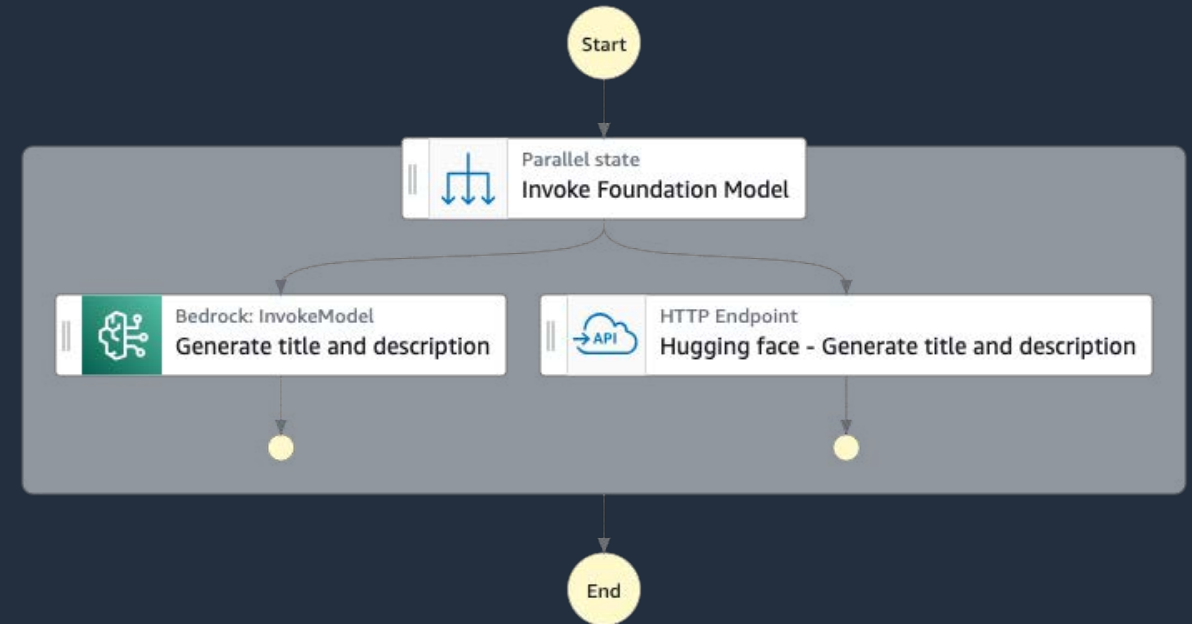
New ExecutionRedriven event

New Redrive count

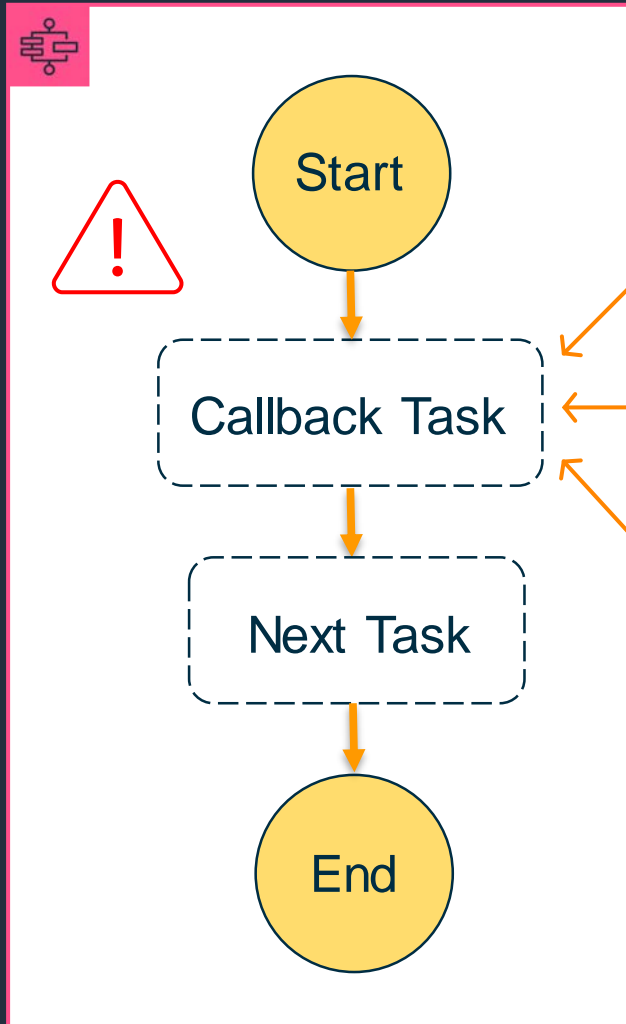
ID	Type	Step	Resource	Redrive attempt	Started After	Timestamp
▶ 13	TaskStarted	Transform More		-	00:00:00.098	Nov 12, 2023,
▶ 14	TaskFailed	Transform More		-	00:00:00.237	Nov 12, 2023,
▶ 15	WaitStateAborted			-	00:00:00.254	Nov 12, 2023,
▶ 16	TaskStateAborted			-	00:00:00.254	Nov 12, 2023,
▶ 17	ParallelStateFailed	Generate Multiple		-	00:00:00.254	Nov 12, 2023,
▶ 18	ExecutionFailed			-	00:00:00.283	Nov 12, 2023,
	ExecutionRedriven			#1	00:02:51.447	Nov 12, 2023,
▶ 20	TaskScheduled	Transform More	Lambda Log group	#1	00:02:51.472	Nov 12, 2023,
▶ 21	TaskStarted	Transform More		#1	00:02:51.558	Nov 12, 2023,
▶ 22	TaskSucceeded	Transform More		#1	00:02:51.671	Nov 12, 2023,
▶ 23	TaskStateExited	Transform More		#1	00:02:51.695	Nov 12, 2023,
▶ 24	WaitStateEntered	Completion step		#1	00:02:51.695	Nov 12, 2023,
▶ 26	WaitStateExited	Completion step		#1	00:02:56.754	Nov 12, 2023,
▶ 27	ParallelStateSucceeded	Generate Multiple		#1	00:02:56.754	Nov 12, 2023,

Building a generative AI application

- Create multiple titles and descriptions for video
- Ask human to provide feedback
- Create an avatar for the video



Wait for callback—human/external process



AWS Lambda



Amazon ECS



AWS Fargate



AWS Step Functions



Amazon SQS



Amazon SNS



Amazon API Gateway



Amazon EventBridge

Callback pattern

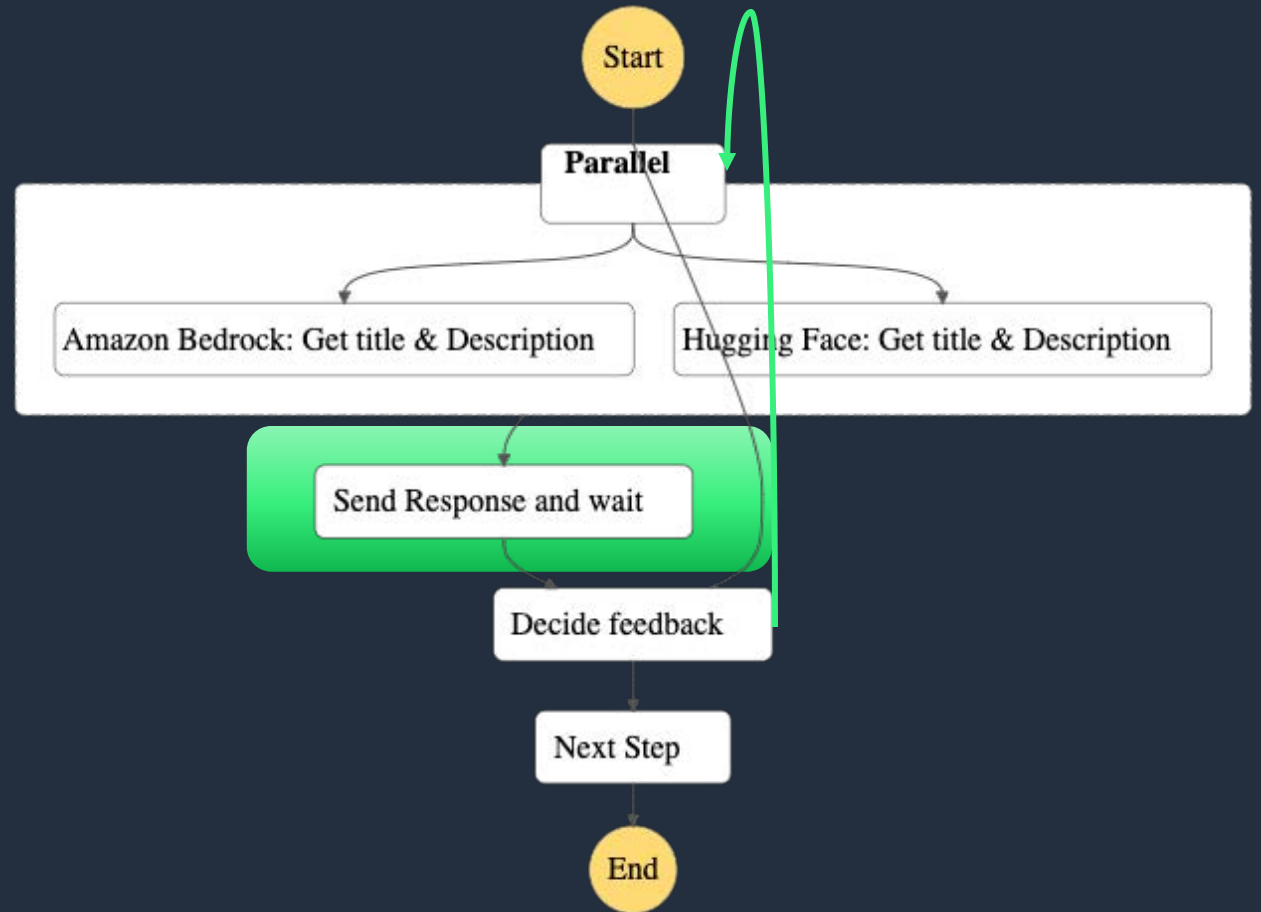
Call an external resource with a token. Pause workflow for a callback event with the token.

Wait for as long as you need—minutes, days, weeks, or months:

- Human activity
- Third-party API
- Legacy application

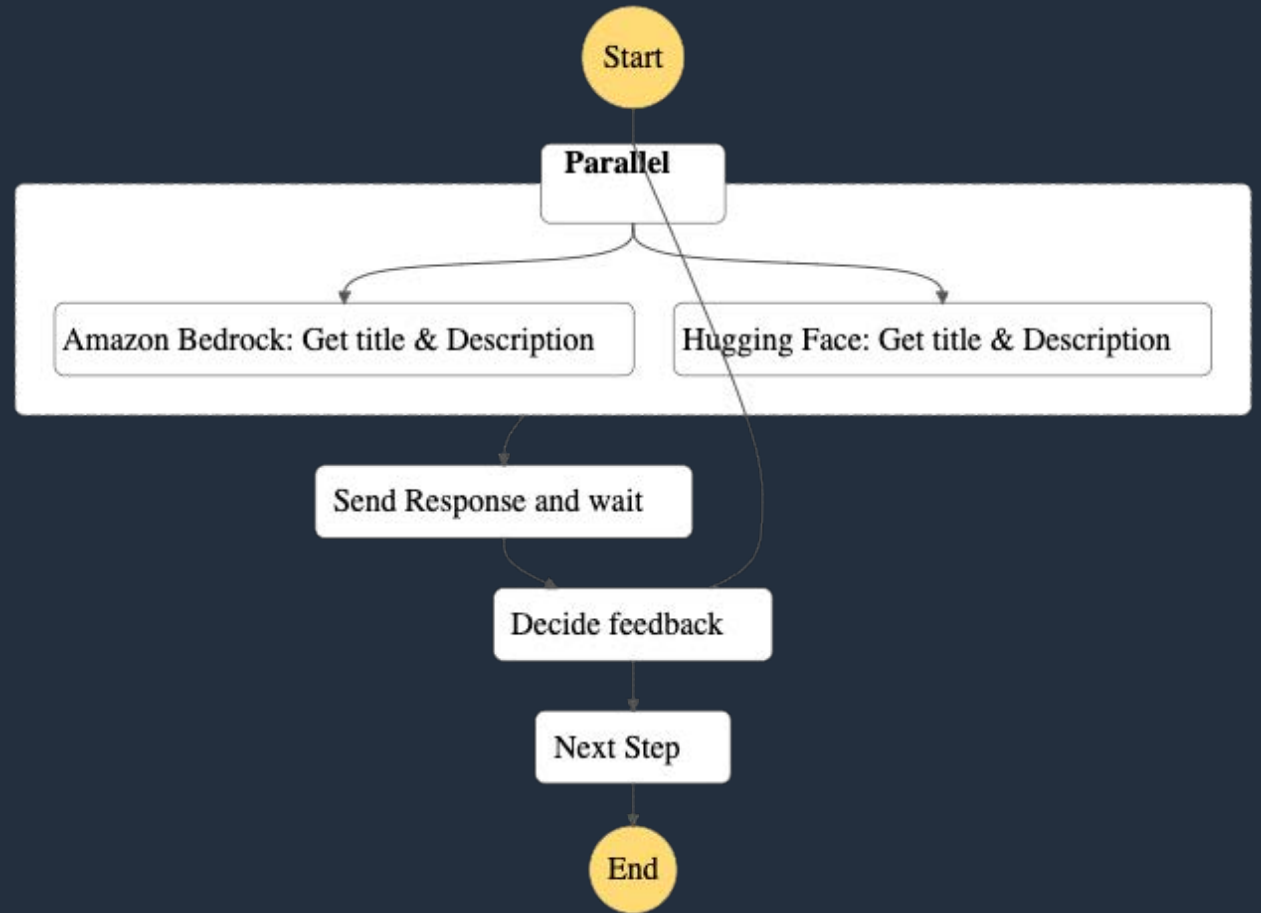
Requirement 2: Human feedback loop

- Feedback loop
 - Async channels—email, WebSocket
 - API to send response back
- Decisions using choice state
- Looping to regenerate

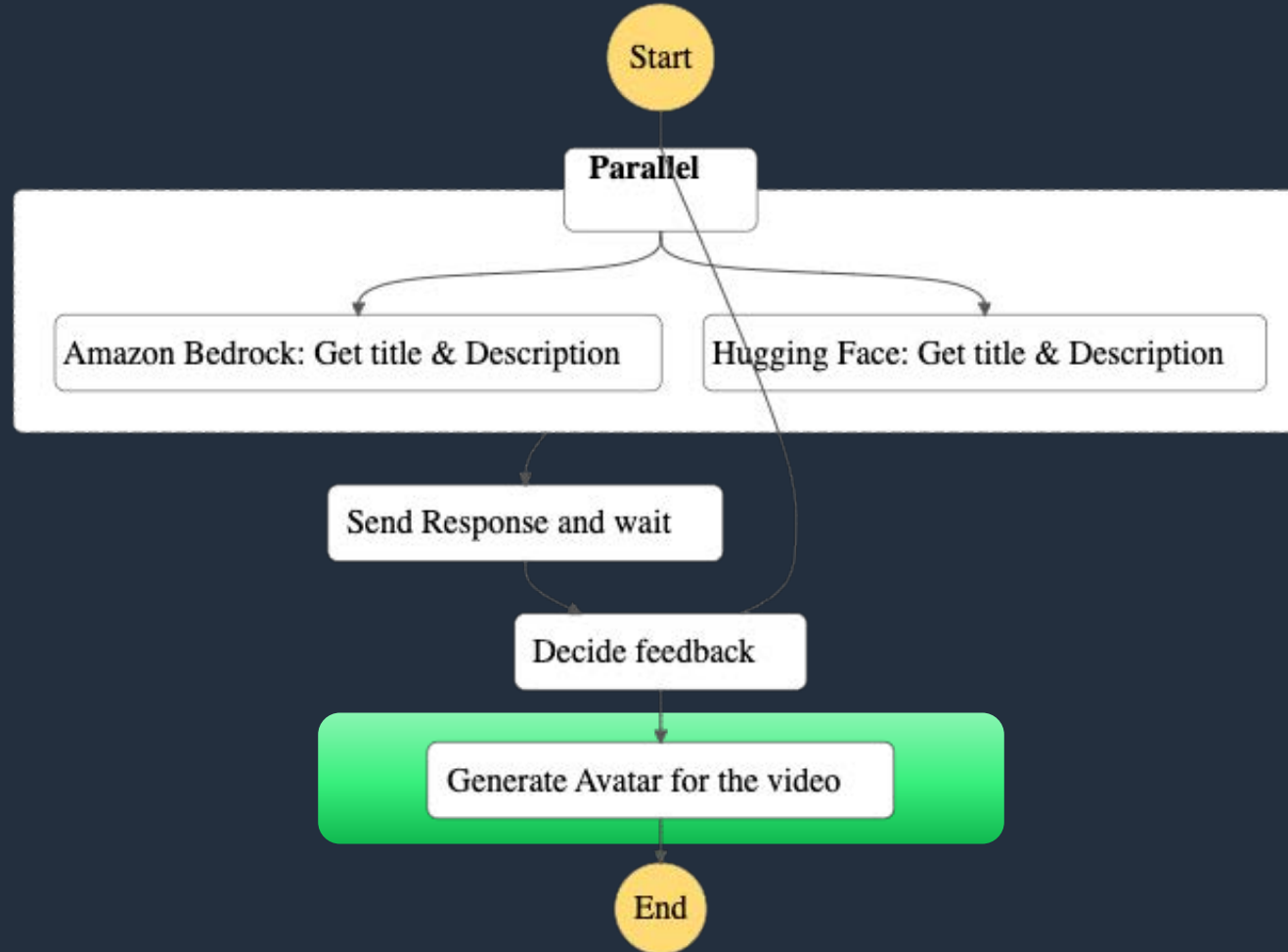


Building a generative AI application

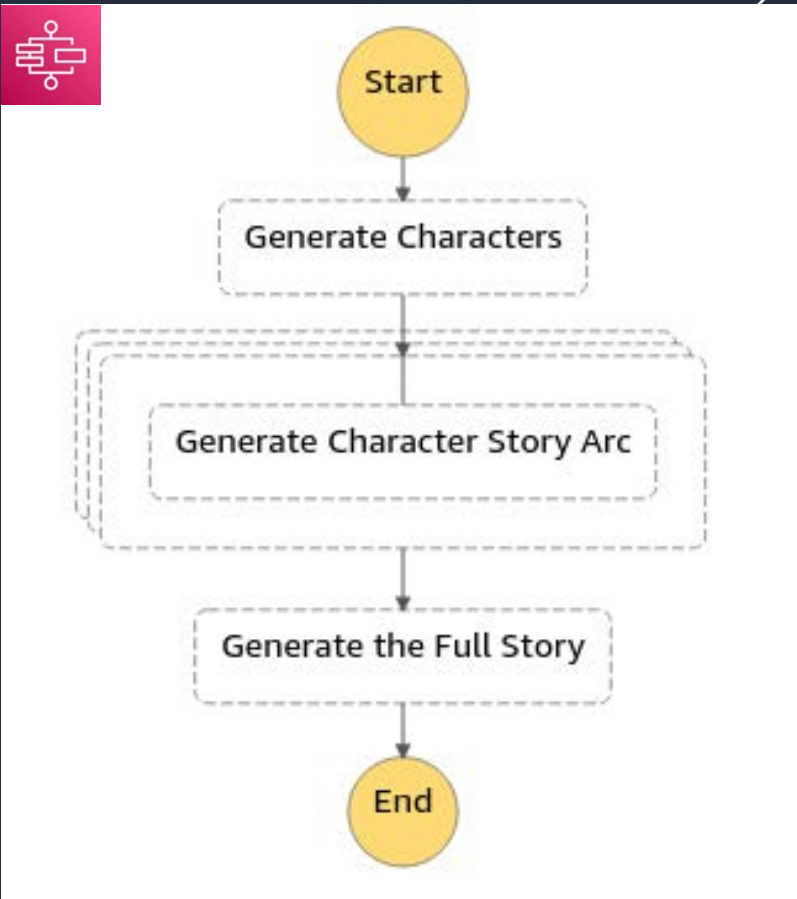
- Create multiple titles and descriptions for video
- Ask human to provide feedback
- Create an avatar for the video



Requirement 3: Generate avatar



Prompt chaining



You are an award-winning fiction writer, and you are writing a new story about {story_description}.

Before writing the story, describe five characters that will be in the story.

Your response should be formatted as a JSON array, with each element in the array containing a "name" key for the character's name and a "description" key with the character's description.

An example of a valid response is below, inside...



Amazon Bedrock
ANTHROP\Claude

Prompt chaining

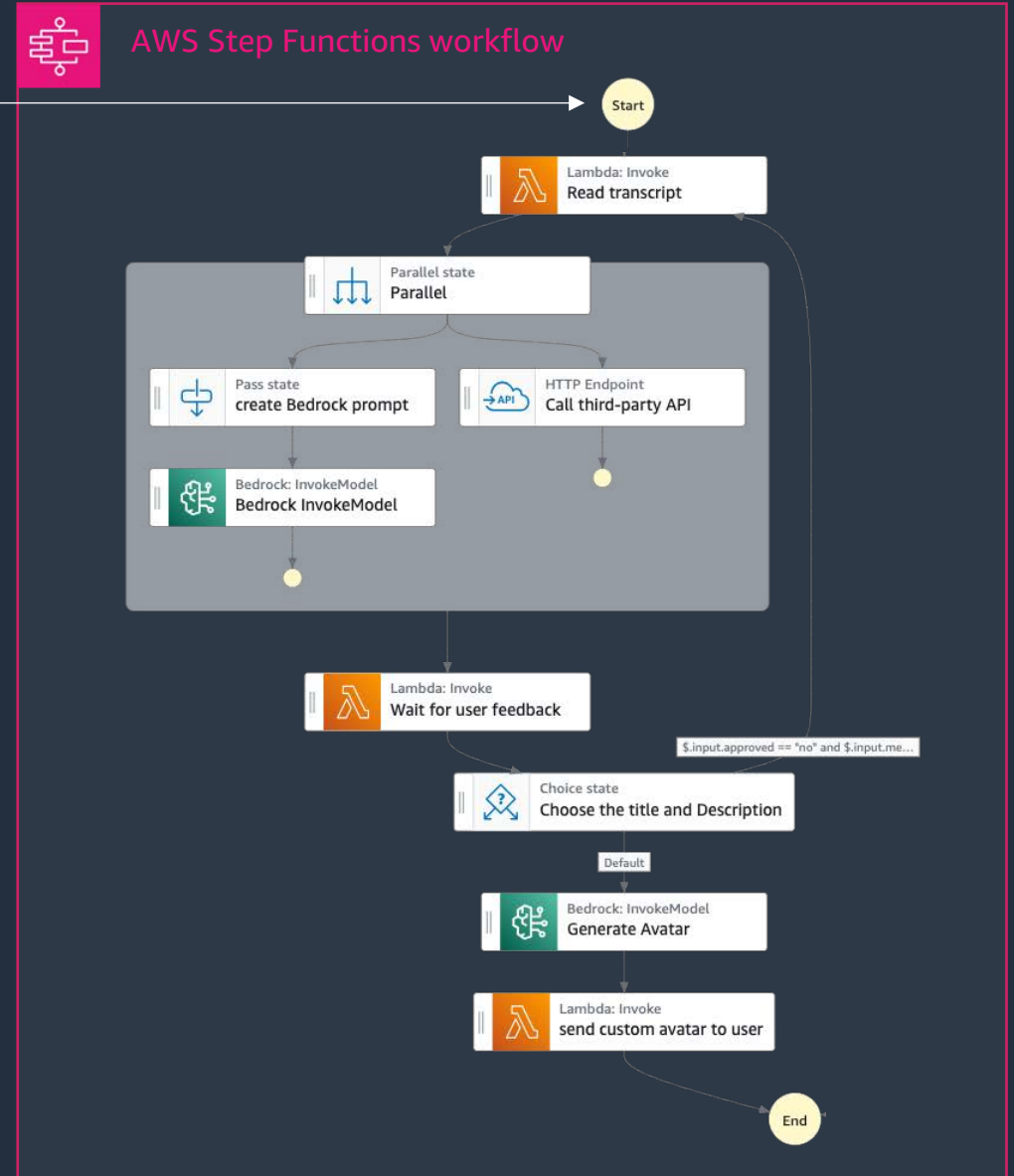
- Connect multiple prompts to generate complex content
- Feed response from one model to the next

Use cases

- Writing blogs and articles
- Response validation
- Conversational LLM

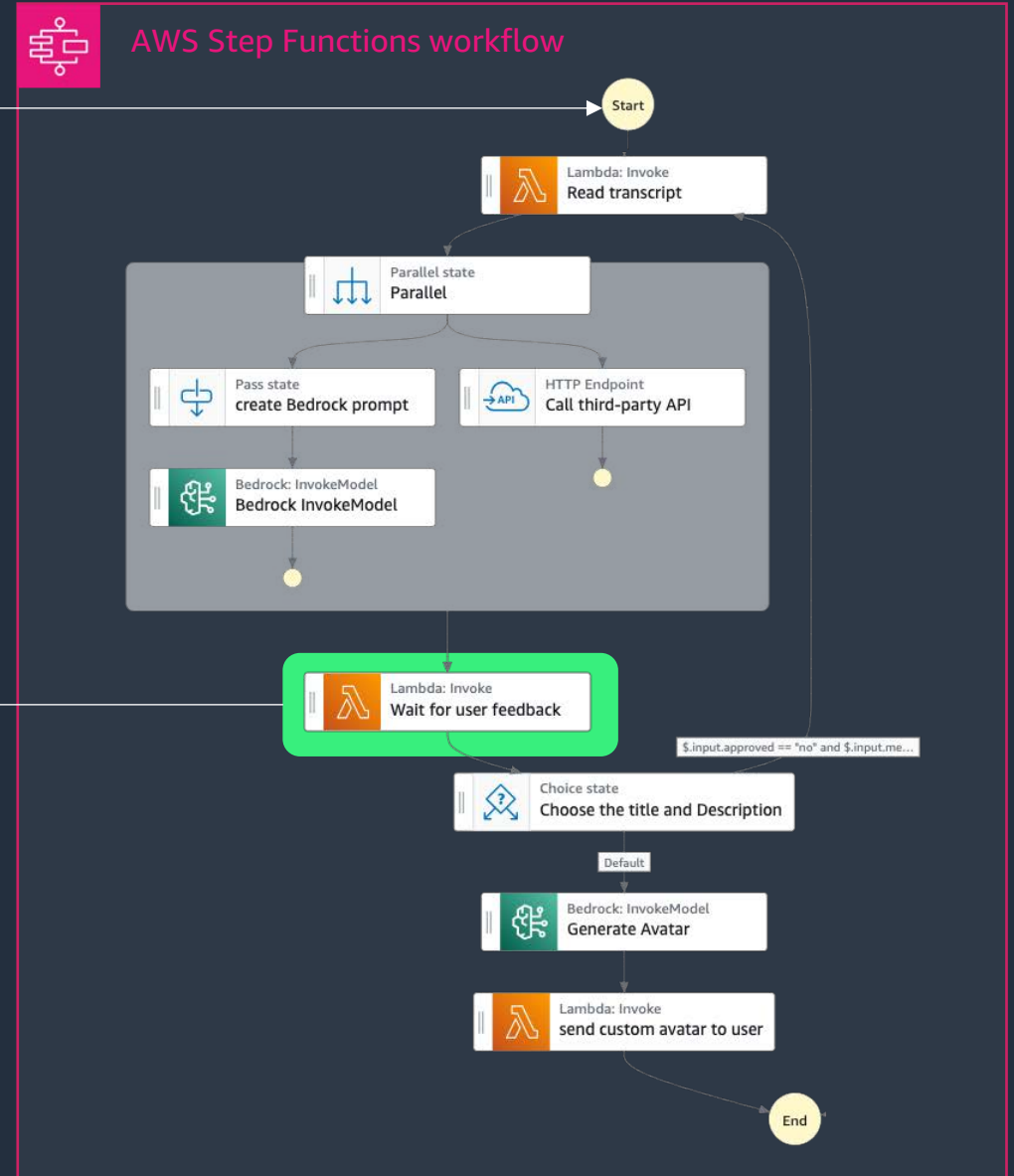
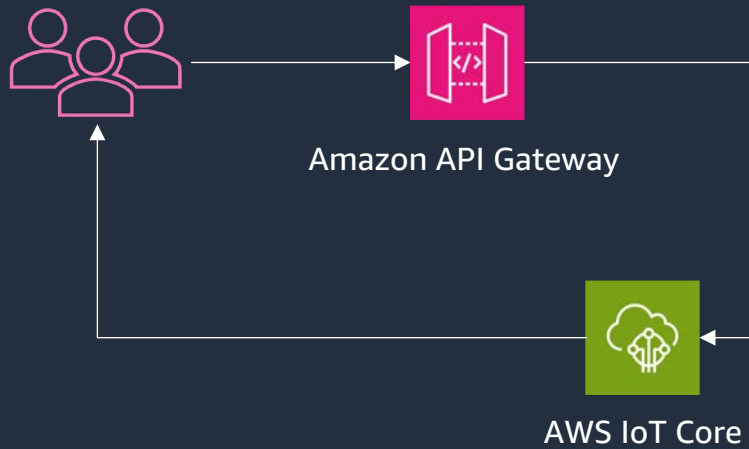
Architecture

- 1. Generate title, description...
- 2. Invoke workflow



Architecture

- 3. Send the title, description to user
- 4. Send the chosen title and description along with token



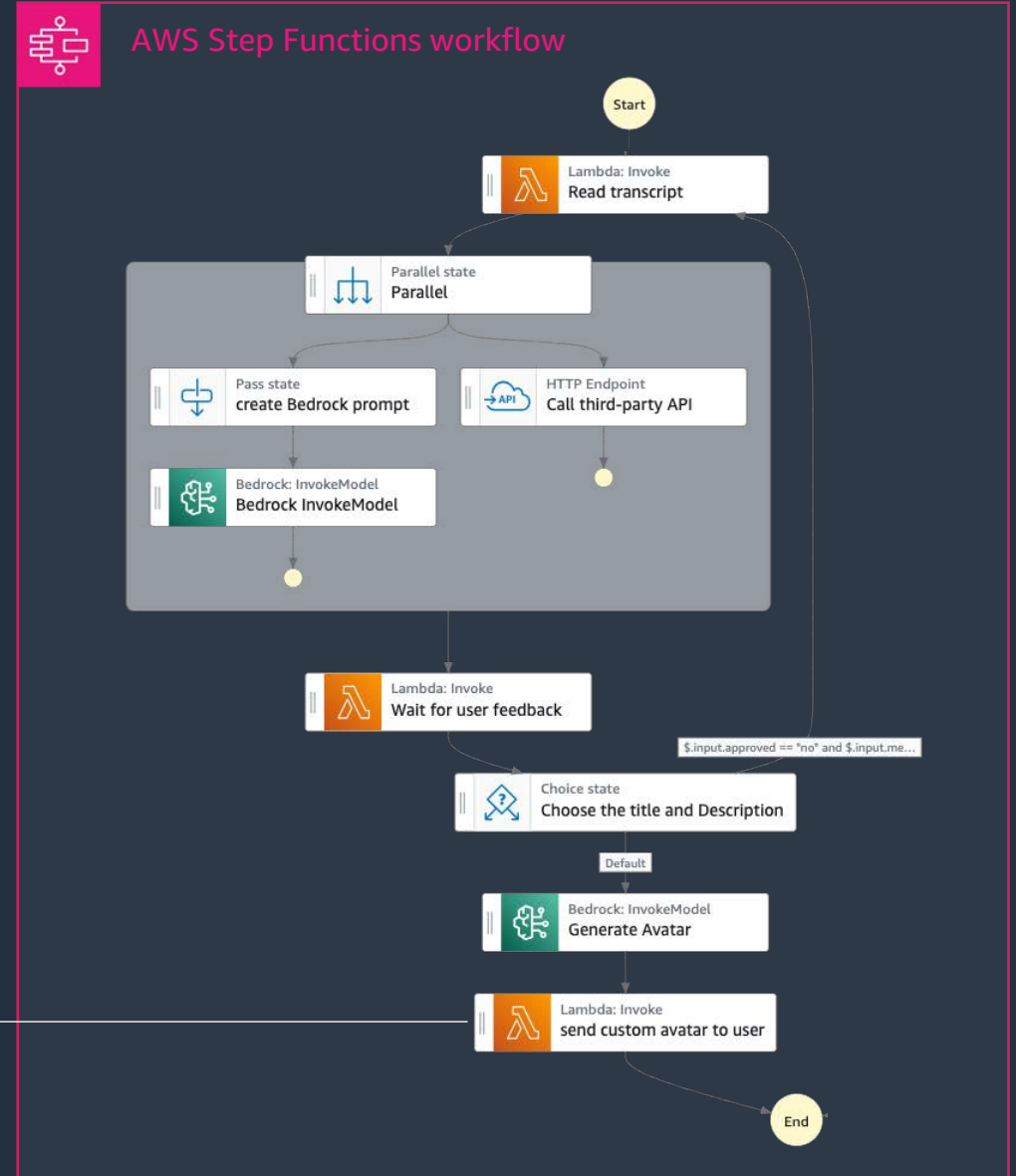
Architecture

5. Send S3 pre-signed URL of avatar to the user



AWS IoT Core

5



Step Functions - Elevating workflows with Generative AI



Generate title, description and avatar

Resources



<https://github.com/aws-samples/amazon-bedrock-serverless-prompt-chaining>



Thank you!

Mohammed Fazalullah Qudrath
Sr Developer Advocate, MENAT
AWS