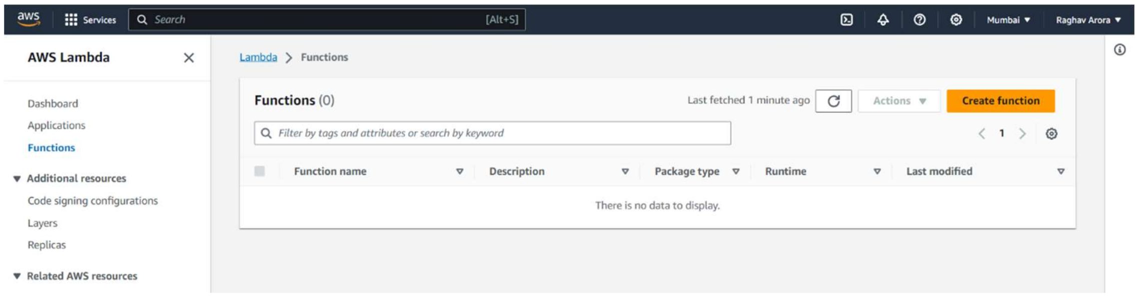
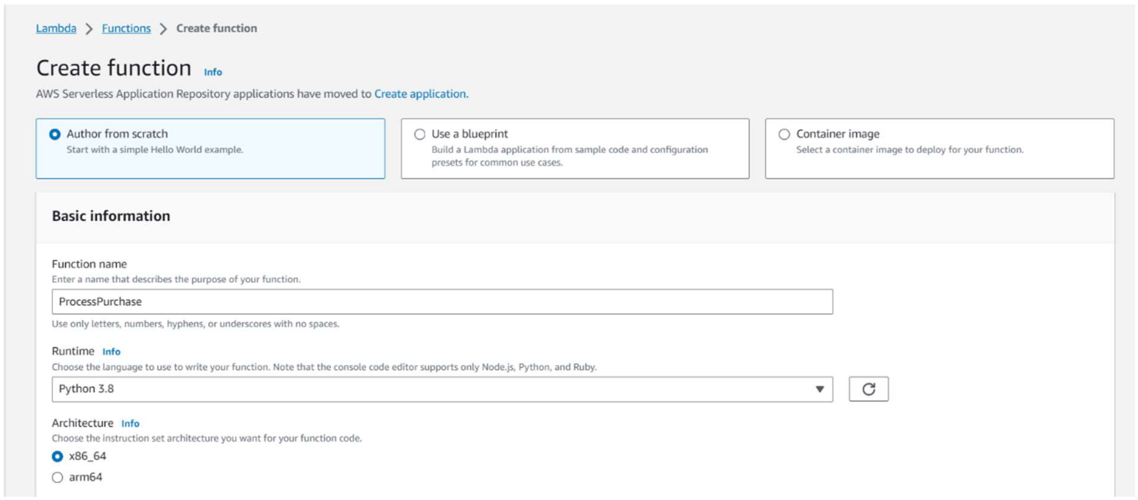
Codes: https://github.com/Raghavarora09/AWS-Step-Functions-with-Lambda.git

Create a new lambda function.



There is no data to display.

Select the runtime as Python 3.8


Lambda > Functions > Create function

Create function Info

AWS Serverless Application Repository applications have moved to Create application.

[✓ Author from scratch] Start with a simple Hello World example.

Use a blueprint: Build a Lambda application from sample code and configuration presets for common use cases.

Container image: Select a container image to deploy for your function.

Basic information

Function name: Enter a name that describes the purpose of your function.

ProcessPurchase

Use only letters, numbers, hyphens, or underscores with no spaces.

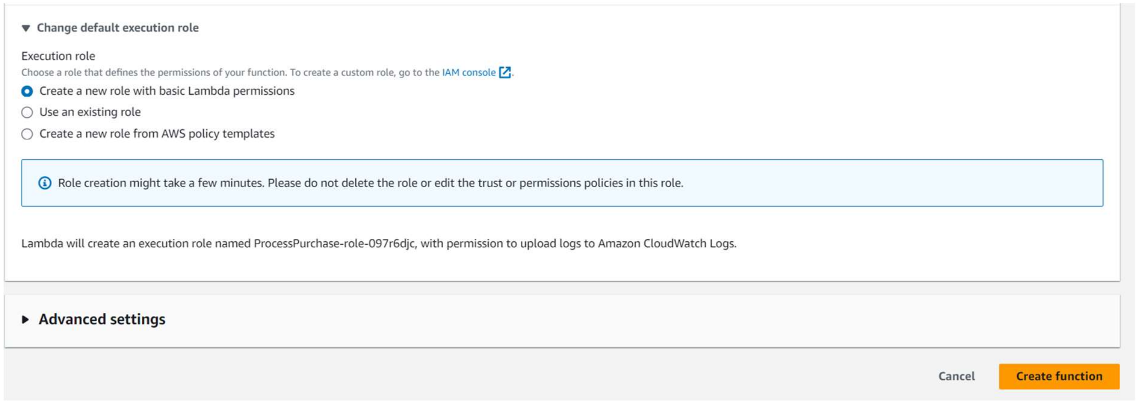
Runtime Info: Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Python 3.8

Architecture Info: Choose the instruction set architecture you want for your function code.

[✓ x86_64] arm64

Click on create.


Change default execution role

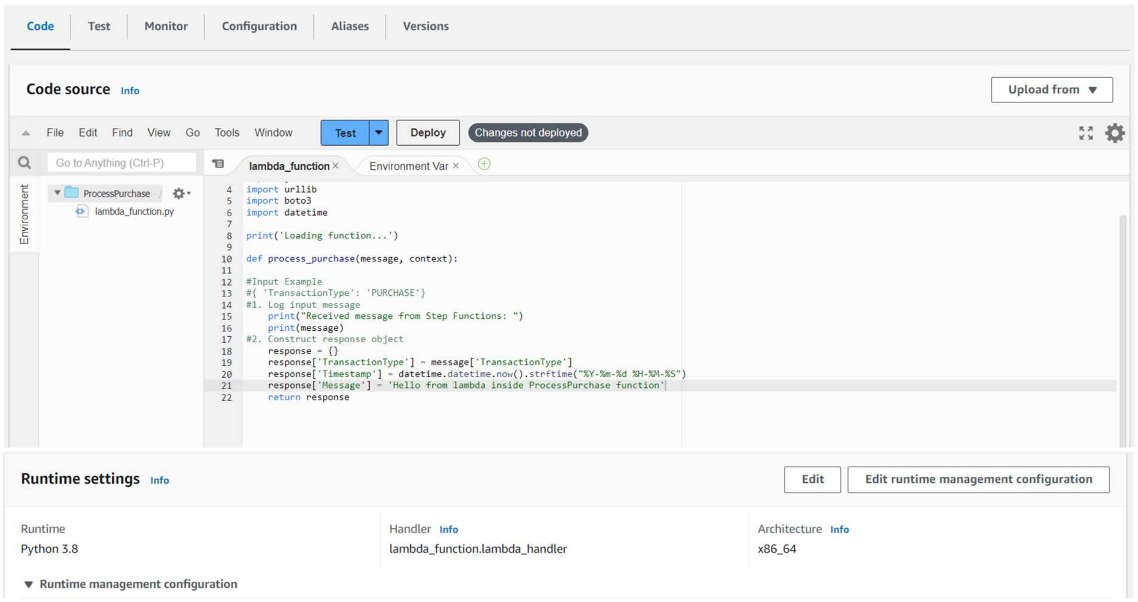
Execution role: Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.

[✓ Create a new role with basic Lambda permissions]
Use an existing role
Create a new role from AWS policy templates

Role creation might take a few minutes. Please do not delete the role or edit the trust or permissions policies in this role.

Lambda will create an execution role named ProcessPurchase-role-097r6djc, with permission to upload logs to Amazon CloudWatch Logs.

Add the lambda function code and in the runtime settings click on edit to change the lambda handler

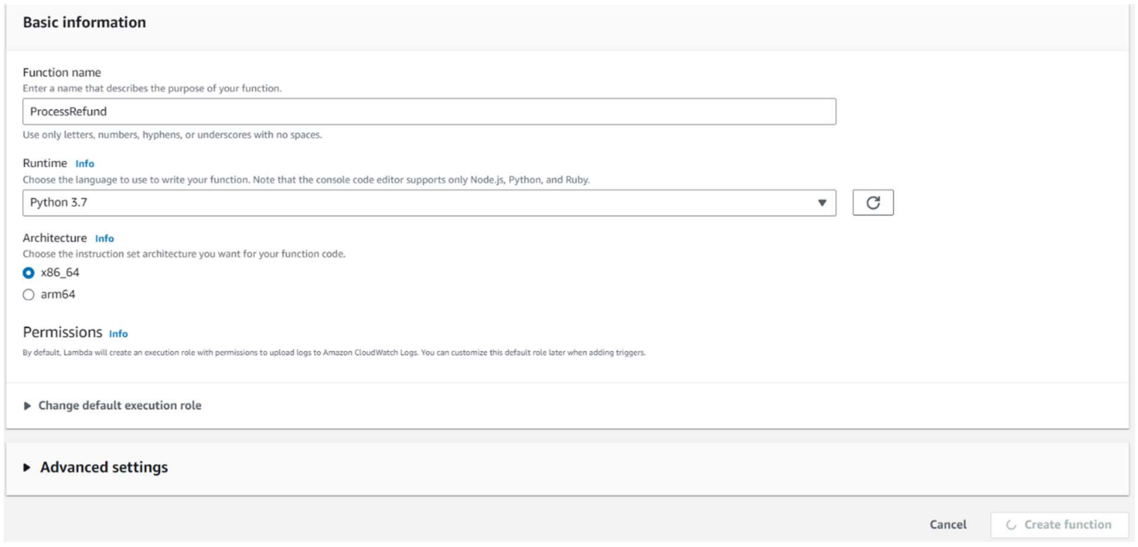


```
import urllib
import boto3
import datetime

print('Loading function...')

def process_purchase(message, context):

#Input Example
#{ 'TransactionType': 'PURCHASE'}
#1. Log input message
    print("Received message from Step Functions: ")
    print(message)
#2. Construct response object
    response = {}
    response['TransactionType'] = message['TransactionType']
    response['Timestamp'] = datetime.datetime.now().strftime("%Y-%m-%d %H-%M-%S")
    response['Message'] = 'Hello from lambda inside ProcessPurchase function'
    return response
```

Runtime	Handler	Architecture
Python 3.8	lambda_function.lambda_handler	x86_64

Give the handler name as lambda_function.process_purchase(which will be same as the function in the lambda code) and save it.

Now create another lambda function and name it ProcessRefund



Function name
Enter a name that describes the purpose of your function.
ProcessRefund
Use only letters, numbers, hyphens, or underscores with no spaces.

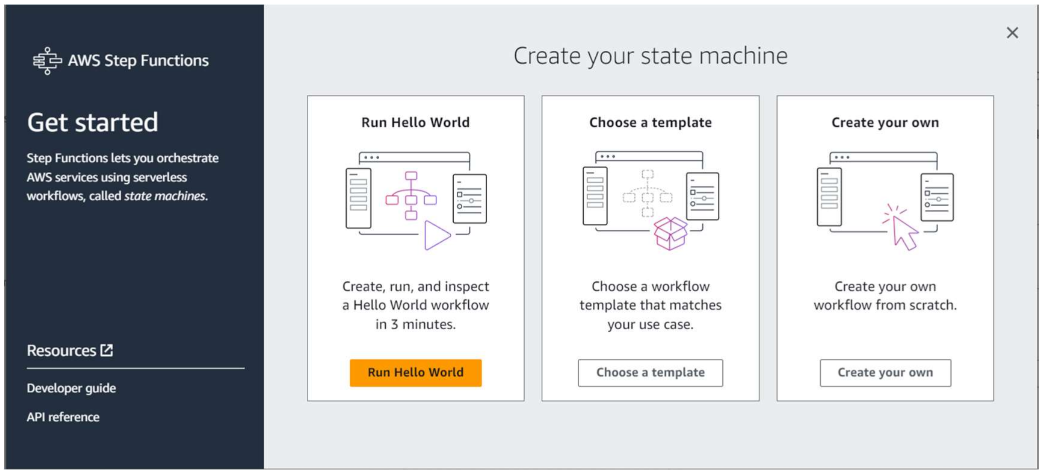
Runtime Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.7

Architecture Info
Choose the instruction set architecture you want for your function code.
[✓ x86_64]
arm64

Permissions Info
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

And add the lambda code and change the lambda handler in the same way.

Now go to Step Functions and select Create your own template.



Step Functions lets you orchestrate AWS services using serverless workflows, called *state machines.*

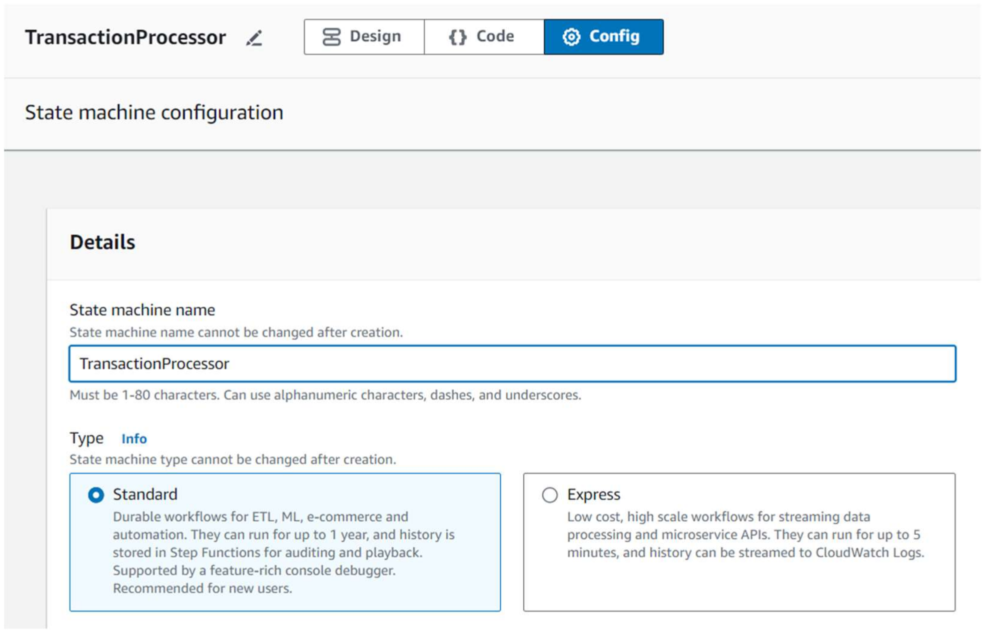
Create your state machine

Create, run, and inspect a Hello World workflow in 3 minutes.

Choose a workflow template that matches your use case.

Create your own workflow from scratch.

Give the name as TransactionProcessor



TransactionProcessor

State machine configuration

Details

State machine name

State machine name cannot be changed after creation.

TransactionProcessor

Must be 1-80 characters. Can use alphanumeric characters, dashes, and underscores.

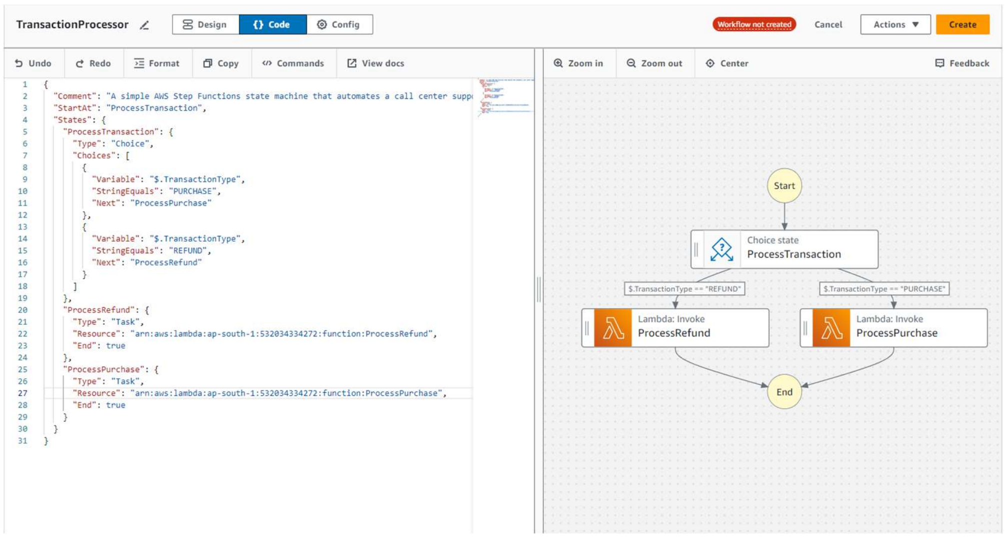
Type

State machine type cannot be changed after creation.

[✓ Standard] Durable workflows for ETL, ML, e-commerce and automation. They can run for up to 1 year, and history is stored in Step Functions for auditing and playback. Supported by a feature-rich console debugger. Recommended for new users.

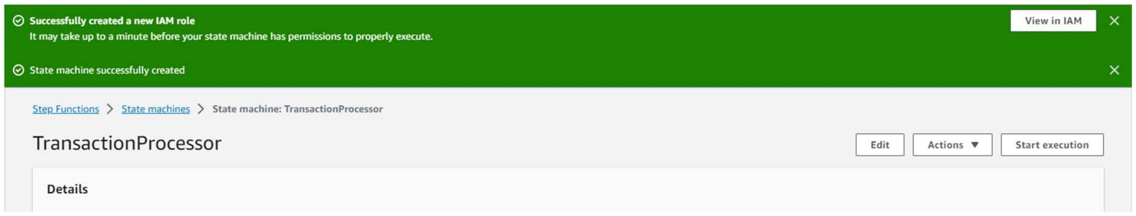
Express Low cost, high scale workflows for streaming data processing and microservice APIs. They can run for up to 5 minutes, and history can be streamed to CloudWatch Logs.

Add the code and replace the resources with the respective ARN's of lambda functions.



```
{
  "Comment": "A simple AWS Step Functions state machine that automates a call center supp
  "StartAt": "ProcessTransaction",
  "States": {
    "ProcessTransaction": {
      "Type": "Choice",
      "Choices": [
        {
          "Variable": "$.TransactionType",
          "StringEquals": "PURCHASE",
          "Next": "ProcessPurchase"
        },
        {
          "Variable": "$.TransactionType",
          "StringEquals": "REFUND",
          "Next": "ProcessRefund"
        }
      ]
    },
    "ProcessRefund": {
      "Type": "Task",
      "Resource": "arn:aws:lambda:ap-south-1:532034334272:function:ProcessRefund",
      "End": true
    },
    "ProcessPurchase": {
      "Type": "Task",
      "Resource": "arn:aws:lambda:ap-south-1:532034334272:function:ProcessPurchase",
      "End": true
    }
  }
}
```

And click on Create



Successfully created a new IAM role
It may take up to a minute before your state machine has permissions to properly execute.

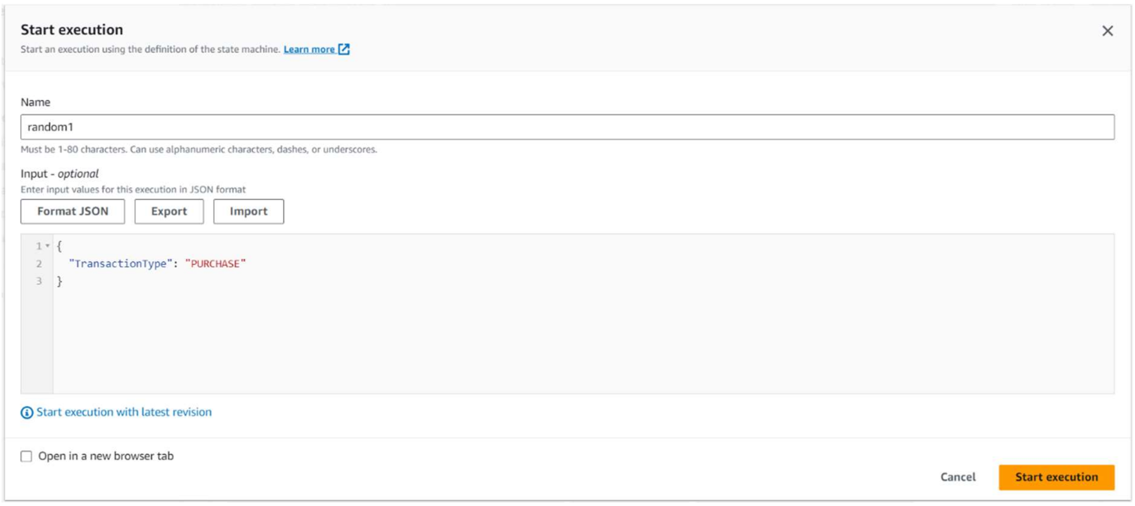
State machine successfully created

Step Functions > State machines > State machine: TransactionProcessor

TransactionProcessor

Details

Now start execution and give the input



Start execution

Start an execution using the definition of the state machine. Learn more

Name

random1

Must be 1-80 characters. Can use alphanumeric characters, dashes, or underscores.

Input - *optional*

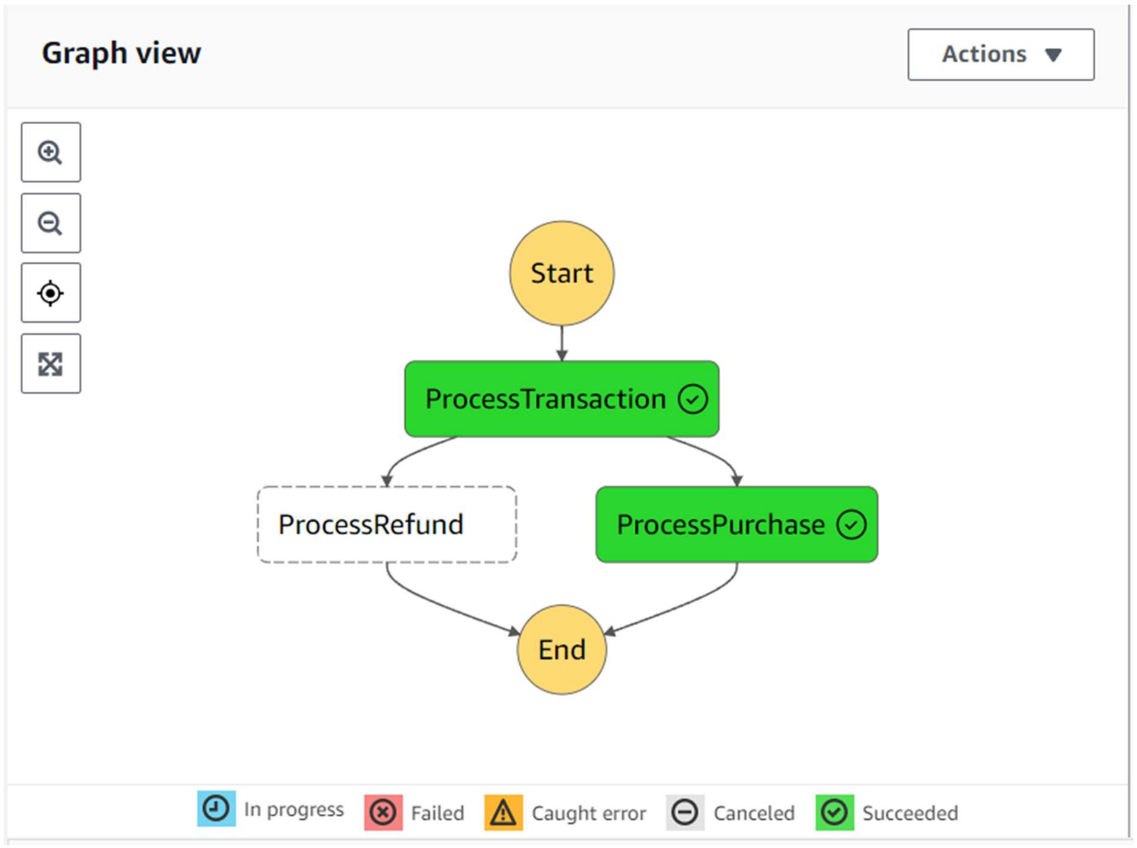
Enter input values for this execution in JSON format

```
{
  "TransactionType": "PURCHASE"
}
```

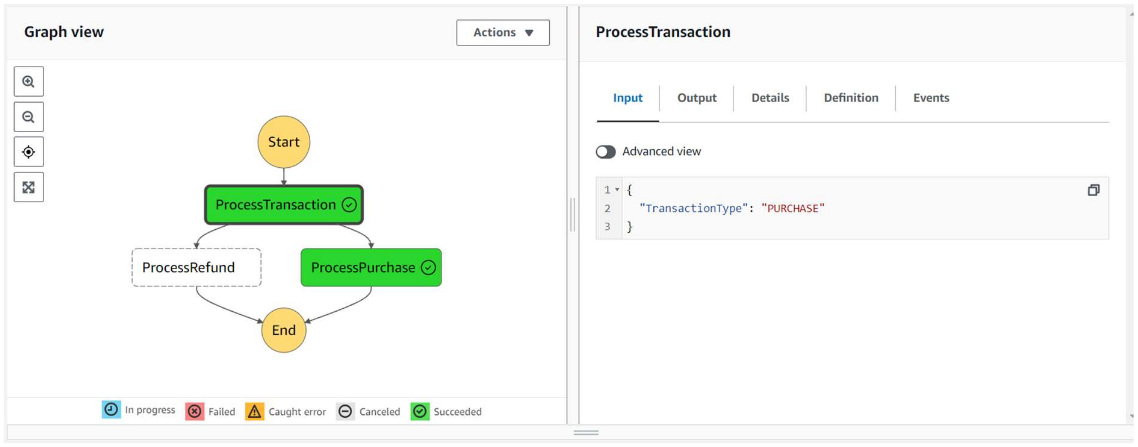
Start execution with latest revision

☐ Open in a new browser tab

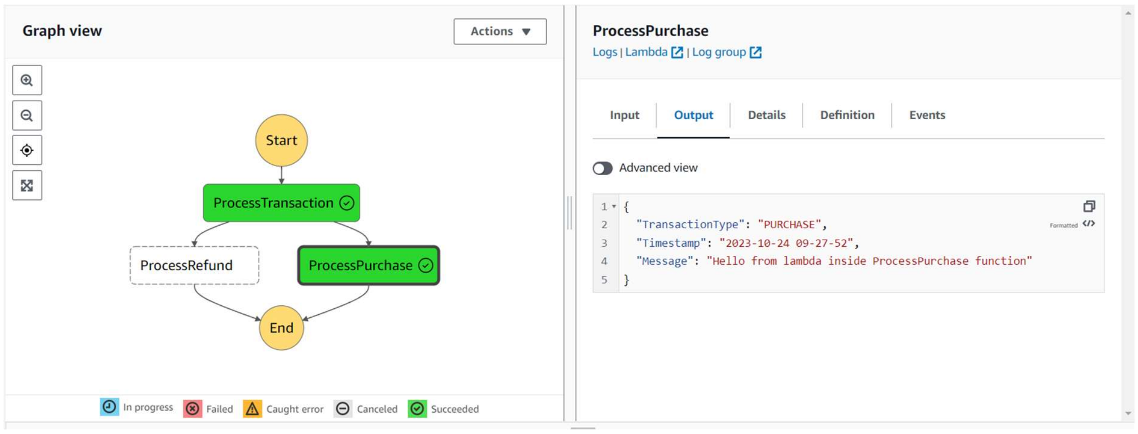
The graph will be highlighted as to which state is being used.



You can select a state to get the information as to what’s the Input and Output and other details.

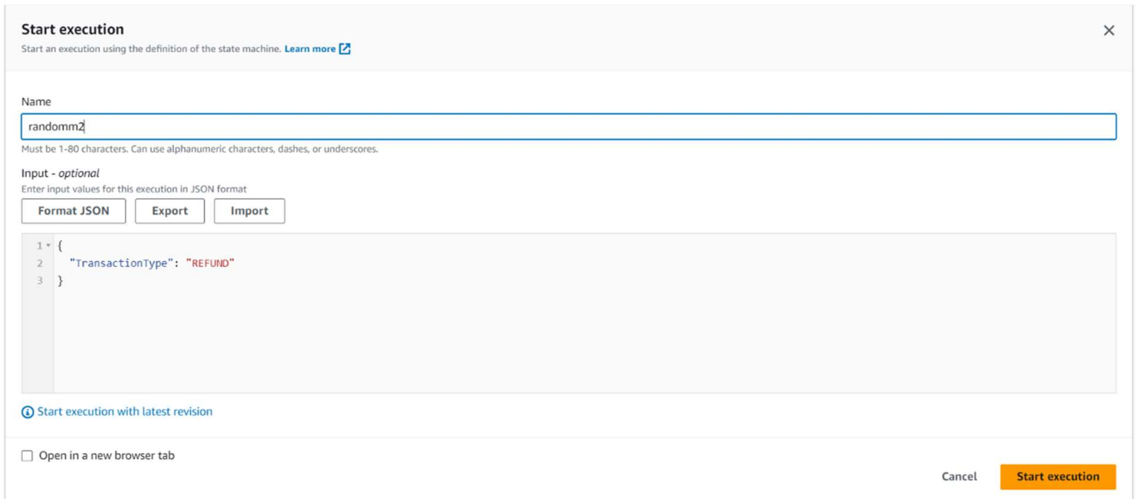


```
{
  "TransactionType": "PURCHASE"
}
```



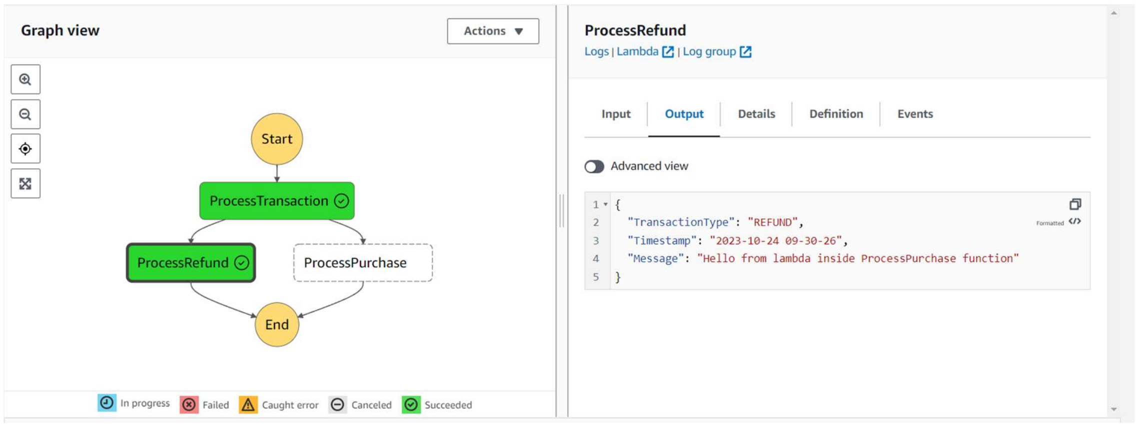
```
{
  "TransactionType": "PURCHASE",
  "Timestamp": "2023-10-24 09-27-52",
  "Message": "Hello from lambda inside ProcessPurchase function"
}
```

Now do the same for REFUND input.


Start execution
Start an execution using the definition of the state machine. Learn more
Name
Must be 1-80 characters. Can use alphanumeric characters, dashes, or underscores.
Input - optional
Enter input values for this execution in JSON format
Format JSON
Export
Import
Start execution with latest revision
Open in a new browser tab
Cancel
Start execution

Name: randomm2

```
{
  "TransactionType": "REFUND"
}
```



```
{
  "TransactionType": "REFUND",
  "Timestamp": "2023-10-24 09-30-26",
  "Message": "Hello from lambda inside ProcessPurchase function"
}
```