AWS Lambda Connected System

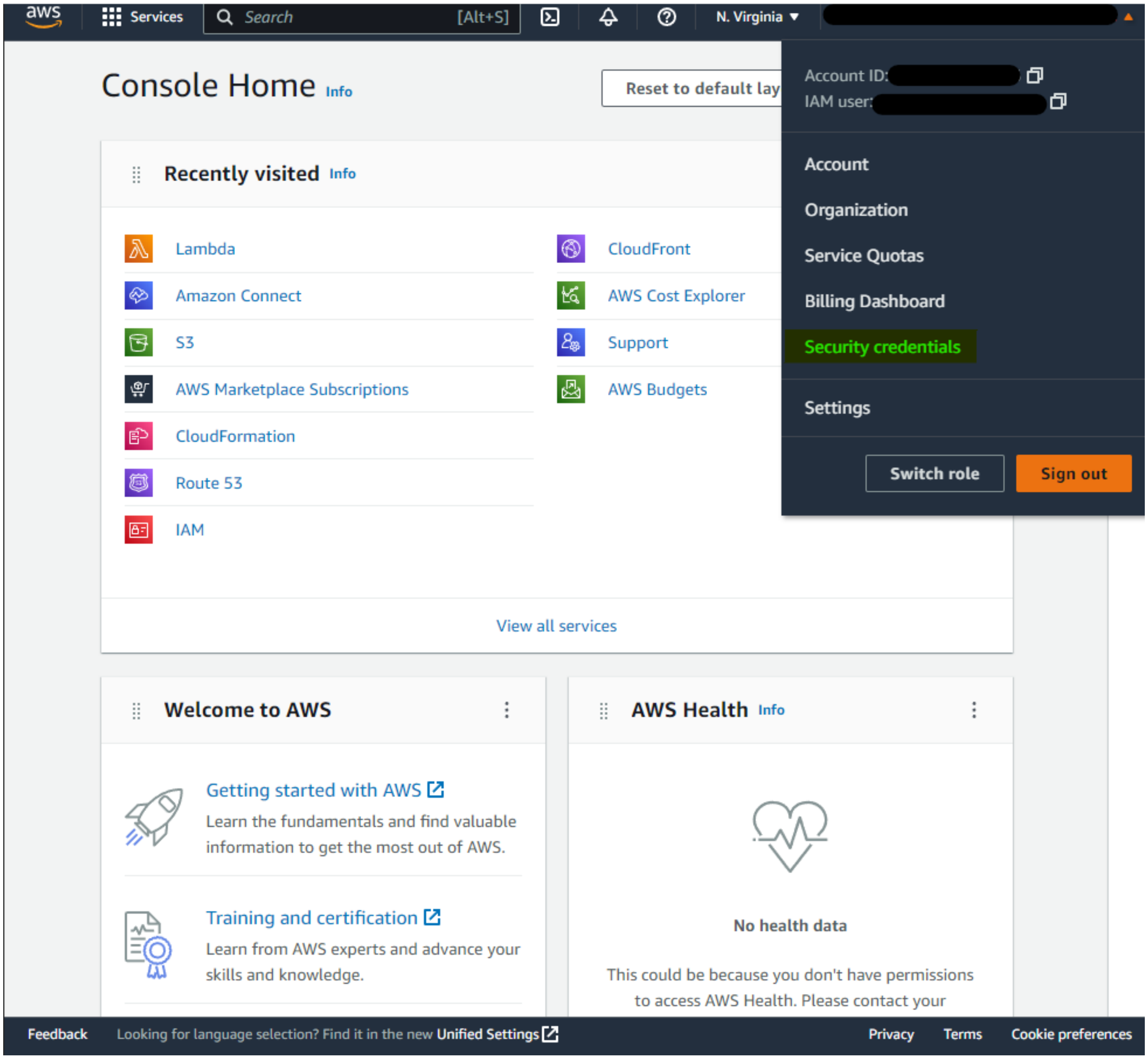
Overview

The AWS Lambda plugin provides the integrations to invoke and list the available functions in the AWS Lambda service in the AWS instance. To connect with the AWS instance, the connected system must be authenticated with the Access Key ID and the Secret Access Key.

Creating an Access Key in AWS Instance

1. Login to your AWS Console.

2. Click on your username and in the appearing pop up select Security Credentials.



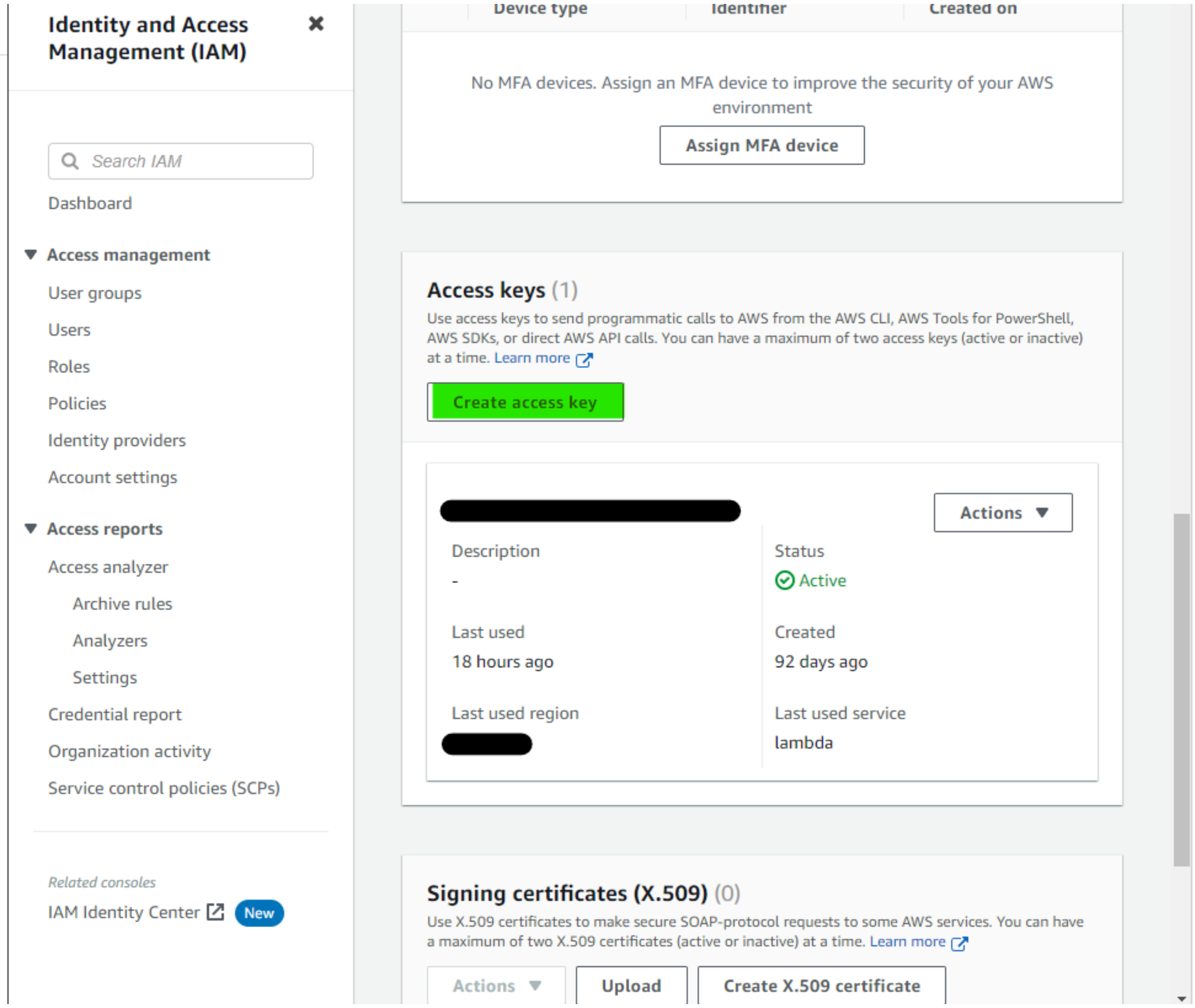
Learn the fundamentals and find valuable information to get the most out of AWS.

Learn from AWS experts and advance your skills and knowledge.

This could be because you don't have permissions to access AWS Health. Please contact your

Looking for language selection? Find it in the new Unified Settings

3. In the Security credential window, scroll down to the Access keys section and click on the Create Access Key button to create a new one.



No MFA devices. Assign an MFA device to improve the security of your AWS environment

Access keys (1)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. Learn more

[redacted]

Description
-

Status
✓ Active

Last used
18 hours ago

Created
92 days ago

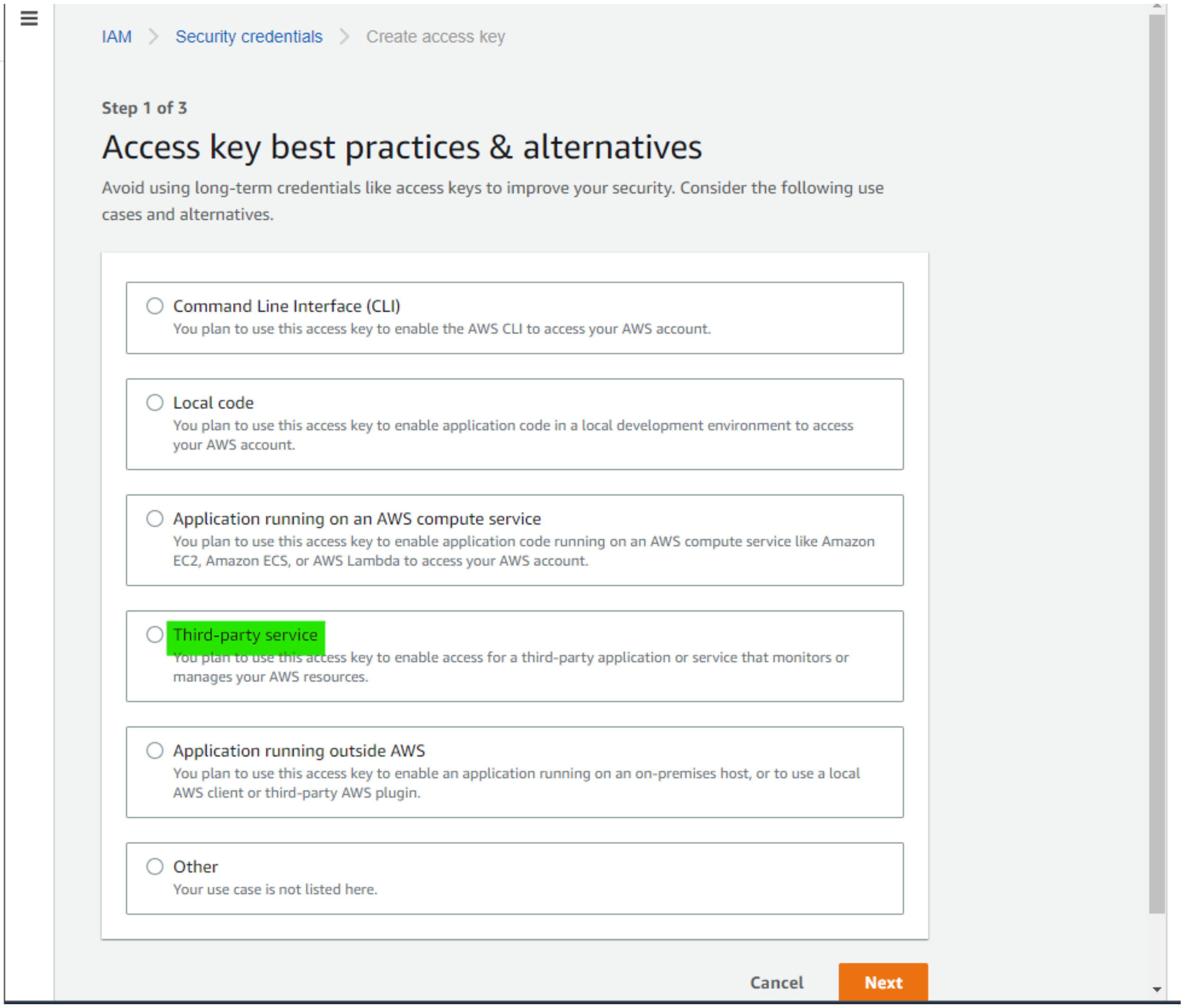
Last used region
[redacted]

Last used service
lambda

Signing certificates (X.509) (0)

Use X.509 certificates to make secure SOAP-protocol requests to some AWS services. You can have a maximum of two X.509 certificates (active or inactive) at a time. Learn more

4. In the appearing list of options, select Third-party service and click on Next.



Step 1 of 3

Access key best practices & alternatives

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

○ Command Line Interface (CLI)
You plan to use this access key to enable the AWS CLI to access your AWS account.

○ Local code
You plan to use this access key to enable application code in a local development environment to access your AWS account.

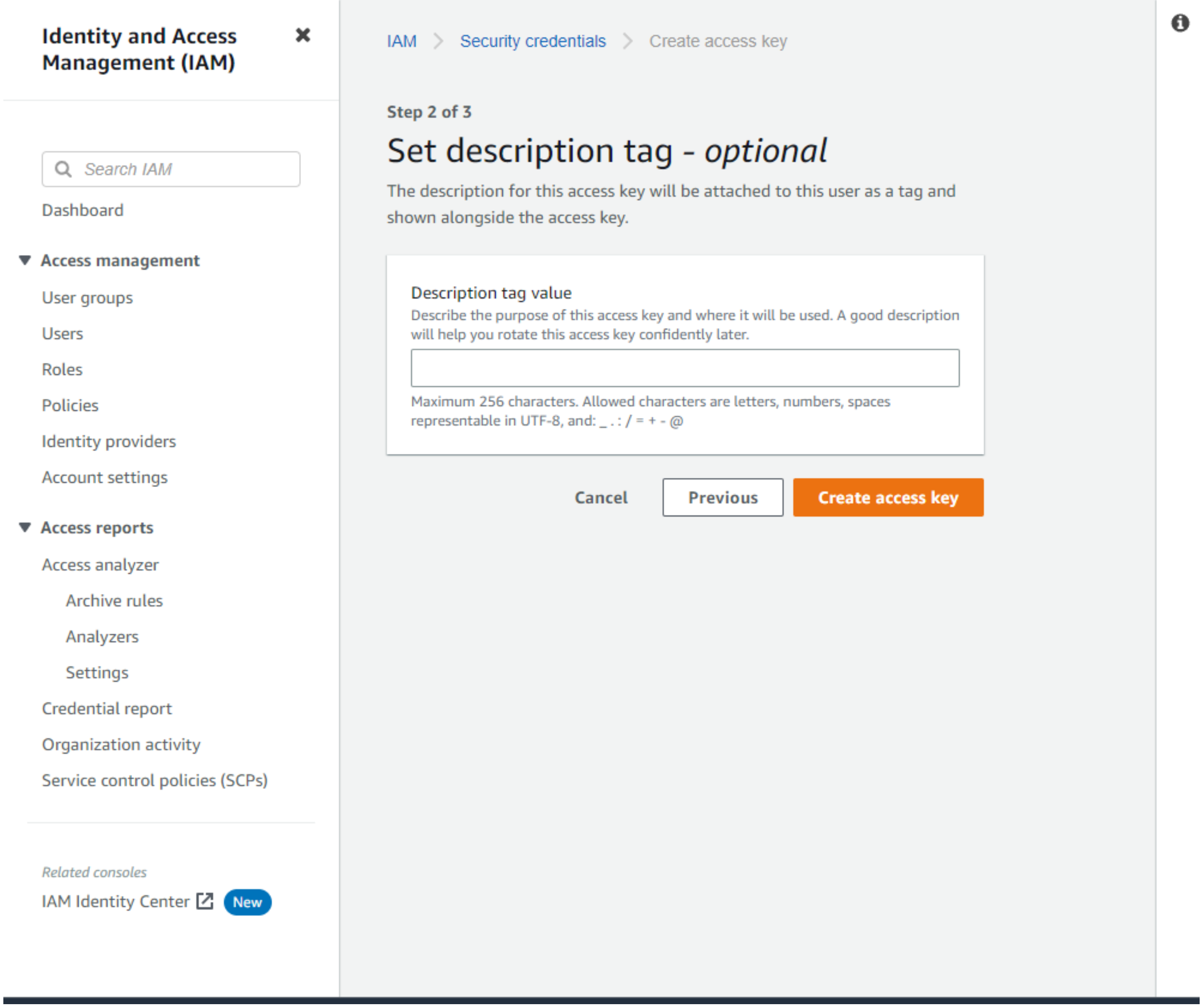
○ Application running on an AWS compute service
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

○ Third-party service
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.

○ Application running outside AWS
You plan to use this access key to enable an application running on an on-premises host, or to use a local AWS client or third-party AWS plugin.

○ Other
Your use case is not listed here.

5. In the next window, provide a description tag. This is an optional step. Click on Create Access key to proceed.



Step 2 of 3

Set description tag - *optional*

The description for this access key will be attached to this user as a tag and shown alongside the access key.

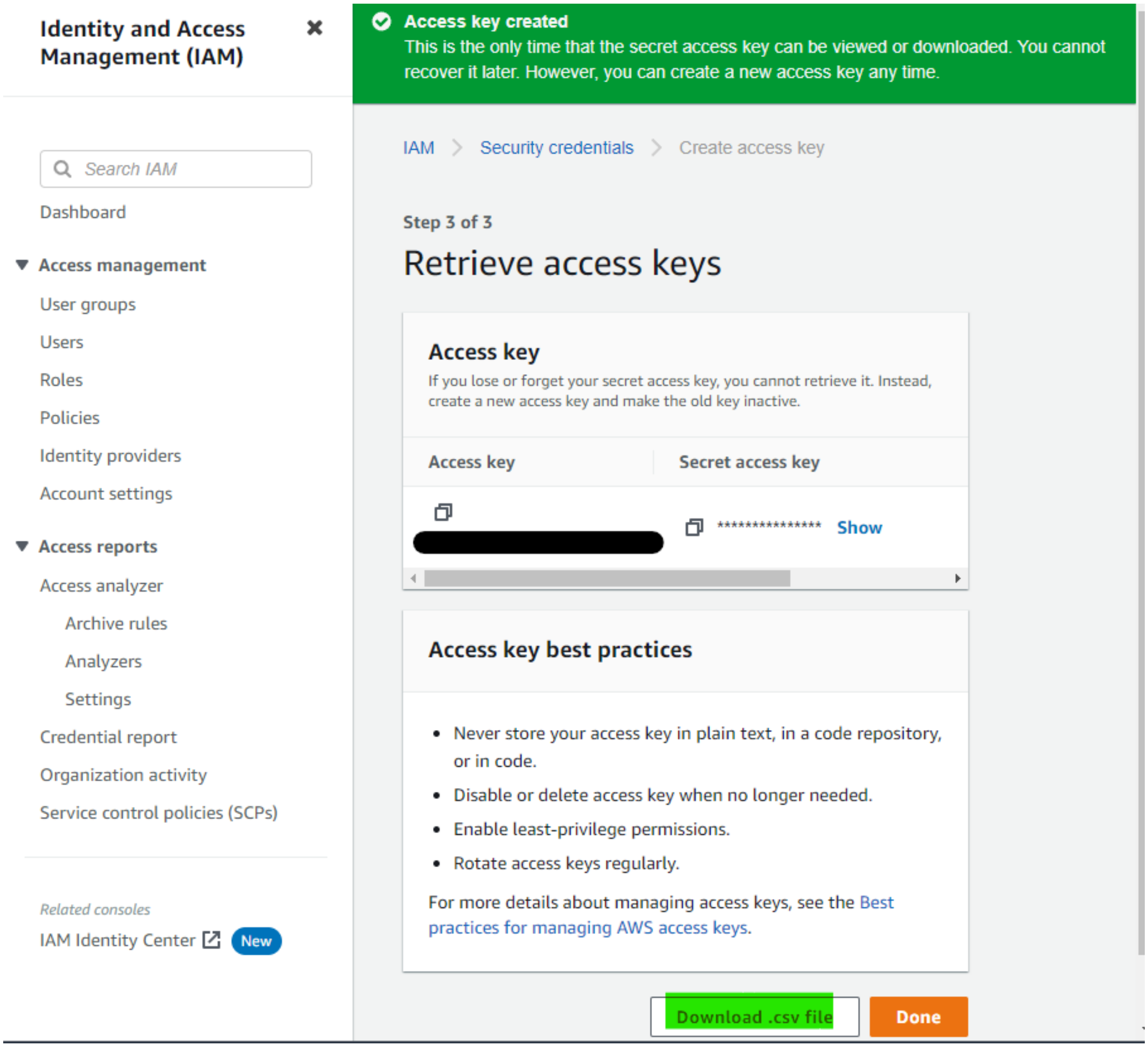
Description tag value

Describe the purpose of this access key and where it will be used. A good description will help you rotate this access key confidently later.

Maximum 256 characters. Allowed characters are letters, numbers, spaces representable in UTF-8, and: _ . : / = + - @

6. The Access Key ID and the Secret access key will be displayed. The credentials can also be downloaded as a CSV file by clicking on the Download .csv file button. Please make sure that the credentials are noted which will not be visible again once the Done

button is clicked.



Access key created

This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.

Step 3 of 3

Retrieve access keys

Access key

If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key	Secret access key
[redacted]	*************** Show

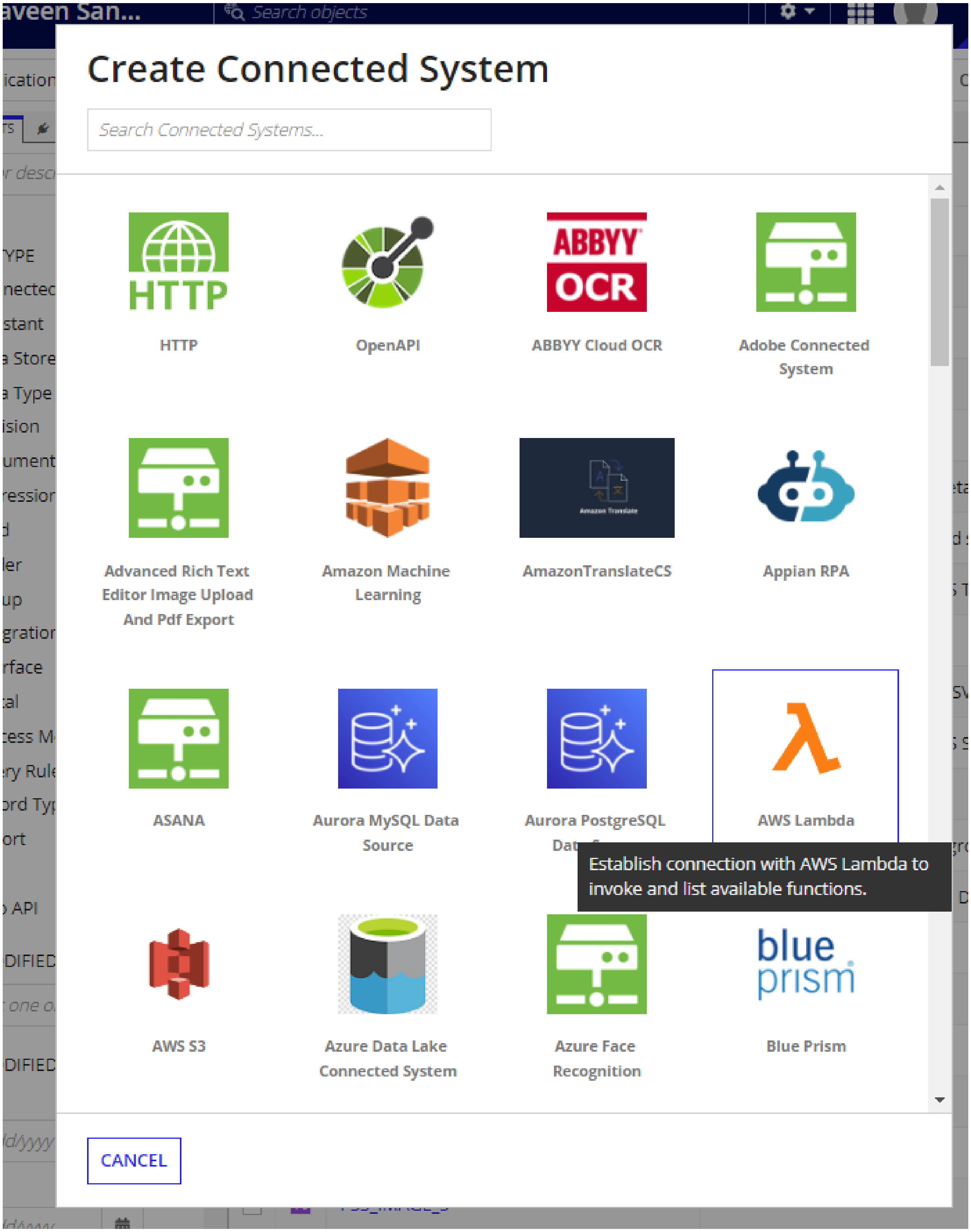
Access key best practices

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the Best practices for managing AWS access keys.

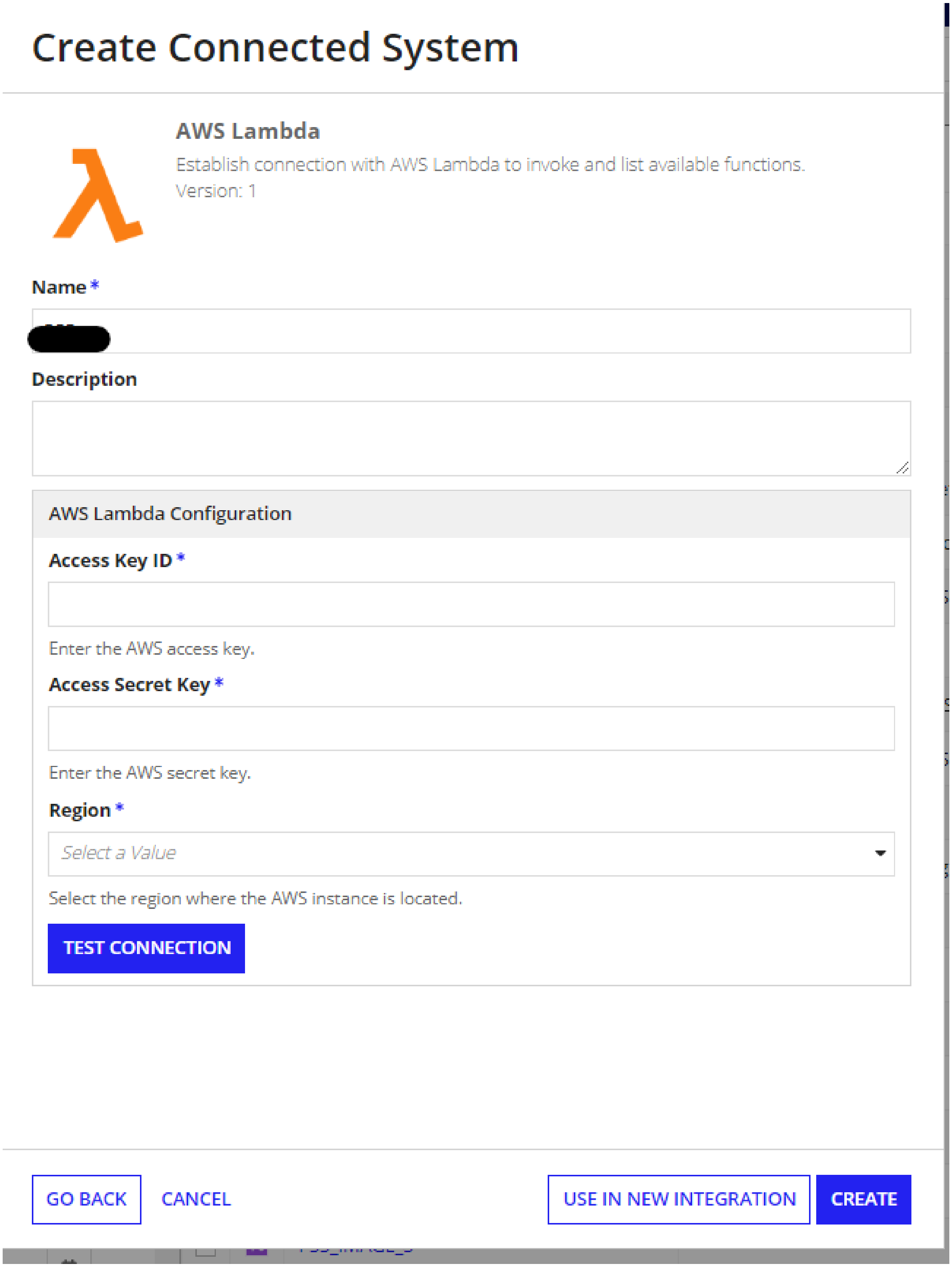
7. Make sure that the Access key is active.

Connected System



Create Connected System

Establish connection with AWS Lambda to invoke and list available functions.



Create Connected System

AWS Lambda

Establish connection with AWS Lambda to invoke and list available functions.
Version: 1

Name *

[redacted]

Description

AWS Lambda Configuration

Access Key ID *

Enter the AWS access key.

Access Secret Key *

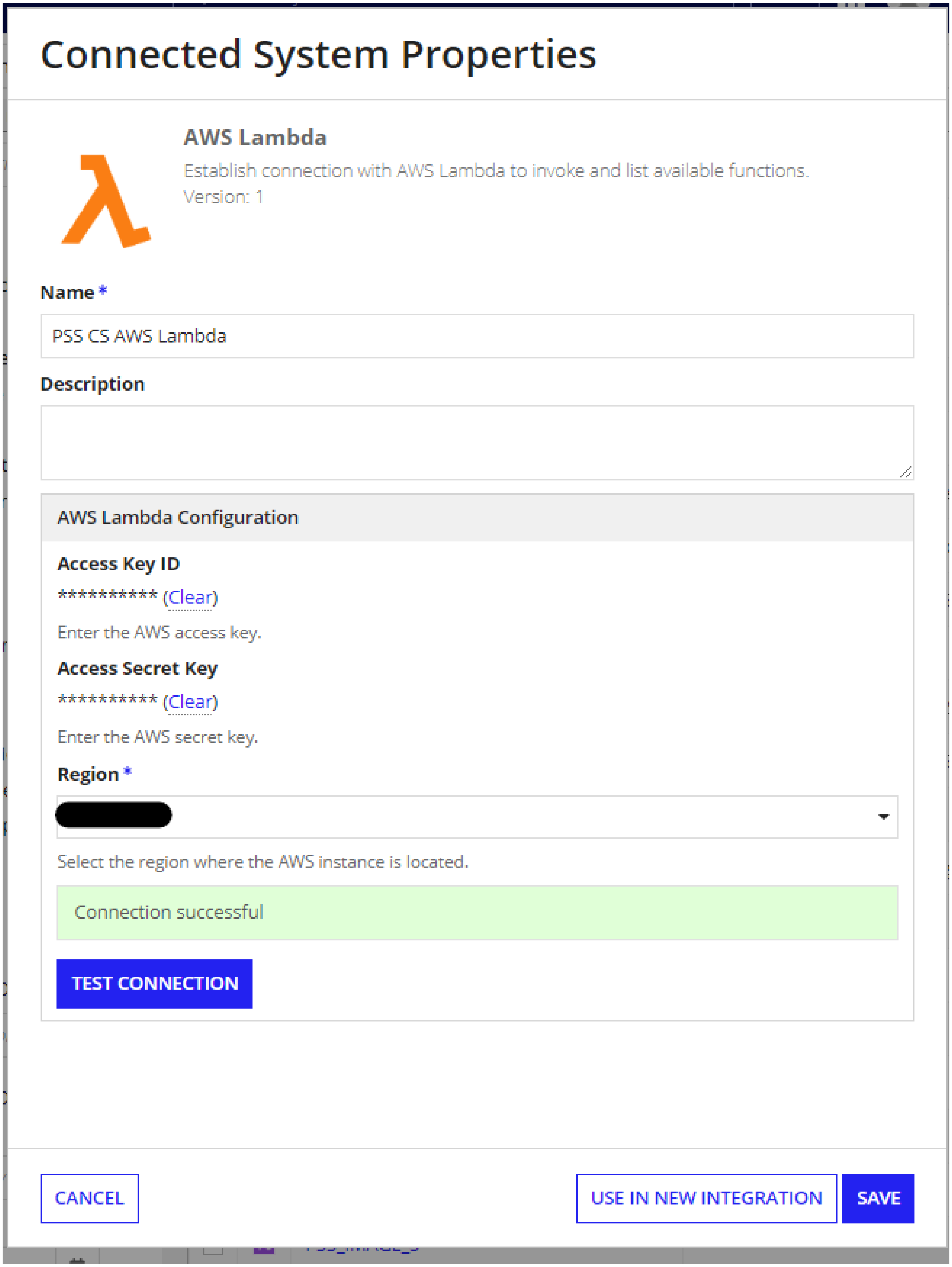
Enter the AWS secret key.

Region *

Select a Value

Select the region where the AWS instance is located.

Provide the Access Key ID and the Secret Access key obtained from the AWS console. Select the AWS instance region and click on test connection to verify the credentials.



Connected System Properties

AWS Lambda

Establish connection with AWS Lambda to invoke and list available functions.

Version: 1

Name *

PSS CS AWS Lambda

Description

AWS Lambda Configuration

Access Key ID

*********** (Clear)

Enter the AWS access key.

Access Secret Key

*********** (Clear)

Enter the AWS secret key.

Region *

[redacted]

Select the region where the AWS instance is located.

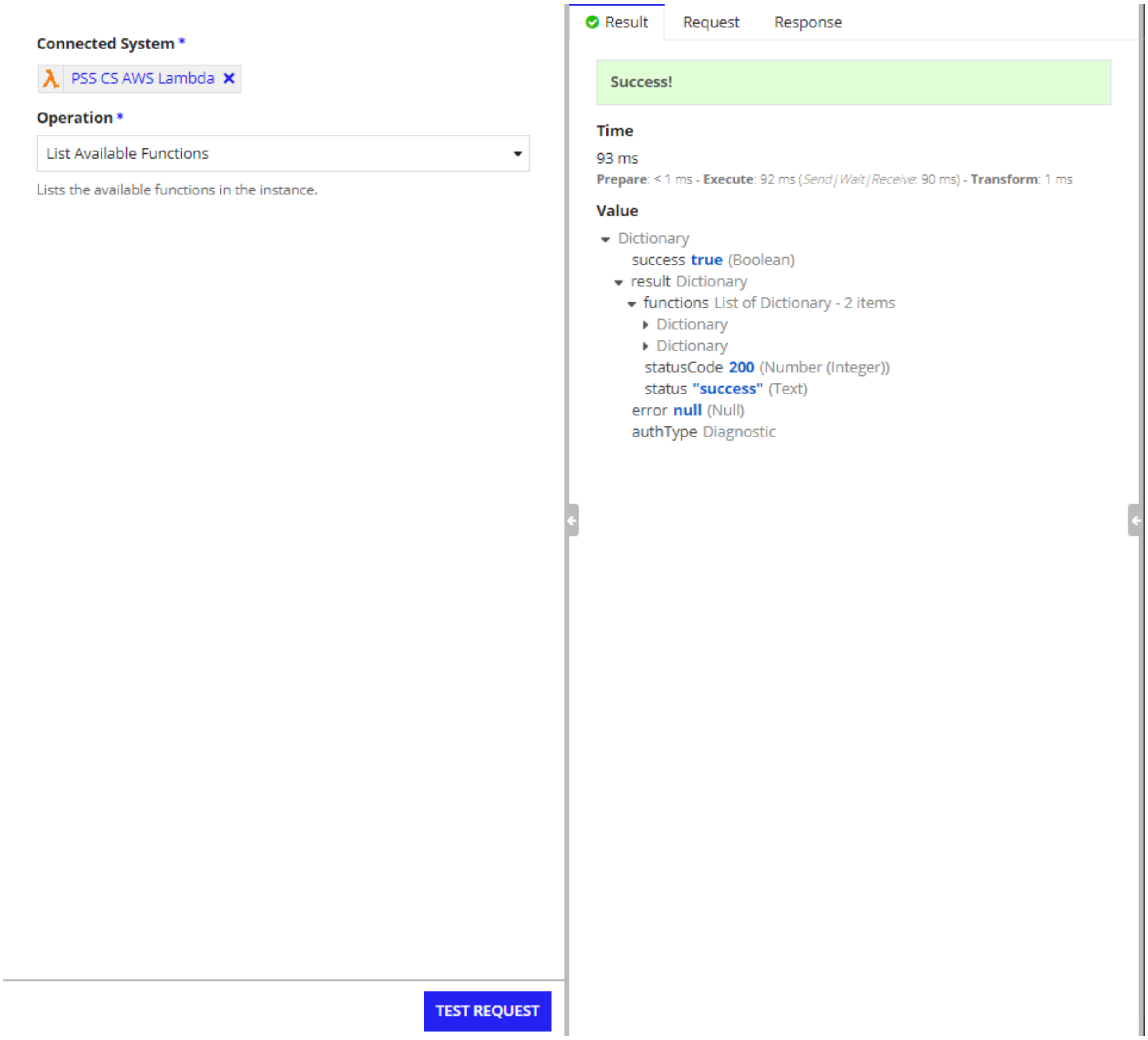
Connection successful

Available Integrations

1. List Available Function
2. Invoke Function

List Available Function

This integration lists all the available functions for the provided credentials in the AWS Lambda service.



Connected System *
PSS CS AWS Lambda

Operation *
List Available Functions

Lists the available functions in the instance.

Success!

Time

93 ms

Prepare: < 1 ms - **Execute**: 92 ms (*Send / Wait / Receive*: 90 ms) - **Transform**: 1 ms

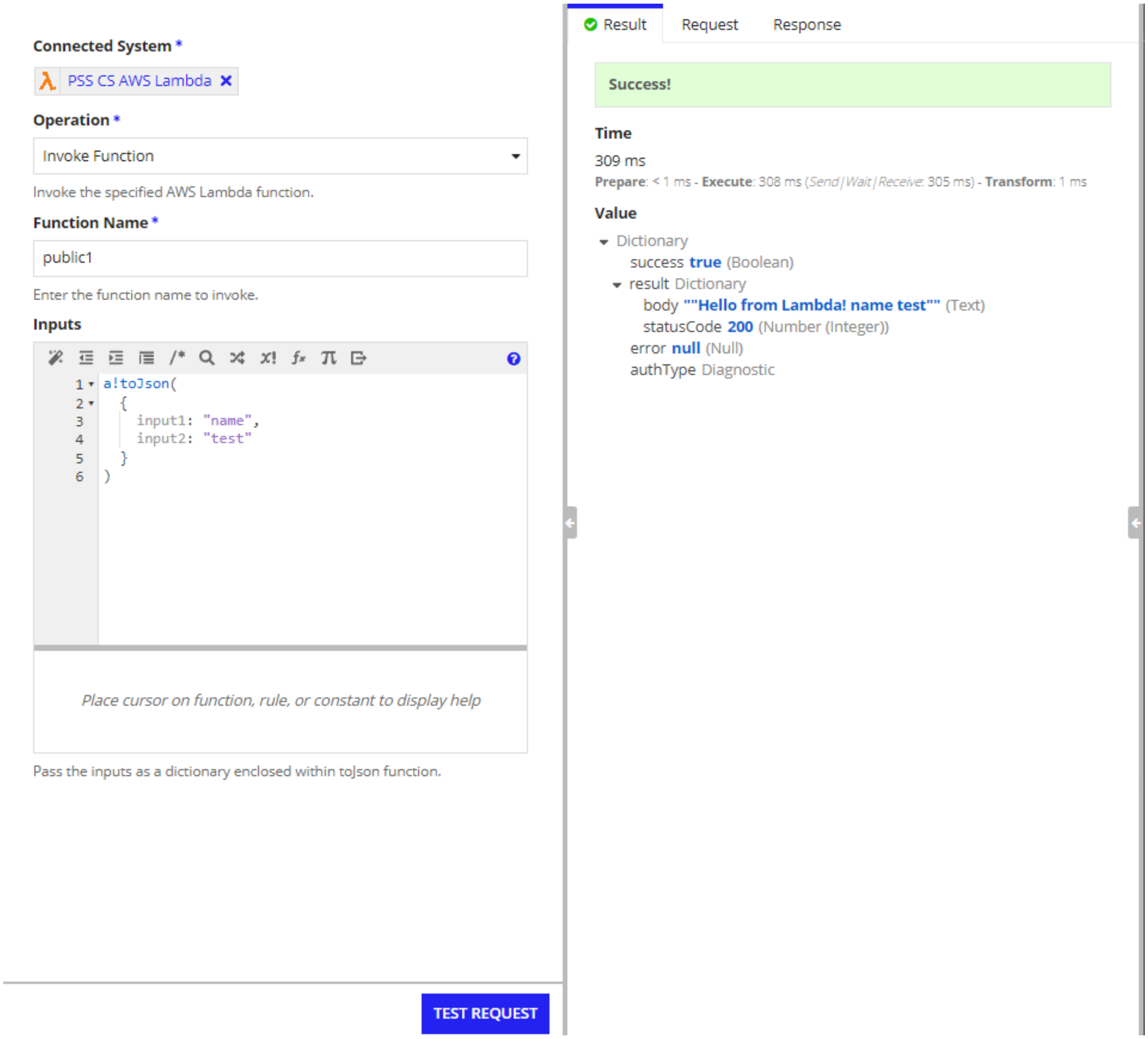
Value

```
▾ Dictionary
    success true (Boolean)
  ▾ result Dictionary
    ▾ functions List of Dictionary - 2 items
      ▸ Dictionary
      ▸ Dictionary
      statusCode 200 (Number (Integer))
      status "success" (Text)
    error null (Null)
    authType Diagnostic
```

Invoke Function

The Invoke Function integration invokes the specified function and returns the output. It has two parameters:

1. **Function Name(required):** the name of the AWS Lambda function to invoke.
2. **Inputs(optional):** the inputs to the function. Must be provided as a JSON object. Use the Appian a!toJson method.



Connected System *

PSS CS AWS Lambda ✕

Operation *

Invoke Function

Invoke the specified AWS Lambda function.

Function Name *

public1

Enter the function name to invoke.

Inputs

```
a!toJson(
  {
    input1: "name",
    input2: "test"
  }
)
```

Place cursor on function, rule, or constant to display help

Pass the inputs as a dictionary enclosed within toJson function.

Success!

Time

309 ms

Prepare: < 1 ms - **Execute**: 308 ms (*Send/Wait/Receive*: 305 ms) - **Transform**: 1 ms

Value

```
▾ Dictionary
    success true (Boolean)
  ▾ result Dictionary
      body ""Hello from Lambda! name test"" (Text)
      statusCode 200 (Number (Integer))
    error null (Null)
    authType Diagnostic
```