Guided Lab: Aliases and Versions in AWS Lambda

Description

In AWS Lambda, versions and aliases are closely related tools that help you manage and deploy your functions more effectively. Versions allow you to lock in a specific iteration of your function, while aliases act as pointers to these versions, making it easier to manage different environments such as development, staging, and production. Aliases can also be used for weighted traffic routing, enabling gradual rollouts or A/B testing of new function versions.

This lab will guide you through the process of creating and managing versions and aliases in AWS Lambda. You will also learn how to implement weighted traffic routing between different versions of your Lambda function.

Prerequisites

This lab assume you have basic understanding of AWSLambda and Node.JS programming language.

If you find any gaps in your knowledge, consider taking the following lab:

- **Creating an AWS Lambda function**
- **Creating a NodeJS Function in AWS Lambda**

Objectives

By the end of this lab, you will be able to:

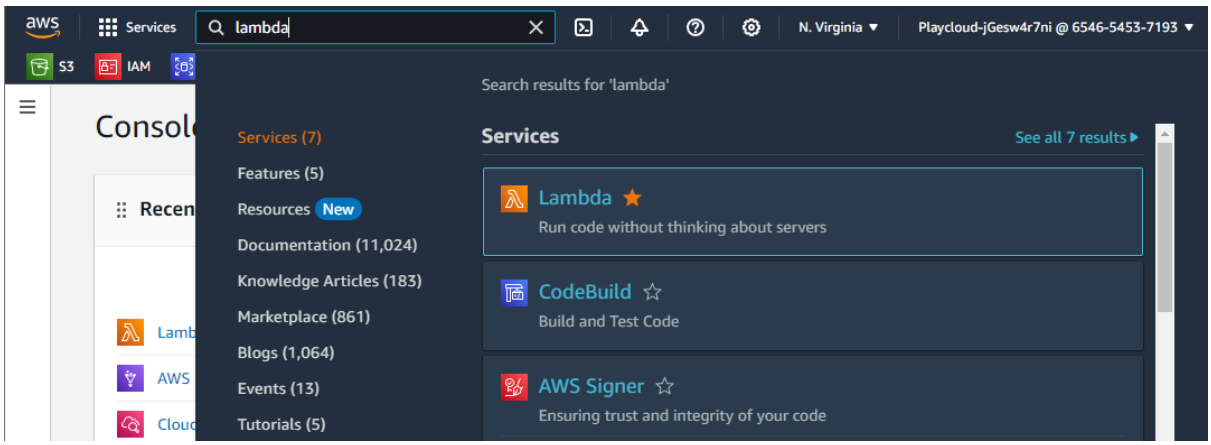
- **Create a version of your Lambda function.**
- **Modify the function code and publish a new version.**
- **Create and update an alias to point to specific versions.**
- **Implement weighted traffic routing between different versions using aliases.**

Lab Steps

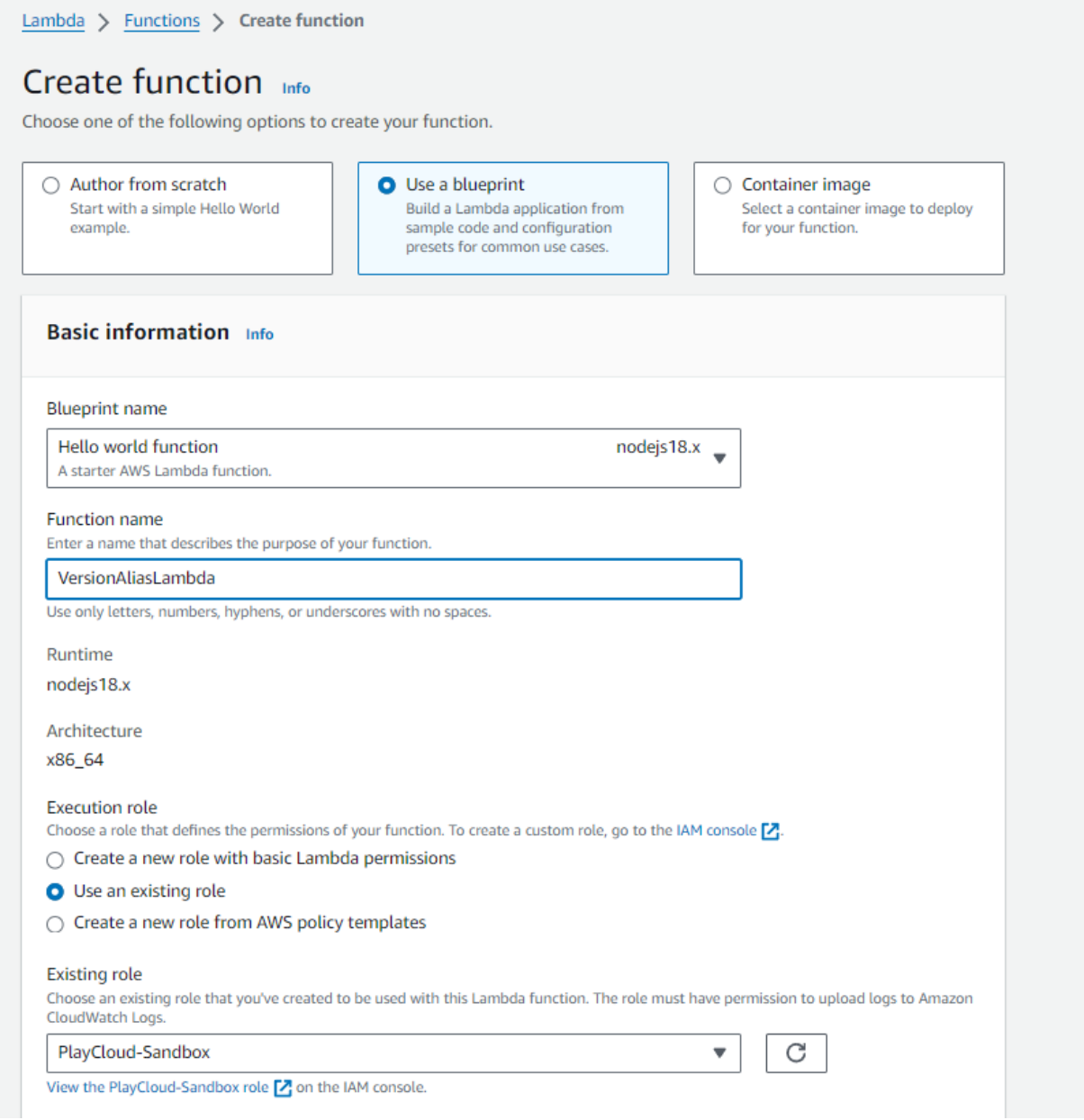
Creating and Testing an AWS Lambda Function

1. CREATE AN AWS LAMBDA FUNCITON

- **Navigate to AWS Lambda Console**

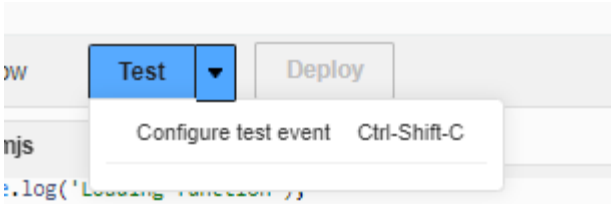


- **Create Function using the following confgurations:**
    - **Select Use a blueprint**
    - **Blueprint name : Hello world function nodejs18.x**
    - **Function name: Enter a name for your function (e.g., VersionAliasLambda).**
    - **Execution role:**
        - **Select Use an Existing Role: PlayCloud-Sanbox**



-
    - **Click on Create Function**

2. Once your function is created. Click the arrow dropdown of the BLUE Test button



3. Click on Configure test event, and follow the configuration below:

- **Event name: VersionTest**
- **Paste the following JSON to teh Event JSON field:**

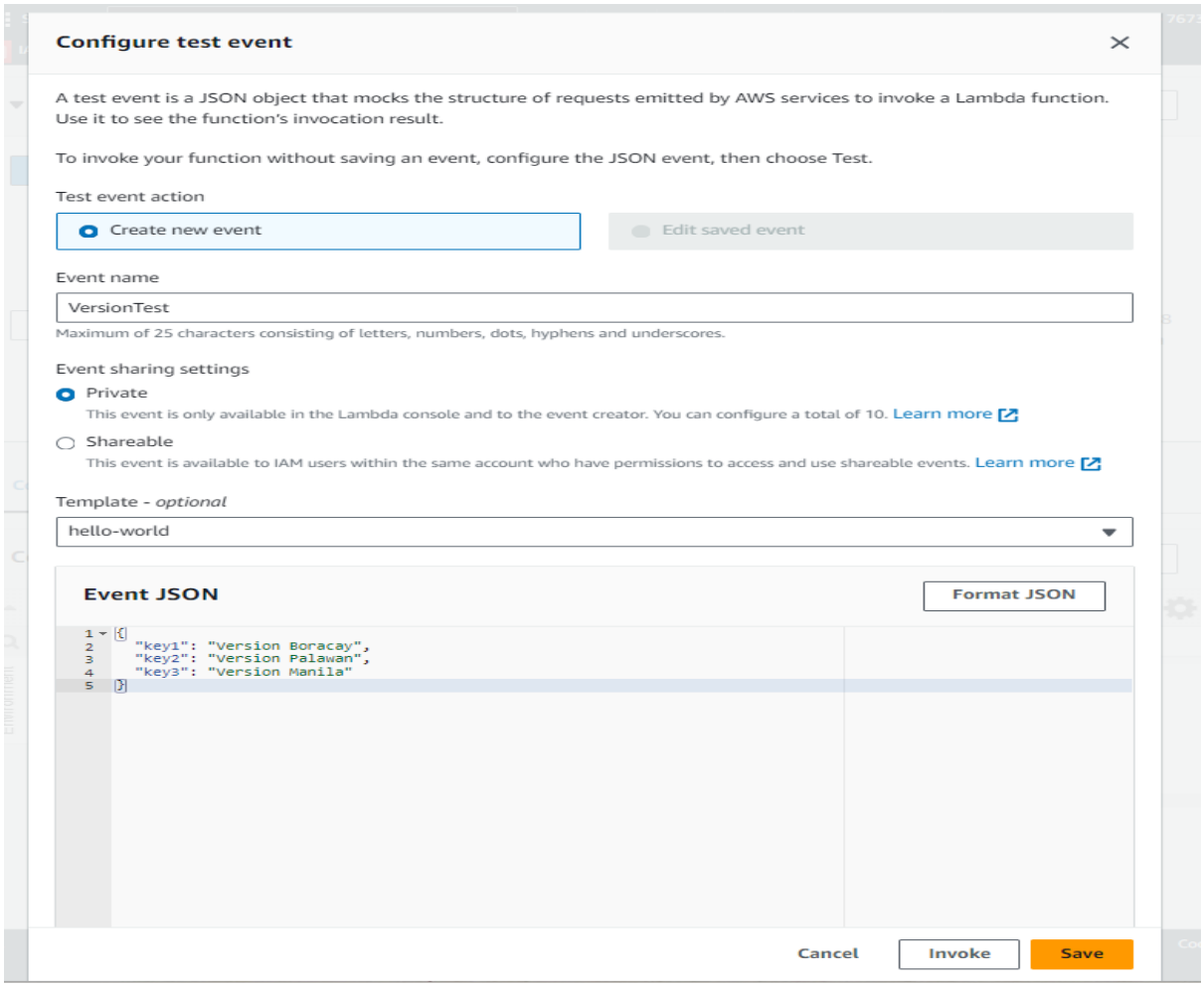
{

"key1": "Version Boracay",

"key2": "Version Palawan",

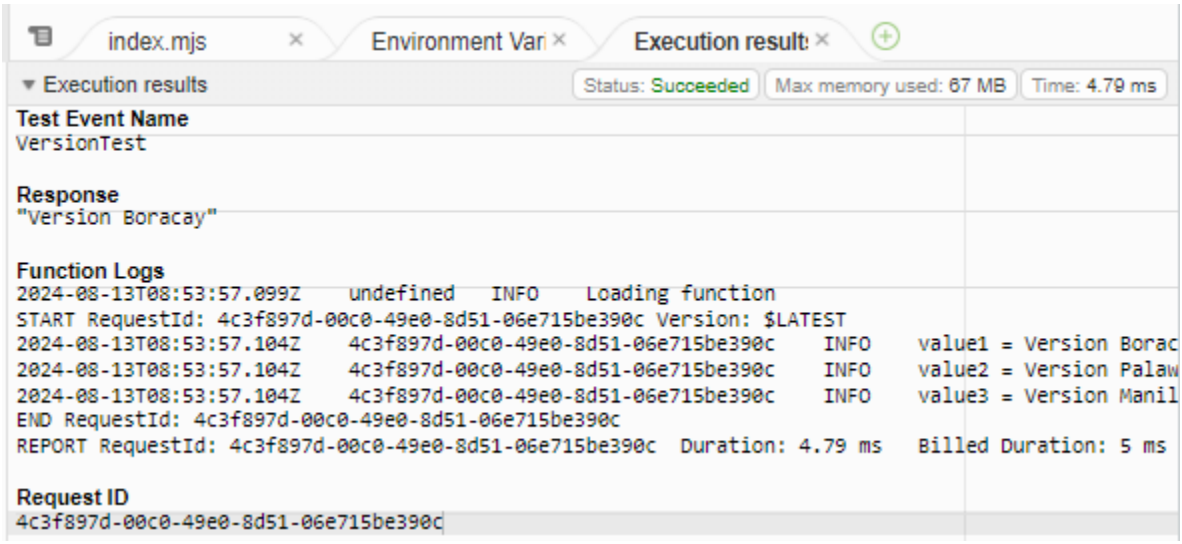
"key3": "Version Manila"

}



- **Click on Save**

4. Now, click on Test. The current response should be:

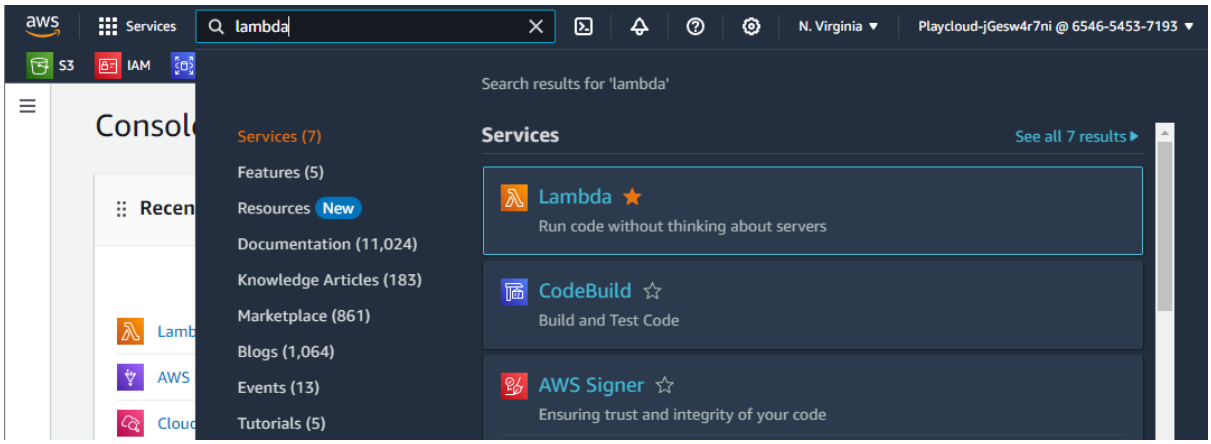


Lab Steps

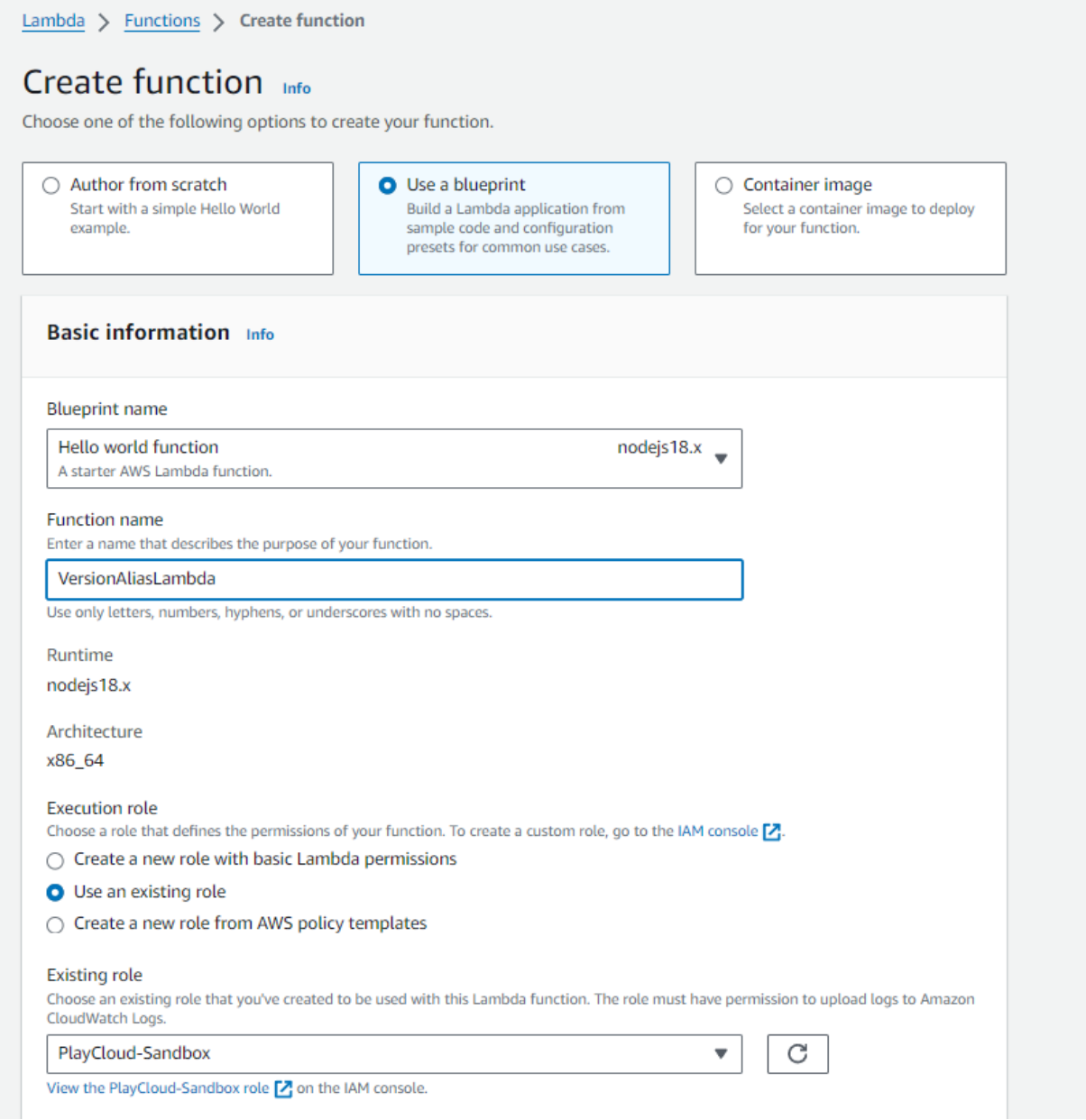
Creating and Testing an AWS Lambda Function

1. CREATE AN AWS LAMBDA FUNCITON

- Navigate to AWS Lambda Console

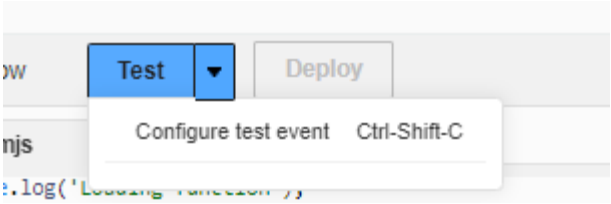


- Create Function using the following confgurations:
    - Select Use a blueprint
    - Blueprint name : **Hello world function nodejs18.x**
    - Function name: Enter a name for your function (e.g., **VersionAliasLambda**).
    - **Execution role:**
        - Select Use an Existing Role: **PlayCloud-Sanbox**



-
    - Click on **Create Function**

2. Once your function is created. Click the arrow dropdown of the BLUE **Test button**



3. Click on **Configure test event,** and follow the configuration below:

- Event name: VersionTest
- Paste the following JSON to teh **Event JSON field:**

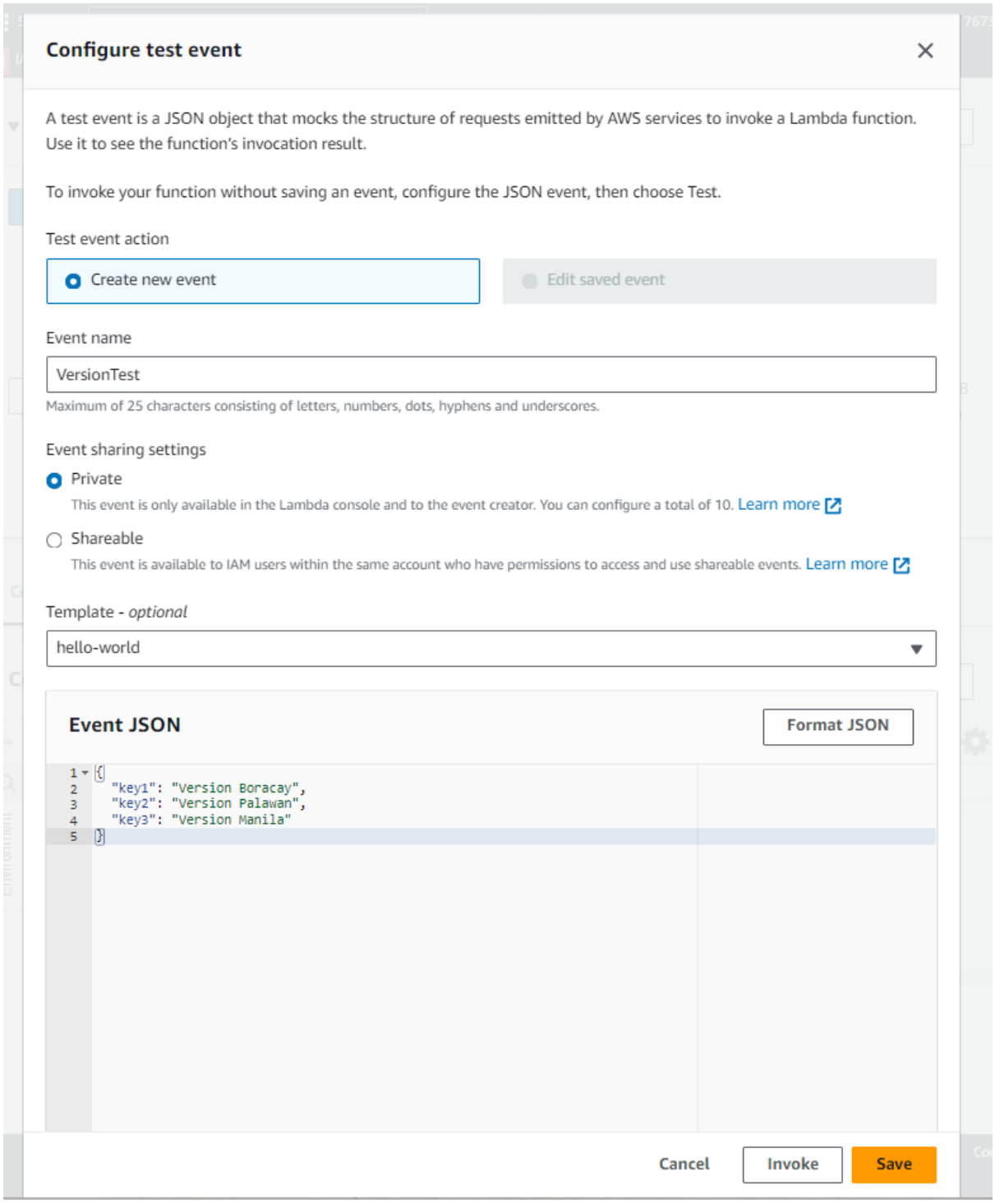
{

"key1": "Version Boracay",

"key2": "Version Palawan",

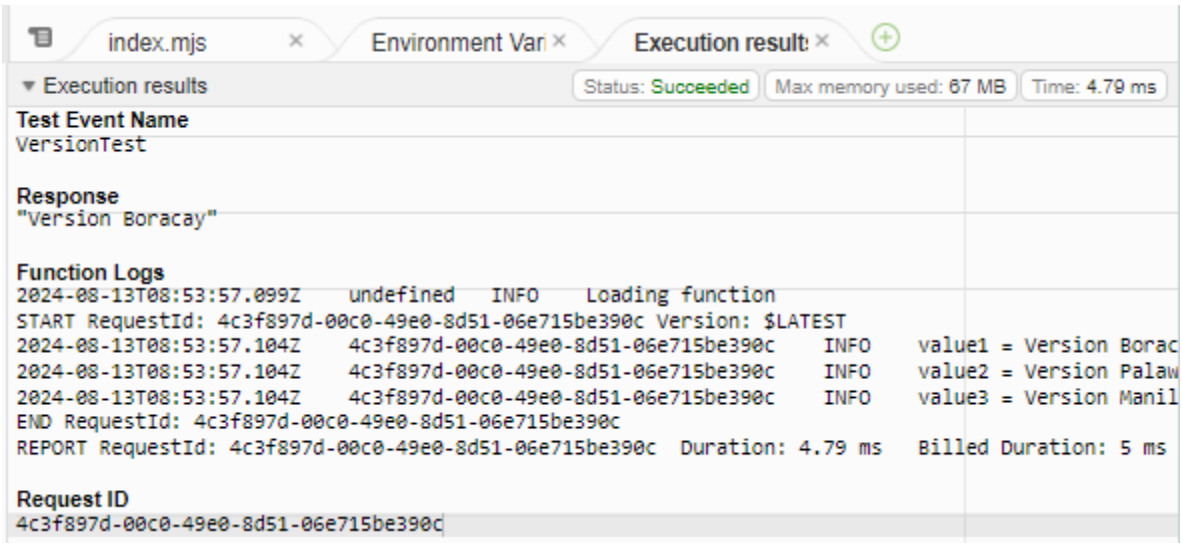
"key3": "Version Manila"

}



- Click on Save

4. Now, click on **Test.** The current response should be:



Create a New Version

1. **Modify the Function Code:**

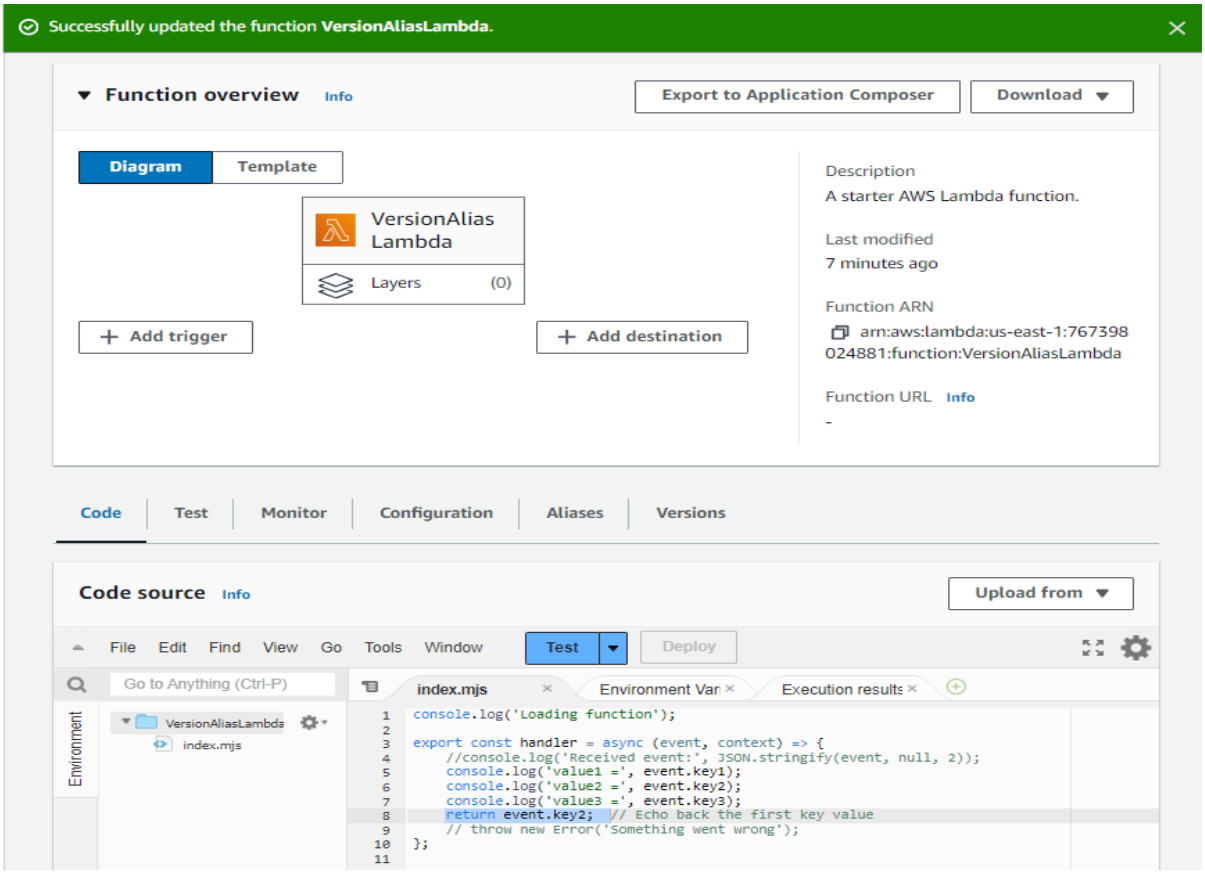
- Scroll down to the **Code source** section.

***Note**: We are using the **old console editor** for this lab. You can switch to the **new or old editor** as you desire; the process remains the same, but the interface may look slightly different.*

- Locate **Line 8** in the code and modify it to:

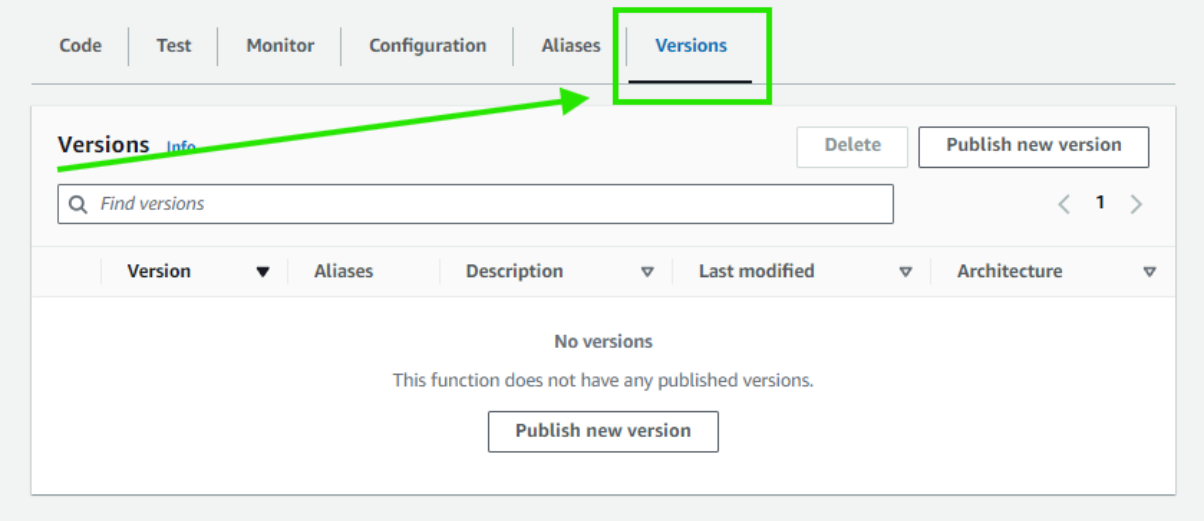
return event.key2;

- Click on **Deploy** to apply the changes.

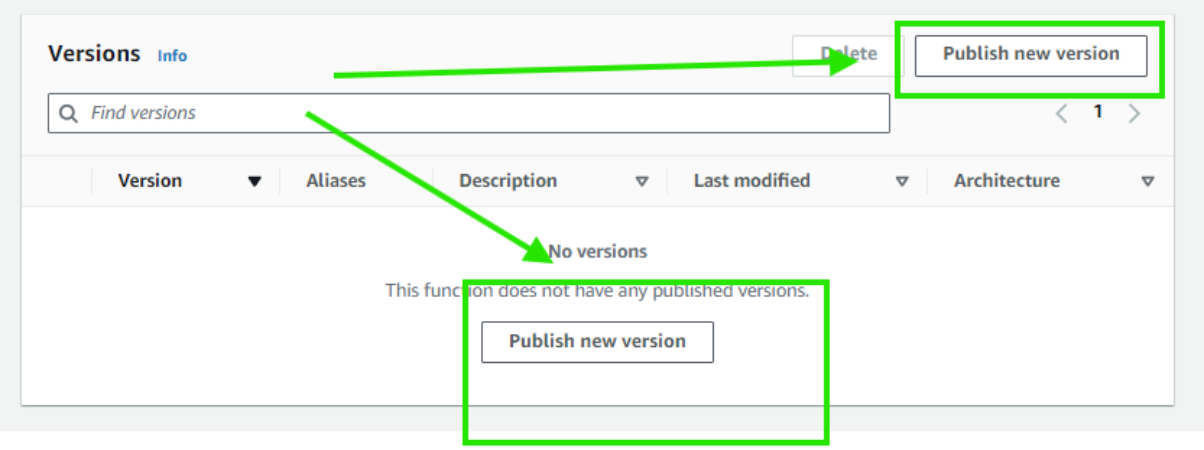


2. **Publish a New Version:**

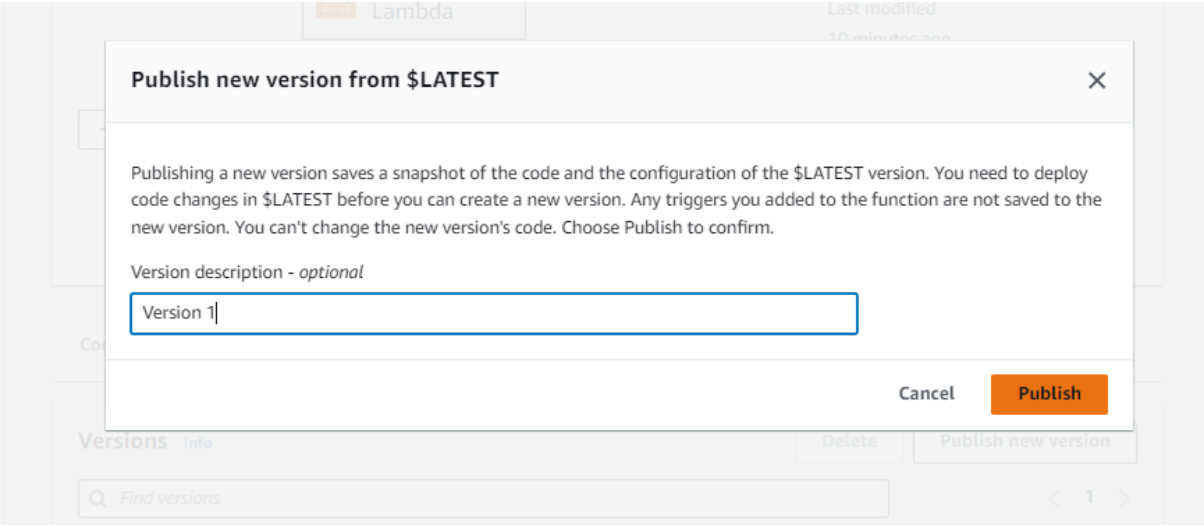
- After deploying the changes, navigate to the Versions Tab:



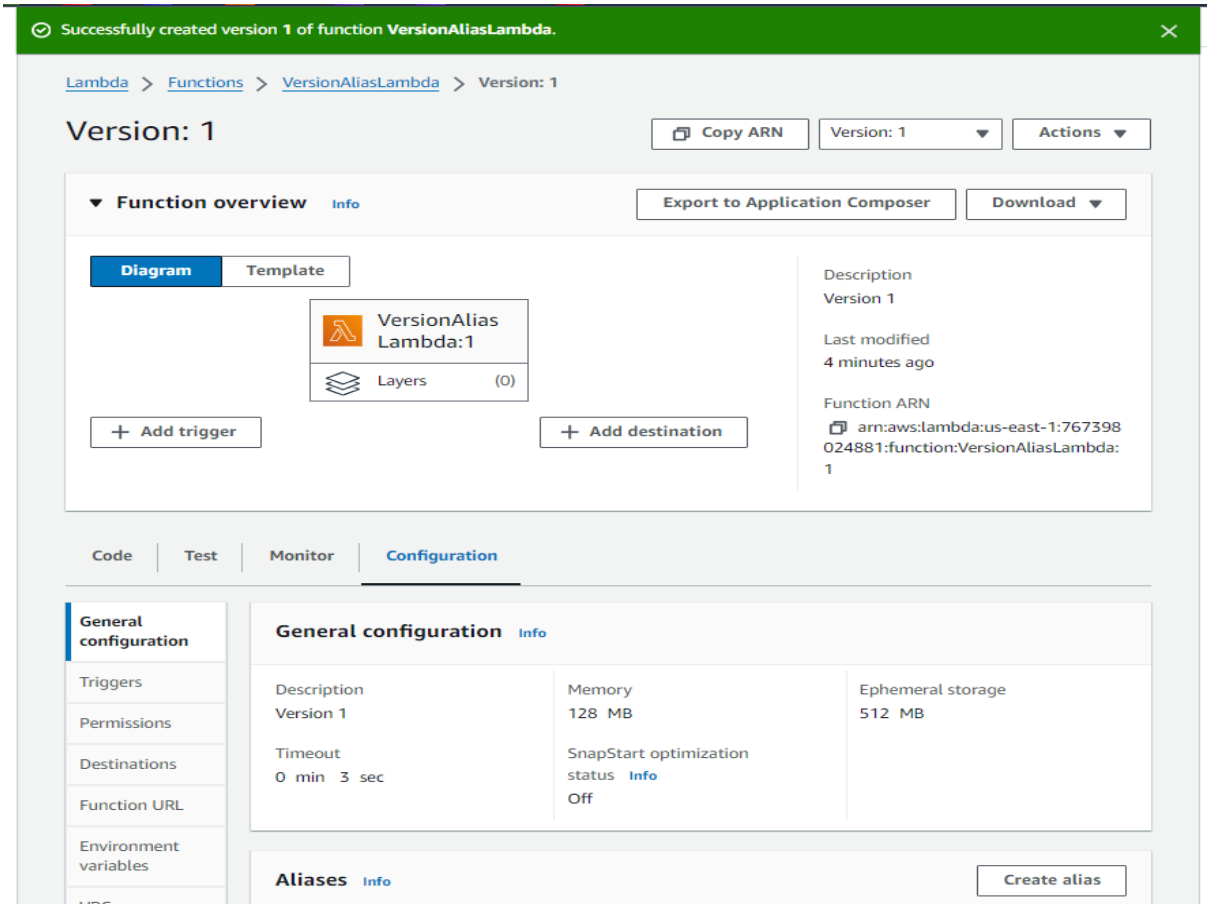
- Click on either of the two **Publish new version**



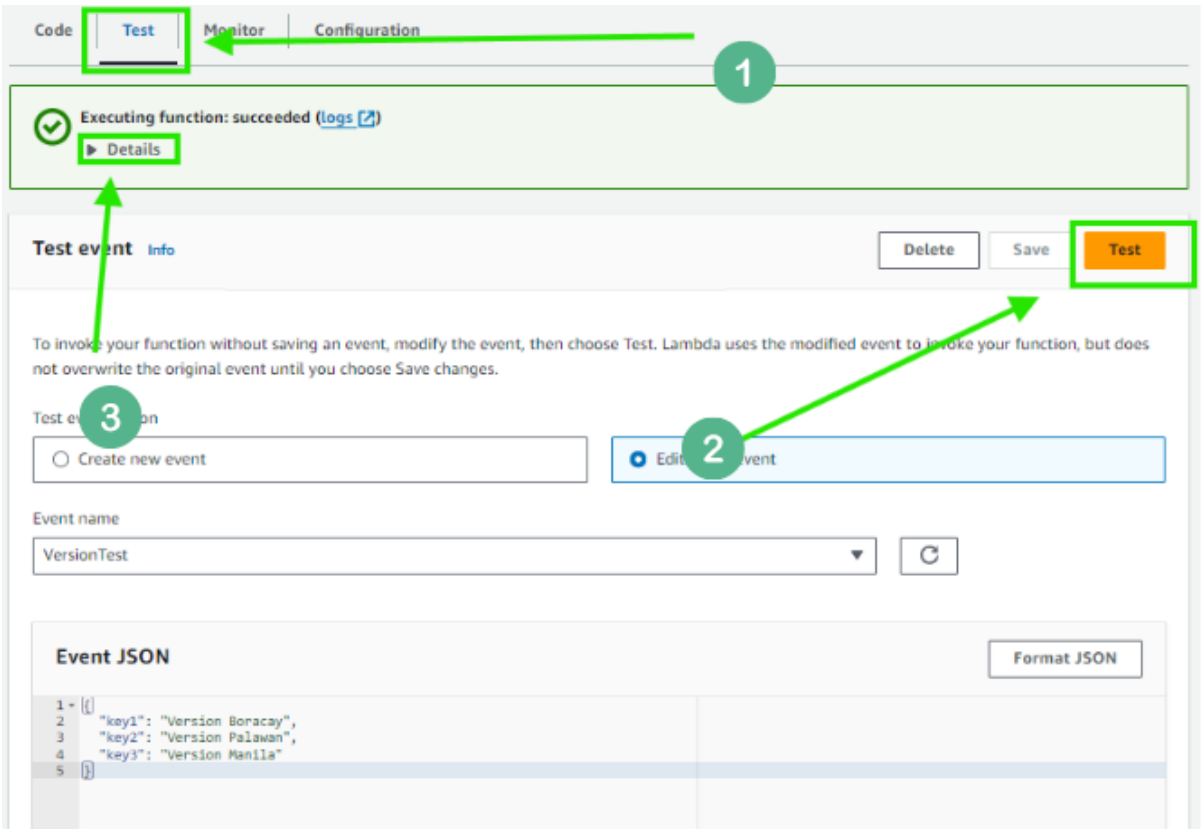
- Add an optional description (e.g., Version 1) and click **Publish**.

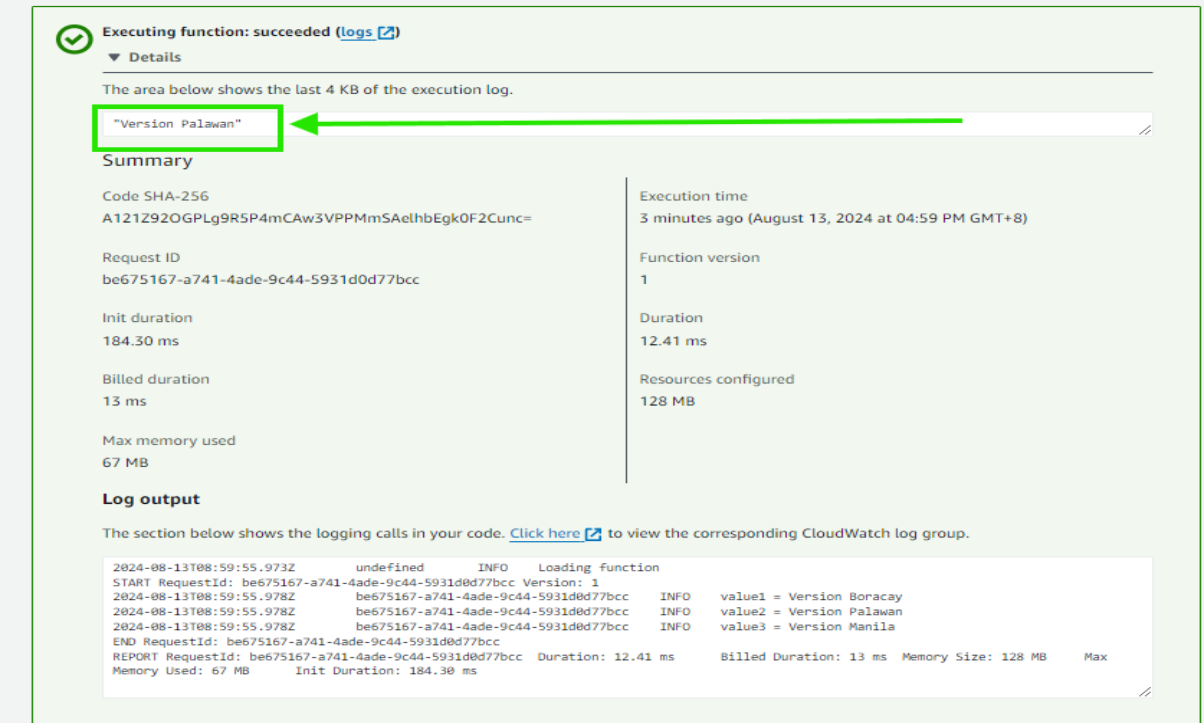


3. You will be redirected to this page:



4. Navigate to the **Test Tab**, click on **Test**, and click the Dropdown **Details**



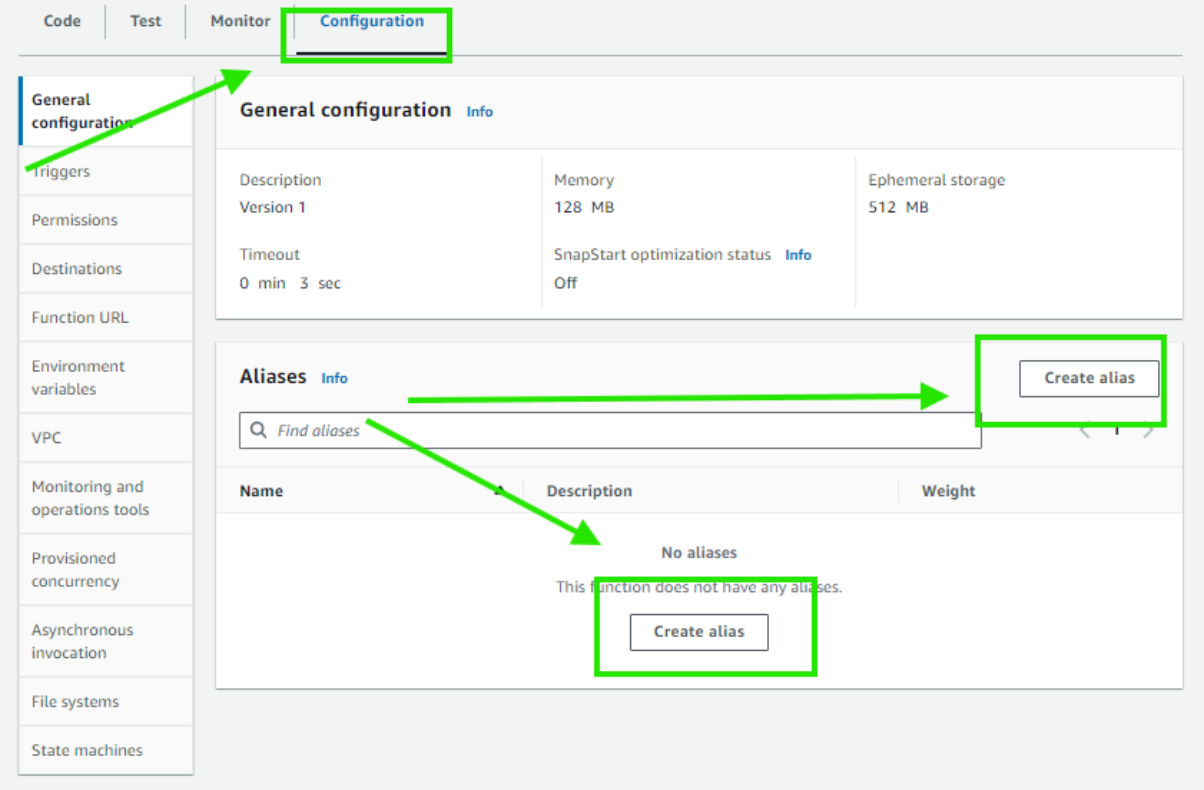


Create an Alias

Lambda aliases provide a flexible way to manage function versions. An alias serves as a reference to a particular version of your Lambda function and can be modified to point to different versions. Users interact with the function through the alias's Amazon Resource Name (ARN). When a new version is deployed, you can adjust the alias to direct traffic to this version or balance the load between the existing and new versions.

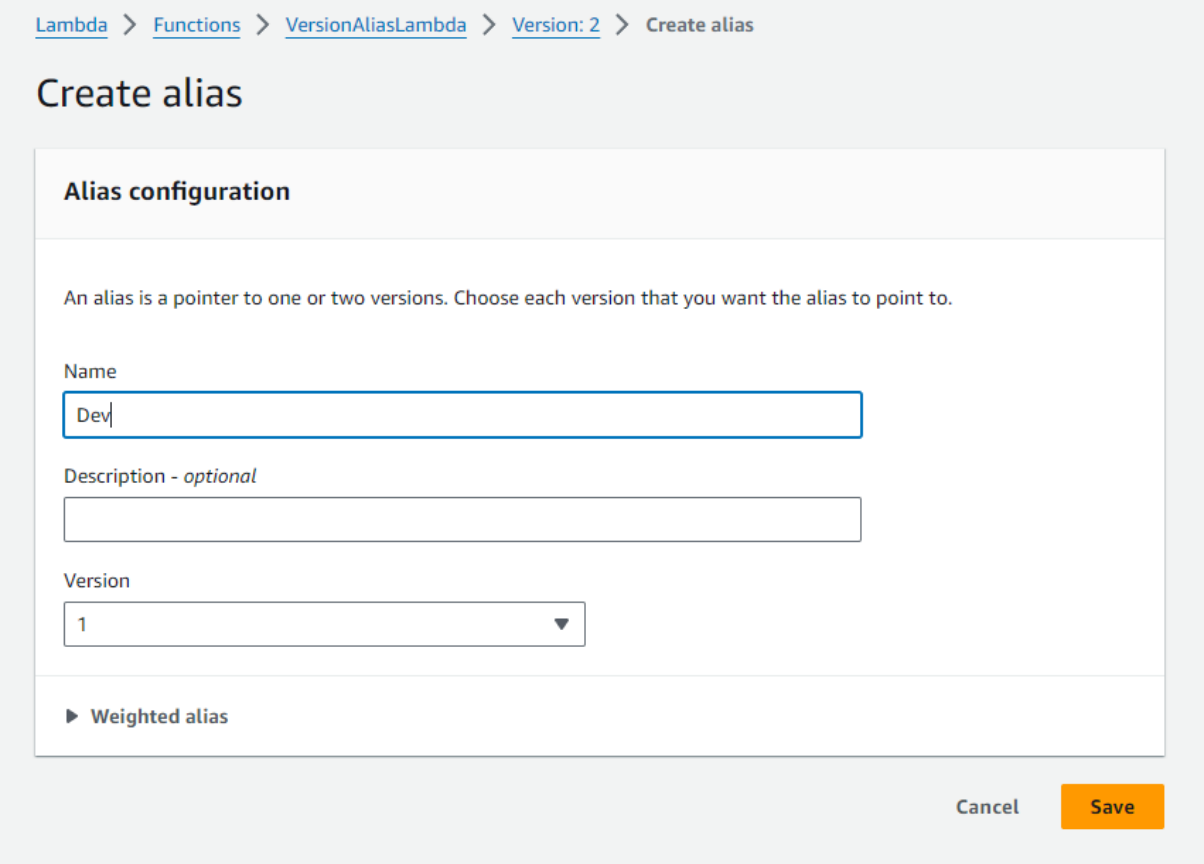
---

1. Navigate to the Configuration Tab of your Lambda function's dashboard, and click **Create alias**.



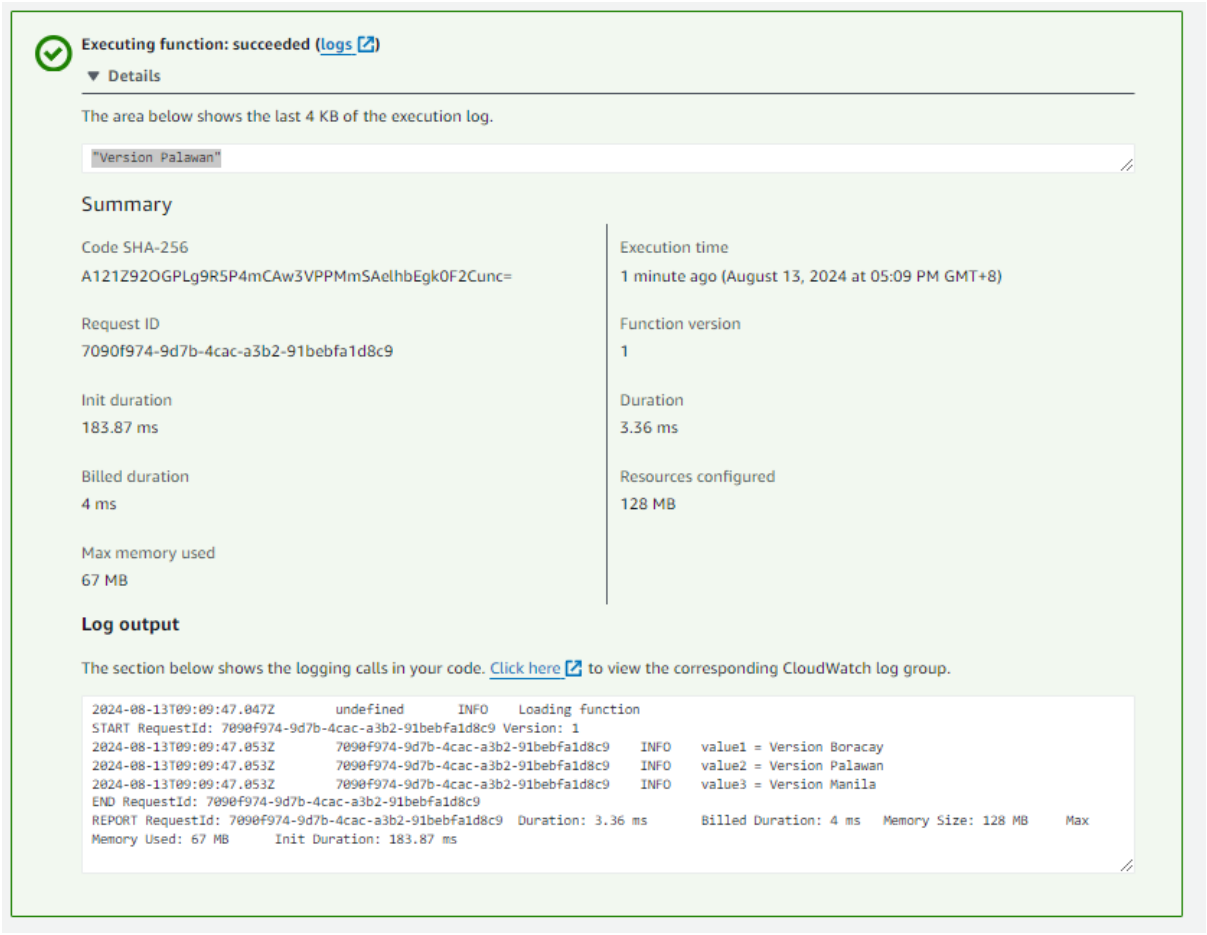
2. Follow the following setting:

- **Name:** Enter a name for the alias (e.g., Dev).
- **Version:** Select the version you just published (e.g., 1).



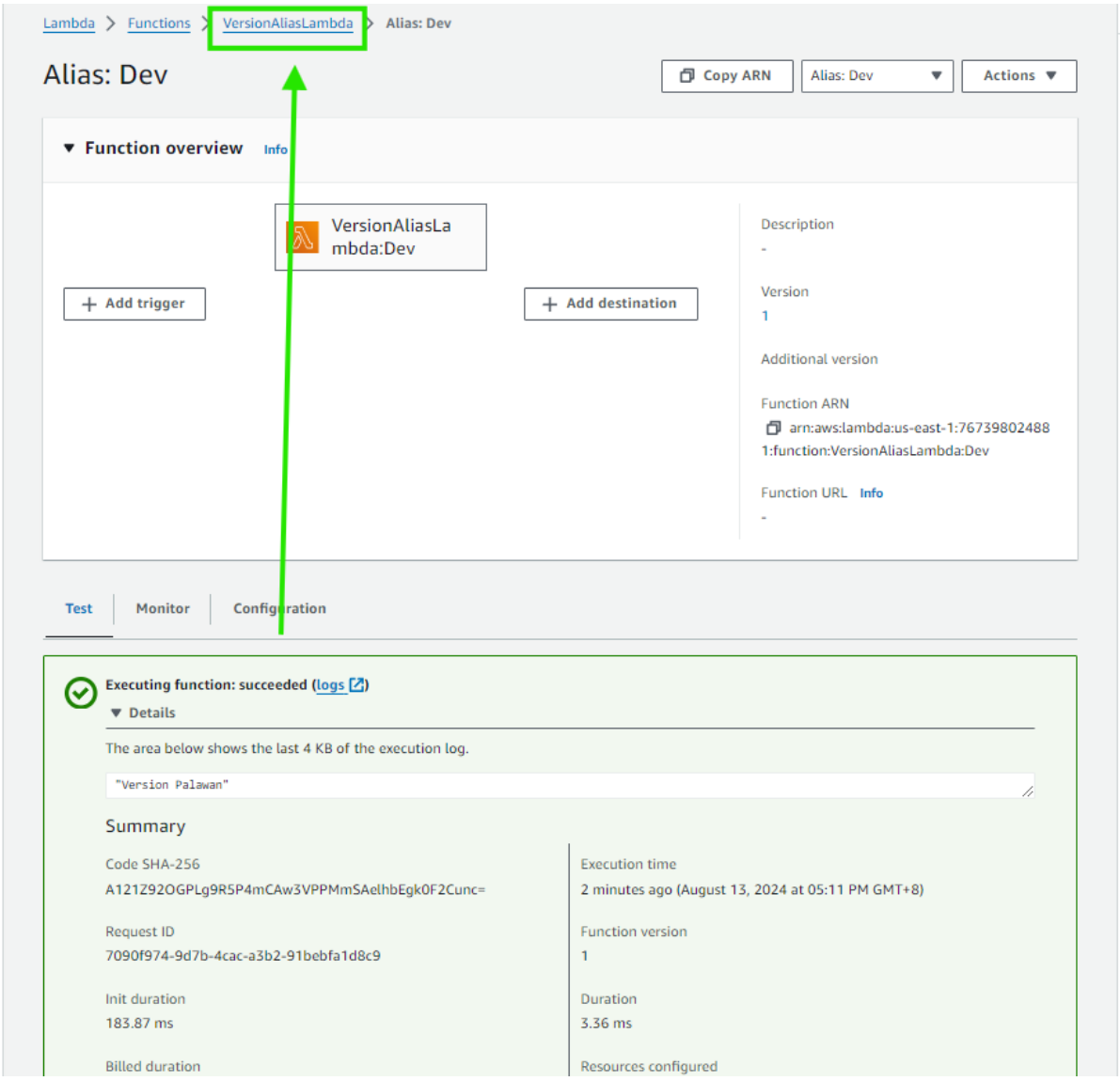
- Click **Save**.

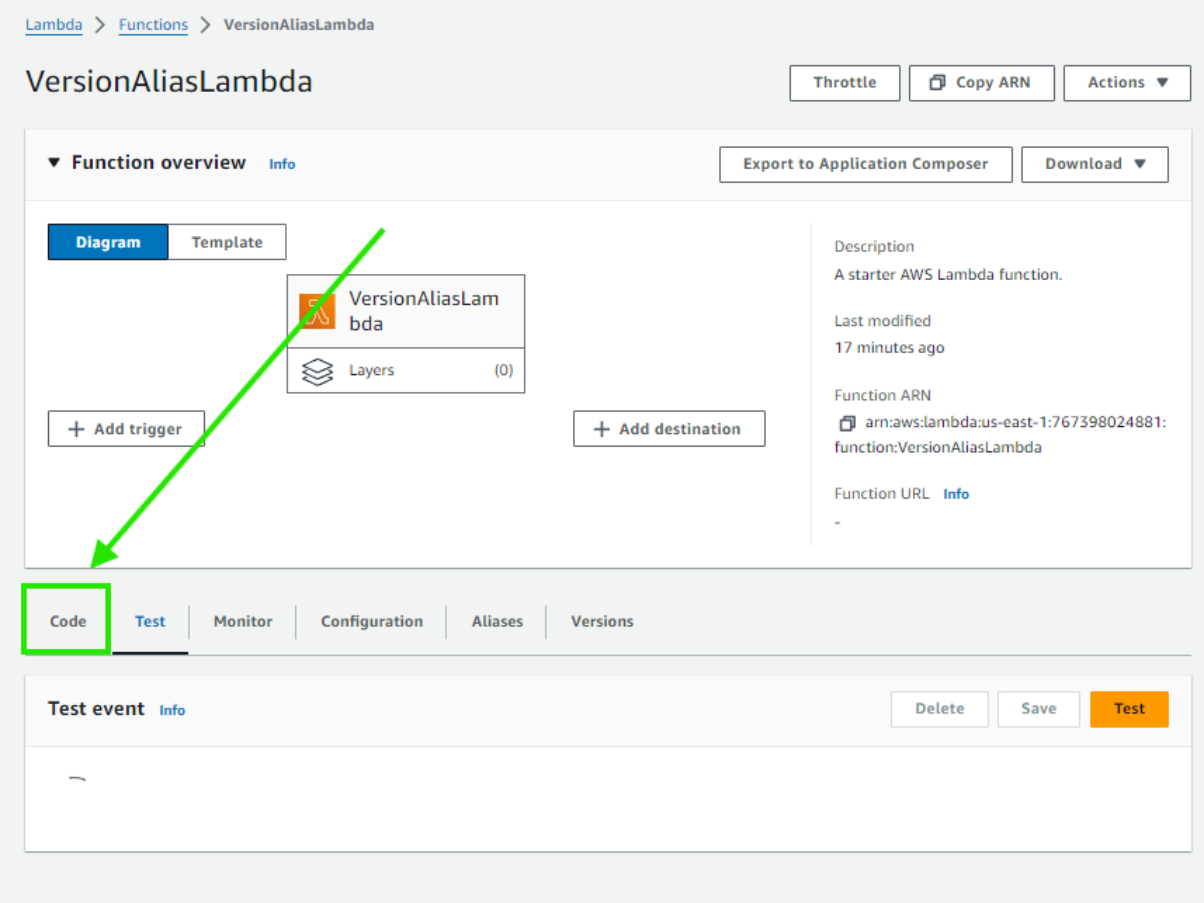
3. Click on **Test** and observe that the function returns "**Version Palawan**" the same as our previous test in the Create a New Version Step Section



Modify the Function and Create a New Version

1. Go back to the **Code source** section.

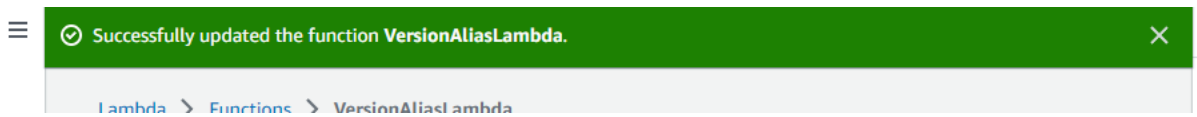

Lambda > Functions > VersionAliasLambda > Alias: Dev
Alias: Dev
Copy ARN
Alias: Dev
Actions
Function overview Info
VersionAliasLambda:Dev
+ Add trigger
+ Add destination
Description
-
Version
1
Additional version
Function ARN
arn:aws:lambda:us-east-1:767398024881:function:VersionAliasLambda:Dev
Function URL Info
-
Test
Monitor
Configuration
Executing function: succeeded (logs)
Details
The area below shows the last 4 KB of the execution log.
"Version Palawan"
Summary
Code SHA-256
A121Z92OGPLg9R5P4mCAw3VPPMmSAelhbEgk0F2Cunc=
Request ID
7090f974-9d7b-4cac-a3b2-91bebfa1d8c9
Init duration
183.87 ms
Billed duration
Execution time
2 minutes ago (August 13, 2024 at 05:11 PM GMT+8)
Function version
1
Duration
3.36 ms
Resources configured



- Change the return statement to:

return event.key3;

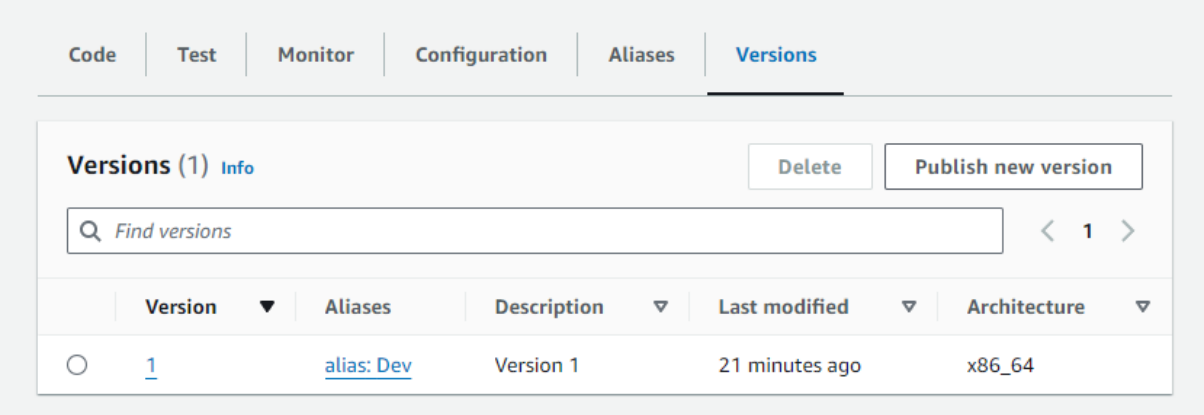
- Click **Deploy** to apply the changes.



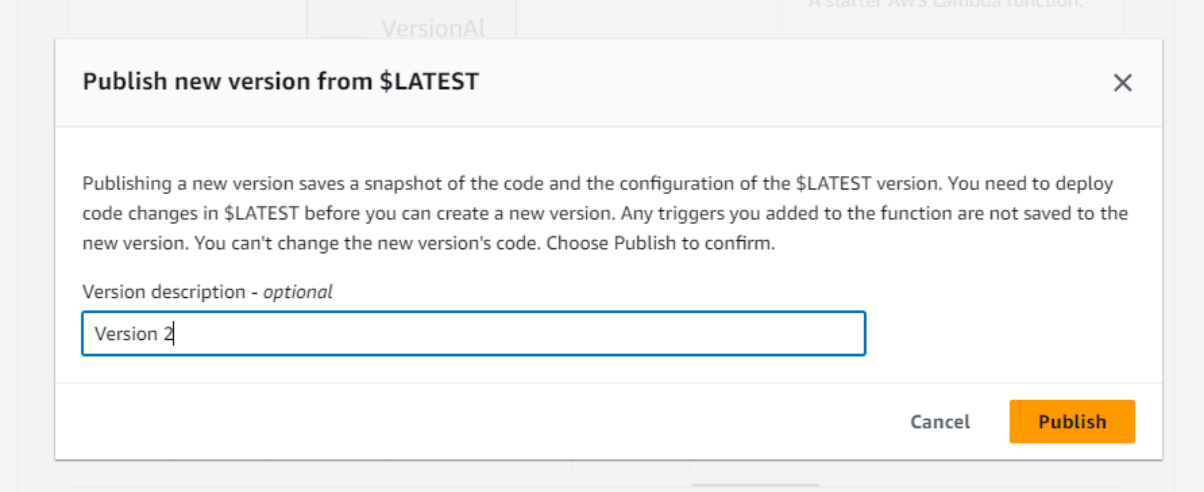
2. **Publish the New Version:**

- As before, go to the **Versions Tab ,** and click on **Publish new version.**

***Note**: You should be able to view the latest version we published.*

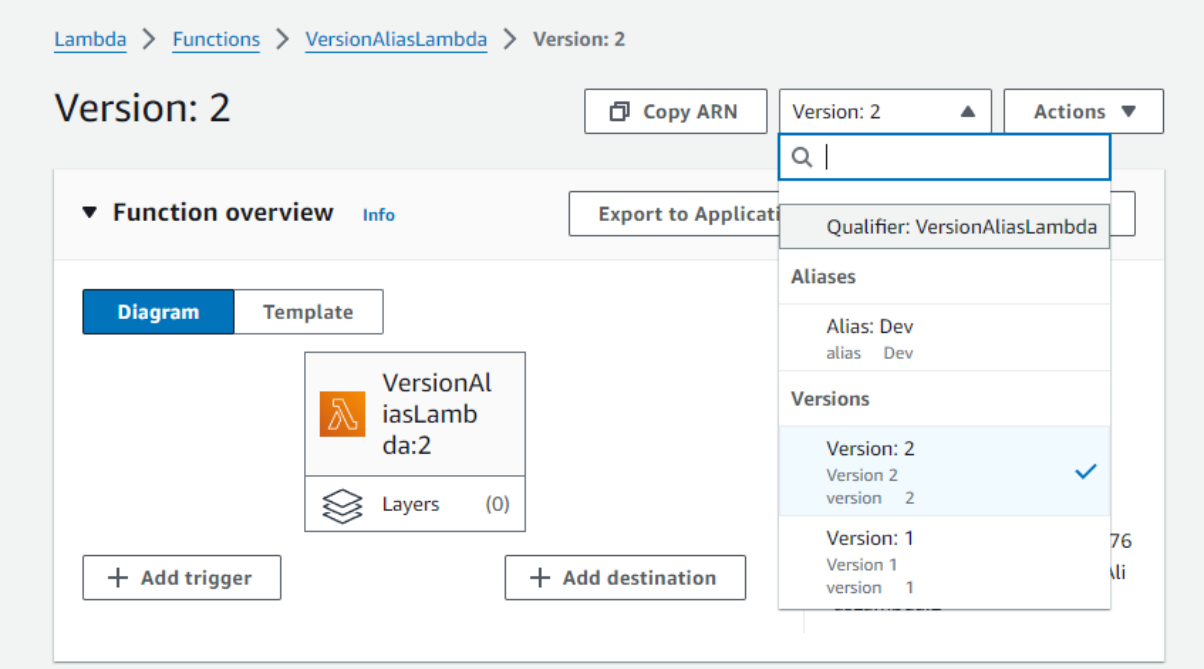


- Describe it as Version 2 and click **Publish**.

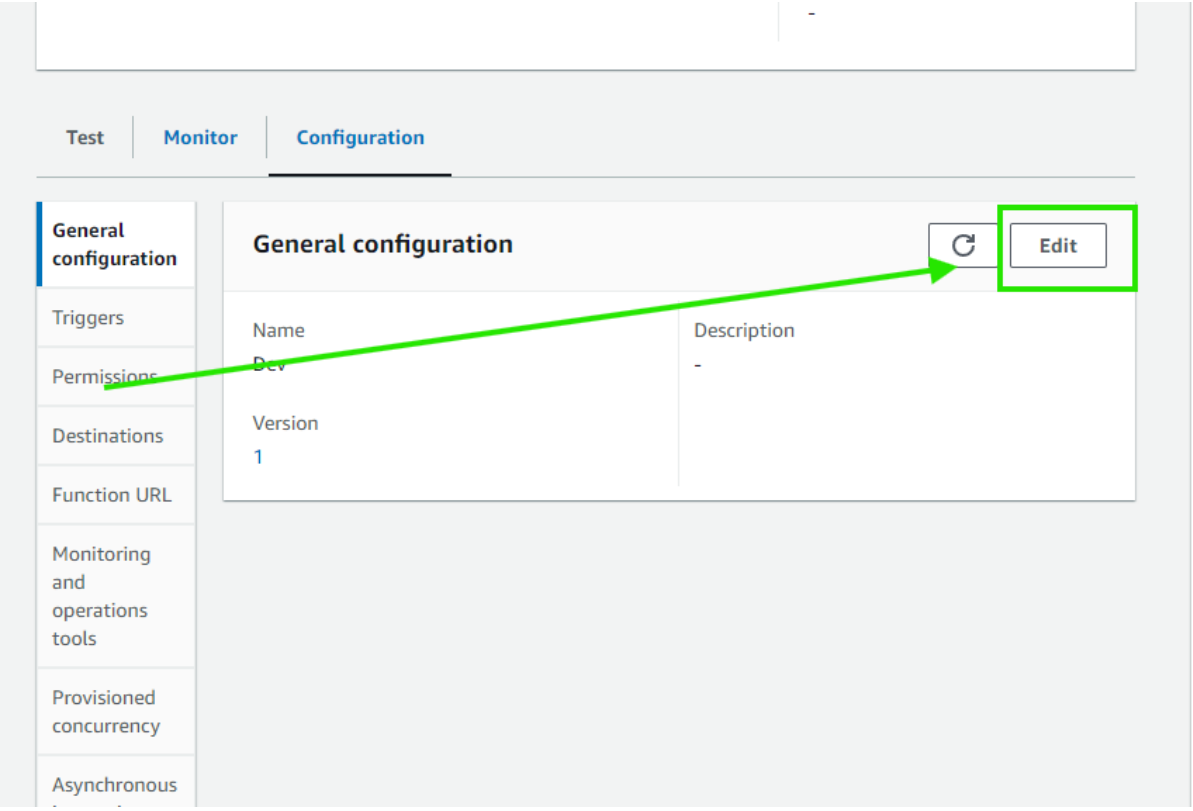


3. **Update the Alias:**

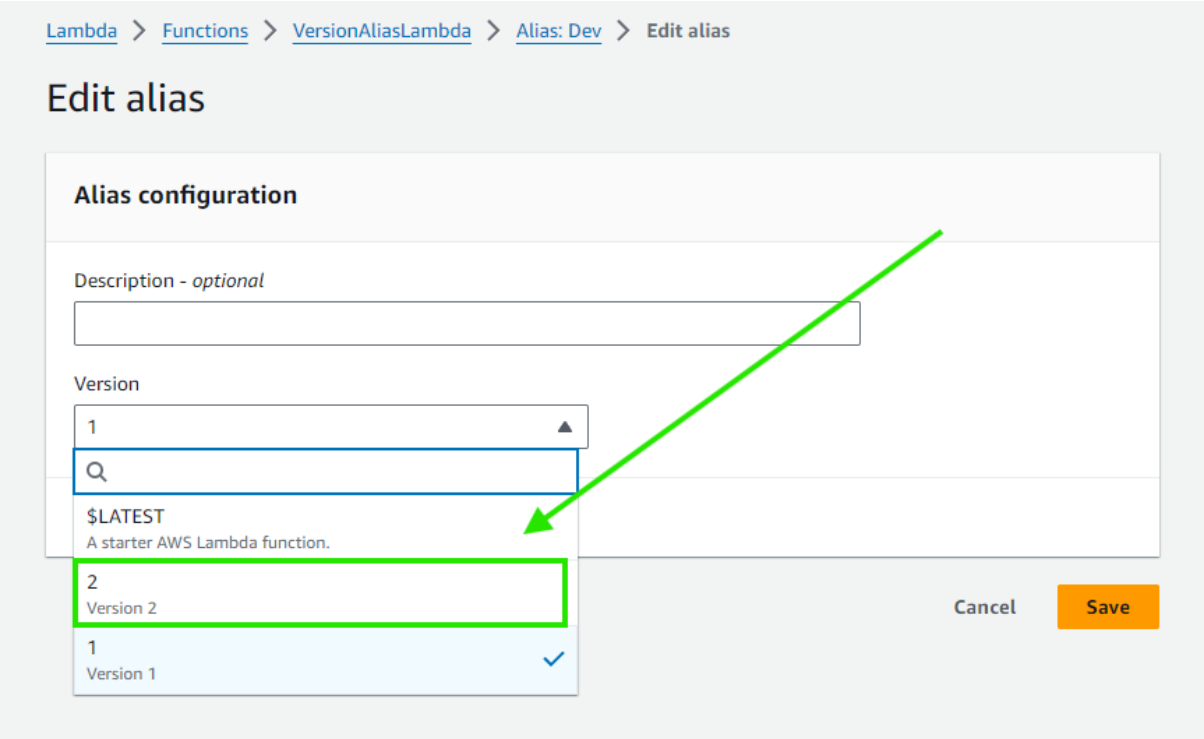
- Scroll up and Click on the **Version: 2 Dropdown**



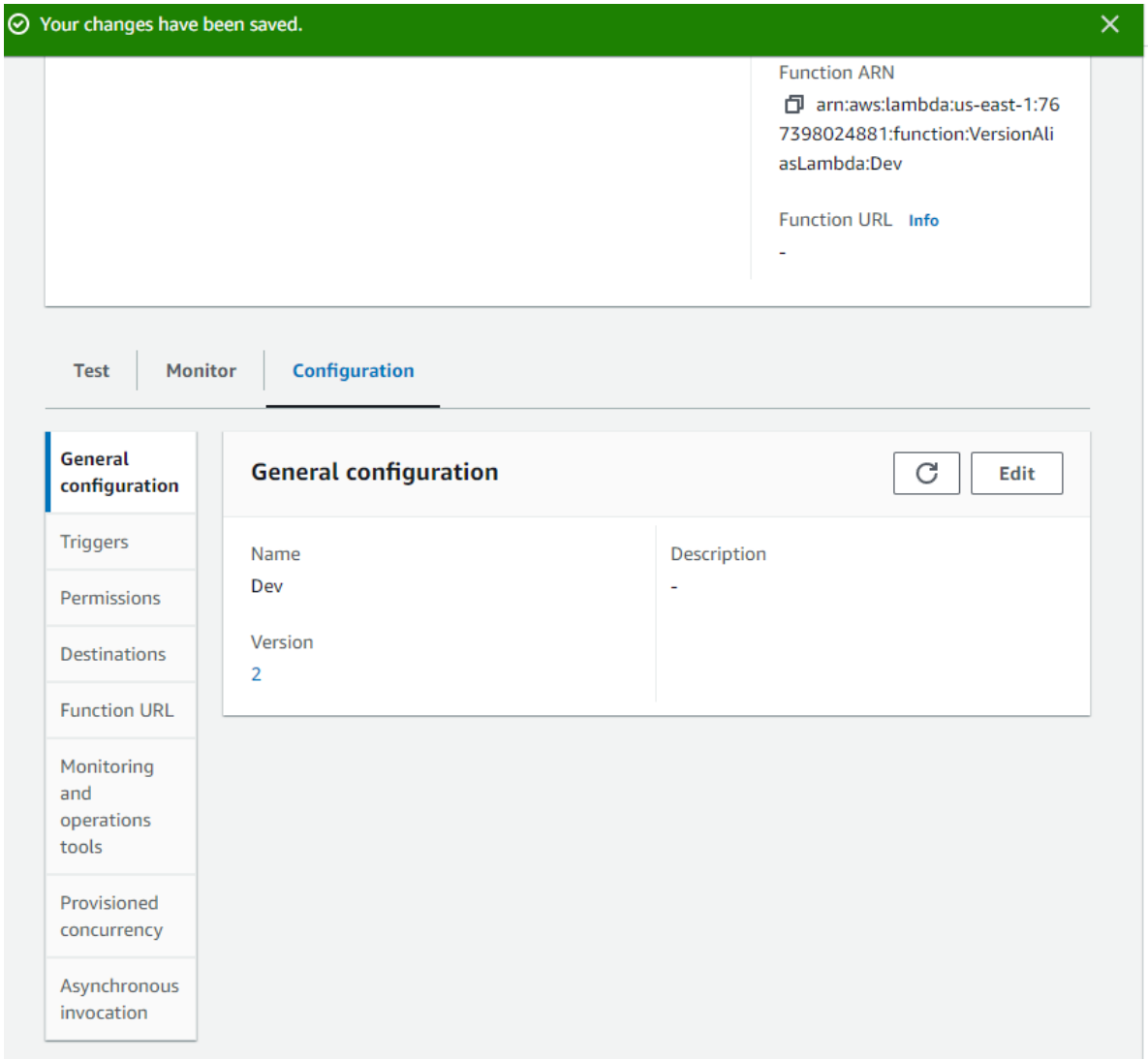
- Select **Alias: Dev**, scroll down and click on **Edit**



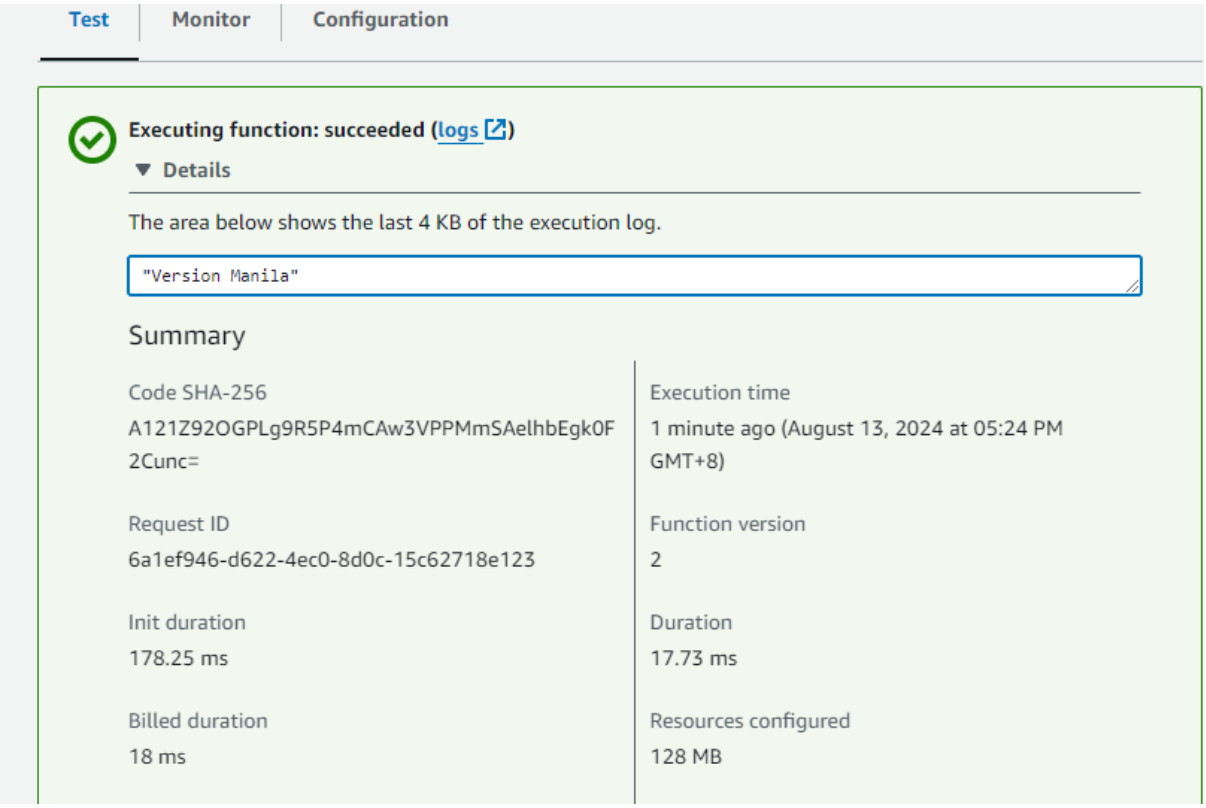
- Change the version it points to from 1 to 2.



- Click **Save** to update the alias.

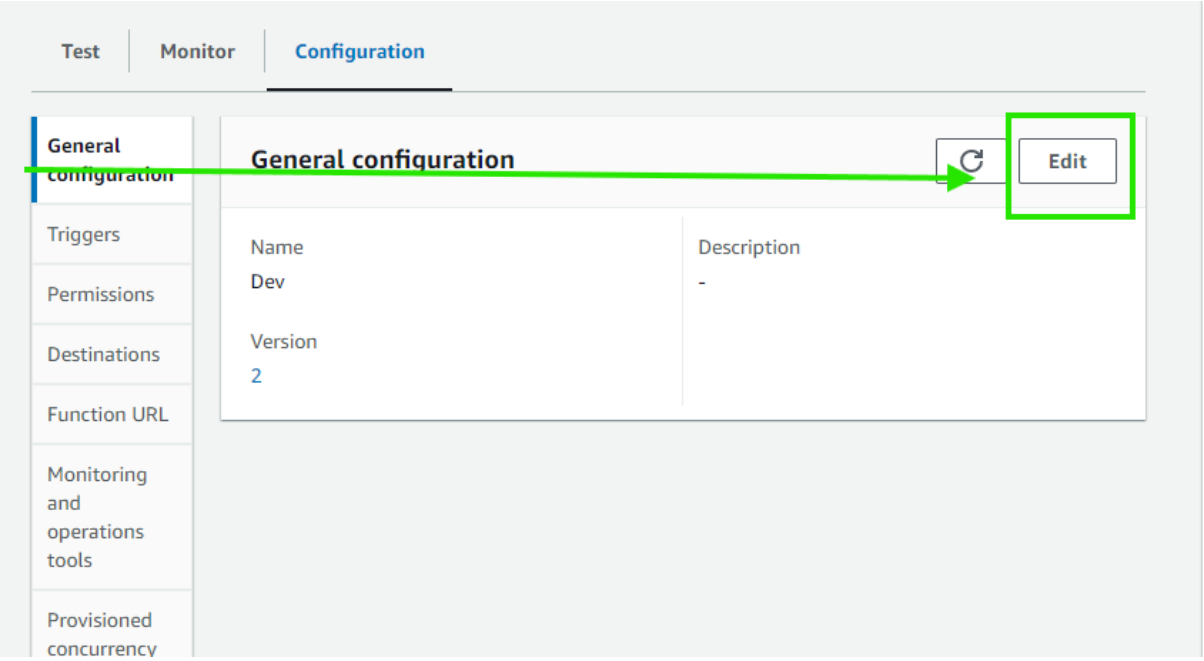


4. Click on **Test** and observe that the function returns "**Version Manila**" the same as our previous test in the Create a New Version Step Section

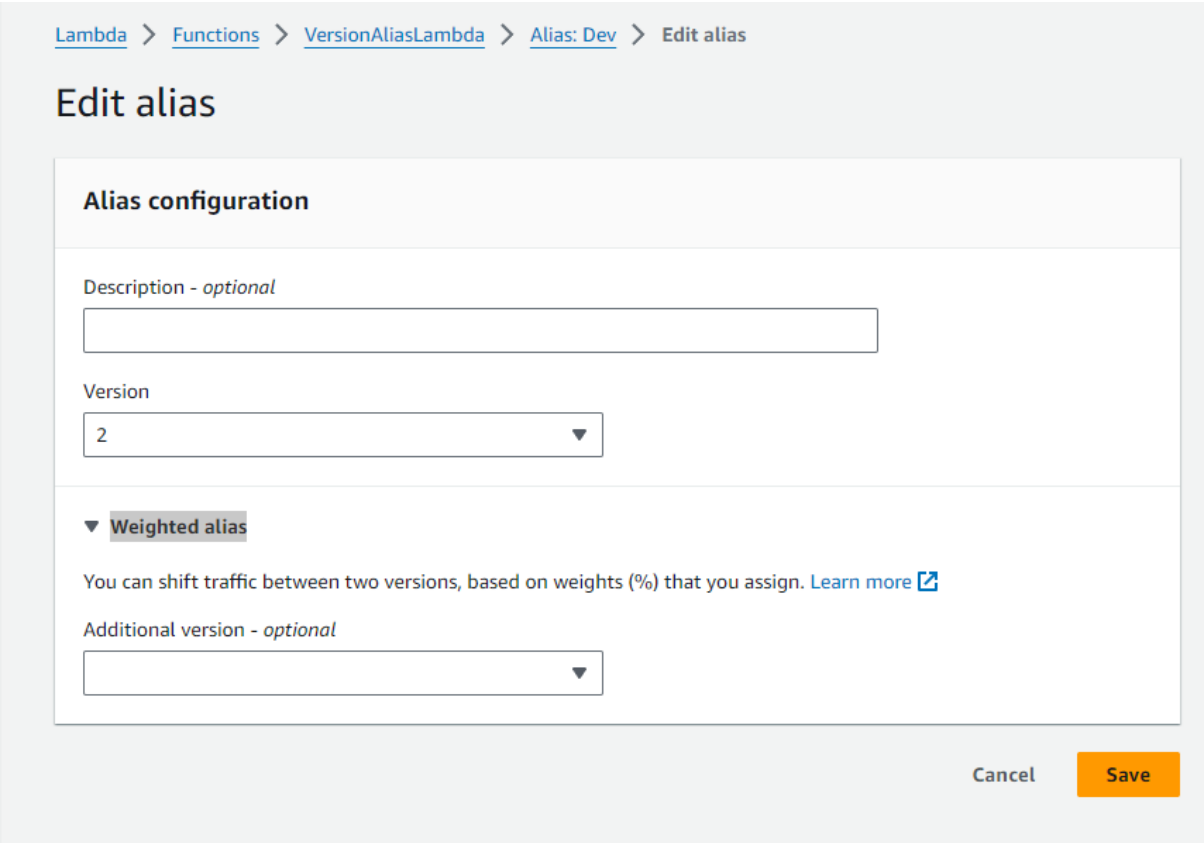


Implement Weighted Traffic Routing

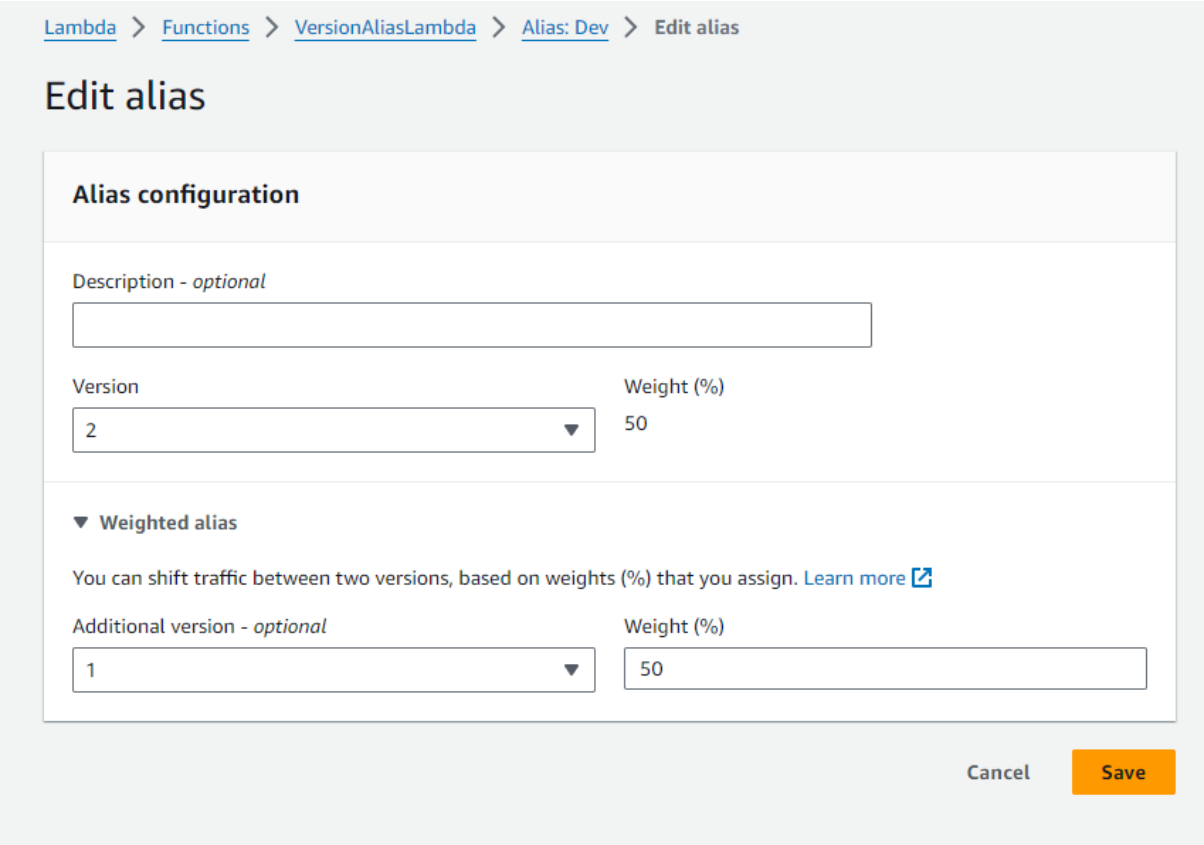
1. Go to the **Configuration tab** and click on **Edit**



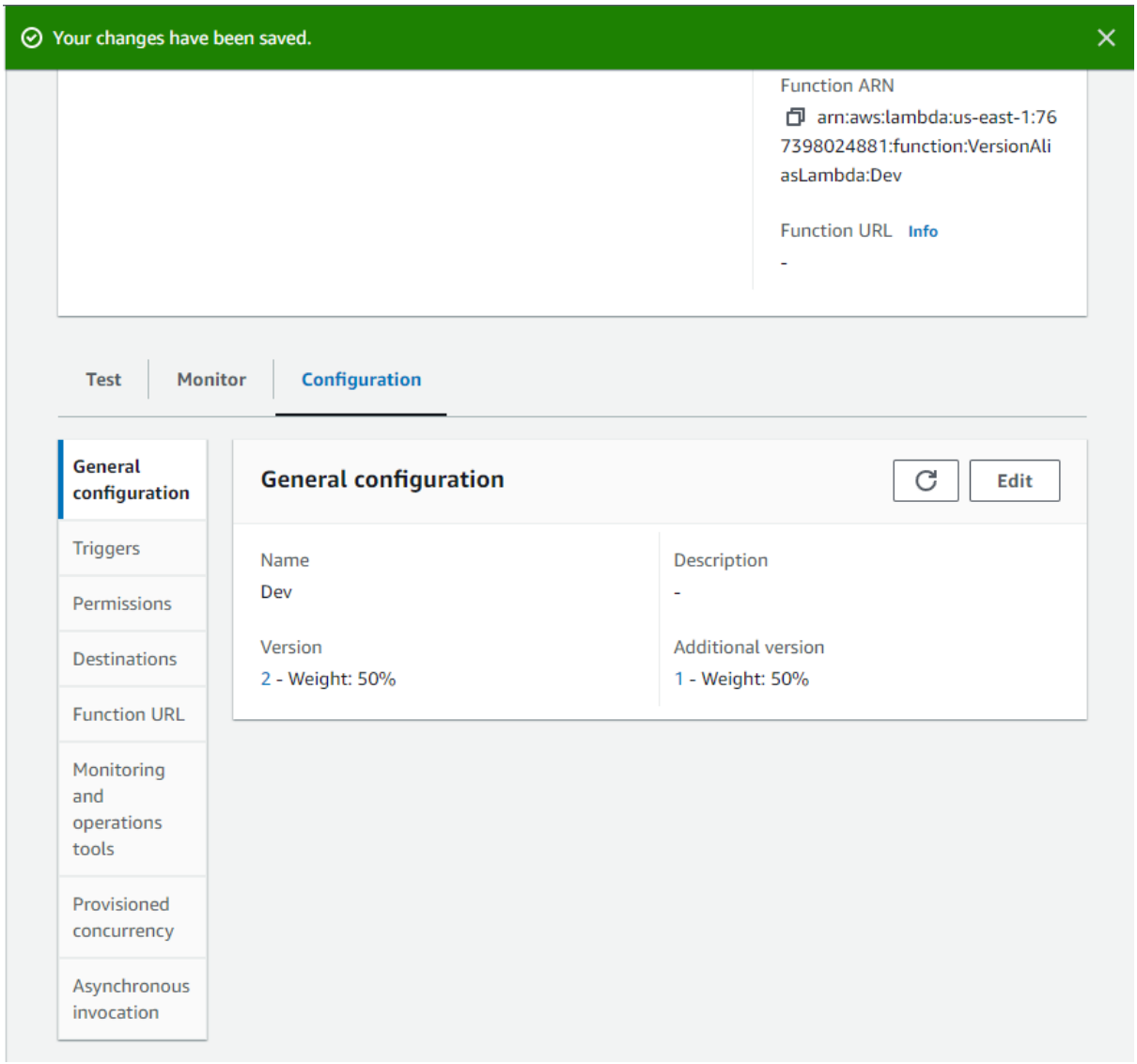
2. Click on **Weighted alias Dropdown**



- Set the traffic weight between the two versions (e.g., 50% to Version 2 and 50% to Version 1).

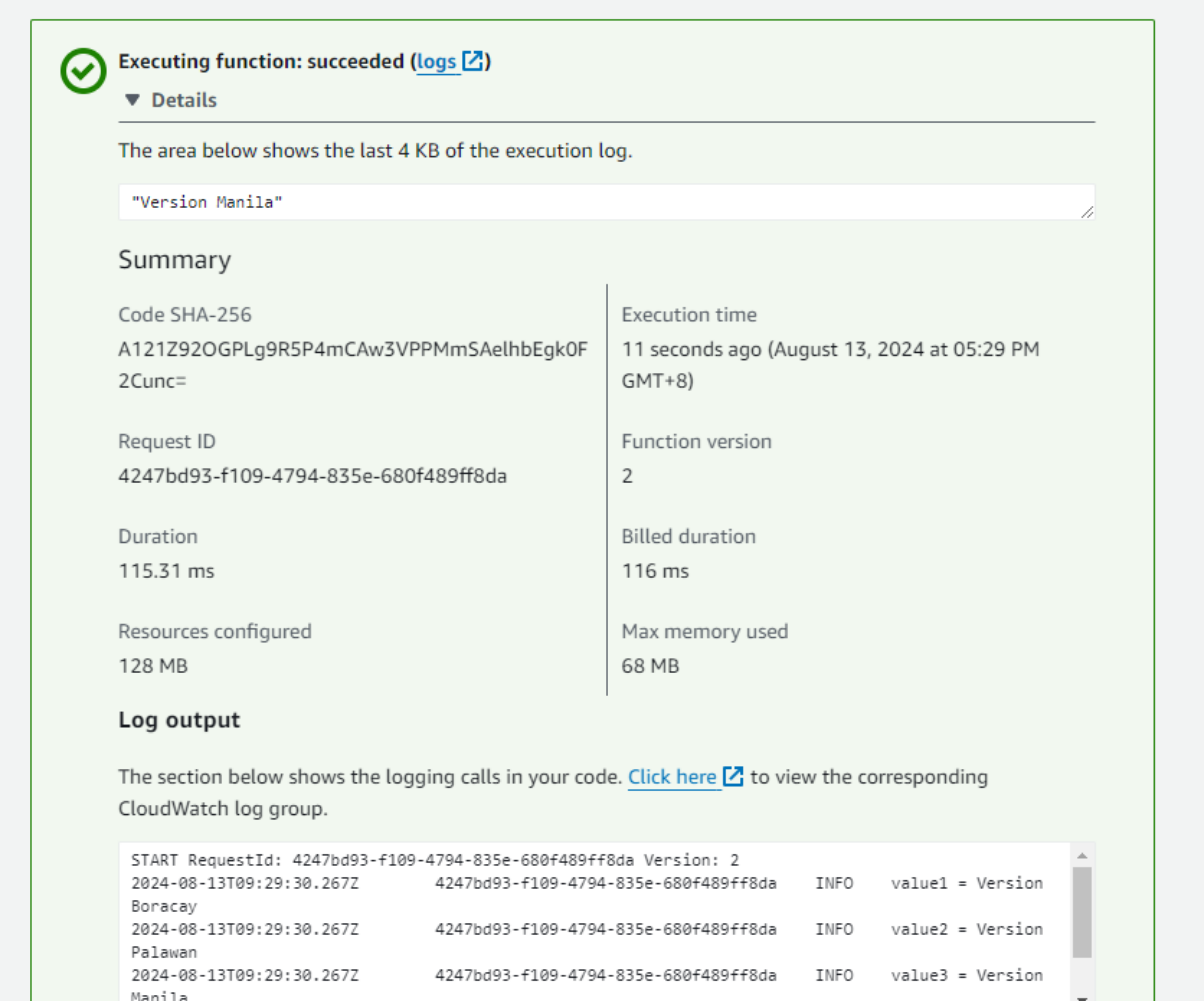


- Click **Save**.

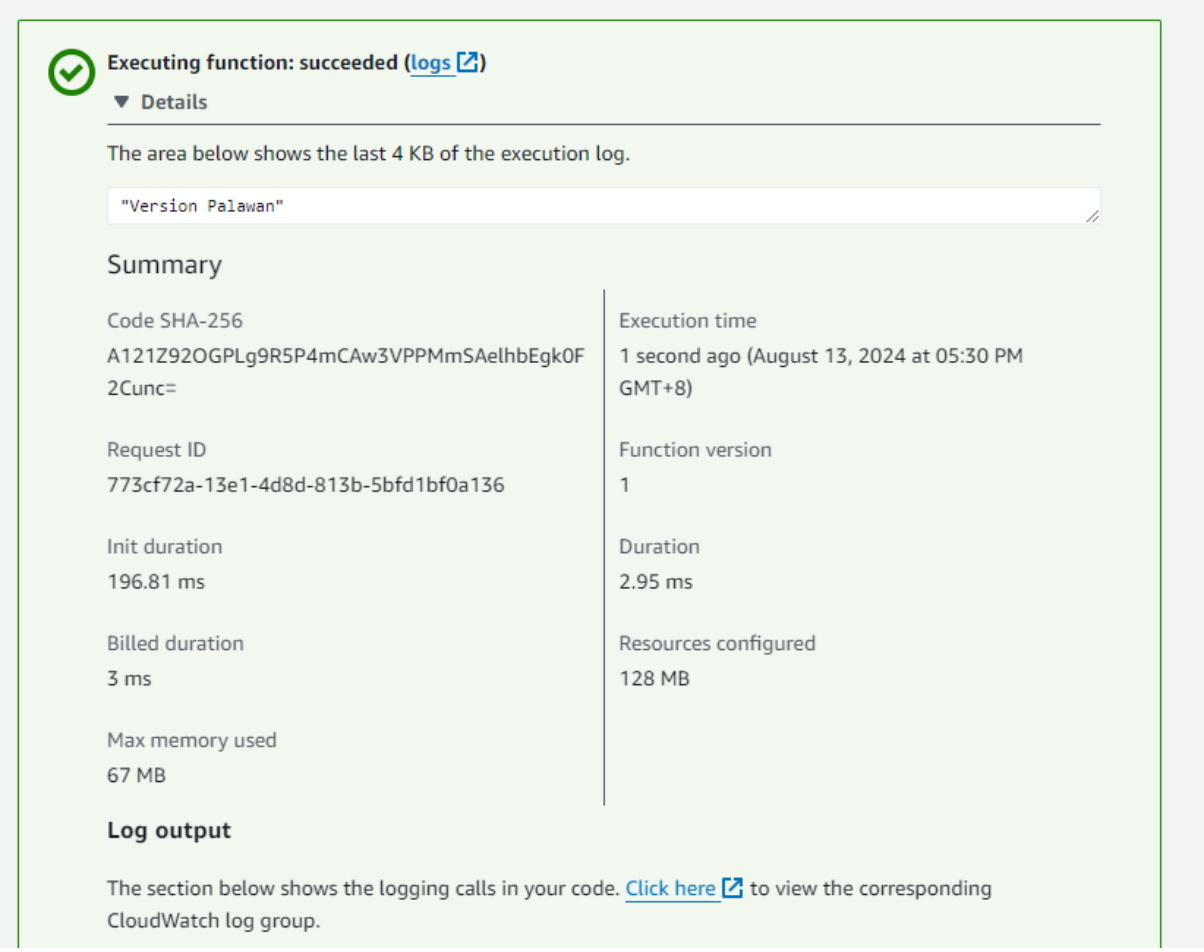


3. **Test the Weighted Routing:**

- Use the **Test** button with the same test event as before.



- Run the test multiple times to observe that the responses are split according to the traffic weight.



That’s it! Congratulations! You just learned how to manage versions and aliases in AWS Lambda, including how to create new versions, set up aliases, and implement weighted traffic routing.

This lab is an introduction to using versions and aliases in AWS Lambda, providing you with essential skills for managing function deployments and testing different versions in a controlled manner. As you continue, you can explore more advanced versioning strategies and alias configurations to further enhance your serverless applications. Happy Learning!