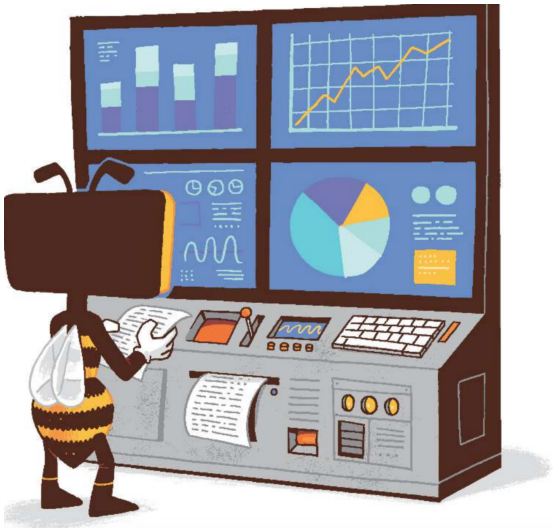
Exploring the AWS Lambda Web Console

In this lesson, you will learn how to check out the configurations of your Lambda functions and how to monitor them using the AWS Lambda Web Console.

> *This chapter explains the basic monitoring and logging features of AWS Lambda to help with troubleshooting. You will also learn how to simulate AWS services locally to speed up development work.*



In the previous chapter, you deployed a 'hello world' service that doesn't do much, but it comes with the full operational support required for serving millions of concurrent users. That's one of the best things about AWS Lambda. A five-minute hack is usually insecure, does not scale well, and generally is not ready for production usage. Monitoring, scaling, fault tolerance, load balancing, and centralized logging come straight out of the box with Lambda. Serverless is like a toy that has the batteries included, instantly ready to play with. (Note, though, that application developers still need to set up the correct security policies, which is covered in Chapter 7)

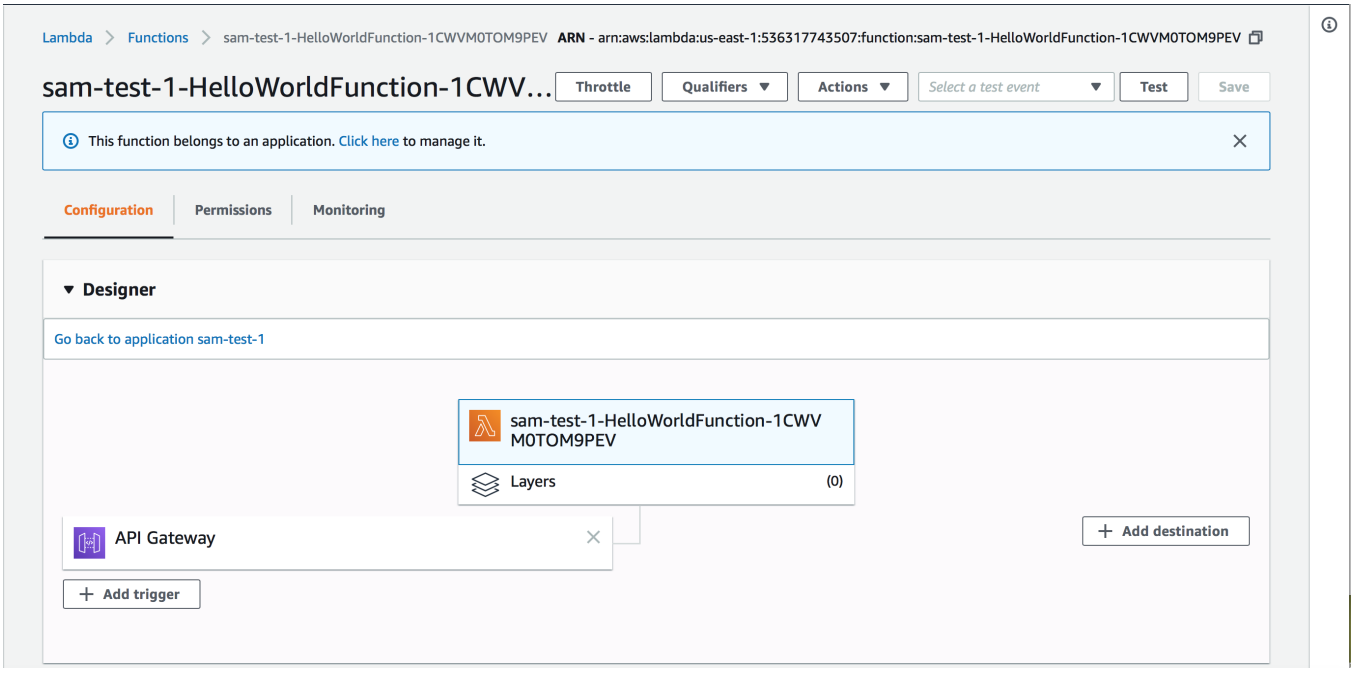
Check out the operational status of your Lambda function by finding it in the AWS Web Console:

1. Open https://aws.amazon.com in a browser, and sign in if necessary.
2. Be sure to select the region where you deployed the stack, in the top-right

corner. For example, if you deployed to `us-east-1`, the right region to select is '*US East (N. Virginia)*'.

3. Select *CloudFormation* from the service list.
4. Select a stack (you created `sam-test-1` in the previous chapter).
5. Open the *Resources* tab and click on the link in the *Physical ID* column corresponding to `HelloWorldFunction` (the one that has `AWS::Lambda::Function` in the `Type` column).

You should see the main screen of the Lambda console for your function, which lists the function configuration. You can use this screen to set function parameters or even edit and replace the source code. Editing function properties directly in the web console might be useful for quick experimentation. I strongly suggest using SAM and CloudFormation to manage functions that you actually care about, in order to perform reliable and reproducible deployments.

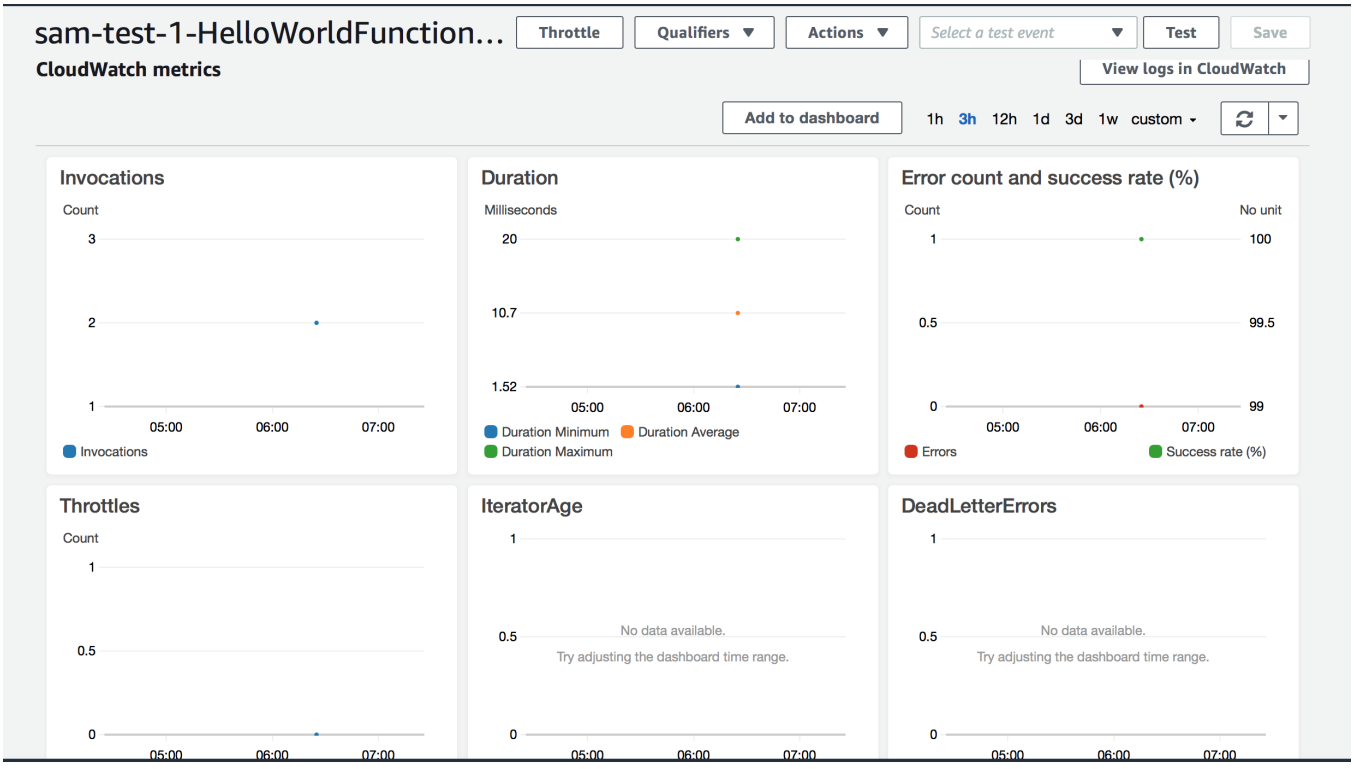


Lambda console shows the configuration of a function, along with authorized incoming and outgoing connections.

The Lambda web console, importantly, shows the services authorised to call a function, below and to the left of the function box. In the figure above, you can see that in the stack so far, only API Gateway can invoke the function. SAM automatically set up the authorisation because the application template contains API events. The web page also shows authorised outgoing connections from a function, below and to the right of the function box. Although the application template does not request any specific outgoing

connections, SAM allowed the function to call one other service, Amazon CloudWatch Logs, as you can see in the figure above. Because Lambda functions run on auto-scaling infrastructure, it's not possible to connect to containers and inspect them directly. System logs are the most important way of observing function behaviour. SAM assumes that, in most cases, developers want to capture function system logs and sets up the CloudWatch Logs service for that purpose. Reliable centralised logging for a highly distributed system is a huge technical challenge, but with Lambda, that comes with the basic setup and is included in the price.

Open the *Monitoring* tab and you'll see a bunch of useful operational information about your function. For example, you should see the number of previous invocations, how long they took to execute, and the number of failed invocations.



The Monitoring tab shows operational statistics, such as invocation and error counts.

In the next lesson, you'll learn how to retrieve execution logs for the Lambda functions. See you there!