

| Business Template  **AWS Lambda and Step Functions** |
| --- |
|  |

# Tasks

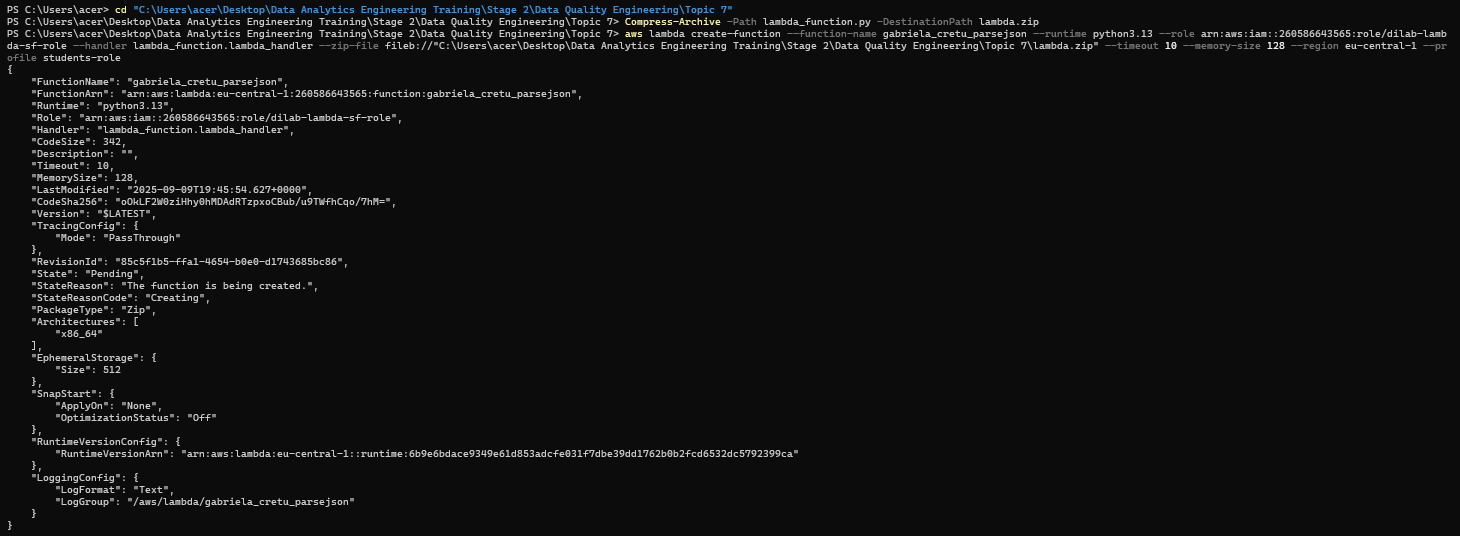
Note: For both tasks, please assign IAM role *dilab-lambda-sf-role* on your Lambda functions and Step Function state machines.

## 1. AWS Lambda service

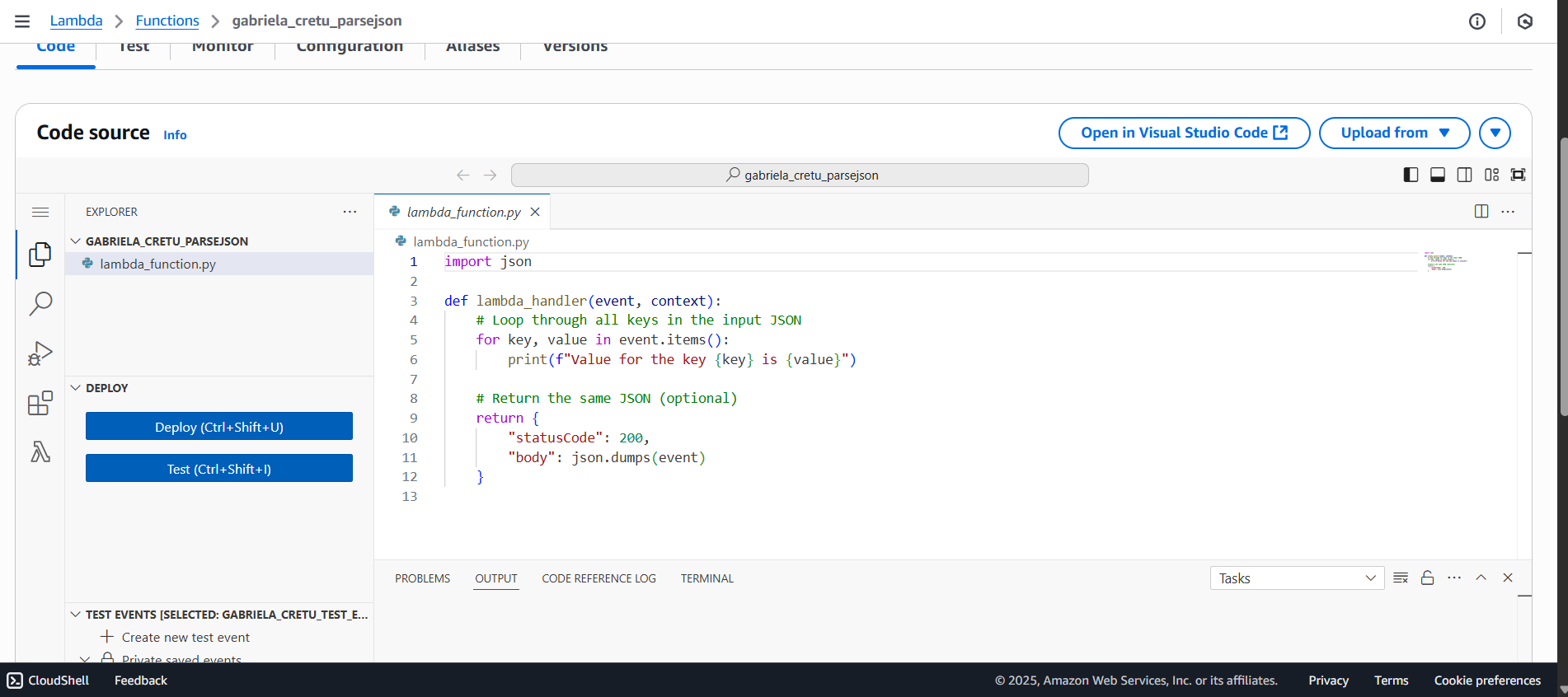
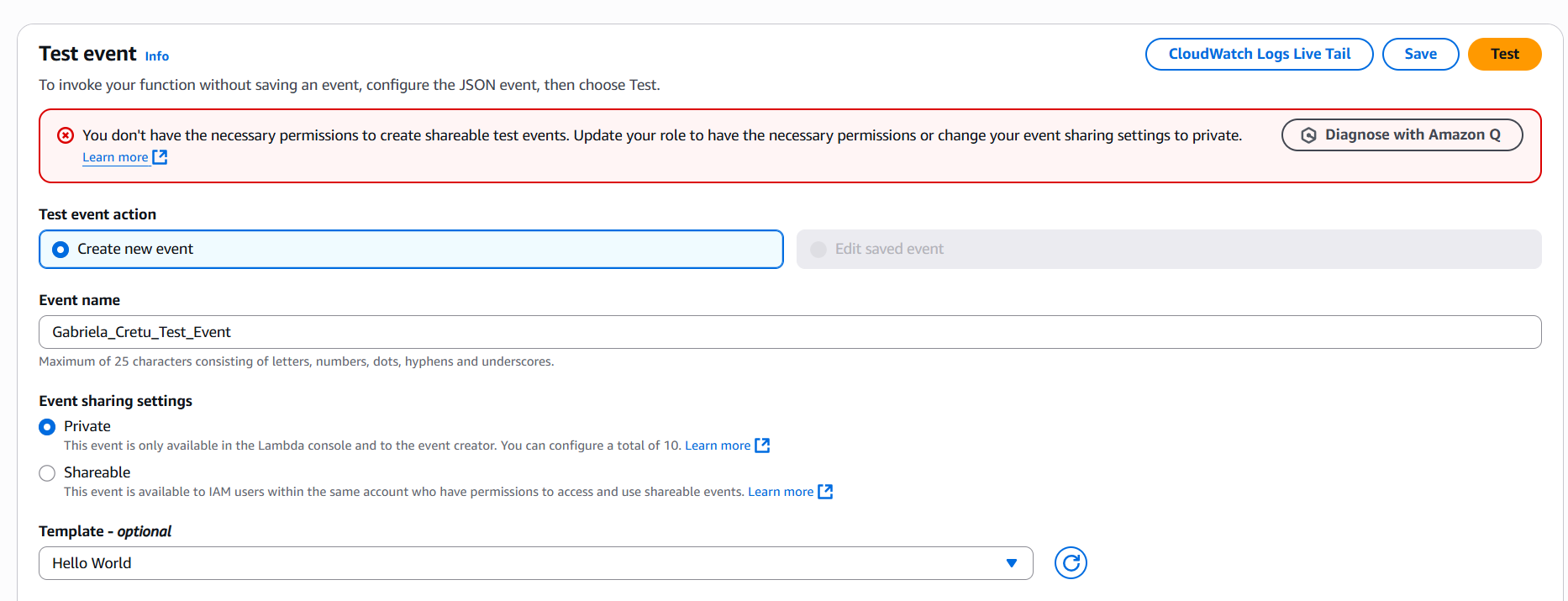
a. Create your first Lambda and show how you will parse test JSON using Response or Function logs. The name mask of Lambda should be <your\_name\_surname>\_<anything\_else>. Example of the result:

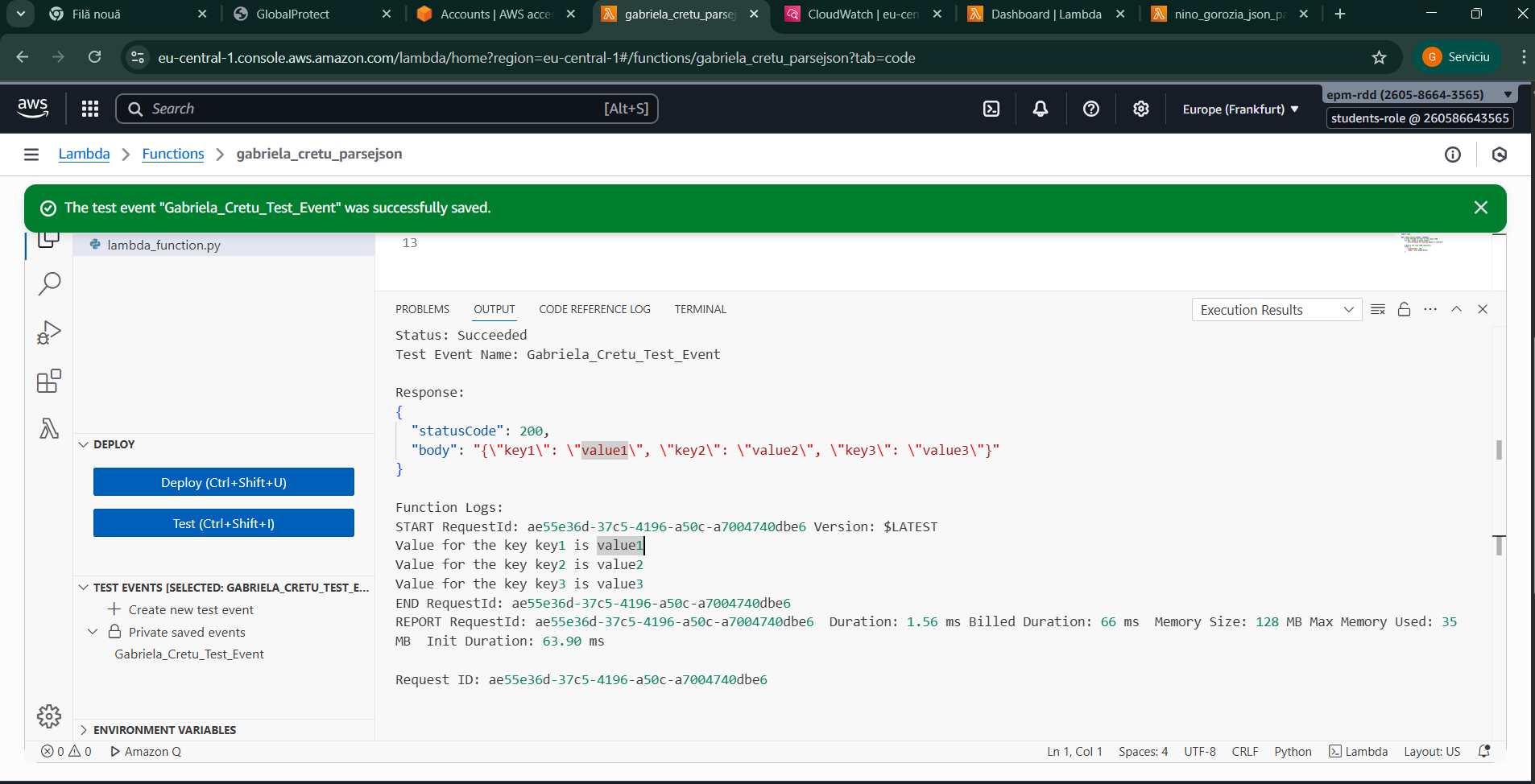
*arn:aws:lambda:eu-central-1:260586643565:function:gabriela\_cretu\_parsejson*

Initially, I created this function using the CLI. I first wrote the Python code in Notepad++ and then zipped it. After preparing the zip file, I uploaded it to the appropriate folder in the CLI. From there, I was able to create the Lambda function based on my Python code.



After that, I created a new test event using the AWS Lambda function creation interface. I then ran the simple function, which executed successfully.

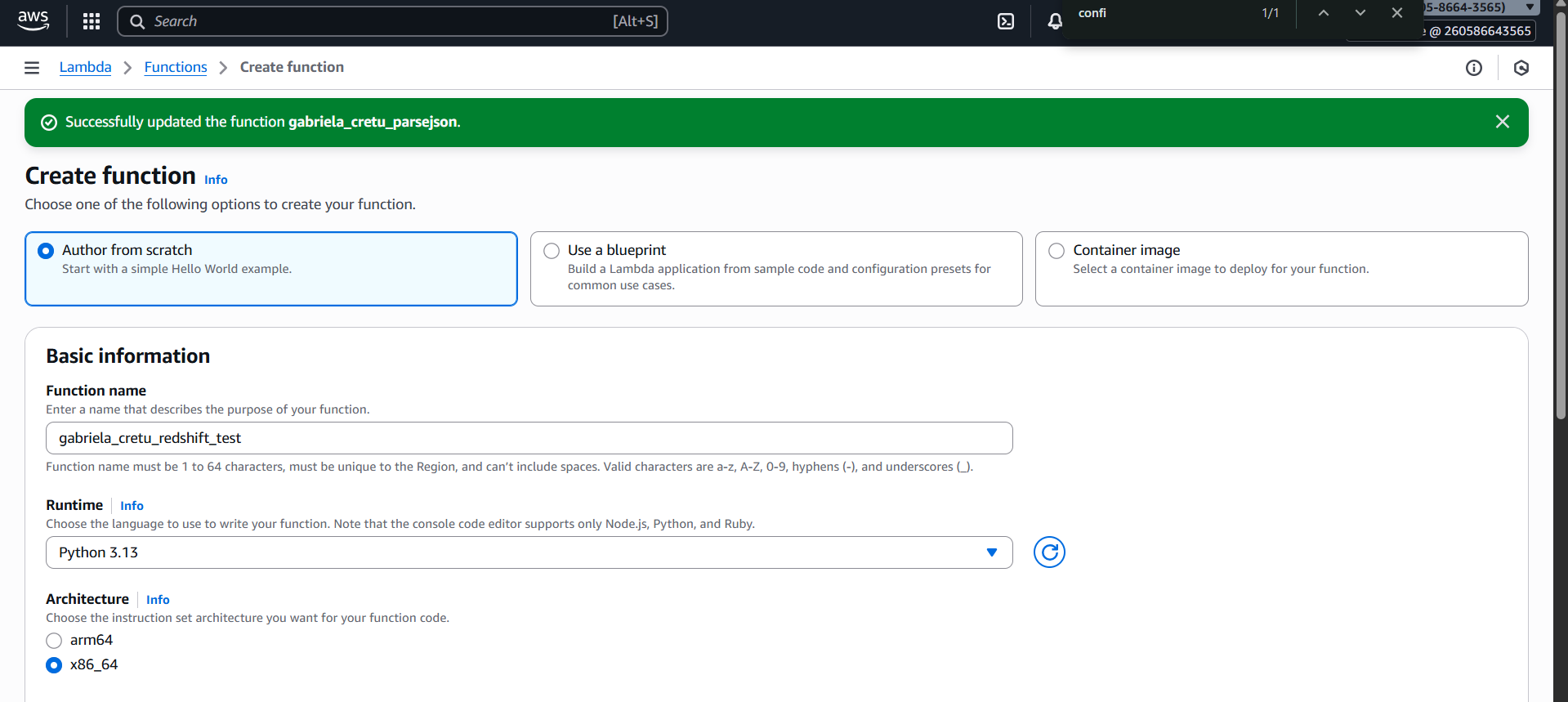




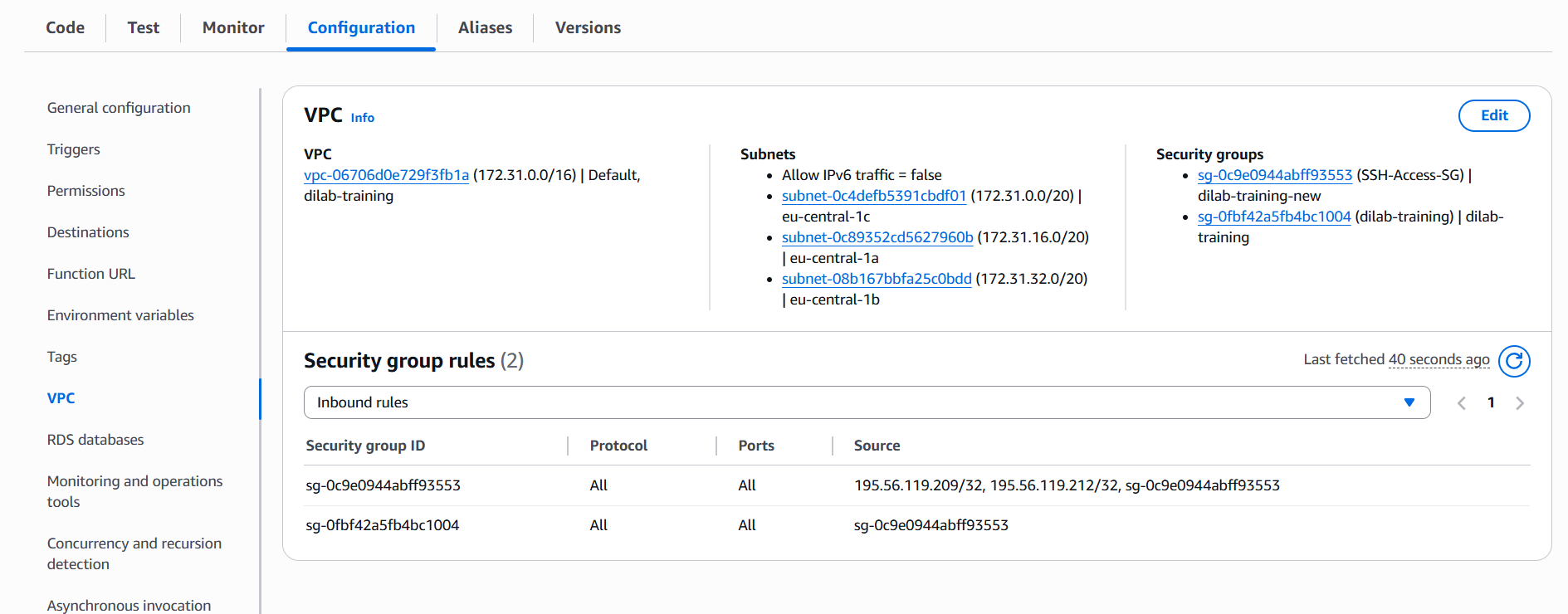
## 2. AWS Step Function service

Note: To connect to Redshift, please assign to Lambda a Layer, VPC, subnets, and security group. You can find respective names in the Lambda function *epam-aleh-shylin-test-lambda.*

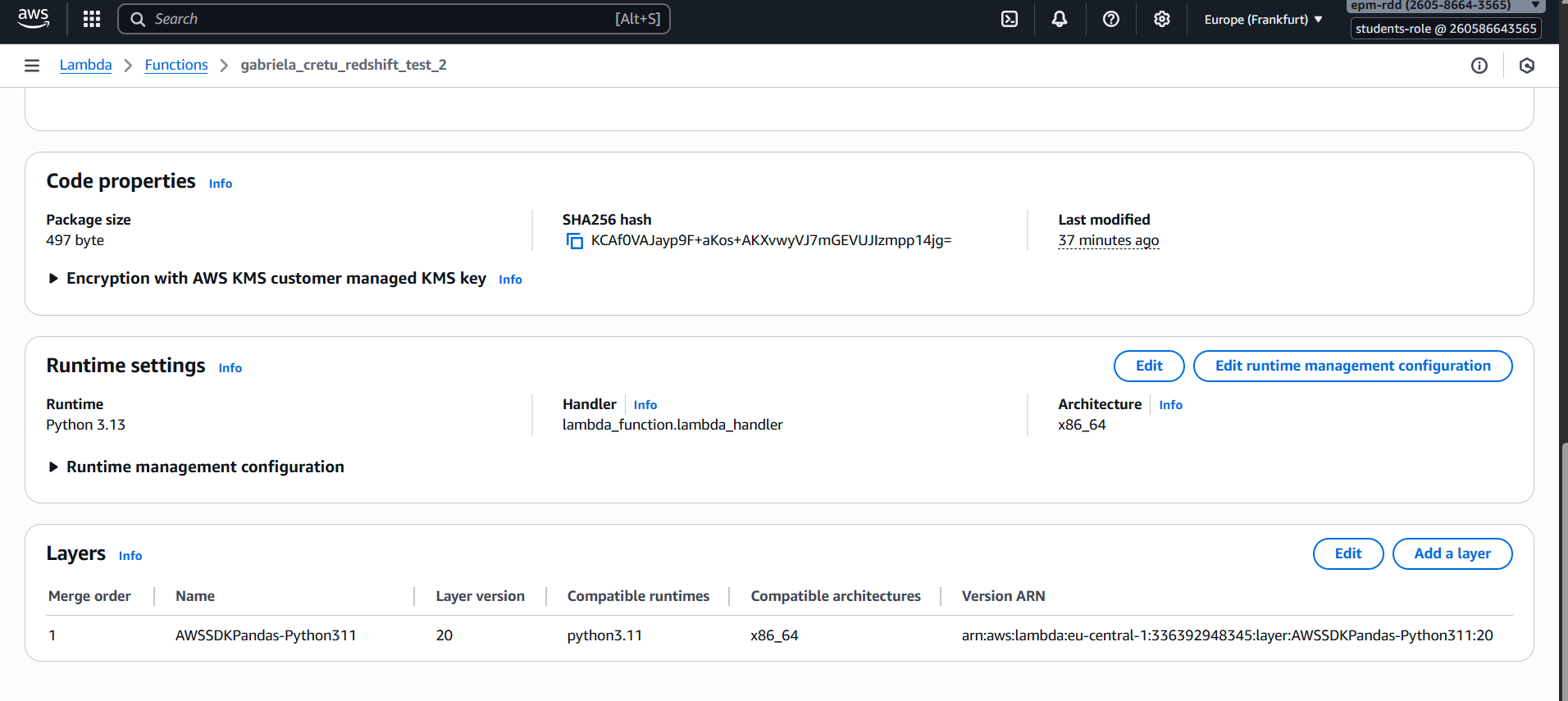
First, here is the format for creating a sample Lambda function. Keep in mind that we also needed to assign the IAM role dilab-lambda-sf-role.

**

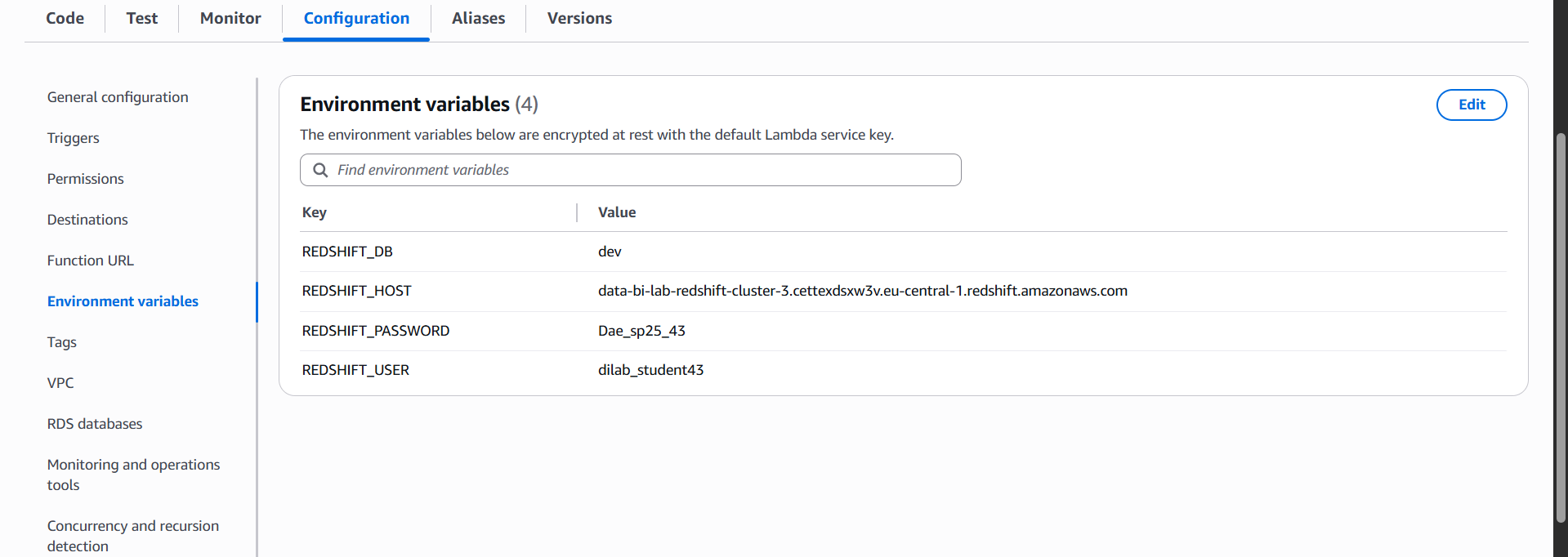
Based on the requirements, I assigned different subnets, security groups, a VPC, and a layer—similar to the setup in the Lambda function epam-aleh-shylin-test-lambda—to each of my Lambda functions. I did this after creating the functions by navigating to the configuration section, selecting VPC, and editing the settings.



Similarly, for the layer, I went to the Layers section in the Lambda console and added my new layer, following the same setup as the epam function.

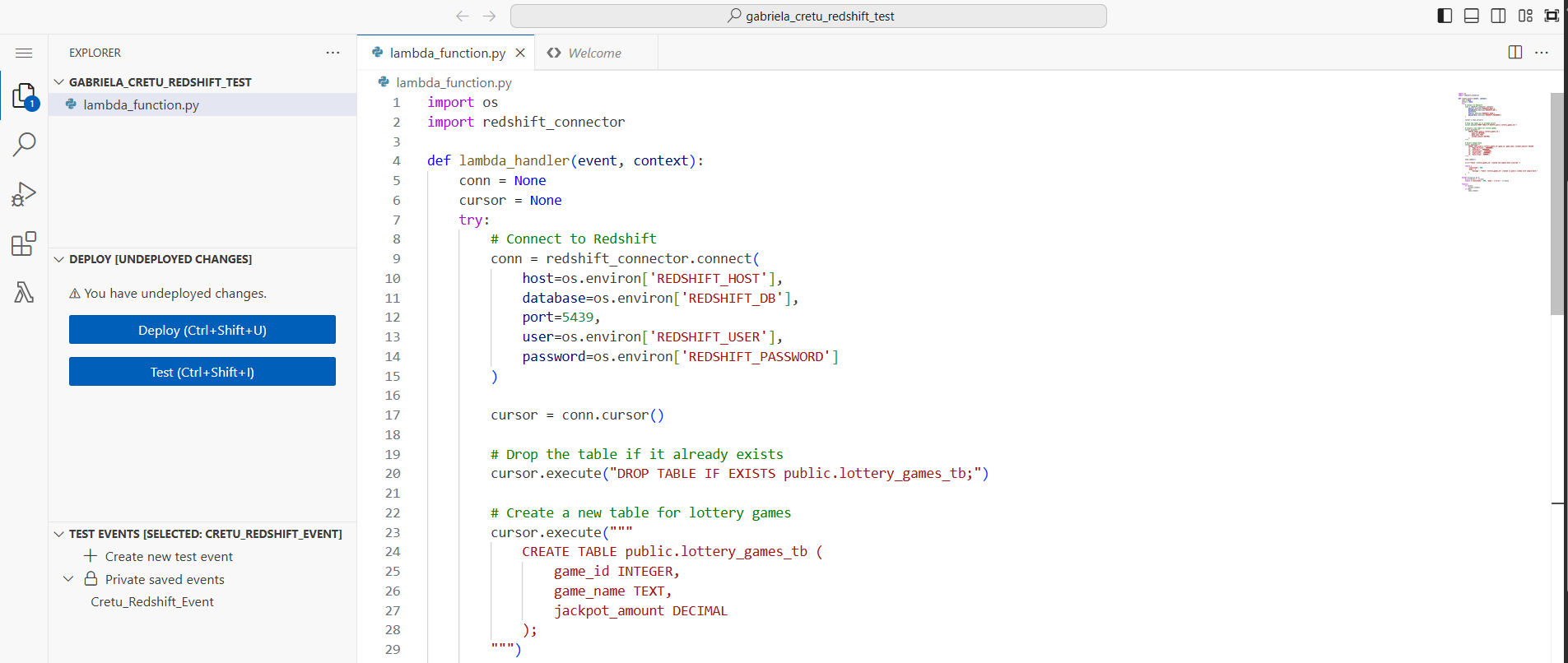


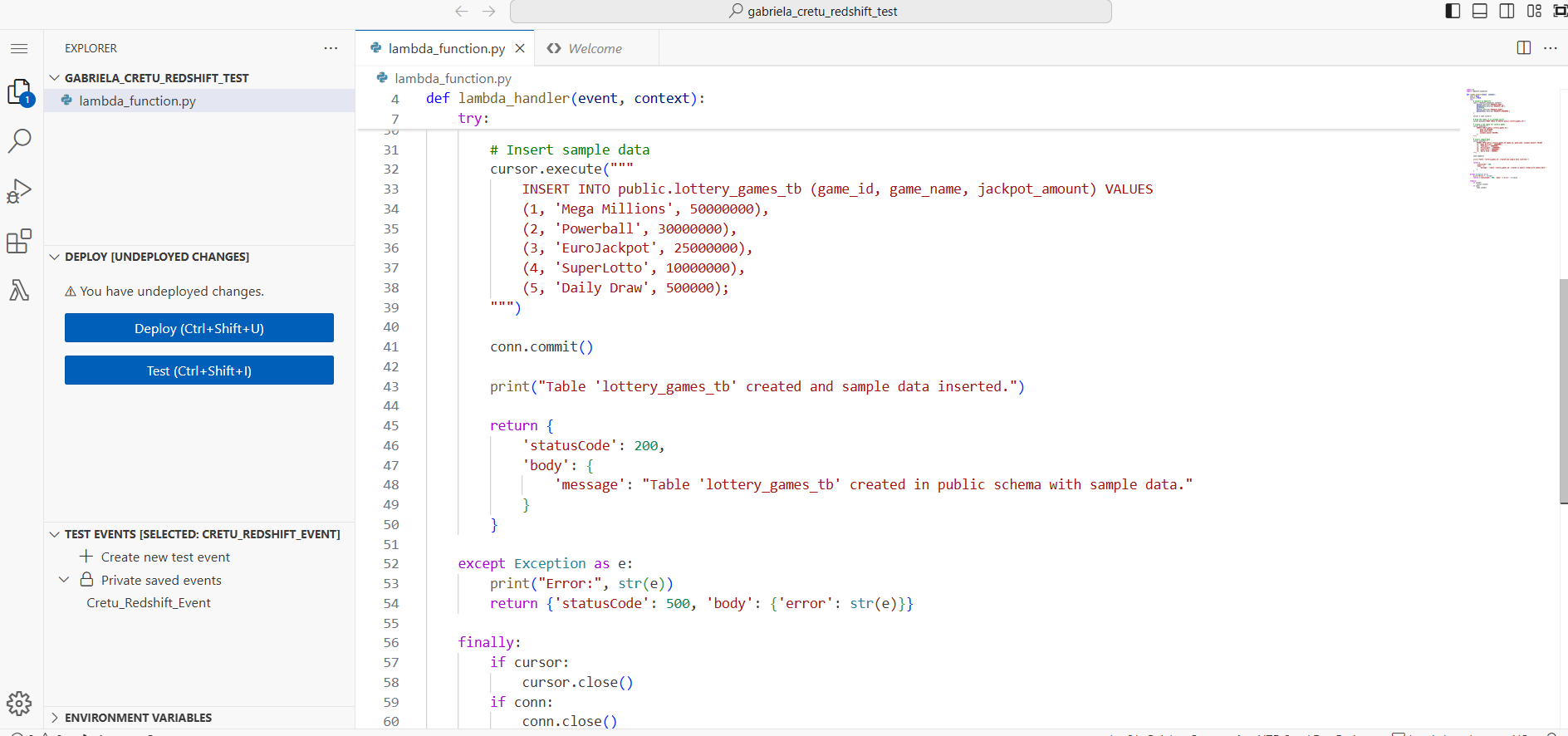
After that, to avoid hardcoding my variables, I defined environment variables. I did this by first accessing the ARN of the active cluster and then using my credentials.



a. Create a Step Function that will have the following steps:

Before running these steps, I executed a separate function that created a new table in my schema—a lottery table. While I was preparing this function, the connection to Redshift failed before executing the rest of the queries, so I will just attach the code.





i. Perform the first test for the table from Redshift using AWS Lambda.

Here is the first piece of code, which essentially checks whether the jackpot prize value for a specific game is 3,000,000.



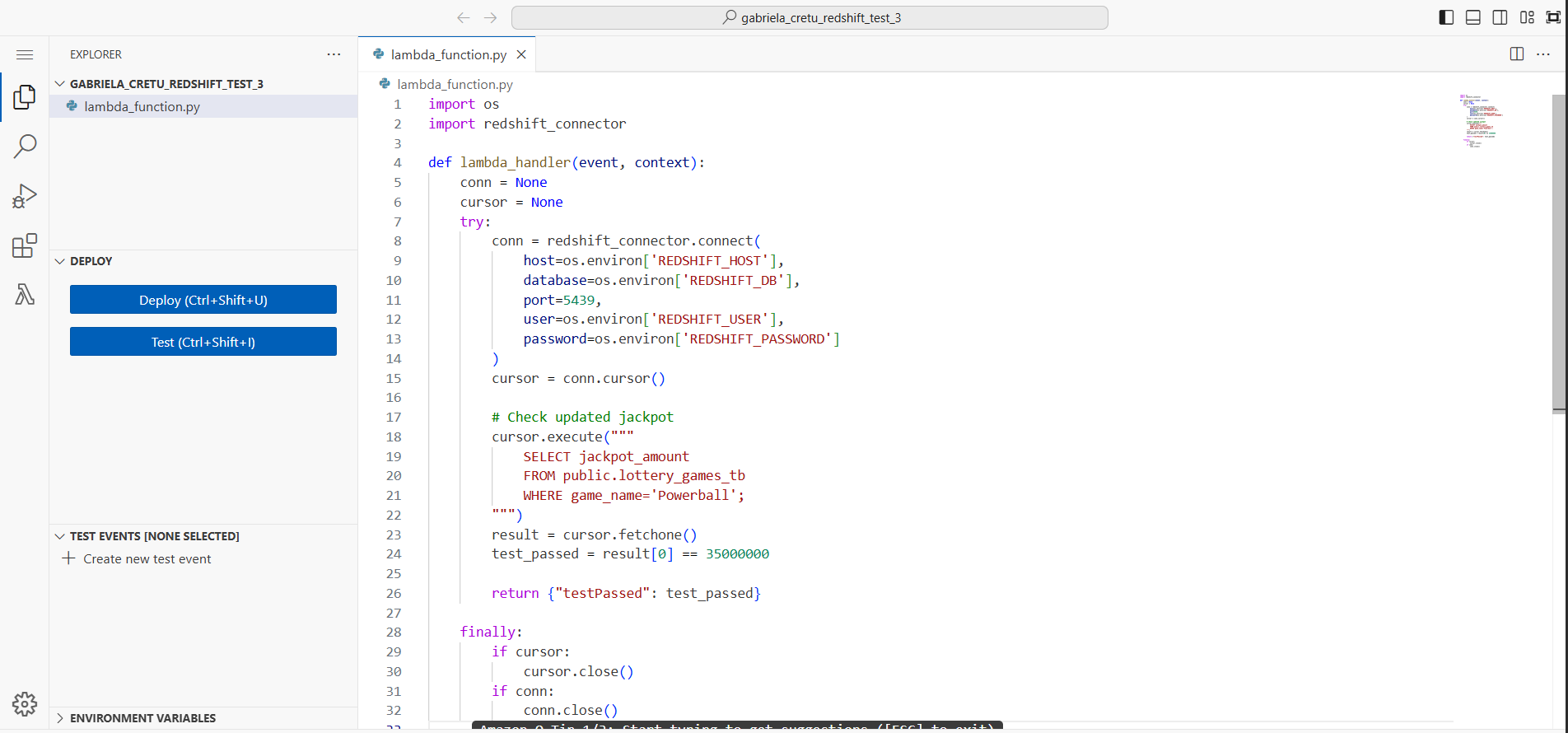
ii. Change data in the Redshift table using AWS Lambda.

Then, I updated the value to 3,500,000.



iii. Perform a second test for the same table from Redshift using AWS Lambda

After that, I verified that the value had been updated correctly.



iv. If the first and third steps are passed then Step Function completes successfully, otherwise – Step Function fails.

Here, you can see the workflow of my Step Function.

arn:aws:iam::260586643565:role/dilab-lambda-sf-role

