operationalizing-an-aws-ml-project

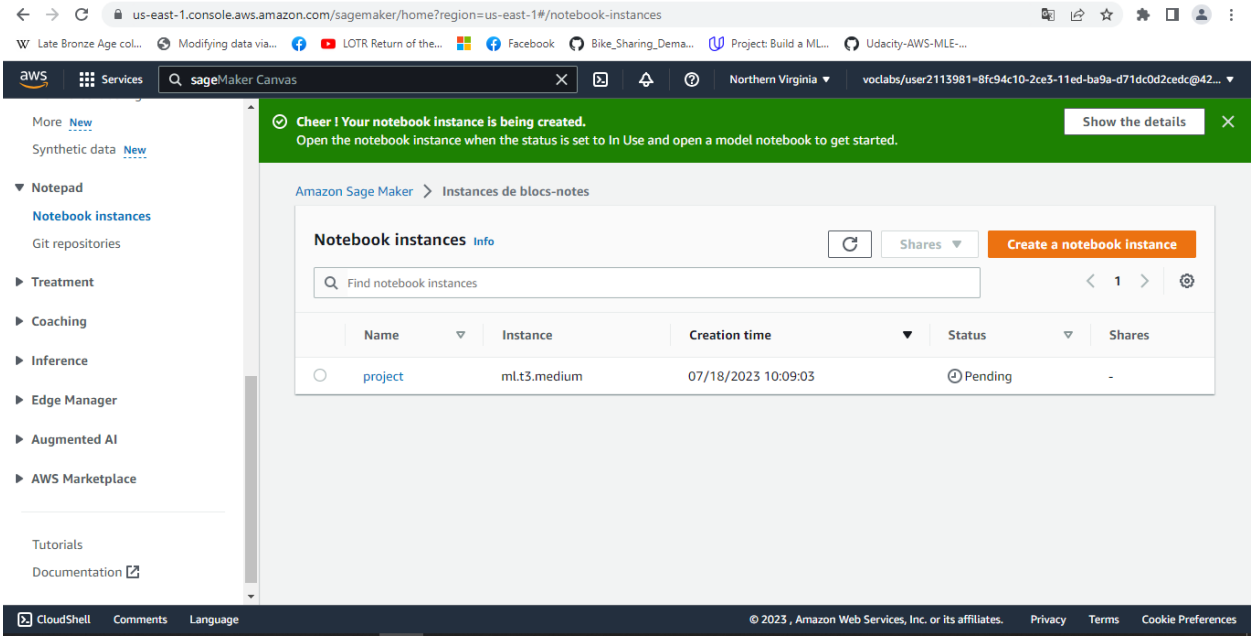
Dog Image Classification

In this project, you will accomplish the following tasks:

1. Utilize SageMaker to train and deploy a model, selecting the most suitable instance types. Configure multi-instance training within your SageMaker notebook.
2. Modify your SageMaker notebooks to facilitate training and deployment on EC2 instances.
3. Establish a Lambda function associated with your deployed model. Configure autoscaling for the deployed endpoint and manage concurrency for the Lambda function.
4. Ensure that the security on your ML pipeline is set up properly

Step 1: Training and deployment on Sagemaker

Created sagemaker notebook instance, I selected an ml.t3.medium instance for the notebook. The ml.t3.medium instance type is a balanced choice offering a good blend of memory, CPU, and cost efficiency. It comes with 2 virtual CPUs and 4GB of memory, making it a suitable choice for running Jupyter notebooks and conducting exploratory data analysis. This is generally sufficient for most data pre-processing tasks and developing the model before training.

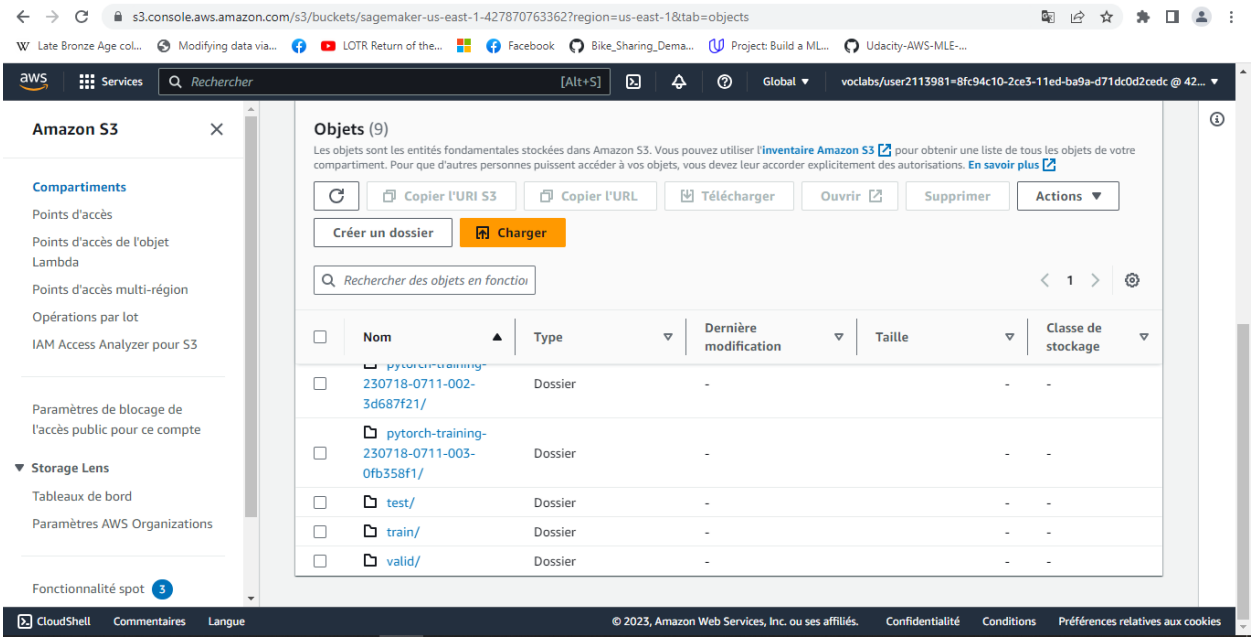


And I used a S3 bucket name "sagemaker-us-east-1-427870763362" and upload data into it using the following code:

!wget https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/dogImages.zip

!unzip dogImages.zip

!aws s3 cp dogImages s3://sagemaker-us-east-1-427870763362/ --recursive



For this model, there are two hyperparameters: learning rate and batch size.

```
hyperparameter_ranges = {
        "learning_rate": ContinuousParameter(0.001, 0.1),
        "batch_size": CategoricalParameter([32, 64, 128, 256, 512]),
}
```

I used a py script (hpo.py) as entry point to the estimator, this script contains the code need to train model with different hyperparameters values.

```
estimator = PyTorch(
entry_point="hpo.py",
base_job_name='pytorch_dog_hpo',
role=role, framework_version="1.4.0",
instance_count=1,
instance_type="ml.g4dn.xlarge",
py_version='py3' )
```

tuner = HyperparameterTuner(

estimator, objective_metric_name,

hyperparameter_ranges, metric_definitions,

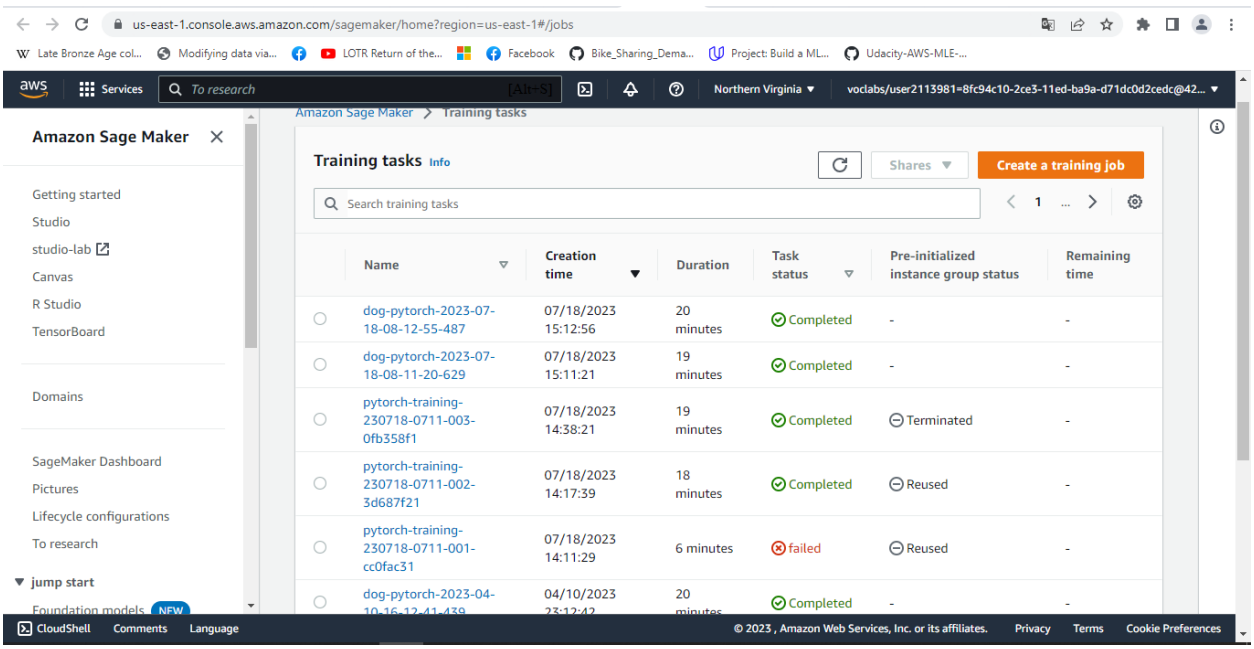
max_jobs=2,

max_parallel_jobs=1, # you once have one ml.g4dn.xlarge instance available
objective_type=objective_type)

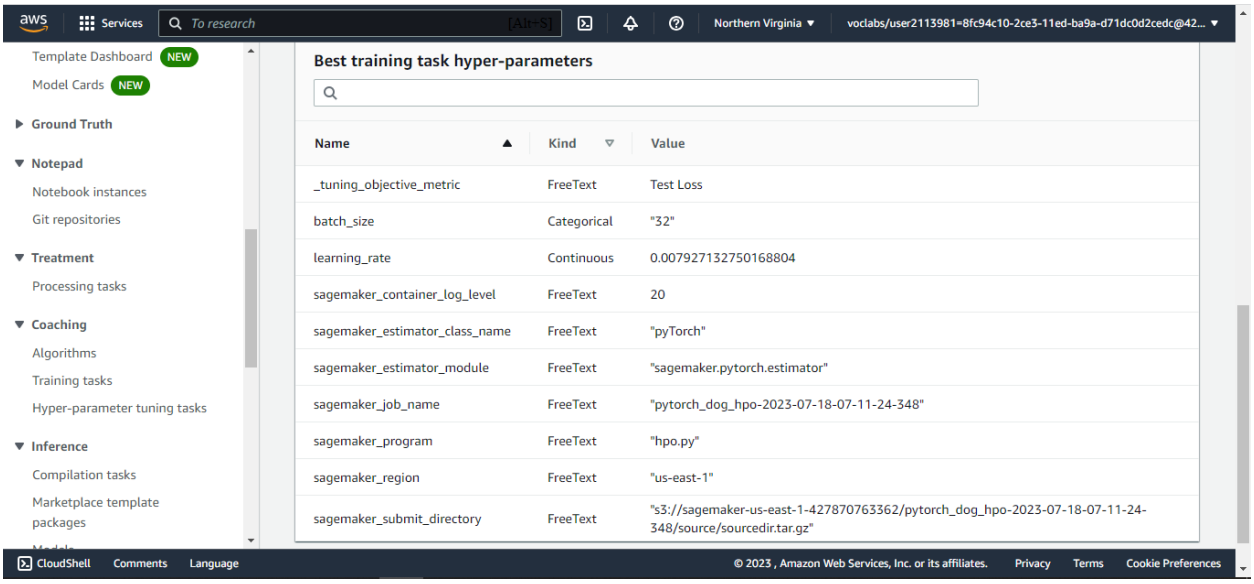
Here I passed some paths to our S3 which will be used by the notebook instance to get data, save model and output

os.environ['SM_CHANNEL_TRAINING']='s3:// sagemaker-us-east-1-427870763362/'
os.environ['SM_MODEL_DIR']='s3:// sagemaker-us-east-1-427870763362/model/'
os.environ['SM_OUTPUT_DATA_DIR']='s3:// sagemaker-us-east-1-427870763362/output/'
tuner.fit({"training": "s3:// sagemaker-us-east-1-427870763362/"})

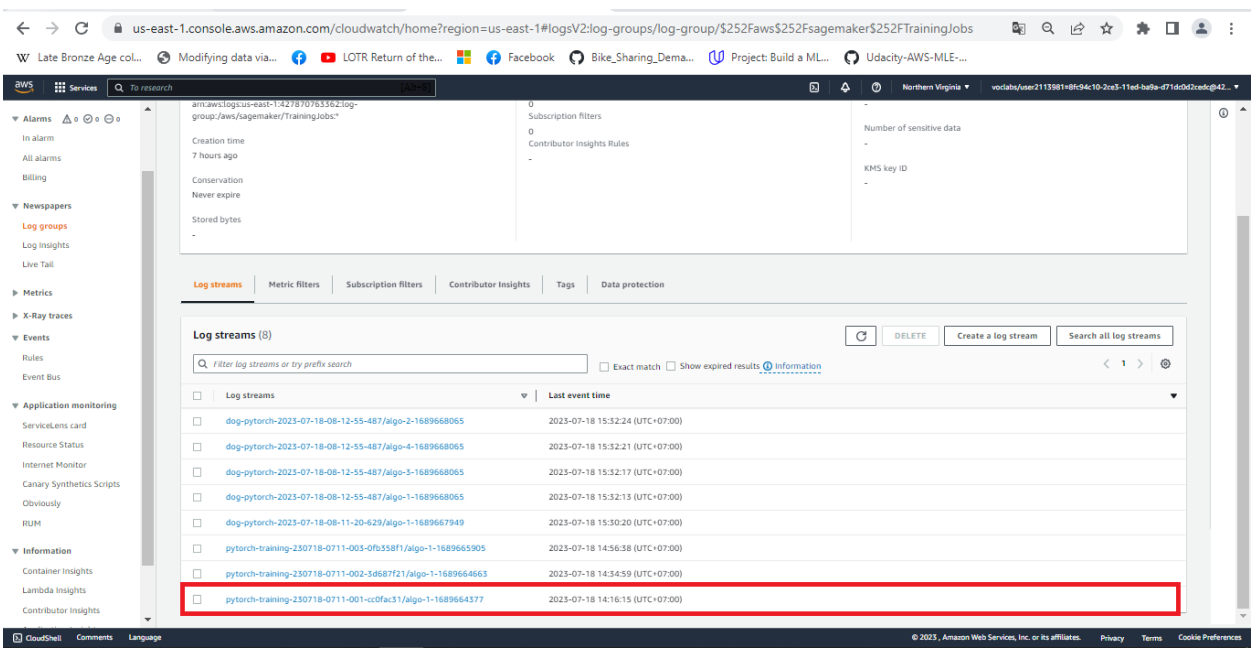
I started the model training we can see the training job status at SageMaker -> Training -> Training Jobs



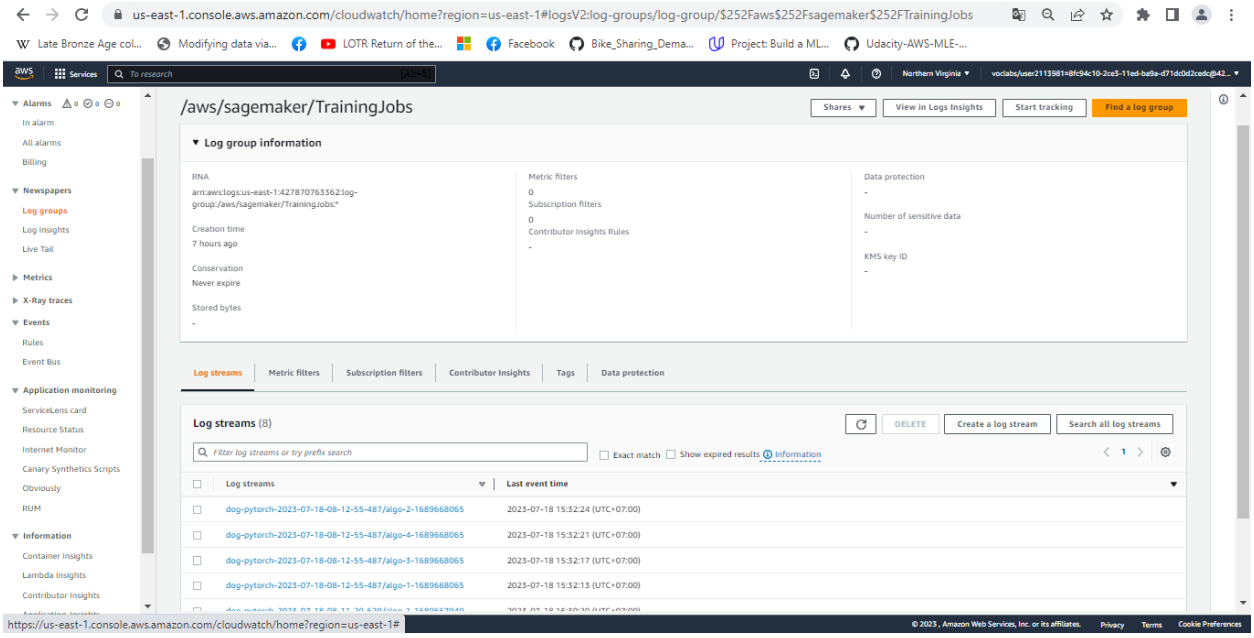
I got the best model:



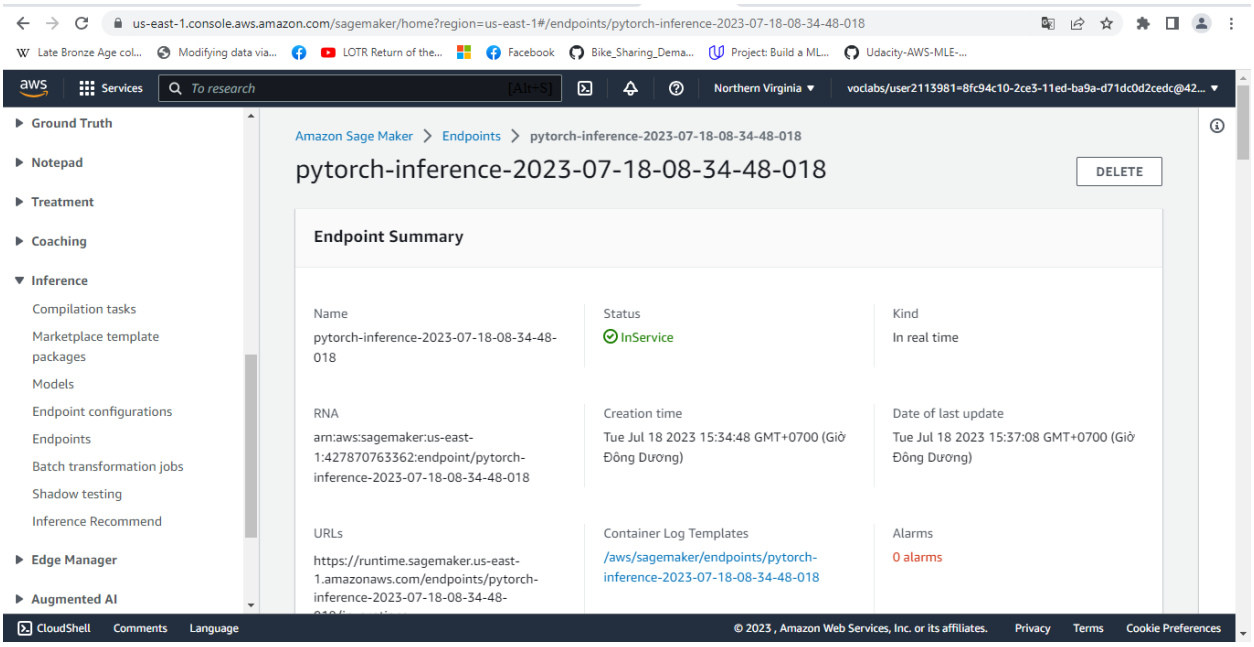
- **Single instance training with best hyperparameters values**



- **Multi-instance training with best hyperparameters values (4 instances)**

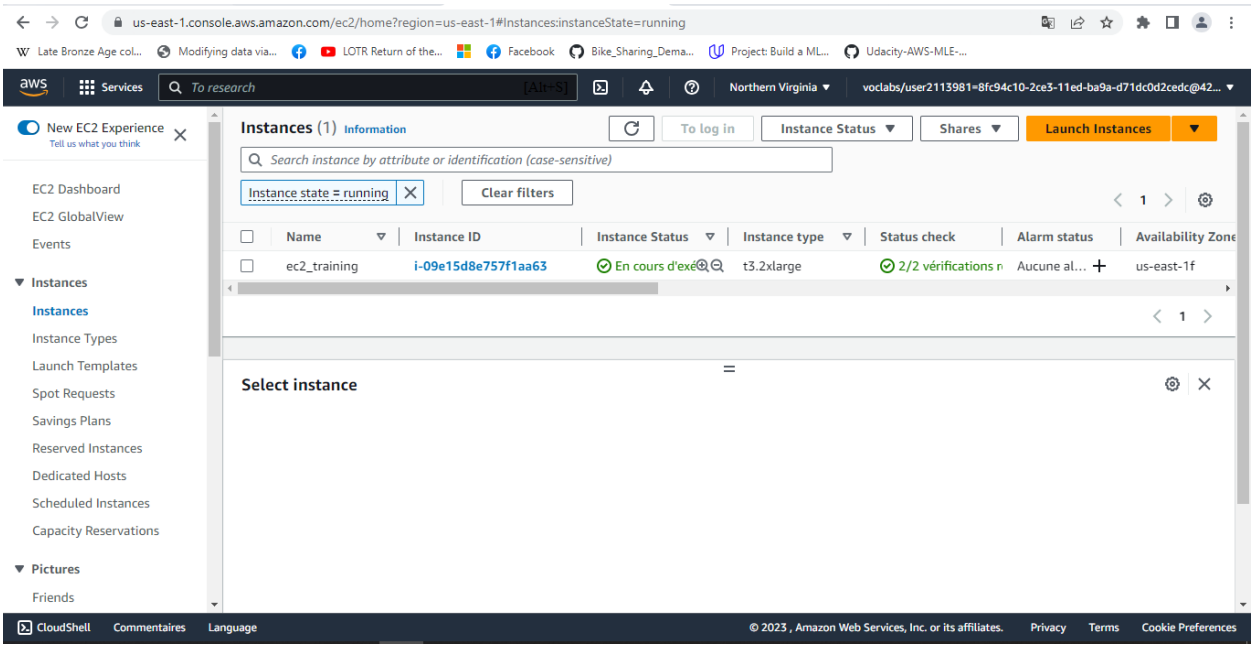


- **Deployment**

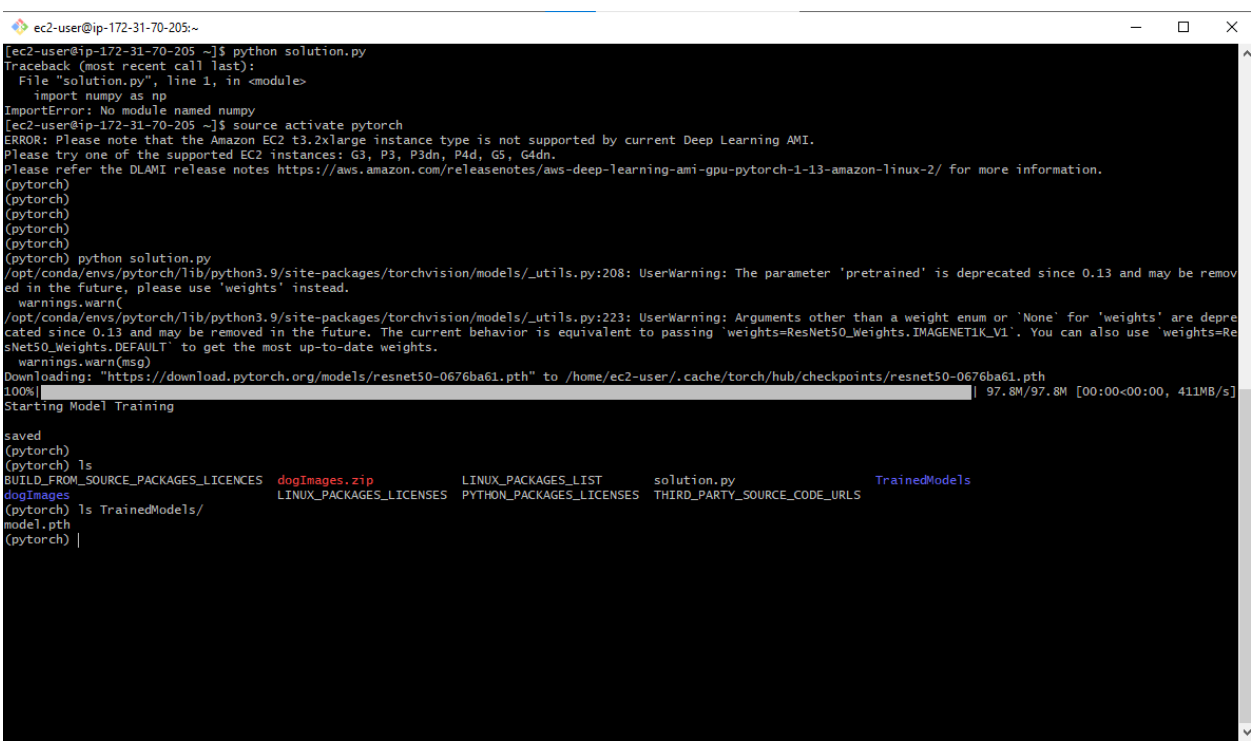


Step 2: EC2 Training

I chose AMI with "Deep Learning AMI GPU PyTorch 2.0" and instance type selected was t3.2xlarge because it is low cost and sufficient to train model.

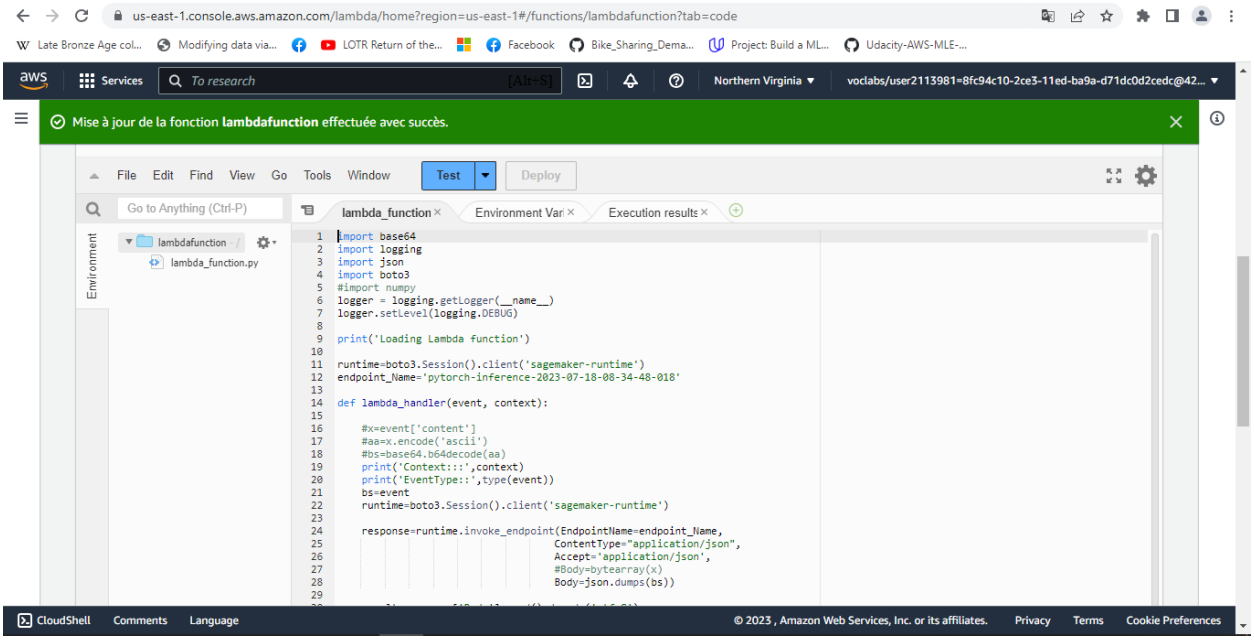


After that, I conduct training job on ec2 instance



Step 3: Lambda function setup

After training and deploying your model, setting up a Lambda function is an important next step. Lambda functions enable your model and its inferences to be accessed by API's and other programs, so it's a crucial part of production deployment.



Step 4: Lambda policy and testing

- Vulnerability Assessment:

Granting "Full Access" privileges to roles or accounts can introduce significant security risks. It is important to follow the principle of least privilege and only provide the necessary permissions required for a specific role or account to perform its intended tasks. Granting "Full Access" unnecessarily can potentially be exploited by malicious actors if they gain access to those roles or accounts.

Old and inactive roles also pose security risks. These roles may still have permissions associated with them, and if they are compromised, an attacker can potentially use them to gain unauthorized access to resources or perform malicious actions. It is essential to regularly review and delete any old or unused roles to mitigate these risks.

Two security policy has been attached to the role are:

Basic lambda function execution

Sagemaker endpoint invocation permission

```
{
"Version": "2012-10-17",
"Statement": [
```

{
"Sid": "VisualEditor0",
"Effect": "Allow",
"Action": "sagemaker:InvokeEndpoint",
"Resource": "arn:aws:sagemaker:us-east-1:856800247221:endpoint/pytorch-inference-2023-07-17-11-56-14-371"
}
]
}

- Creating policy with permission to only invoke specific endpoint

us-east-1.console.aws.amazon.com/iamv2/home?region=us-east-1#/policies

Late Bronze Age col... Modifying data via... LOTR Return of the... Facebook Bike_Sharing_Dema... Project: Build a ML... Udacity-AWS-MLE-...

aws Services Search [Alt+S] Global voclabs/user2113981=8fc94c10-2ce3-11ed-ba9a-d71dc0d2cedc @ 42...

Identity and Access Management (IAM)

Unable to load search

- Dashboard
- Access management
  - User groups
  - Users
  - Roles
  - Policies
  - Identity providers
  - Account settings
- Access reports
  - Access analyzer
    - Archive rules
    - Analyzers

IAM > Policies

Policies (1111) Info

A policy is an object in AWS that defines permissions.

Actions | Create policy

Filter policies by property or policy name and press enter.

1 2 3 4 5 6 7 ... 56

Policy name	Type	Used as
AmazonSageMaker-ExecutionPolicy-20230718T093807	Customer managed	Permissions polic
AWSLambdaBasicExecutionRole-c028a06e-9a2b-4178-9175-8463bc874...	Customer managed	Permissions polic
Pvoclabs1	Customer managed	Permissions polic
Pvoclabs2	Customer managed	Permissions polic
Pvoclabs3	Customer managed	Permissions polic
robomaker_students	Customer managed	None
voc-cancel-cred	Customer managed	Permissions polic

CloudShell Feedback Language © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

us-east-1.console.aws.amazon.com/iamv2/home?region=us-east-1#/roles/details/AmazonSageMaker-ExecutionRole-20230718T093807?section=permissions

Late Bronze Age col... Modifying data via... LOTR Return of the... Facebook Bike_Sharing_Dema... Project: Build a ML... Udacity-AWS-MLE-...

aws Services Search [Alt+S] Global voclabs/user2113981=8fc94c10-2ce3-11ed-ba9a-d71dc0d2cedc @ 42...

Identity and Access Management (IAM)

Unable to load search

- Dashboard
- Access management
  - User groups
  - Users
  - Roles
  - Policies
  - Identity providers
  - Account settings
- Access reports
  - Access analyzer
    - Archive rules
    - Analyzers

Last activity: 33 minutes ago

Maximum session duration: 1 hour

Permissions | Trust relationships | Tags | Access Advisor | Revoke sessions

Permissions policies (4) Info

You can attach up to 10 managed policies.

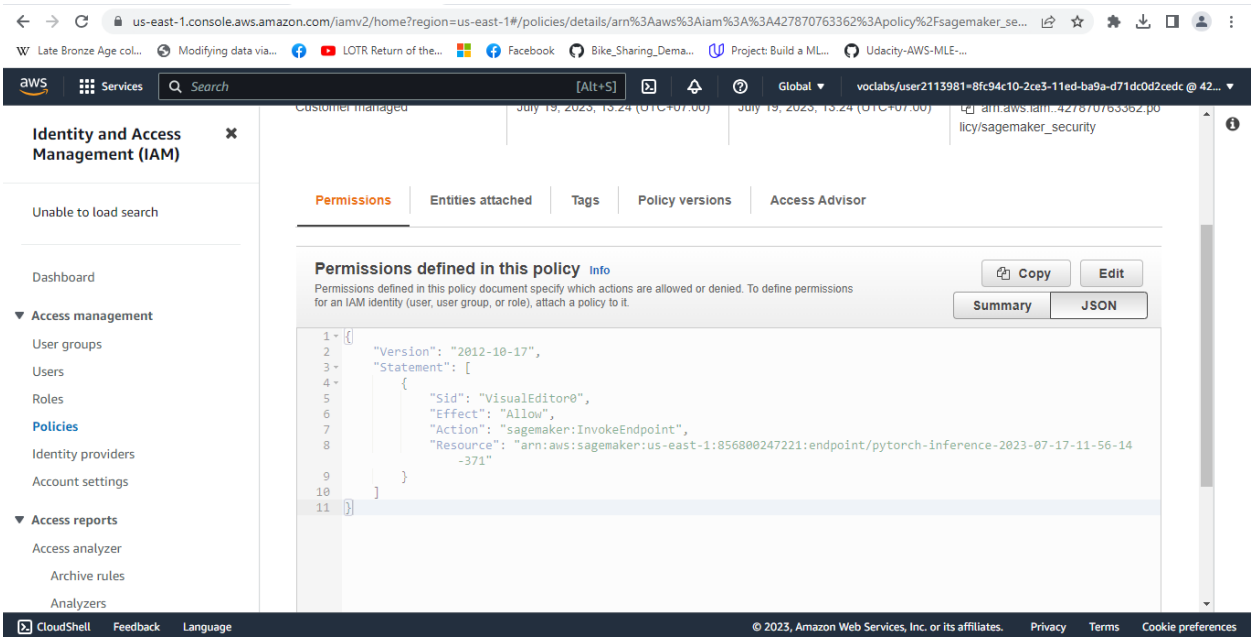
Simulate | Remove | Add permissions

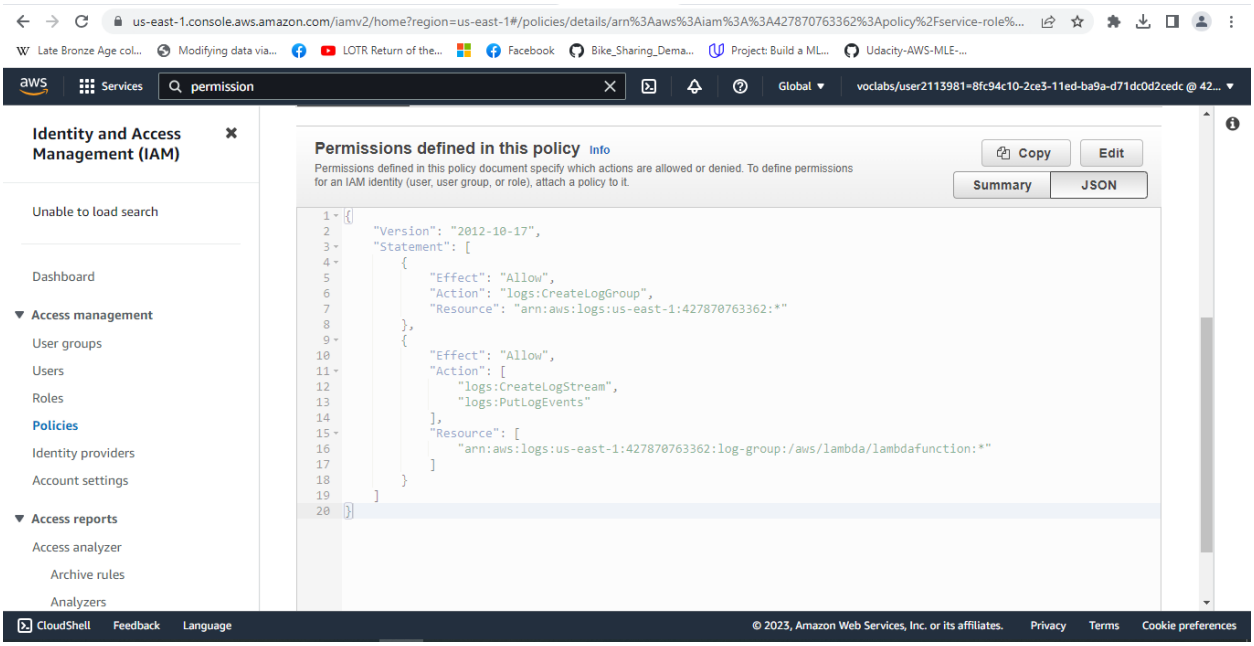
Filter policies by property or policy name and press enter.

1

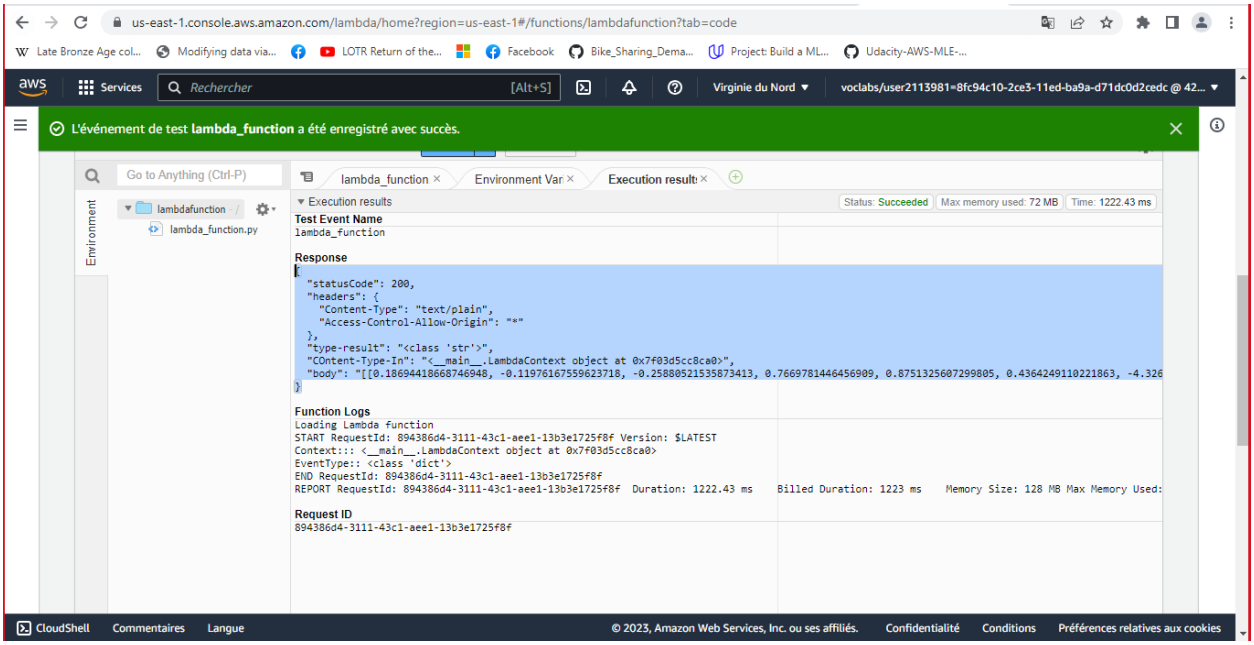
Policy name	Type	Description
AmazonSageMaker-ExecutionPolicy-20230718T093807	Customer managed	
AmazonSageMakerFullAccess	AWS managed	Provides full access to Amazon SageMa...
AmazonSageMakerCanvasFullAccess	AWS managed	Provides full access to Amazon SageMa...
AmazonSageMakerCanvasAIServicesAccess	AWS managed	Provides permissions for Amazon Sage...

CloudShell Feedback Language © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences





Testing lambda function



- Response

{

"statusCode": 200,

"headers": {

"Content-Type": "text/plain",

"Access-Control-Allow-Origin": "*"

},

"type-result": "<class 'str'>",

"COntent-Type-In": "<__main__.LambdaContext object at 0x7f7ee0d91ca0>",

"body": "[[0.18694418668746948, -0.11976167559623718, -0.25880521535873413, 0.7669781446456909, 0.8751325607299805, 0.4364249110221863, -4.326269149780273, -2.502095937728882, -0.7320882678031921, 0.17710088193416595, 0.675487220287323, 0.6102117300033569, -0.7135155200958252, -2.1245930194854736, -5.474817752838135, -4.405975818634033, -1.644808053970337, 0.3950308561325073, -1.7279447317123413, 0.6936320662498474, 0.7821153402328491, 0.8201605081558228, 0.49822020530700684, -4.432534694671631, -1.3959944248199463, -0.17860844731330872, -4.12421178817749, -1.7433632612228394, 0.8464545607566833, -0.6990833282470703, 0.05725167691707611, -1.8189820051193237, -0.02967911958694458, -

3.9690301418304443, -0.006218254566192627, -1.1326336860656738, -1.951277732849121, -0.3474602699279785, -1.9994728565216064, -3.490974187850952, -2.8858368396759033, -0.6671701073646545, 0.37207409739494324, -1.1812468767166138, 0.09818802773952484, -2.07558536529541, -1.2915866374969482, -0.8028730154037476, -2.4953577518463135, -4.881816387176514, 0.7013586163520813, -3.5340988636016846, -1.0568047761917114, 0.6765141487121582, -0.5022785067558289, -0.3398802578449249, -3.0912108421325684, -3.270838737487793, -0.028150461614131927, -4.1644721031188965, -0.788272500038147, -1.7746328115463257, -3.5130953788757324, -0.7117965221405029, -1.4560455083847046, -0.5636162161827087, 0.34504300355911255, 0.41097337007522583, -2.820807695388794, -0.09915725886821747, 0.6034729480743408, -2.4122331142425537, -0.4129945933818817, 0.20166942477226257, -1.142094612121582, -0.034452736377716064, -0.07025730609893799, -2.42698335647583, 0.6532773971557617, -3.8123397827148438, -0.6482104659080505, -3.053157329559326, -1.385493516921997, 0.3254506289958954, -3.495389223098755, 0.032257337123155594, -0.15192192792892456, -0.09814624488353729, -0.20918098092079163, -1.6140245199203491, -0.7781722545623779, 0.3407379388809204, -0.6098639965057373, -3.2056283950805664, -0.29552778601646423, -1.2895311117172241, -0.618022084236145, 0.5042839050292969, -2.7377214431762695, -2.289071798324585, -6.158321857452393, -0.9612634778022766, -2.2826547622680664, -0.6516165733337402, -4.319814205169678, 0.4649618864059448, -1.3943252563476562, 0.37625259160995483, 0.6220213174819946, 0.43092969059944153, -1.0943121910095215, 0.07643745839595795, -2.2445759773254395, -1.6854450702667236, 0.37922239303588867, -2.584455728530884, -0.001870766282081604, -0.3041623532772064, -2.7006101608276367, -2.610447406768799, -2.541846752166748, -3.0037319660186768, 0.46628397703170776, -0.519836962223053, -0.5407978892326355, -1.6393791437149048, -1.953521966934204, -1.1705448627471924, 0.5698442459106445, -1.814894437789917, -1.347900390625, -2.4129226207733154, -2.578855037689209]]"

}

Step 5: Lambda concurrency setup and endpoint auto-scaling

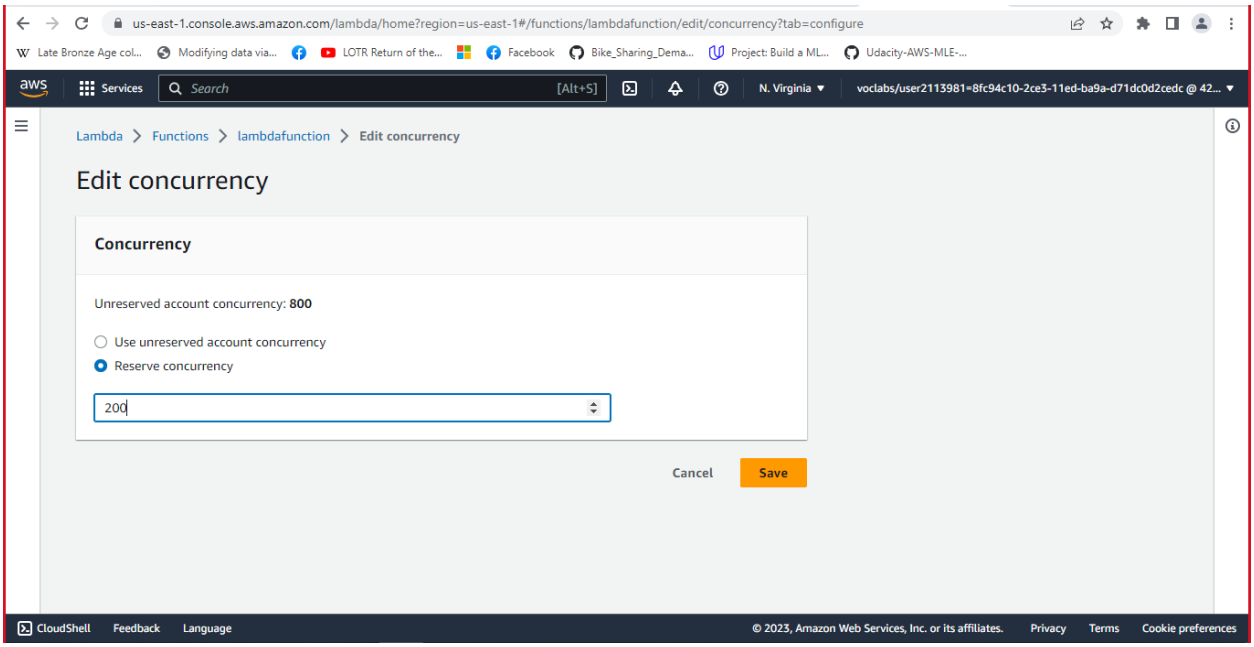
Vulnerability Assessment:
Granting "Full Access" has the potential to be exploited by malicious actors. Roles that are old and inactiv

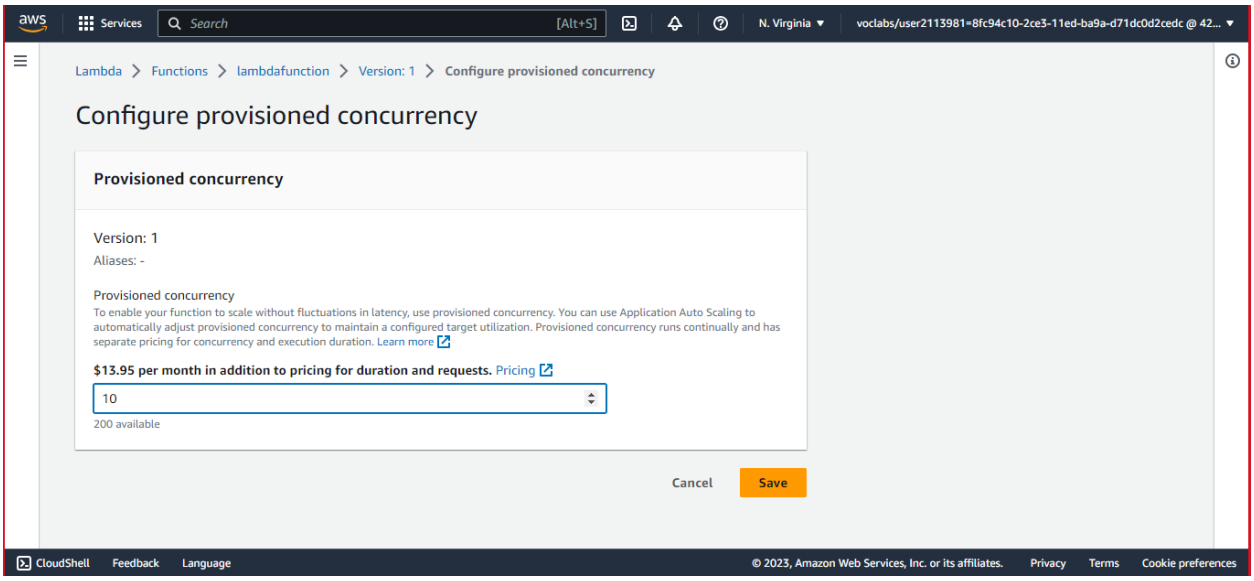
e pose a risk of compromising the lambda function, and it is essential to delete such roles. Additionally, r oles with policies that are no longer in use may lead to unauthorized access, making it crucial to remove these policies.

By default a Lambda Function can only respond one request at once. One way to change that is to use concurrency so that the Lambda Function can be responds to multiple requests at once.

To set up concurrency on your Lambda function, you will need to open your Lambda function in the Lambda section of AWS. Next, you should open the Configuration tab. Then, you should configure a Version for your function, in the Version section of the Configuration tab.

After configuring a Version for your Lambda function, navigate to the Concurrency section of the Configuration tab of your function. Use this section to configure concurrency for your Lambda function





In addition to setting up concurrency for your Lambda function, you should also set up auto-scaling for your deployed endpoint.