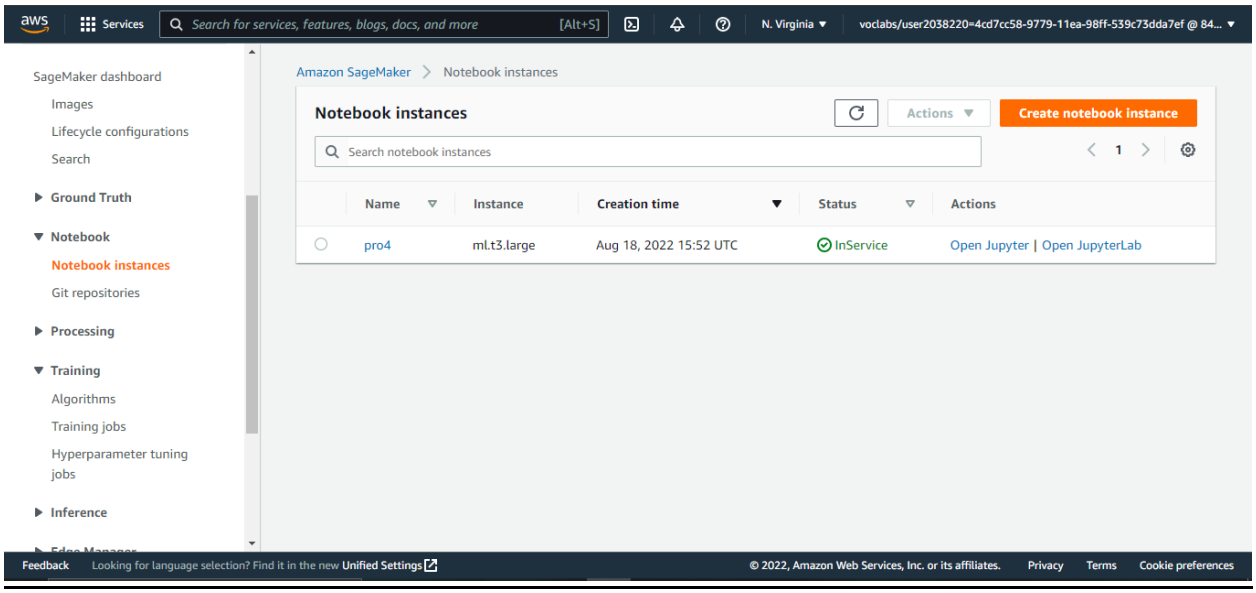
Operationalizing an AWS ML Project

1- **Notebook instance**:

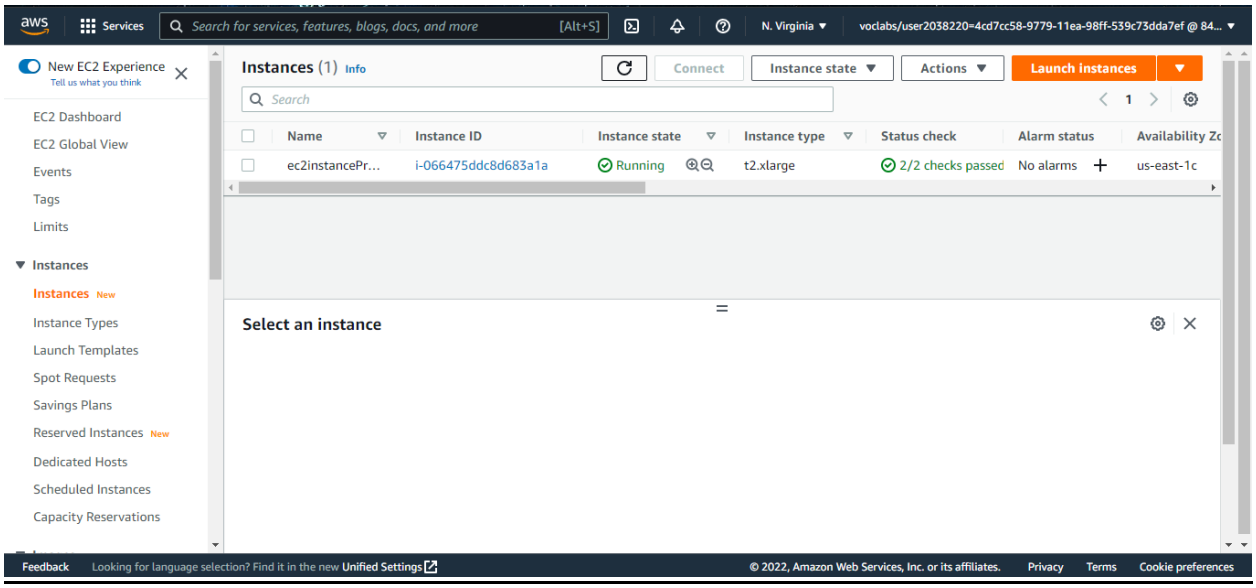
- I selected ml.t3.large instance type (vCPU=2 and Memory=8GiB) because we will train on large data.



- I used ml.m5.xlarge as it has more resources for training an image classification model.

2- **EC2**:

- I selected t2.xlarge EC2 instance type as it provides moderate CPU and Memory requirements for price per hour less than 0.20 Dollars which is good cost to run data processing and model training.



- **With this configuration model training took approximately twenty minutes which is fairly short duration given the amount of data.**

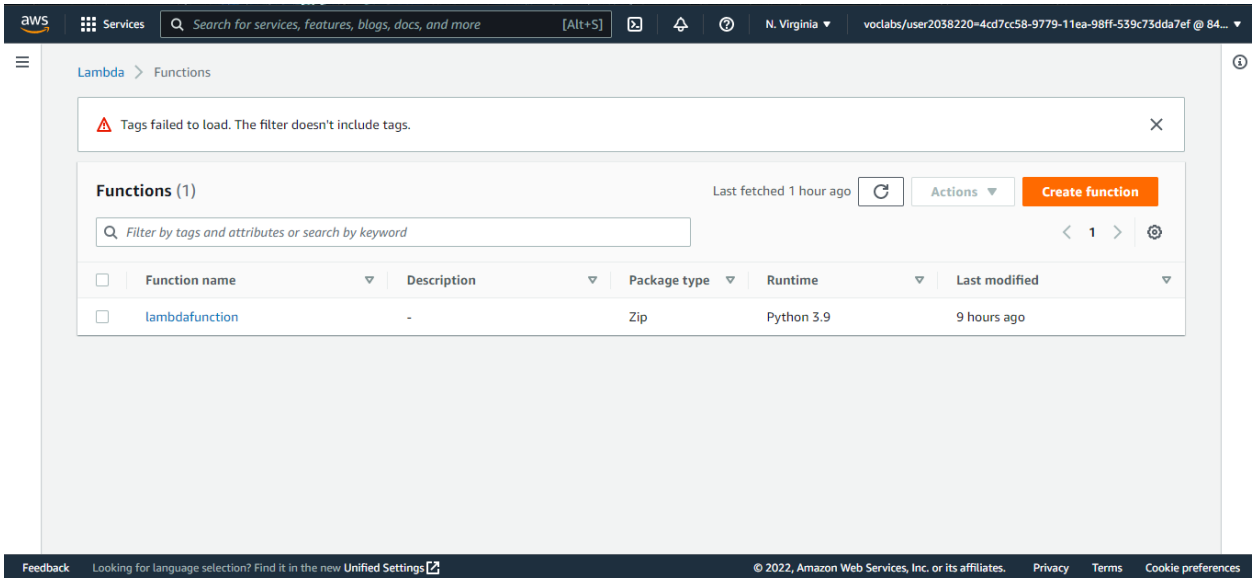
3- **<u>Write about EC2 code and how it compares to the code from Step 1</u>:**

- **Sagemaker libraries are available in sagemaker notebook environment which are not available in EC2 instance.**
- **In the EC2 training script, all the variables like hyperparameters and output locations are mentioned in the script itself and so there is no need for argparse. Meaning while running the EC2 script we do not need to mention any arguments.**
- **In the EC2 script the training happens on the same server on which the script is executed, however in the SageMaker scripts the training job that is invoked, it runs on a separate container than the one on which the SageMaker notebook is running.**

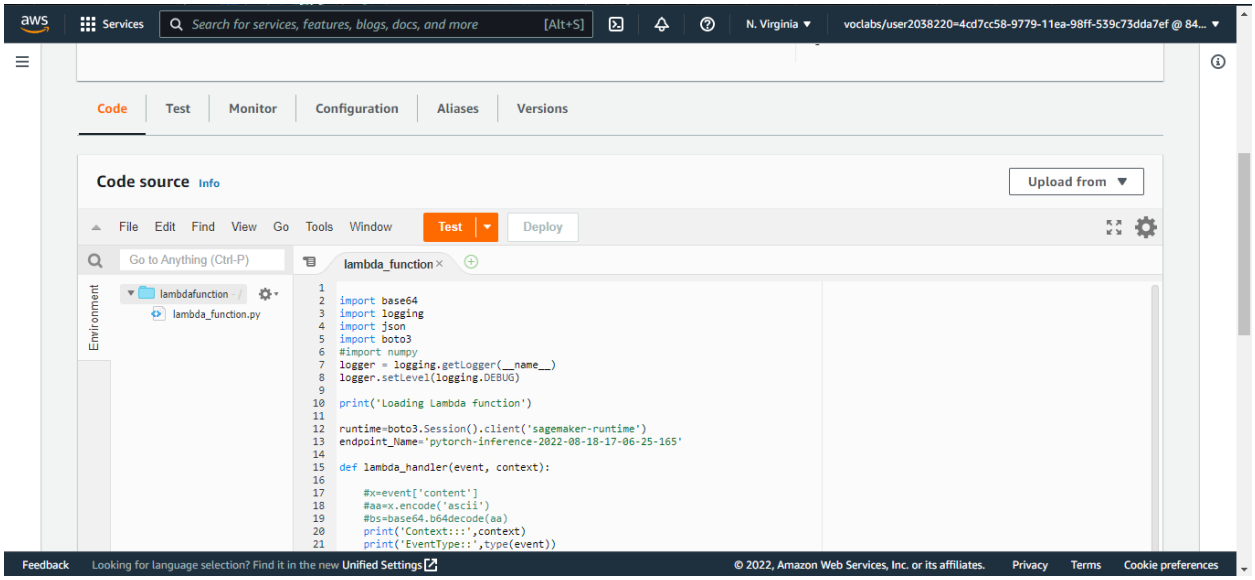
```
[root@ip-172-31-25-58 ~]# python3 solution.py
Downloading: "https://download.pytorch.org/models/resnet50-0676ba61.pth" to /root/.cache/torch/hub/checkpoints/resnet50-0676ba61.pth
100%|                                                                                | 97.8M/97.8M [00:01<00:00, 76.4MB/s
Starting Model Training
saved
[root@ip-172-31-25-58 ~]# ls
dogImages  dogImages.zip  solution.py  TrainedModels
[root@ip-172-31-25-58 ~]# cd TrainedModels
[root@ip-172-31-25-58 TrainedModels]# ls
model.pth
[root@ip-172-31-25-58 TrainedModels]#
```

4- **<u>Lambda Function</u>:**

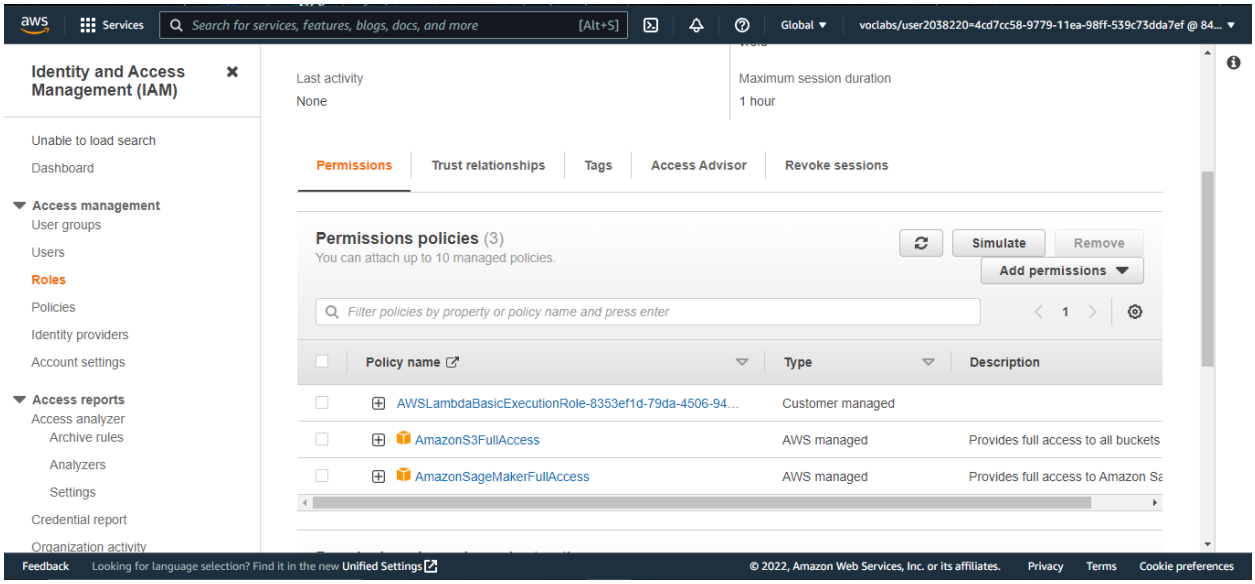
- **lambda function is used for invoking deployed endpoints.**



- **lambda function expects the image inputs in json format, which is used to invoke the model's deployed endpoint.**



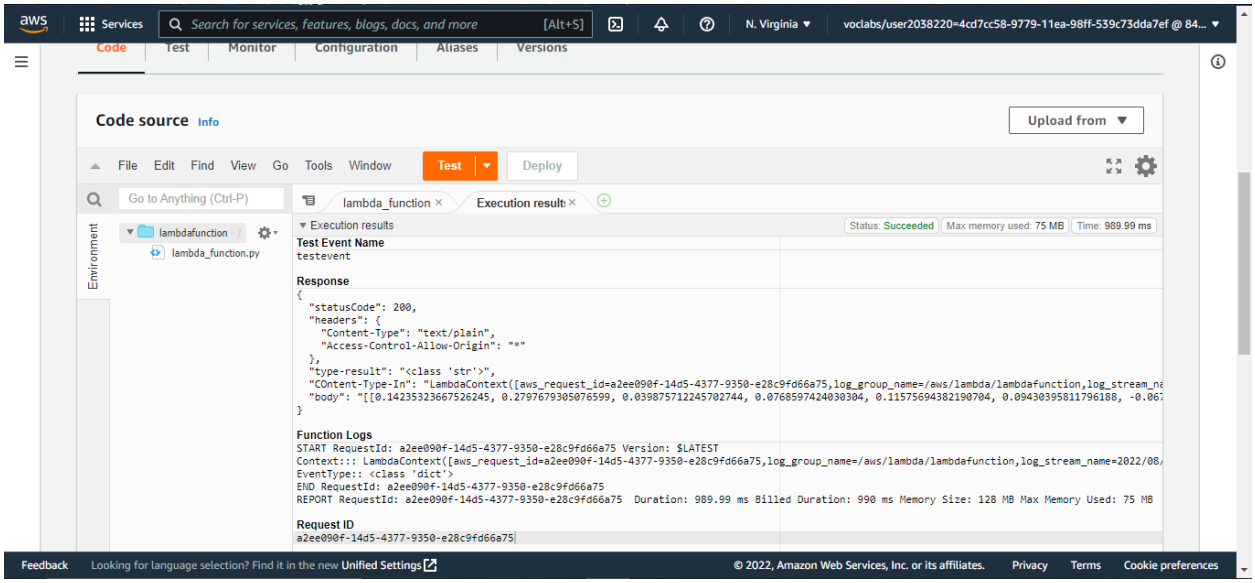
- **We will only use the multi-instance training jobs endpoint for invoking the endpoint.**
- **We have to add the "SageMakerFullAccess" policy role to the lambda function's role to invoke the endpoint.**



- **Result of lambda function test: Response: { "statusCode": 200,"headers": { "Content-Type": "text/plain", "Access-Control-Allow-Origin": "*" }, "type-result": "<class 'str'>", "COntent-Type-In": "<__main__.LambdaContext object at 0x7f3802e37b20>", "body": "[[0.1423585706138611, 0.279767677658081, 0.03987571521492004,**

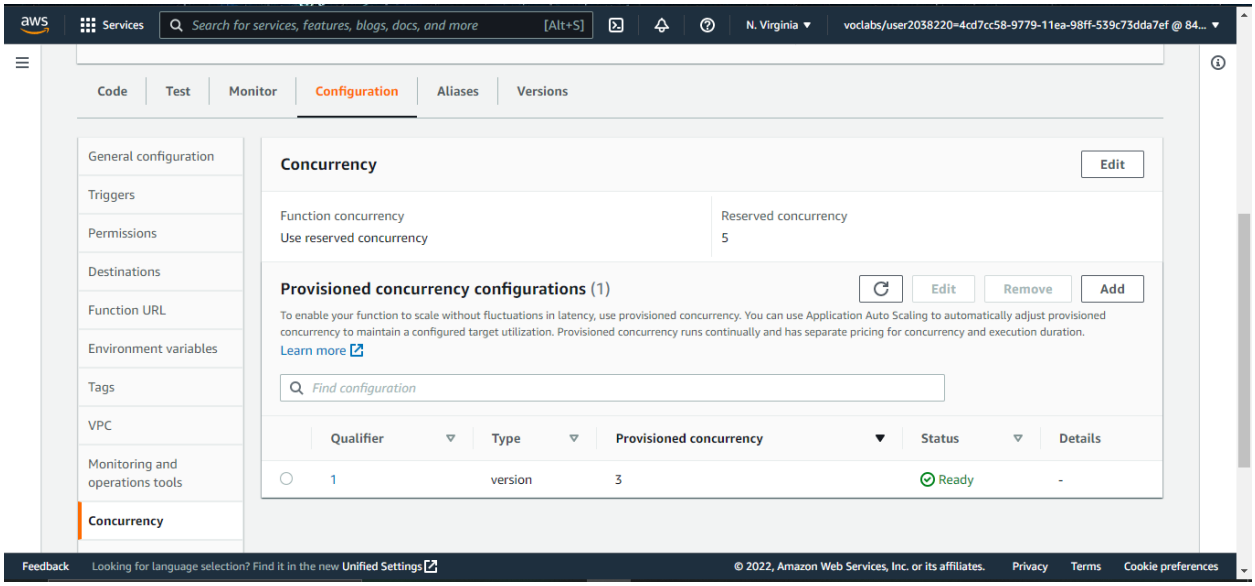
0.0768599638557434, 0.115757938709259, 0.09430169671177864, -0.0672649701833725, 0.5599482655525208, -0.004726633429527283, 0.26198041439056396, 0.44466689229011536, 0.6322798728942871, 0.23320285975933075, 0.6302286982536316, 0.6538736820220947, 0.503573477268219, 0.5196398496627808, 0.2090550810098648, 0.282077431678772, 0.5264465808868408, 0.5401625633239746, 0.24061383306980133, 0.5089805126190186, 0.6206284761428833, -0.016930846497416496, 0.008964188396930695, 0.6029733419418335, -0.25446122884750366, 0.7258913516998291, 0.2825552821159363, 0.504551887512207, 0.5072543025016785, 0.041132666170597076, 0.5149539709091187, 0.337560772895813, 0.6005417108535767, 0.3293423652648926, 0.39723503589630127, 0.6054755449295044, 0.3309067189693451, -0.06646368652582169, 0.6616803407669067, 0.26549050211906433, 0.6130195260047913, 0.24571208655834198, 0.6165071129798889, 0.31599143147468567, 0.4608200490474701, 0.2141287475824356, 0.20662344992160797, 0.38995176553726196, 0.27706003189086914, 0.2728506028652191, 0.438828706741333, 0.3287105858325958, 0.591151237487793, 0.6151048541069031, 0.3438055217266083, 0.24811337888240814, 0.47794631123542786, 0.11838075518608093, 0.3076281249523163, 0.34239906072616577, -0.02150164544582367, 0.09787628799676895, -0.02746940776705742, -0.05814646556973457, 0.44791263341903687, 0.2661702334880829, 0.28706124424934387, 0.4709480404853821, 0.2293863445520401, 0.0467015765607357, 0.3039526641368866, 0.17233839631080627, 0.561221718788147, 0.2366209775209427, 0.19137583673000336, 0.5240403413772583, 0.255370557308197, 0.49720728397369385, 0.4307907819747925, 0.12711751461029053, 0.299511849802185, 0.02879352681338787, 0.30821502208709717, 0.4563232958316803, -0.04539699074085236, 0.3409106731414795, 0.6171738505363464, -0.14309659600257874, 0.1574305146932602, -0.07284145057201385, 0.14746177196502686, 0.29568302631378174, 0.12199152261018753, 0.4861227571964264, 0.24723970890045166, 0.2210666537284851, -0.09887337684631348, 0.36195996403694153, -0.21740089356899261, 0.29109564423561096, 0.04220682382583618,

-0.10023950040340424, 0.24861504137516022, 0.24743123352527618, -0.29791533946990967, 0.19722631573677063, -0.05723617225885391, 0.23458585143089294, 0.34250569343566895, -0.03882203251123428, -0.004864443093538284, 0.5508145689964294, -0.1445448249578476, 0.257218599319458, 0.345347136259079, -0.1522929072380066, 0.017558464780449867, -0.18726523220539093, -0.08899061381816864, 0.1663469523191452, 0.3544248342514038, -0.018308008089661598, -0.15787075459957123, 0.19376394152641296, -0.1657421886920929, 0.167301744222641, 0.3652785122394562, -0.235820472240448, -0.24553430080413818, -0.20475737750530243]]".



5- **<u>Writeup about whether you think your AWS workspace is secure, and whether you think there are any vulnerabilities</u>:**
   - **I have enabled Sagemaker full access for Lambda function which can be further restricted to limited Sagemaker resources such as endpoint so that other sagemaker resources can be secured from lambda function execution.**
   - **I have launched EC2 in default VPC and made it accessible to public. This is not good security practice. EC2 instance can be created inside private VPC and subnets and access to instance can be whitelisted from specific IP address.**
6- **<u>Concurrency and auto-scaling</u>:**

- **Before adding in configs for Concurrency and Auto-scaling for our lambda functions we will first create a version config for our lambda function.**
- **For the lambda function we have set the reserved concurrency to be 5. This implies that the lambda function would be able to handle up to 5 requests concurrently at the same time. This would help lower latency issues in situations when there is higher traffic than usual.**
- **I expect 3 lambda function to run concurrently so I set provisioned concurrency to 3 which allows at most 3 lambda function to run concurrently.**



- **I have set the max instance count to 3 for Auto-scaling, as considering the current requirement, auto scaling on 3 instances with a scale-in and scale-out cool down time of 30 seconds should be a good config.**