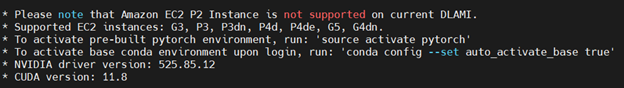
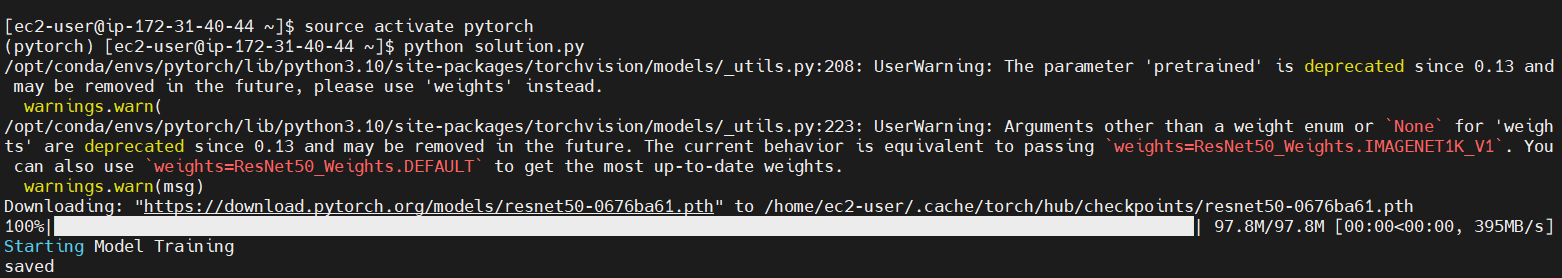
# Step1 - Training and deployment on Sagemaker

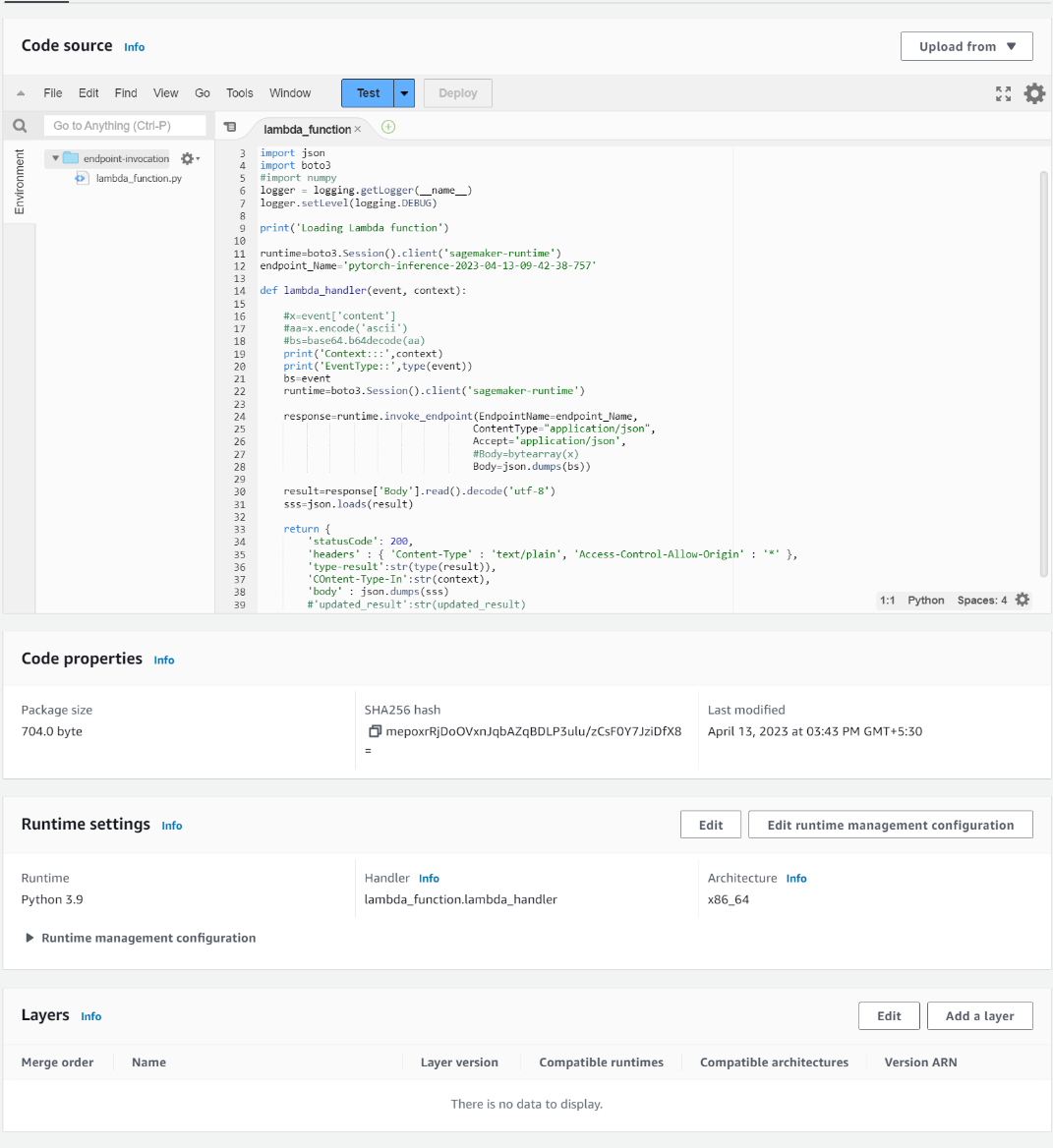
* Above is the Sagemaker Studio Notebook instance screenshot I have been used to train the model
* Justification about the selected EC2 Instance Type:
* According to the above note deplayed on DLAMI instances in ec2 G3,P3,P3de, G5, G4dn instance tpes are supported for the Deep Learning AMI GPU PyTorch 2.0.0 (Amazon Linux 2) 20230406Deep Learning AMI GPU PyTorch 2.0.0 (Amazon Linux 2) 20230406. Therefore G3 instance type has been chosen to provide the optimal level of requirement for the AMI and On demand instance has been chosen due to spot instance are not defaulting supported for G3 the service quota for the G3 instance should be increased to achieve this.

# Step2 - EC2 Training

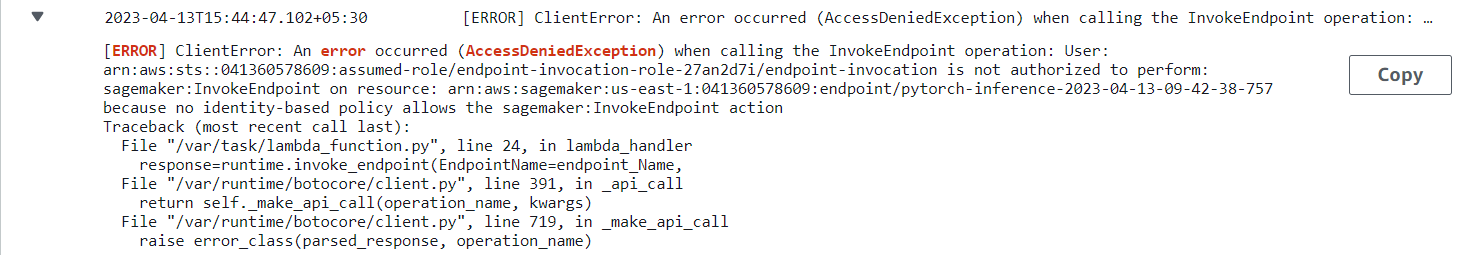
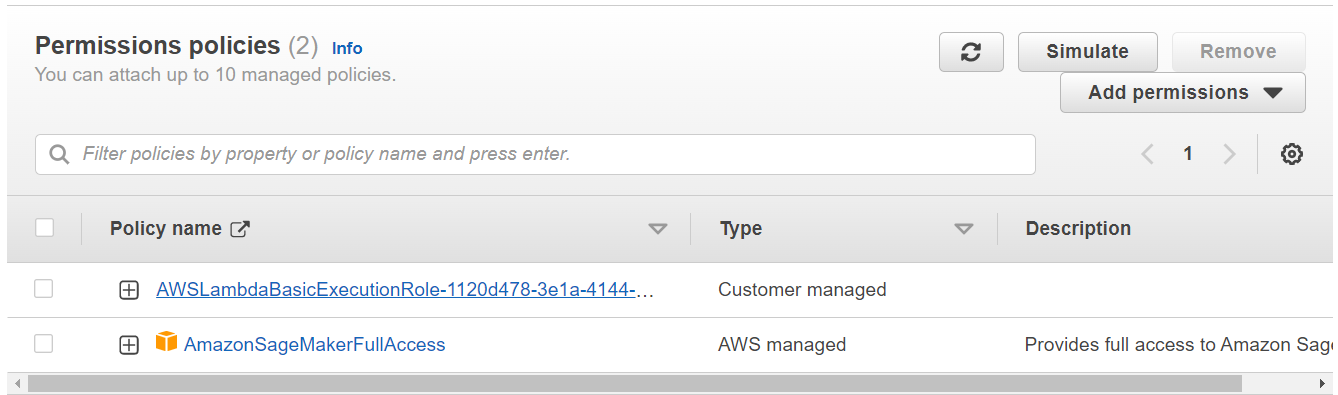
* The EC2 Code Compared to the SageMaker Code:

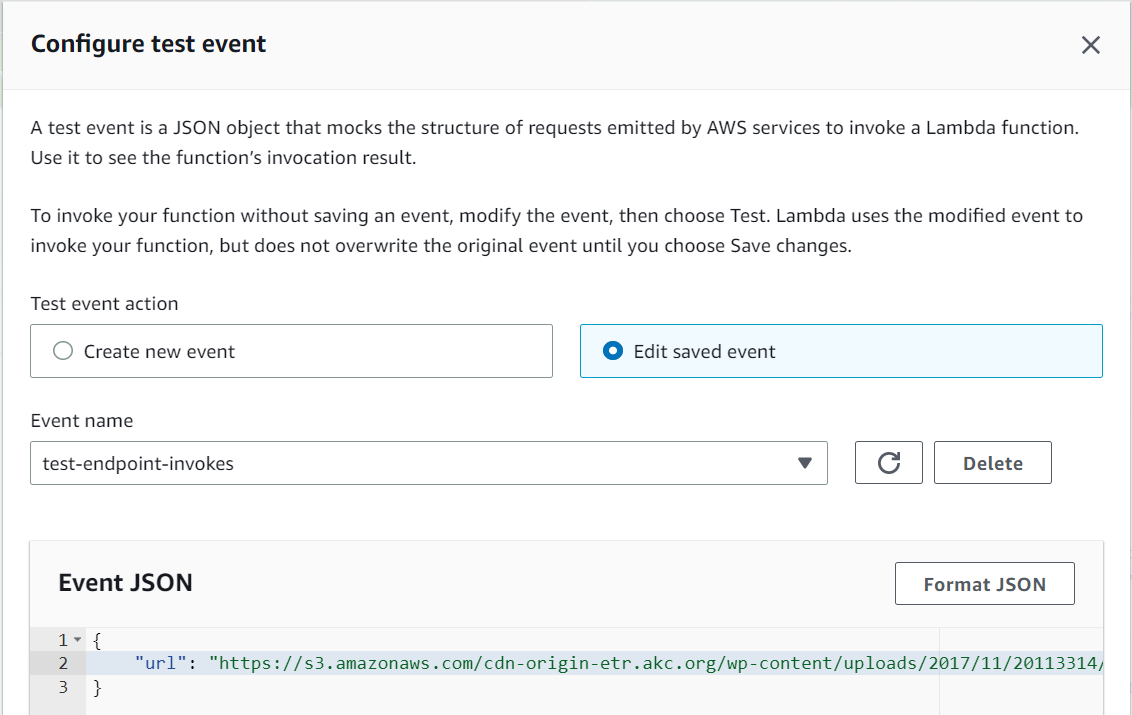
Both Sagemaker and EC2 code for train the model are similar instead the followings; - In Sagemaker we first do the hyperparameter tuning for and select the best parameter values for the model training but in ec2 instance training job we didn’t do such kind of hyper parameter tuning.. - Also in Sagemaker we have to store data in S3 bucket, and have to import them into sagemaker using sagemaker method to train the models. But in EC2 instance it is more similar to train the model inside an own machine.

# Step3 - The Lambda Function

* The lambda function has been deployed with the provided code by updating the endpoint name as above.

# Step4 - Security & Testing

* Intial function didn’t have the necessary permission to access to the Sagemaker. Therefore, Sagemaker full access permission has been attached to the endpoint-invocation-role-27an2d7i role.
* After attaching the role the endpoint invokation was successfully done with the “url”: “https://s3.amazonaws.com/cdn-origin-etr.akc.org/wp-content/uploads/2017/11/20113314/Carolina-Dog-standing-outdoors.jpg”.



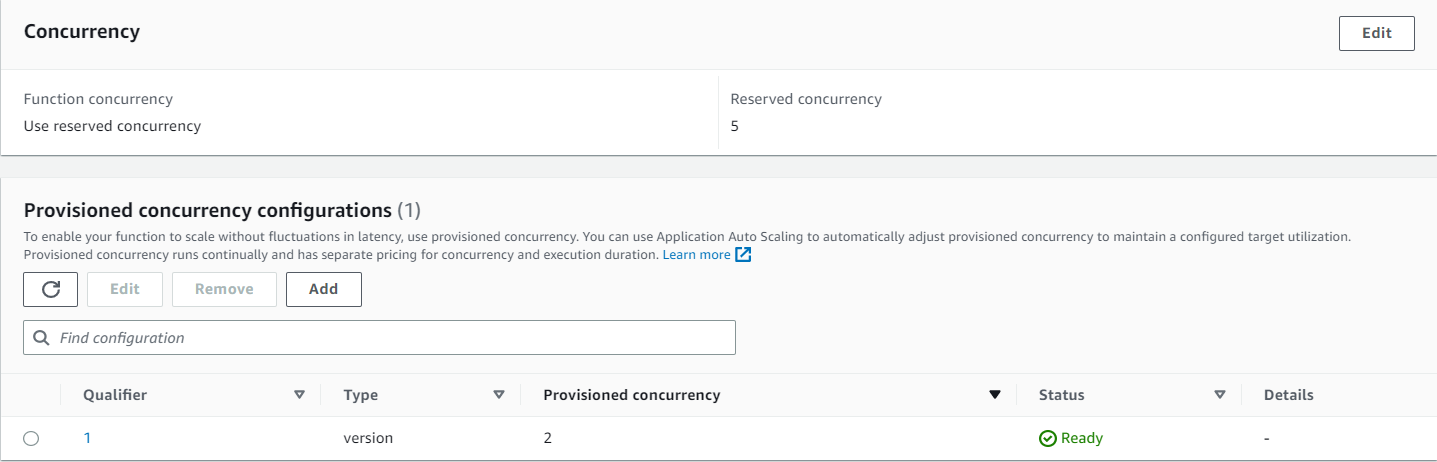
* The test event results are as below:

Test Event Name  
test-endpoint-invokes  
  
Response  
{  
 "statusCode": 200,  
 "headers": {  
 "Content-Type": "text/plain",  
 "Access-Control-Allow-Origin": "\*"  
 },  
 "type-result": "<class 'str'>",  
 "COntent-Type-In": "LambdaContext([aws\_request\_id=b037da00-0ad4-4465-b22e-fcd0737d280d,log\_group\_name=/aws/lambda/endpoint-invocation,log\_stream\_name=2023/04/13/[$LATEST]7863e0ff818b43e8918b5089be04594f,function\_name=endpoint-invocation,memory\_limit\_in\_mb=128,function\_version=$LATEST,invoked\_function\_arn=arn:aws:lambda:us-east-1:041360578609:function:endpoint-invocation,client\_context=None,identity=CognitoIdentity([cognito\_identity\_id=None,cognito\_identity\_pool\_id=None])])",  
 "body": "[[-0.01797017827630043, 0.029482660815119743, 0.08906541764736176, -0.06038343161344528, -0.01794680394232273, -0.46629634499549866, 0.22187794744968414, 0.3008261024951935, -0.11902894824743271, 0.19467774033546448, 0.29246413707733154, -0.2215651273727417, -0.08232477307319641, 0.3711948096752167, 0.24510228633880615, 0.3030194342136383, -0.157492533326149, 0.17989042401313782, 0.05015610158443451, 0.3777329623699188, -0.08977203071117401, 0.007890690118074417, -0.10903298854827881, -0.3213697075843811, 0.12642478942871094, 0.02754545956850052, 0.46439245343208313, -0.004751796834170818, -0.40215542912483215, -0.014899954199790955, 0.08186669647693634, 0.3739027976989746, 0.15897634625434875, 0.37218427658081055, -0.01119186356663704, 0.11083319038152695, 0.06563431769609451, -0.13128426671028137, 0.07534578442573547, 0.28775814175605774, 0.4237826466560364, 0.06632264703512192, 0.033261481672525406, 0.4378844201564789, 0.08703325688838959, -0.49956122040748596, 0.2310592234134674, 0.0436323918402195, -0.15639543533325195, 0.19563926756381989, -0.0959526002407074, 0.03934663534164429, -0.28930965065956116, -0.33469101786613464, 0.10374609380960464, 0.08651759475469589, 0.02198834717273712, -0.4293921887874603, 0.2598825693130493, 0.3884611129760742, -0.06918875873088837, -0.05065305531024933, -0.1261543482542038, -0.3965974450111389, 0.15586964786052704, -0.047885164618492126, -0.18222665786743164, 0.19508181512355804, 0.2846355438232422, 0.10613688081502914, 0.3328869640827179, 0.28651154041290283, 0.03145391866564751, 0.13602443039417267, -0.13934952020645142, 0.0528394840657711, 0.08379044383764267, 0.04540861025452614, -0.03115582838654518, 0.20425722002983093, 0.4191071391105652, -0.26806774735450745, 0.16484081745147705, -0.3541029691696167, -0.18167923390865326, -0.21143047511577606, 0.029214033856987953, -0.010294117033481598, 0.1814262866973877, 0.28048497438430786, -0.5519843697547913, -0.46105435490608215, -0.01761147566139698, 0.0385548397898674, -0.07677969336509705, 0.10123839974403381, -0.026916639879345894, 0.19218841195106506, -0.27591460943222046, -0.5180968046188354, -0.583723783493042, -0.10444218665361404, 0.37789052724838257, -0.10145723074674606, 0.07286269217729568, 0.0658789873123169, -0.36088407039642334, -0.3690186142921448, 0.09292355179786682, -0.15924866497516632, -0.32419759035110474, 0.07097570598125458, -0.20900440216064453, -0.3070576786994934, -0.6248983144760132, -0.23336553573608398, -0.3209134042263031, 0.0326412096619606, -0.24327999353408813, 0.13911588490009308, -0.04725434258580208, 0.01507787685841322, -0.5268176794052124, 0.0761292576789856, -0.08016560971736908, -0.16502171754837036, -0.5888742208480835, -0.024587446823716164, 0.1866881549358368, -0.12257489562034607, -0.14660067856311798, -0.11219187080860138, -0.6710981726646423]]"  
}  
  
Function Logs  
Loading Lambda function  
START RequestId: b037da00-0ad4-4465-b22e-fcd0737d280d Version: $LATEST  
Context::: LambdaContext([aws\_request\_id=b037da00-0ad4-4465-b22e-fcd0737d280d,log\_group\_name=/aws/lambda/endpoint-invocation,log\_stream\_name=2023/04/13/[$LATEST]7863e0ff818b43e8918b5089be04594f,function\_name=endpoint-invocation,memory\_limit\_in\_mb=128,function\_version=$LATEST,invoked\_function\_arn=arn:aws:lambda:us-east-1:041360578609:function:endpoint-invocation,client\_context=None,identity=CognitoIdentity([cognito\_identity\_id=None,cognito\_identity\_pool\_id=None])])  
EventType:: <class 'dict'>  
END RequestId: b037da00-0ad4-4465-b22e-fcd0737d280d  
REPORT RequestId: b037da00-0ad4-4465-b22e-fcd0737d280d Duration: 1001.61 ms Billed Duration: 1002 ms Memory Size: 128 MB Max Memory Used: 68 MB Init Duration: 356.59 ms  
  
Request ID  
b037da00-0ad4-4465-b22e-fcd0737d280d

* The AWS workspace is not 100% secure due to lamda function has full access to the sagemaker and it can be used to delete the deployed endpoint. Therefore, some vulnerabilities may occur and it can be reduce by providing more strict IAM policy to lambdafunction to just allowing the invoking sagemaker endpoint only and deny other permissions using a custome inline policy as below;

{  
 "Version": "2012-10-17",  
 "Statement": [  
 {  
 "Sid": "VisualEditor0",  
 "Effect": "Allow",  
 "Action": "sagemaker:InvokeEndpoint",  
 "Resource": "arn:aws:sagemaker:us-east-1:041360578609:endpoint/pytorch-inference-2023-04-13-14-58-34-578"  
 }  
 ]  
}

# Step5 - Concurrency and auto-scaling

* As the concurrency type I have been chosen the provisioned concurrency which will help to avoid the startup time and which make delay to respond to requests.
* The endpoint autoscaling has been configures as below to add more instances for the endpoint when the Sagemaker Endpoint invokations hit the target value 30 and it will add maximum 2 instance.

