# Step 1: Training and deployment on Sagemaker

Created a Sagemaker instance using instance type of ml.t2.medium, which is enough for the notebook to run and cost effective, the instance did not take much time to be created and starter files were uploaded to the instance.

The provided train_and_deploy notebook was edited to use ml.m5.xlarge instance type for training, and the S3 bucket path was changed to the bucket used in this project, also added code for multi instance training.

# Step 2: EC2 Training

EC2 instance was created using Deep Learning AMI GPU PyTorch 1.13.0, but when running EC2 code it showed that numpy, torch, torchvision and tqdm packages were not found, so: pip3 install *package_name* command were used to install these packages then the code executed and trained and saved the model successfully.

The difference between EC2 code and sagemaker code is that the former is a self-containing script that trains the model using local file system, while sagemaker requires methods to access S3 bucket.

# Step 3: Lambda Function Setup

Lambda function was created using the provided code after editing endpoint_name variable, and the provided json test code was used as well. This fuction required additional permission policies to run.

# Step 4: Security and Testing

My AWS workspace is not secure because while setting up EC2 instance I chose no key pairs and chose to allow connections from any IP address, also Lambda function polices included sagemaker full access.

# Step 5: Autoscaling

Provisioned concurrency for Lambda function was used, the chosen value depends on the expected load, so a value of 1 was used for this function.
As for the sagemaker endpoint, a maximum instance count of 2 was used in the autoscalling configuration, due to the low expected traffic on this project.