The project aimed to train and deploy an image classification model on AWS and the following are steps and decisions I made during the process:

The goal is to minimize cost as much as possible so to start with, I created a notebook instance(ml.m5.xlarge) because this is among the cheapest and a bit faster compared to other cheapest instances type available on AWS.

Waited for some time for the notebook instance to be in service, launched the notebook instance by open Jupyter, and selected conda_python3 to open the environment that allowed allow me to train and deploy the model.

I downloaded the dataset and unzip it and then copied it to S3. The dataset uploaded to my bucket contains train, test, and validation files.

After that, I started the training process with 1 instance and it took 1690 seconds to complete the training job and billed 1690 seconds. I then repeated the same process on multiple instance training and chose 5 instances this time and it turnout that training on multiple instances can accomplish the training job much faster as compared to training on 1 instance. The model artifacts were saved in the S3 bucket which I have created. The next step I did was to deploy the model and then later set up a lambda function for the deployed endpoint.

Meanwhile, I have set up auto-scaling for the deployed endpoint and since we are not expecting much traffic, I have set the auto-scaling as follows: scale down to 120 seconds, scale-out cool down to 30 seconds, and the target value of 20. Doing this to save resources in the AWS account.

Proceeding to the lambda function, testing the lambda function with default policy resulted in an error however when added necessary policies, which in this regard SageMakerFullAccess, the Lambda function returned 200 status code, meaning that the Lambda function can communicate with the deployed endpoint and access other resources in the AWS.

Again, in this project, we do not expect high traffic and for this reason, I have decided to configure reserved concurrency only, choosing this because this is the cheapest compared to provisioned concurrency.

On the other hand, I trained the same model on EC2 and since this project is for exploration and learning purposes, I have opted for spot instance. Spot instances are way cheaper compared to standard instances and overall using this type of instance is a great way to lower costs in the AWS.

Training models in this environment have resulted in saving the model artifacts within the folder instead of S3 and the terminal commands are widely used here to accomplish the task.

After performing the above steps, I stopped the notebook instance and deleted deployed endpoint as well as the lambda function to avoid further charges.

In this project, I used several AWS tools to adjust, improve and prepare the model for production-grade deployment. I have demonstrated the necessary skills required to optimize advanced Machine Learning pipelines for efficiency, speed, and security as done in the real world.