# Operationalizing an AWS ML Project - report

Why did you choose that Sagemaker instance?

I chose a t3.medium instance because it will not require a lot of power, since it will only run the Jupyter notebook from which other resources will be called. Additionally, I used this instance to test and debug some of the functions in the notebook, these were small tests that also run well with this instance's resources.

Why did you choose that particular EC2 instance?

I chose a g4dn.xlarge instance because it supports the kind of AMI that is needed for deep learning model training, it is also relatively cheap, with just enough computer power and memory to complete this project but without consuming the budget.

What are the differences between the code deployed in the EC2 instance and the code deployed in Sagemaker?

Many parameter that are command line variables in the Sagemaker environment are hard-coded in the EC2 code, making the program less flexible. The EC2 code cannot directly use AWS services that are available in Sagemaker, the consequences are, for example, that the model is saved locally (in the EC2 instance) and will require further processing to be deployed in AWS.
Another difference is that the EC2 code needs to be run in one instance, this will affect performance and may not be suitable for large datasets.

Can you describe the Lambda function used in this project?

This Lambda function is simple enough: it calls the endpoint and then outputs the result. It manages that through the AWS SDK and a simple call to `invoke_endpoint`, it then parses the answer as a JSON string that will be returned to the caller of this function.

Do you think your AWS workspace is secure?

I don't think that the policy attached to the Lambda role is the correct one, as it gives the function more permissions than it needs to complete its work, for better security, there would need to be custom policy that enables the Lambda function to invoke only the specified endpoint.

Describe your autoscaling and concurrency configurations

The endpoint can autoscale from 1 to 3 instances when it receives more than 2 simultaneous calls in a row, it will scale up quickly (10 seconds) and scale down slowly (60 seconds) this can be useful in a scenario where the user cares about reducing latency and is not concerned about cost. This can be the case with important services in the application, such as client-facing services.
The concurrency in the Lambda function is only reserved, meaning that it will adjust to demand dynamically, but can be slow if many requests arrive at the same time. I considered this was enough to test and complete this project, but in a situation where the function is saturated, perhaps some added provisioned capacity will be needed.