**Detection of Coronavirus (covid-19) using patient’s Chest X-Ray Image**

**Advanced Systems Project -** **CIS 5690**

**Abel Rozario Leo**

**Email: axl64160@ucmo.edu**

**Date: 05/05/2021**

**ABSTRACT**

The COVID-19 pandemic continues and more cases come to light globally, the use of artificial intelligence or AI-based tools to help detect the disease is being explored more extensively. There are numerous agencies working on testing patients for Covid-19, and giving them proper medication for better health, if and when required. But the problem is, the test takes up to 3–4 days to finalize the report of a patient, which is quite risky as the disease is very contagious. The tool that I developed in this project will help in the detection of potential coronavirus cases, which is a deep neural network (NN) that will look at chest radiography images to determine whether a patient is infected or not. The technique is up on Classification NN for automated detection of COVID-19 from chest X-Ray images. The proposed technique targets the discrimination of COVID-19 infected from Viral infected and Healthy individual. The learning capacity of the proposed NN is enhanced using Channel Boosting to improve the detection rate while maintaining high precision.

**IMPLEMENTATION**

The project was architected and designed using AWS cloud, which comprises of different AWS services in order to automate the complete process, the Services that were used are AWS: IAM Role, SageMaker (Jupyter Notebook), Simple storage Services(S3), Simple Email service (SES), Lambda function (Incorporated NodeJS code), CloudWatch and Word document which consist of the final prediction. Figure 1 shows the AWS architecture

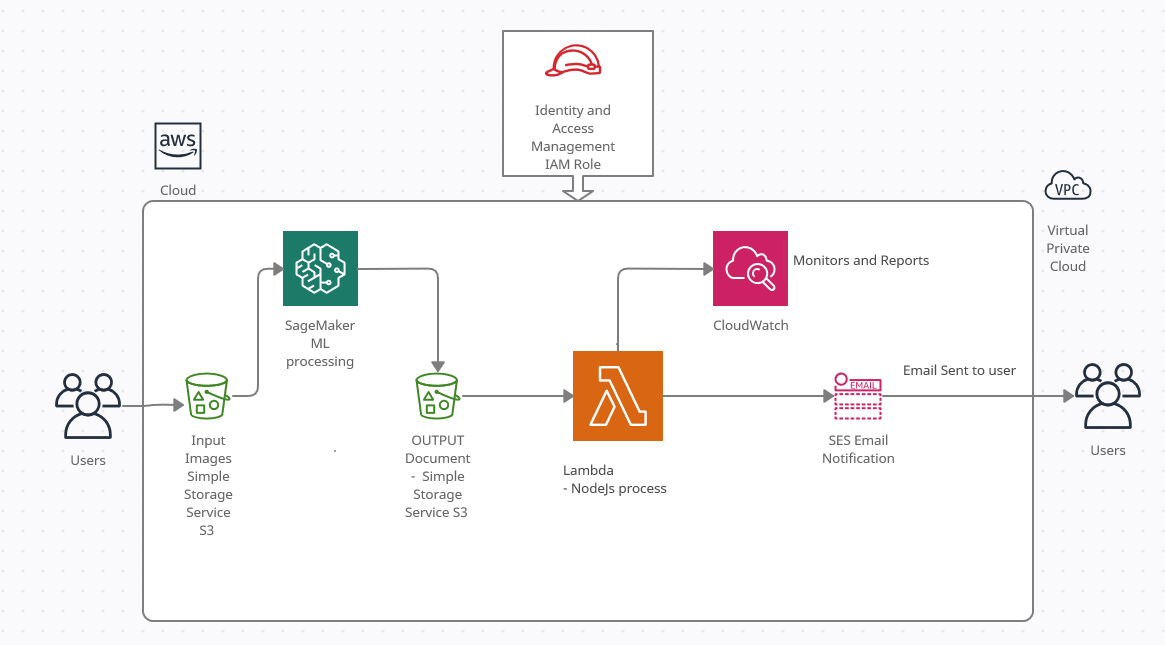


Figure 1 – AWS architecture

* **AWS SageMaker – Jupyter Notepad used for Creating the Neural Network Model**

The tool uses Kaggle dataset that had chest X-Ray Images of different people and the images were mainly classified into 3 classes: <Normal>, <Viral Pneumonia>, <Covid-19>. The data was split into training and testing sets. Where 80% of the datasets were used for Training and 20% for testing finally after the model has been developed. The dataset will take an optional argument transform so that any required processing can be applied to the sample. The challenge is the input images can be of any size so an additional image size transformation was done to make all input images of fixed size. A Data loader was created to allow an easier implementation for yielding a batched sample for every iteration. The overall Sagemaker workflow is shown in Figure 1.

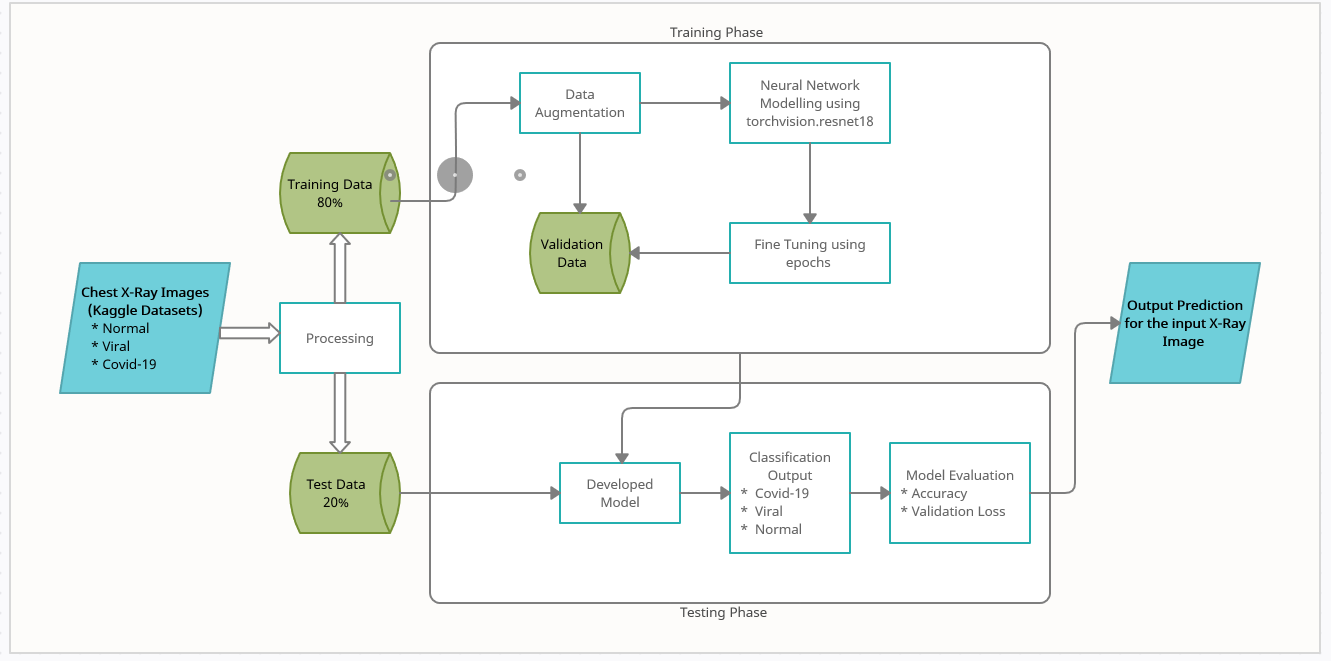


Figure 2

Visualization was done to see the outputs in graphical view. A function that would take the true labels, the predicted labels, and a set of images passed through a data loader iterator, and display a set of images and if the prediction is correct i.e. if the predicted label of the image matches with the true label of the image, then the predicted label would be displayed on the side of the image in green color and if the prediction is wrong, the predicted label would be displayed in red color.

Below Figure 2 show how the Labels and Images are displayed after the Dataloader is done.

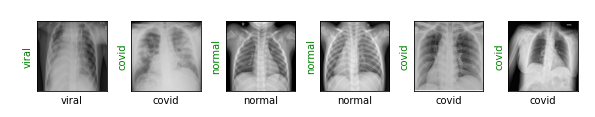


Figure 3

Ones the Data Loader is done, the classification neural network training model called ResNet-18 was used to train the model using the input dataset. It assigns a Cross Entropy Loss to the model for classifying the task and using Adam as an optimizer with a learning rate of 4e-5. Another function was created to do the predictions during training by iterating the data loader and passing a batch of images to the modified ResNet-18 model for prediction (show\_preds function), and display the results using the function that we previously created.

The Finally train(epochs) was used to train and test the model for accuracy and check for training loss. This function loop over the data iterator, and feed the inputs to the network and optimize. If everything works fine, the training will start and an output somewhat like below will be displayed as show in Figure 3 – Training Phase. The model is saved and used against the test dataset(images) and using the predefined prediction function, we can see the predicted output for the test images, and if there are any mismatches it will be displayed in Red if both label and image matches it will display all in Green as shown in Figure 4 – Testing Phase Now this trained model can be used against any input chest X-Ray images to detect if there is any presence of Covid-19 or Viral or its Normal

**train(epochs=1)**

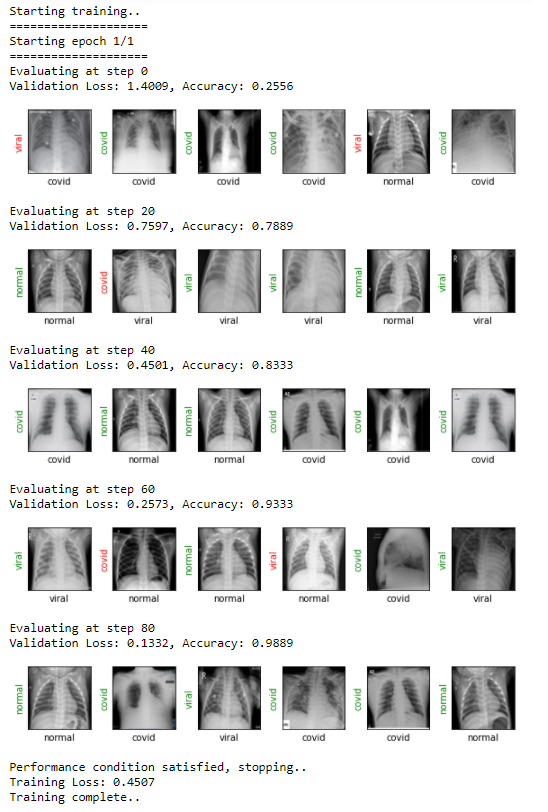


Figure 4 – Training Phase

**show\_preds ()**

**Output**

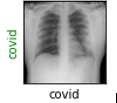


Figure 5 – Testing Phase

**IAM Role:**

IAM role is both an identity and a resource that supports resource-based policies. The IAM role need defined so that all the AWS service within the user account can share the data amongst the assigned services. For security reason the different services in the AWS can’t talk to each other even though they are under the same user account. So, an IAM role needs to be created specifying which all services can share the information. In this project the IAM policy was created for S3, SES and Lambda and SageMaker so that they can share the data amongst each other. Figure 5 shows the IAM Role

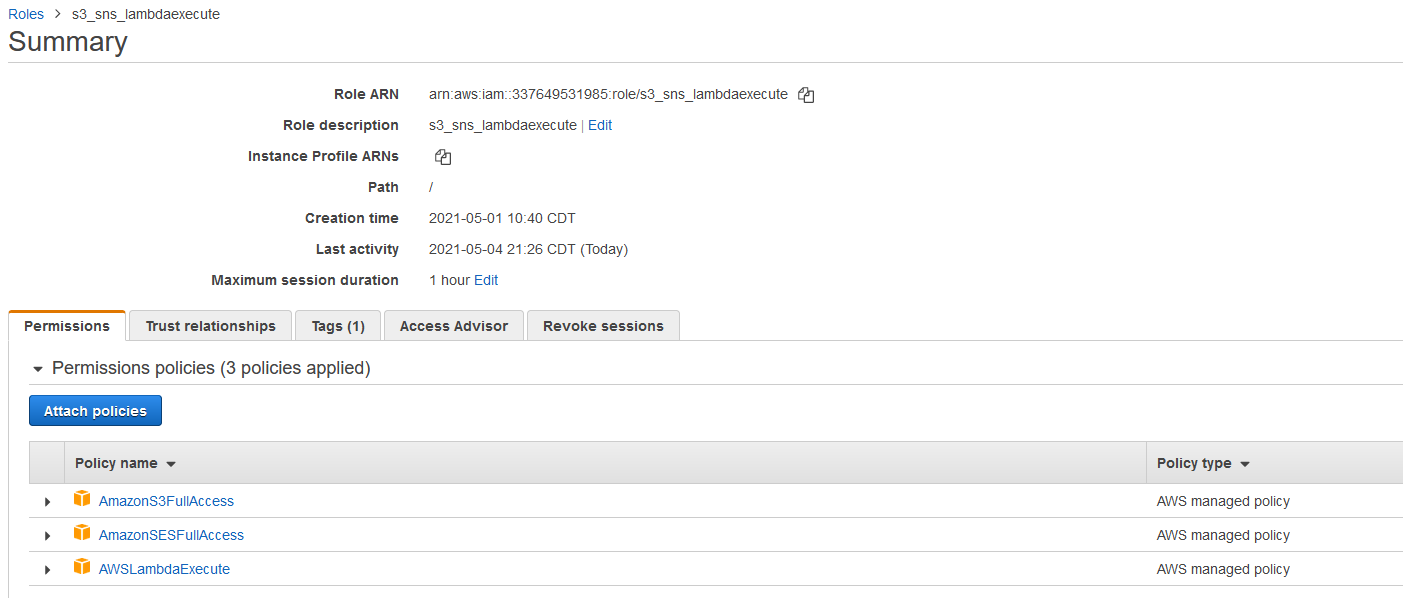


Figure 6 – IAM Role

**Simple Storage Services (S3):**

Amazon S3 bucket is a public cloud storage resource available in Amazon Web Services' (AWS) Simple Storage Service (S3), an object storage offering. Amazon S3 buckets, which are similar to file folders, store objects, which consist of data and its descriptive metadata. In the project I have used S3 to store the input X-Ray image that needs to checked, The image will be passed on to the SageMaker by importing Boto3. Along with the image a Word document contain the Patient information is also passed as an input to the SageMaker, the SageMaker uses this input image and predicts the result using the SageMaker model and updates the word document with the output predicted result and stores the image back to another S3 object bucket. Figure 7 shows the S3 buckets

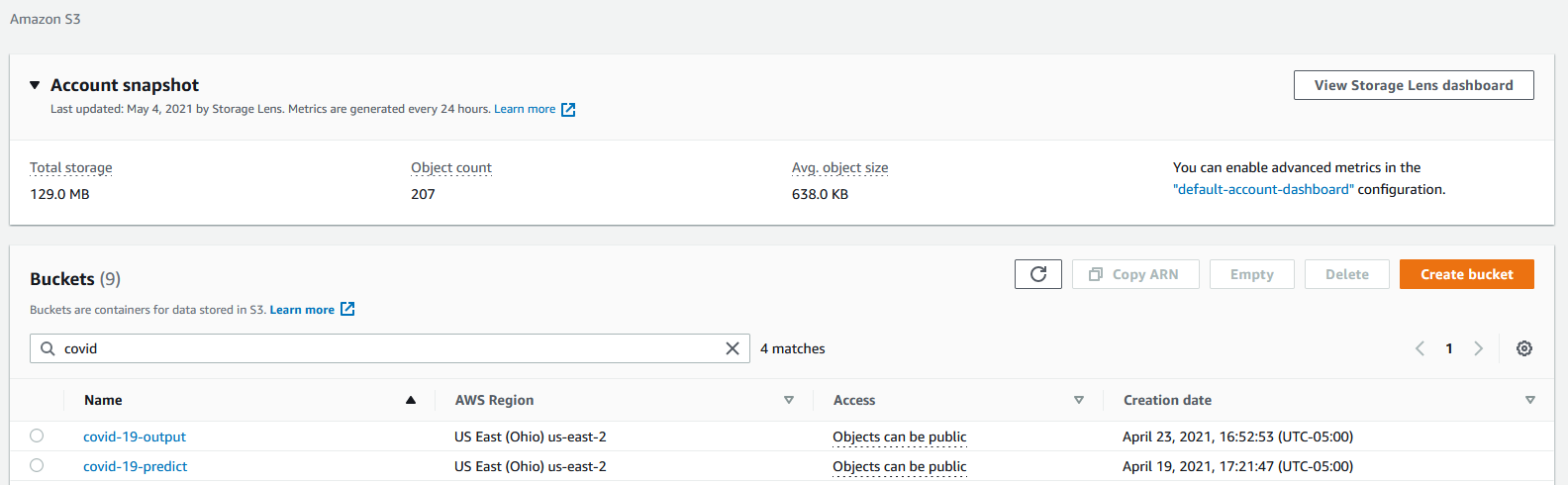


Figure 7 – S3 Bucket

**Simple Email Service (SES)**

SES is a cost-effective email service built on the reliable and scalable infrastructure that Amazon.com developed to serve its own customer base. With Amazon SES, we can send transactional email, marketing messages, or any other type of high-quality content to your customers. In this project SES was used to send the Final report with the prediction to the Patient’s email ID. In order to do that we first need to verify and activate the email id using SES services. Figure 8 shows the SES.

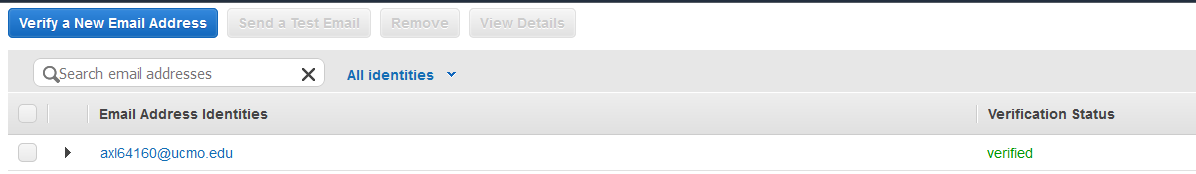
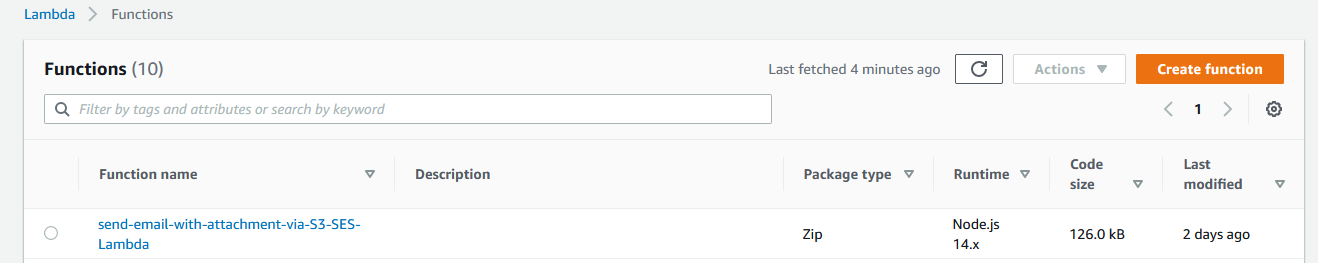
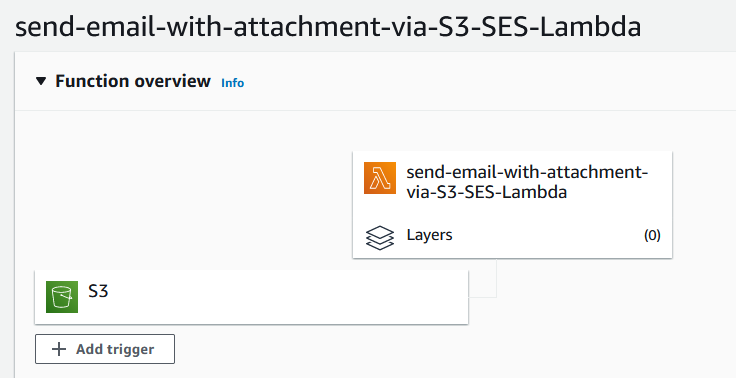


Figure 8 - SES

**Lambda Functions**

AWS Lambda is a serverless compute service that runs your code in response to events and automatically manages the underlying compute resources for you. You can use AWS Lambda to extend other AWS services with custom logic, or create your own back-end services that operate at AWS scale, performance, and security. I have used Lambda to perform the automation role, as once the SageMaker produces the output it saves the output in the S3 Bucket. Using S3 Bucket event trigger I will trigger the Lambda function so that it can send the result via email. To perform this operation in Lambda function I have incorporated NodeJS code, the code is used to pick the document from S3 and draft an email with from and to address, subject and along with the document attachment. NodeJS was specifically used as other services were not compatible to send email with attachment.





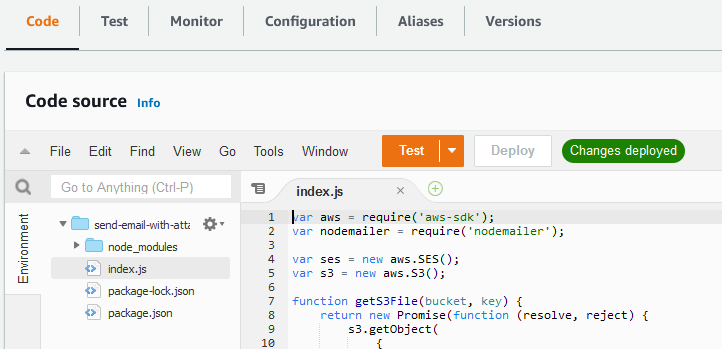
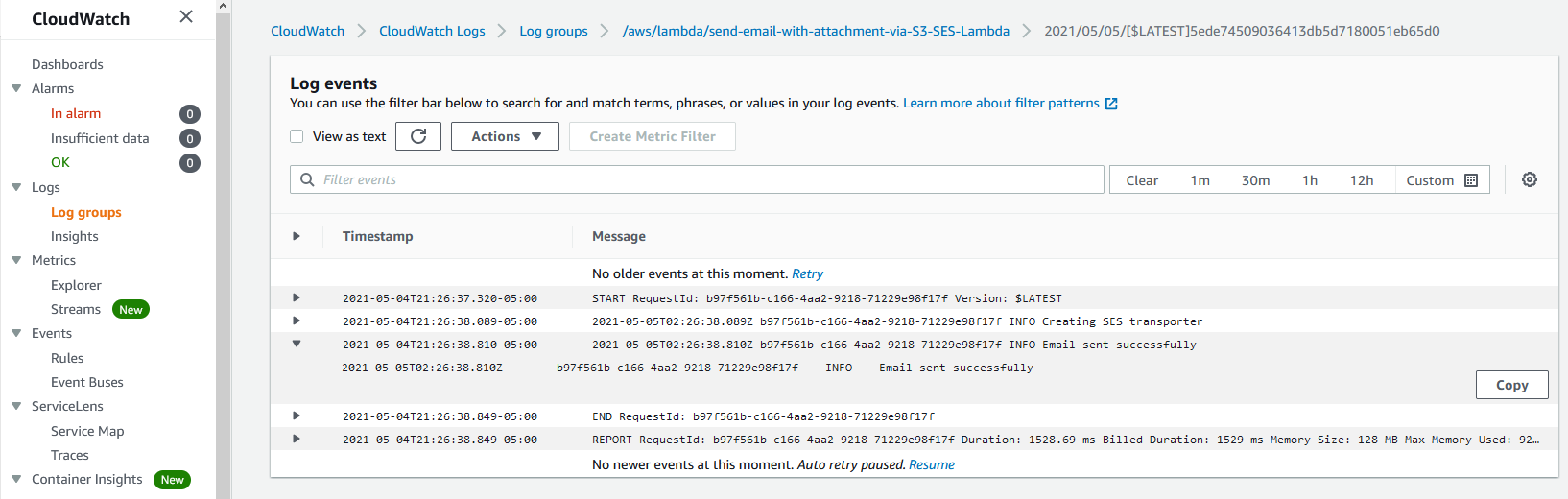


Figure 9 – Lambda Configuration

**CloudWatch**

CloudWatch is a monitoring and management service that provides data and actionable insights for AWS, hybrid, and on-premises applications and infrastructure resources. With CloudWatch, we can collect and access all the performance and operational data in form of logs and metrics from a single platform. In the project I have used Cloudwatch to monitor the status, execution and also to check whether S3 was able to trigger the Lambda function via object event trigger and whether Lambda function was able to trigger SES and send the email to the registered email ID, if there was any configuration error I was able to find out in CloudWatch Log and fix the code. Figure 10 shows the Cloudwatch event Log



**Figure 10 – CloudWatch**

**CONCLUSION:**

In this project, I have successfully shown how we can use Classification neural network model along with AWS services to predict Covid-19 using Chest X-Ray images as well as using the same model we can predict Pneumonia viral infections. This model can also be used in future with some tuning to predict any other infection related to lungs. Project output Report is attached in the word document below.



**REFERENCES:**

1. S. Loussaief and A. Abdelkrim, "Machine learning framework for image classification," *2016 7th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT)*, 2016, pp. 58-61, doi: 10.1109/SETIT.2016.7939841.
2. Swarnambiga Ayyachamy, Varghese Alex, Mahendra Khened, Ganapathy Krishnamurthi, "Medical image retrieval using Resnet-18," Proc. SPIE 10954, Medical Imaging 2019: Imaging Informatics for Healthcare, Research, and Applications, 1095410 (15 March 2019)
3. Joshi A.V. (2020) Amazon’s Machine Learning Toolkit: Sagemaker. In: Machine Learning and Artificial Intelligence. Springer, Cham. https://doi.org/10.1007/978-3-030-26622-6\_24
4. **James Wiggins and Alistair Johnson: “**Perform biomedical informatics without a database using MIMIC-III data and Amazon Athena”

# AWS documentation ‘Image Classification Algorithm using Sagemaker’ - https://docs.aws.amazon.com/sagemaker/latest/dg/image-classification.html

# AWS documentation ‘Custom AWS Lambda runtime’ https://docs.aws.amazon.com/lambda/latest/dg/runtimes-custom.html

# Aws documentation ‘Building Lambda functions with Node.js https://docs.aws.amazon.com/lambda/latest/dg/lambda-nodejs.html