Here the object is once the DNS name of the ALB is invoke, it will call the Lambda Function.

Diagram as below:

A diagram of a company

Description automatically generated

First Create the SQS queue:

A red circle with black text

Description automatically generated

Pull for message should be like the below:

A screenshot of a computer

Description automatically generated

Second: Create the ALB Rules as shown below:

A screenshot of a computer

Description automatically generated

The target group is the Lambda you create, see below:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Lastly the Lambda Code as below (NodeJS)

const AWS = require('aws-sdk');

const sqs = new AWS.SQS();

const SQS\_URL = "https://sqs.us-east-1.amazonaws.com/478614889459/AEB\_Queue";

const randomSayings = [

    "Life is what happens when you're busy making other plans.",

    "The way to get started is to quit talking and begin doing.",

    "Life is what we make it, always has been, always will be.",

    "The purpose of our lives is to be happy.",

    "Life is short, and it's up to you to make it sweet."

];

exports.handler = async function (event, context) {

    const path = event.path;

    switch (path) {

        case '/sendmessage':

            const randomSaying = randomSayings[Math.floor(Math.random() \* randomSayings.length)];

            const messageBody = "Hello World - " + randomSaying + ". This is Lambda function is behind the ALB.";

            const sendMessageParams = {

                MessageBody: messageBody,

                QueueUrl: SQS\_URL

            };

            try {

                await sqs.sendMessage(sendMessageParams).promise();

                return {

                    isBase64Encoded: false,

                    statusCode: 200,

                    headers: {

                        "Content-type": "text/plain"

                    },

                    body: 'Message sent to SQS successfully.'

                };

            } catch (error) {

                console.error("Error sending message to SQS:", error);

                return {

                    isBase64Encoded: false,

                    statusCode: 500,

                    headers: {

                        "Content-type": "text/plain"

                    },

                    body: 'Failed to send message to SQS.'

                };

            }

        case '/getmessages':

            const receiveMessageParams = {

                QueueUrl: SQS\_URL,

                MaxNumberOfMessages: 10  // retrieve up to 10 messages

            };

            try {

                const data = await sqs.receiveMessage(receiveMessageParams).promise();

                const messages = data.Messages || [];

                return {

                    isBase64Encoded: false,

                    statusCode: 200,

                    headers: {

                        "Content-type": "application/json"

                    },

                    body: JSON.stringify(messages)

                };

            } catch (error) {

                console.error("Error retrieving messages from SQS:", error);

                return {

                    isBase64Encoded: false,

                    statusCode: 500,

                    headers: {

                        "Content-type": "text/plain"

                    },

                    body: 'Failed to retrieve messages from SQS.'

                };

            }

        default:

            return {

                isBase64Encoded: false,

                statusCode: 200,

                headers: {

                    "Content-type": "text/plain"

                },

                body: 'This is Lambda function is behind the ALB.'

            };

    }

};

Outputs:

/getmessages

A screenshot of a computer

Description automatically generated

/sendmessage will just output you   
  
“Successfully sent to SQS”