UDACITY

Project# 4

Lamia Zain
lamiahasan4@gmail.com
LinkedIn

August, 16, 2024

Connect Sessions | Purpose

A Connect Session **IS**:

- Focused on learning, encouragement & graduation for a group of students coached by a Udacity Session Lead
- Setting weekly study goals
- Helping each other with progress (including peer to peer)
- Keeping everyone accountable for their responsibilities
- A way to meet individuals in tech field & learn about the industry
- **Mandatory**

A Connect Session **IS NOT**:

- A social meetup
- A study group
- A substitute for online learning
- **Optional**

UDACITY

Let’s check your progress

You are encouraged to spend at lest 10 hours/week to graduate.

Presentation date

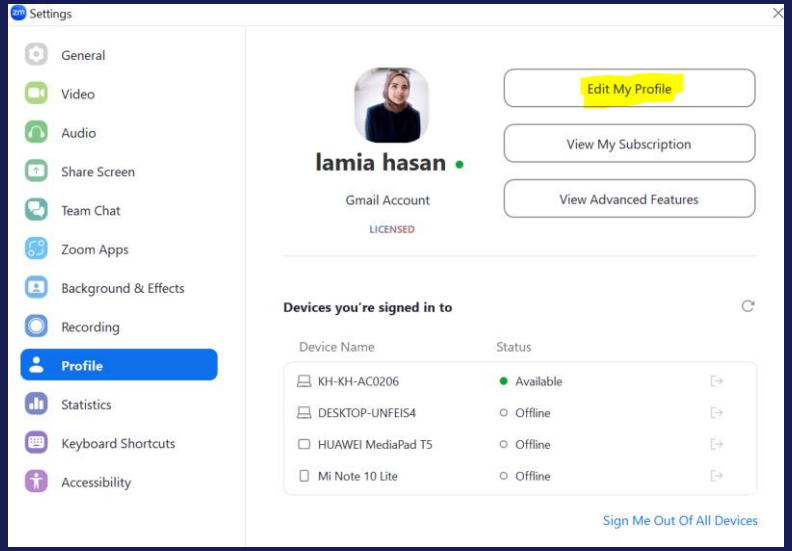
UDACITY

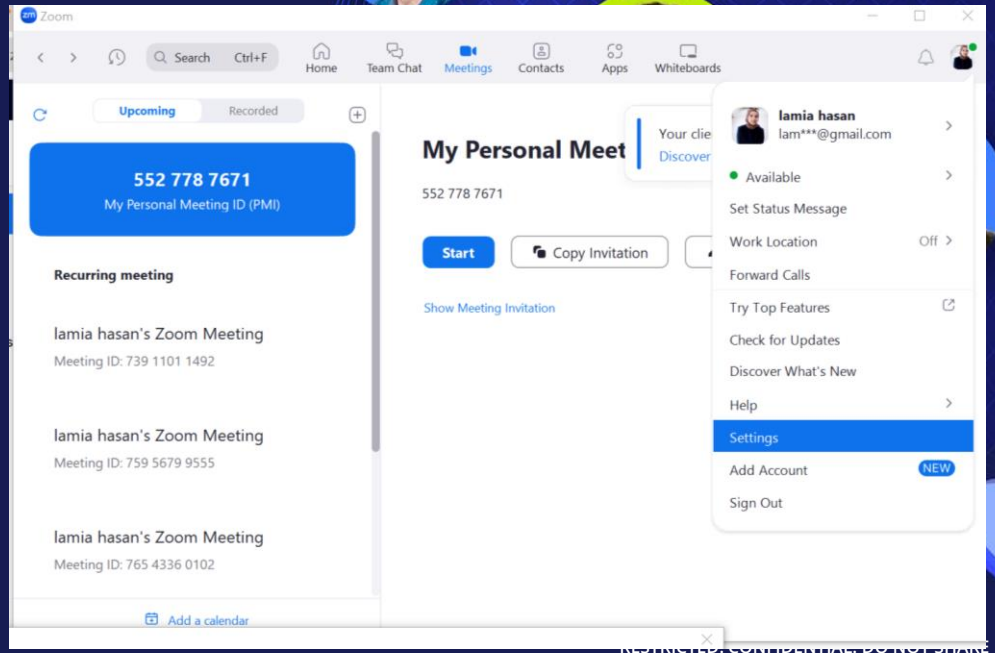
Attendance is taken automatically

Please change your name to be First Name and Last name on Zoom
Like : Lamia Zain

UDACITY

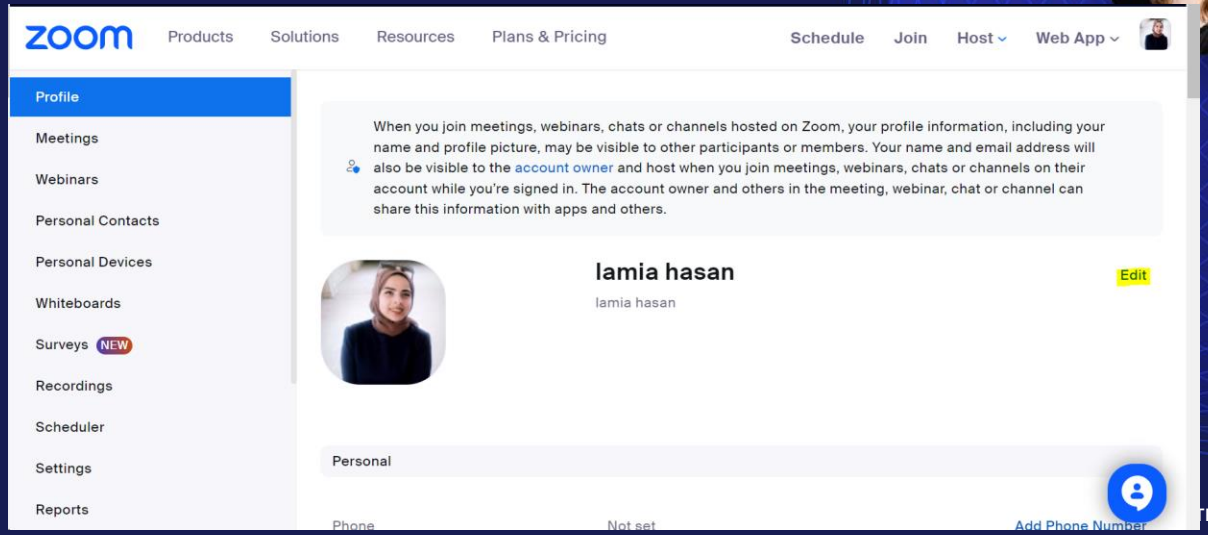
Change your Name on Zoom





UDACITY

Change your Name on Zoom



Session Lead role:

Communication Chart

Issue	Where to go?
Classroom access/ Withdrawal/ Graduation issues/ Plagiarism/ Project Review Inquiries	Email support@udacity.com
Technical Issues, Attendance, Content Related Issues/ Project inquiries	Session Lead
Session Switch/ Community related issues	Community Moderators

Program Timeline & Learning Journey

2024

April | May | June | July | August

Program Kickoff
April 10, 2024

Revoking Deadline
May 12, 2024

End of Program
August 28, 2024

First Project Submission
May 1, 2024

Second Project Submission
June 5, 2024

Third Project Submission
July 10, 2024

Fourth Project Submission
August 21, 2024

Program Period
April 10 - August 28, 2024

Four-weeks Agenda, Weekly schedule

Week 15	Jul 17, 2024			Finish the lessons below from **the Developing your First ML Workflow** Introduction to Developing ML Workflows **[Work on/submit the #4 project: Build a ML Workflow For Scones Unlimited On Amazon SageMaker]**	Developing your First ML Workflow Introduction to Developing ML Workflows
Week 16	Jul 24, 2024			Finish the lessons below from **the Developing your First ML Workflow** SageMaker Essentials **[Work on/submit the #4 project: Build a ML Workflow For Scones Unlimited On Amazon SageMaker]**	Developing your First ML Workflow SageMaker Essentials
Week 17	Jul 31, 2024			Finish the lessons below from **the Developing your First ML Workflow** Designing Your First Workflow **[Work on/submit the #4 project: Build a ML Workflow For Scones Unlimited On Amazon SageMaker]**	Developing your First ML Workflow Designing Your First Workflow
Week 18	Aug 7, 2024			Finish the lessons below from **the Developing your First ML Workflow** Monitoring a ML Workflow **[Work on/submit the #4 project: Build a ML Workflow For Scones Unlimited On Amazon SageMaker]**	Developing your First ML Workflow Monitoring a ML Workflow Project Walkthrough: Build a ML Workflow For Scones Unlimited On Amazon SageMaker

Four-weeks Agenda, Weekly schedule

Week 19	Aug 14, 2024	Aug 14, 2024	**Build a ML Workflow For Scones Unlimited On Amazon SageMaker**	Finish the lessons below from **the Developing your First ML Workflow** **[Work on/submit the #4 project: Build a ML Workflow For Scones Unlimited On Amazon SageMaker]**	Project Walkthrough: Build a ML Workflow For Scones Unlimited On Amazon SageMaker
Week 20	Aug 21, 2024			Prepare any questions you have about the content	Ask me Anything Session
Week 21	**Aug 28, 2024**	**(FINISH & GRADUATE)**			

Student Milestone | Revoking

REVOKING

Revoking is the process by which Udacity removes a student from a Nanodegree program.

AWS reserves the right to revoke you from the program if you do not comply with program requirements.

CRITERIA

Students can be revoked if they fail to:

- Submit Project 1
- Complete the required concepts

The date of **Revoking** for the scholarship is **May 12, 2024**

Code of Conduct | Plagiarism

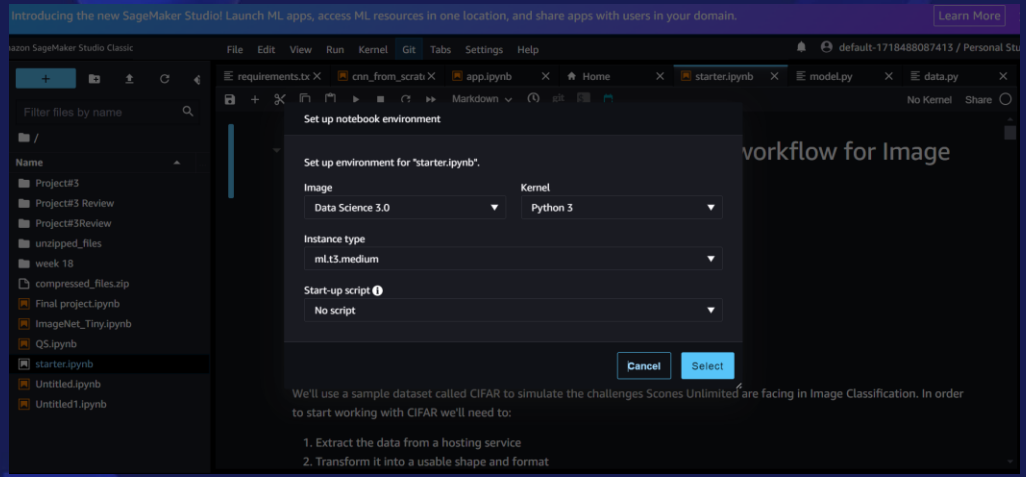
BASIC RULES

- Project submissions must consist of original work
- Submitted projects will be scanned for plagiarism
- Students who are found to have plagiarised will risk their Nanodegree being revoked
- Read the honor code and the rubric carefully for all projects

Project files

You will use sagemaker studio.

With ml.t3.medium instance, data science image



You should go through the following

- Step 1: Data staging (CIFAR100)
- Step 2: Model training and deployment
- Step 3: Lambdas and step function workflow
- Step 4: Testing and evaluation
- Step 5: Optional challenge
- Step 6: Cleanup cloud resources

You should do the following

1. Extract the data from a hosting service (locally).

2. Transform it into a usable shape and format
Instead of having it as a 1D array, we need it to be of shape (3x32x32)
Only select Bikes and Motor Bikes.

3. Load it into a production system (S3 storage).

	filenames	labels	row
23603	b'bicycle_s_002728.png'	8	23603
13860	b'motorcycle_s_001753.png'	48	13860
8585	b'motorcycle_s_000447.png'	48	8585
39781	b'motorcycle_s_002066.png'	48	39781
4367	b'velocipede_s_000586.png'	8	4367

You should do the following

1. Train an image classification model. Set model hyperparameters.
2. Make sure to have a validation accuracy > 80% (Success criteria)
3. Enable data capturing for the endpoint for inputs and outputs to be saved on S3 Bucket.
4. Deploy the model to an endpoint and **Must** print the endpoint.
5. Prepare an image from the dataset, pass it to the endpoint to get some inferences.

```
print(inference)
b'[0.9992424249649048, 0.0007575892377644777]'
```

Three Lambda functions should do the following:

1. First Lambda function Encodes an image (The location of the image on S3 bucket is given as a test event).
2. Second lambda Function Decodes the image and sends it to the endpoint to get the inference.
3. Third lambda function checks for a threshold if not met, an error will be the output.

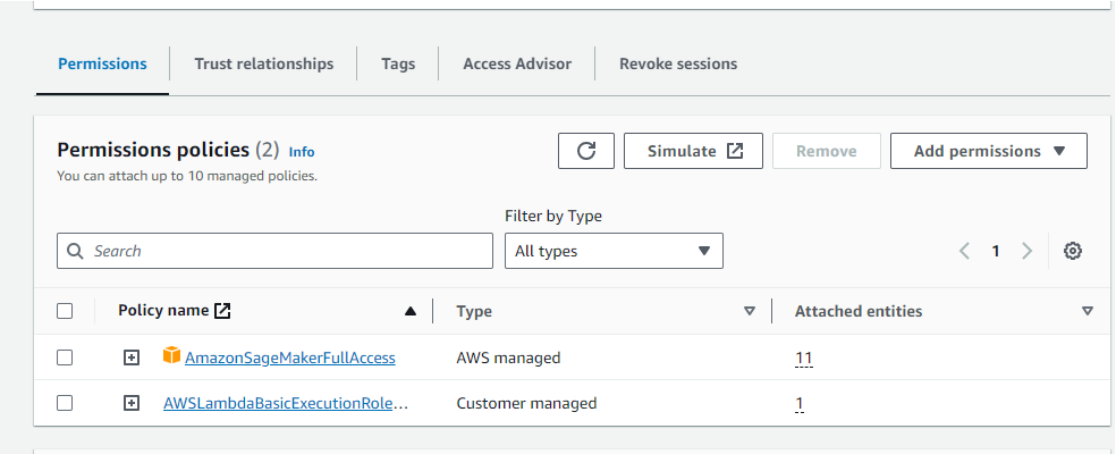
Notes: Give all lambda functions access to SagemakerFullAceess Policy

Notes: Use python3.9 runtime in all lambda functions

Don't forget to add credentials/policies to Lambda IAM role


```
{
  "errorMessage": "An error occurred (403) when calling the HeadObject operation: Forbidden",
  "errorType": "ClientError",
  "requestId": "24e8f35e-bd50-4aad-bb8e-e42a9ae0214d",
  "stackTrace": [
    "  File \"/var/task/lambda_function.py\", line 17, in lambda_handler\n    s3.download_file(bucket,key,'/tmp/image.png')\n",
    "  File \"/var/runtime/boto3/s3/inject.py\", line 190, in download_file\n    return transfer.download_file(\n",
    "  File \"/var/runtime/boto3/s3/transfer.py\", line 326, in download_file\n    future.result()\n",
    "  File \"/var/runtime/s3transfer/futures.py\", line 103, in result\n    return self._coordinator.result()\n",
    "  File \"/var/runtime/s3transfer/futures.py\", line 266, in result\n    raise self._exception\n",
    "  File \"/var/runtime/s3transfer/tasks.py\", line 269, in _main\n    self._submit(transfer_future=transfer_future, **kwargs)\n",
    "  File \"/var/runtime/s3transfer/download.py\", line 354, in _submit\n    response = client.head_object(\n",
    "  File \"/var/runtime/botocore/client.py\", line 530, in _api_call\n    return self._make_api_call(operation_name, kwargs)\n",
    "  File \"/var/runtime/botocore/client.py\", line 960, in _make_api_call\n    raise error_class(parsed_response, operation_name)\n"
  ]
}
```


First Lambda Function



Second Lambda Function, Creating deployment package

Error when importing sagemaker

lambda_function × | Environment Vari × | Execution results ×

```python
import json
import sagemaker
import base64
from sagemaker.serializers import IdentitySerializer

# Fill this in with the name of your deployed model
#ENDPOINT = ## TODO: fill in

def lambda_handler(event, context):

    # Decode the image data
    #image = base64.b64decode(## TODO: fill in)
```

lambda_function × | Environment Vari × | Execution result: ×

▾ Execution results

Test Event Name

test1

Response

```
{
  "errorMessage": "Unable to import module 'lambda_function': No module named 'sagemaker'",
  "errorType": "Runtime.ImportModuleError",
  "requestId": "7ad528cc-e3fa-4887-8707-532f610d7395",
  "stackTrace": []
}
```

Function Logs

The next function is responsible for the classification part - we're going to take the image output from the previous function, decode it, and then pass inferences back to the the Step Function.

Because this Lambda will have runtime dependencies (i.e. the SageMaker SDK) you'll need to package them in your function. *Key reading:* https://docs.aws.amazon.com/lambda/latest/dg/python-package-create.html#python-package-create-with-dependency

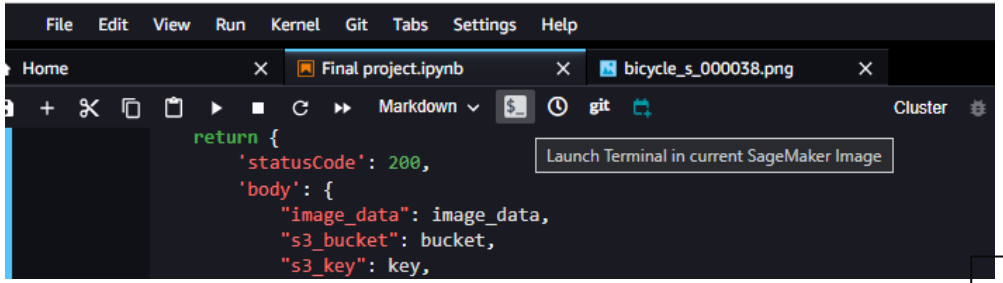
Create a new Lambda function with the same rights and a descriptive name, then fill in the starter code below for your classifier Lambda.

```python
import json
import sagemaker
import base64
from sagemaker.serializers import IdentitySerializer

# Fill this in with the name of your deployed model
ENDPOINT = ## TODO: fill in
```

As directed in project we can create a deployment package to solve this problem

1- Create a Virtual Environment in the root directory:
Go to terminal (img:1)



Check the current python version
python3 --version

If python isn't installed, install it by running this command
apt-get install python3.9

```
root@sagemaker-data-scienc-ml-t3-medium-ccb588b5efaf671be41927273f0c:~# python3 --version
Python 3.9.2
root@sagemaker-data-scienc-ml-t3-medium-ccb588b5efaf671be41927273f0c:~#
```

- Install venv

apt-get update
apt-get install python3-venv

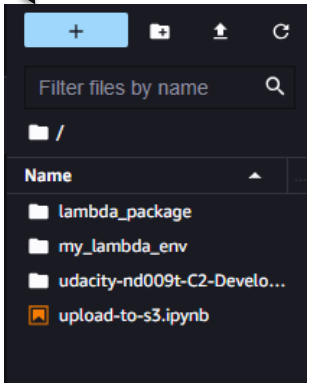
```
root@sagemaker-data-scienc-ml-t3-medium-ccb588b5efaf671be41927273f0c:~# apt-get install python3-venv
```

1- Create a Virtual Environment in the root directory:

- Create a virtual environment with name **my_lambda_env.** You will find a folder created in the root directory with the same name

python3 -m venv my_lambda_env

```
root@sagemaker-data-scienc-ml-t3-medium-ccb588b5efaf671be41927273f0c:~# python3 -m venv my_lambda_env
root@sagemaker-data-scienc-ml-t3-medium-ccb588b5efaf671be41927273f0c:~#
```



- On Debian/linux, activate the VM,

source my_lambda_env/bin/activate

```
root@sagemaker-data-scienc-ml-t3-medium-ccb588b5efaf671be41927273f0c:~# source my_lambda_env/bin/activate
(my_lambda_env) root@sagemaker-data-scienc-ml-t3-medium-ccb588b5efaf671be41927273f0c:~#
```

- Check again for the Python version of the virtual environment. This will be the same version of the lambda function

```
(my_lambda_env) root@sagemaker-data-scienc-ml-t3-medium-ccb588b5efaf671be41927273f0c:~# python3 --version
Python 3.9.2
```

2- Install sagemaker:

- Make sure pip is installed

pip –version OR pip3 --version

If pip isn't installed, install it by running this command
apt-get install pip3

```
(my_lambda_env) root@sagemaker-data-scienc-ml-t3-medium-ccb588b5efaf671be41927273f0c:~# pip --version
pip 20.3.4 from /root/my_lambda_env/lib/python3.9/site-packages/pip (python 3.9)
```

- Install sagemaker.

pip install sagemaker

```
(my_lambda_env) root@sagemaker-data-scienc-ml-t3-medium-ccb588b5efaf671be41927273f0c:~# pip install sagemaker
```

All of the following dependencies will be installed.

```
Installing collected packages: six, urllib3, rpds-py, python-dateutil, jmespath, attrs, referencing, dill, botocore, zipp, tzdata, s3transfer, pytz,
ppft, pox, numpy, multiprocess, jsonschema-specifications, contextlib2, tblib, smdebug-rulesconfig, schema, PyYAML, protobuf, platformdirs, pathos, p
andas, packaging, jsonschema, importlib-metadata, google-pasta, cloudpickle, boto3, sagemaker
```

The following 2 steps can be done using file explorer or terminal:

- Go to site packages of the virtual environment

cd my_lambda_env/lib/python3.9/site-packages

```
(my_lambda_env) root@sagemaker-data-scienc-ml-t3-medium-ccb588b5efaf671be41927273f0c:~# cd my_lambda_env/lib/python3.9/site-packages
```

- Create a lambda_function,py file

touch lambda_function.py

If touch isn't installed, u know what to do

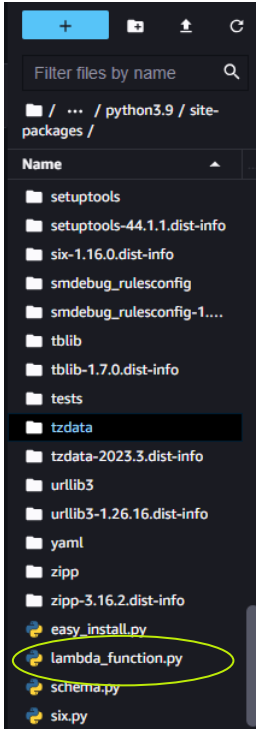
3- Create la,bda_function.py:

- Now you can find the lambda_function.py in site-packages, Open it from the file explorer and type the content of your second lambda function.
- Suppose we have this simple Function. Save it.

```python
import json
import sagemaker
import base64
from sagemaker.serializers import IdentitySerializer


def lambda_handler(event, context):

    return {
        'statusCode': 200,
        'body': json.dumps(event)
    }
```



4- Zip the site packages folder.

- Install zip package.

apt-get install zip

- Zip the site-packages in a lambda_deployment_package.zip file. First make sure you are in the site-packages folder.

```
(my_lambda_env) root@sagemaker-data-scienc-ml-t3-medium-ccb588b5efaf671be41927273f0c:~/my_lambda_env/lib/python3.9/site-packages#
```

- Type, zip -r lambda_deployment_package.zip .

```
(my_lambda_env) root@sagemaker-data-scienc-ml-t3-medium-ccb588b5efaf671be41927273f0c:~/my_lambda_env/lib/python3.9/site-packages# zip -r lambda_deplo
yment_package.zip .
```

This will compress all files including lambda_function.py in the lambda_deployment_package.zip

5- Uploading the created .zip package to lambda

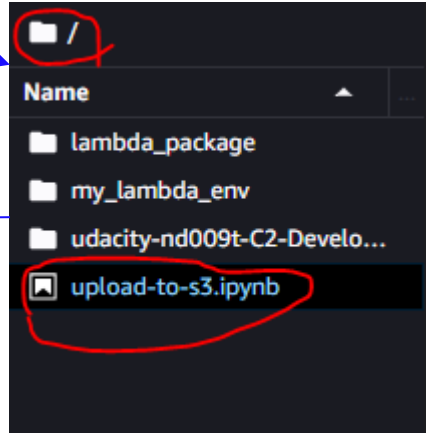
This file is > 50 MB. Uploading it to lambda function as file.zip will generate a size limit error. We will have to upload the file to S3 bucket and provide its URI to the Lambda function.

- Go to the root directory and create a jupyter notebook
- Provide this code to the notebook.

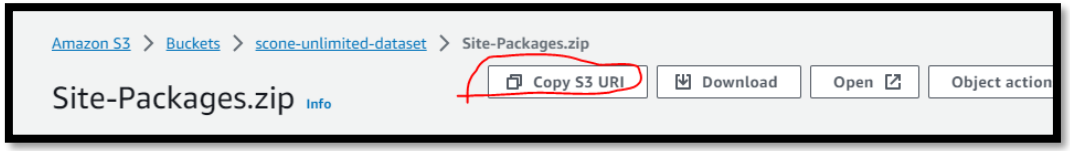
```python
import boto3

# Configure AWS credentials (replace with your credentials)

local_file_path = 'my_lambda_env/lib/python3.9/site-packages/lambda_deployment_package.zip'
#change the bucket name sothat it's similar to your preferred bucket
bucket_name = 'scone-unlimited-dataset'
file_name = 'Site-Packages.zip'
# Create an S3 client
s3 = boto3.client('s3')
s3.upload_file(local_file_path, bucket_name, file_name)
```

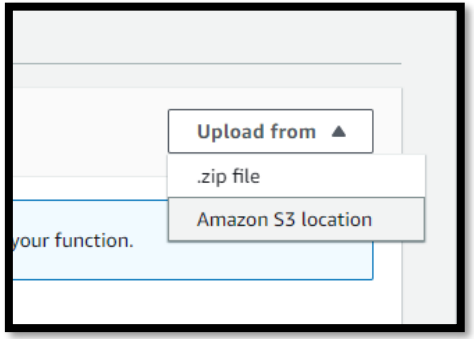


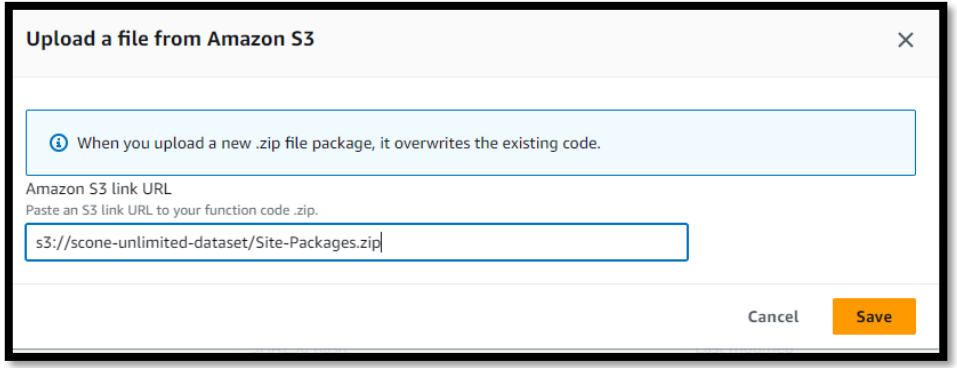
- Go to the file in the S3 bucket and copy its URI



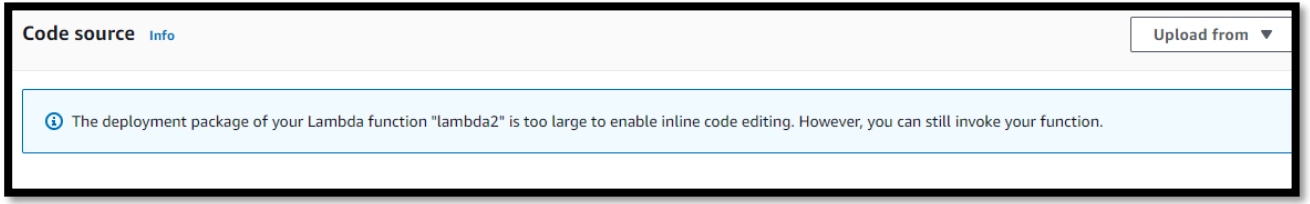
5- Uploading the created .zip package to lambda

- Create a lambda function and upload your package by providing the S3 URI. **Make sure your runtime of the lambda function is Python3.9.**



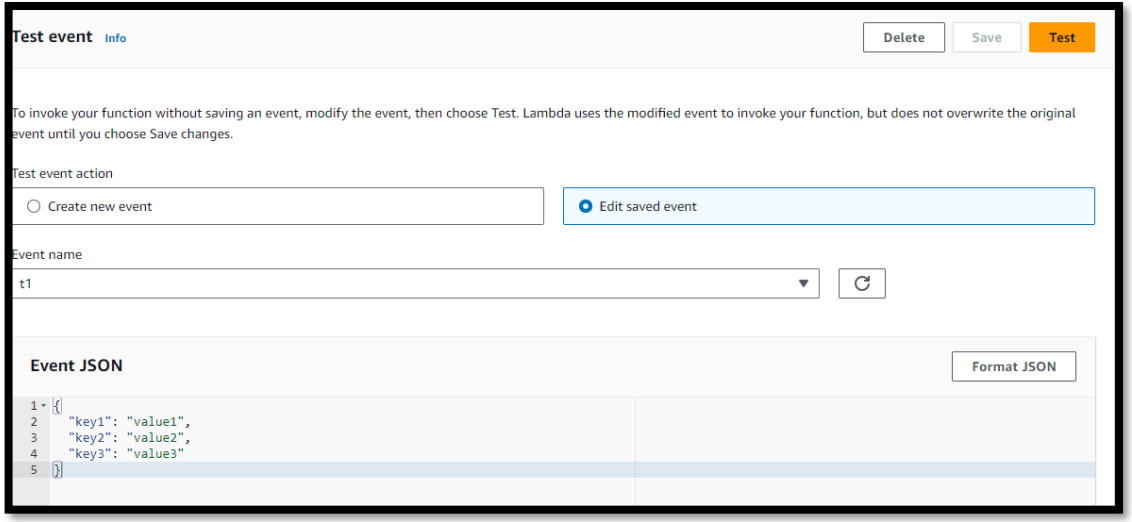


- It will show you this message which means that you can't see the content on the lambda_function inline. However, you can invoke it.



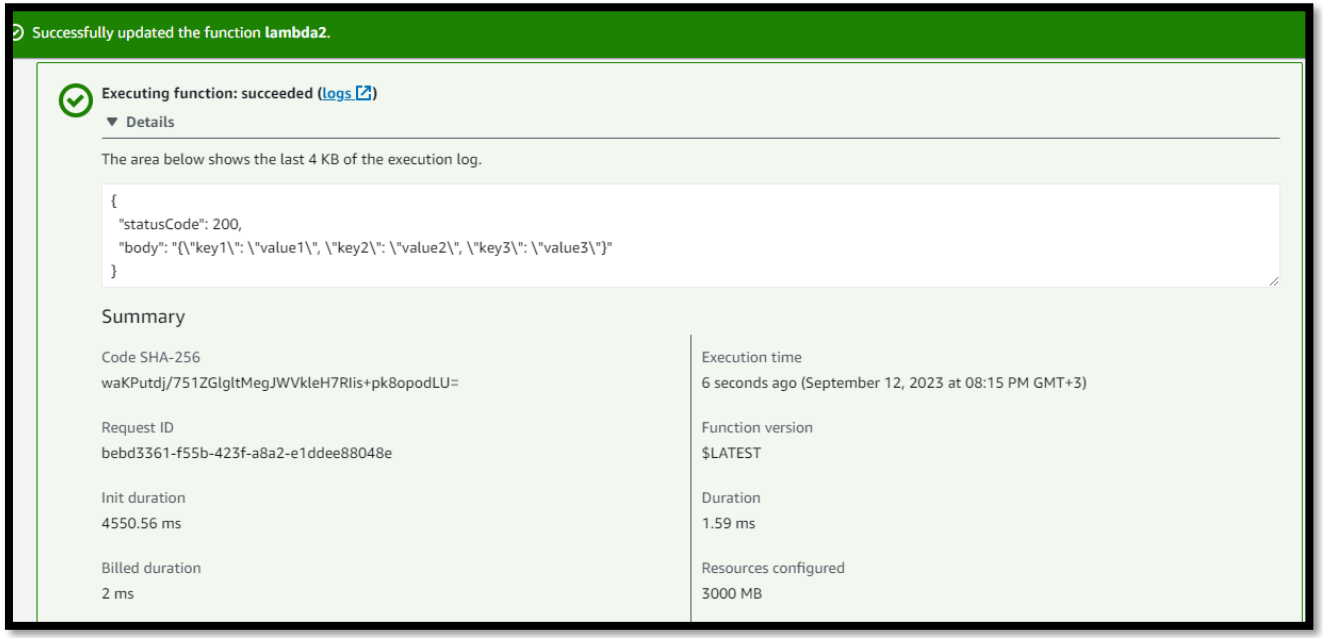
6- Invoke your lambda function

- This Lambda function we provided is very simple that imports some libraries and returns a status code and event content if it's successfully executed.
- For this demo, no need to configure a test event. Use the default one and test.



6- Invoke your lambda function

- As you can see, we successfully imported sagemaker SDK and or returned dictionary is displayed.



Notes:

- You have to edit this lambda_function to perform the task required in project#4.

- The only problem with this solution is that at each time you do edits to your lambda_function, you will have to compress the site-packages again and upload to S3.
- The good thing is you now know how to use deployment packages that aren't installed in your lambda function.

Another solution is to create a boto3 client

```
import json
import boto3
import base64

# Fill this in with the name of your deployed model
ENDPOINT = 'image-classification-2022-11-25-00-24-07-438'  ## TODO: fill in
runtime = boto3.Session().client('sagemaker-runtime')

def lambda_handler(event, context):

    # Decode the image data
    image = base64.b64decode(event['body']['image_data'])

    # Make a prediction:
    #inferences  = predictor.predict(image)## TODO: Process the payload with your predictor## TODO: fill in

    image_data = event['body']['image_data']
    response = runtime.invoke_endpoint(EndpointName = ENDPOINT, ContentType = 'image/png',Body = image)
    predictions = json.loads(response['Body'].read().decode())
    # We return the data back to the Step Function #Create a new key to the event named   "inferences"
    event['body']["inferences"] = predictions
    #print(json.dumps(event))

    return {
        'statusCode': 200,
        'body': event}
```

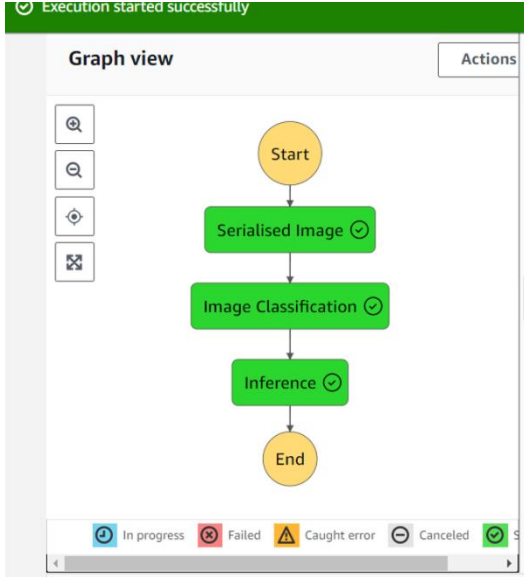
Testing and Evaluation

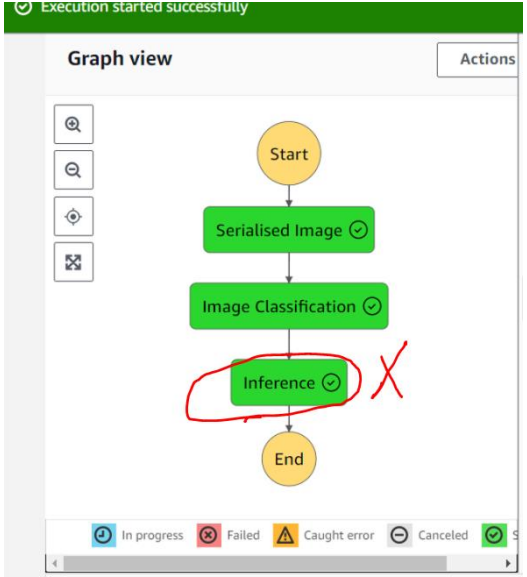
Retreive the captured data from S3 Bucket and visiualize graphs.

Submit the following files

Files to be submitted

- Project Notebook with all comments applied.
- Three Python scripts for the three lambda functions in a python file called lambda.py.
- JSON file of the step function definition(Export execution details)
- Screenshot of the step function working and failing. (Export execution Graph)





Files to be submitted

- Project Notebook
- Three Python scripts for the three lambda functions in a python file called lambda.py.
- JSON file of the step function (Export execution details)
- Screenshot of the step function working and failing. (Export execution Graph)

Step Functions > State machines > SageMakerProcessingTrainingWorkflow7 >
Execution: 7623af58-b957-4f4e-8bde-d0c73b2c126c

Execution: 7623af58-b957-4f4e-8bde-d0c73b2c126c

Edit state machine | New execution | Actions

Details | Execution input and output | Definition

Copied to clipboard

Formatted

```
3   "States": {
4     "SageMaker pre-processing step": {
5       "Resource": "arn:aws:states:::sagemaker:createProcessingJob.sync",
6       "Parameters": {
7         "ProcessingJobName": "Preprocess4",
8         "ProcessingInputs": [
9           {
10            "InputName": "input-1",
11            "AppManaged": false,
12            "S3Input": {
13              "S3Uri": "s3://breast-cancer-20/train.csv",
```

Break (10 minutes)

Satisfaction Survey

Thank you

UDACITY