

Managed Cloud Services

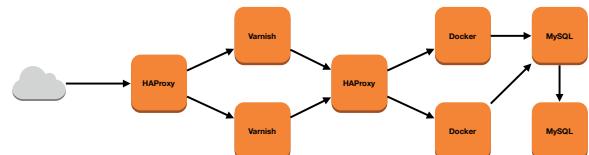
When you don't want to run it yourself

Managed Cloud Services

Virtual Servers vs Cloud Services

- All the pieces of internet applications began as discrete software run on a server you managed
- Everyone had to be at least an intermediate level sysadmin
- Managed Cloud Services aim to take away the “undifferentiated heavy lifting” from your application stack

VM Centric Architecture

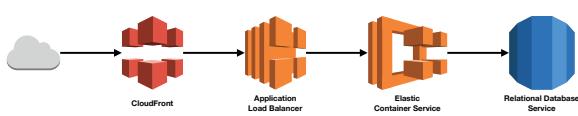


Managed Cloud Services

	VM / EC2	AWS Service
Database	MySQL	RDS MySQL
Load Balancer	HAProxy	Elastic Load Balancer Application Load Balancer
Docker	Docker	Elastic Container Service
Caching	Varnish	CloudFront

4

Cloud Centric Architecture



5

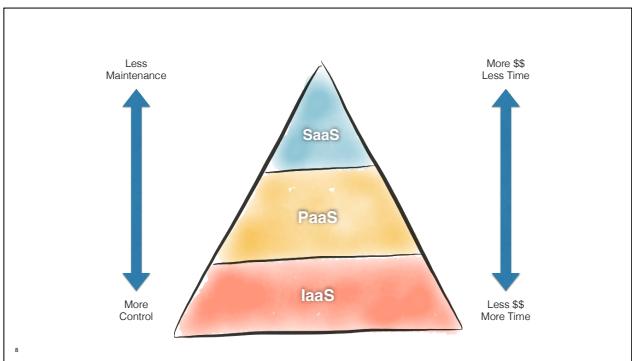
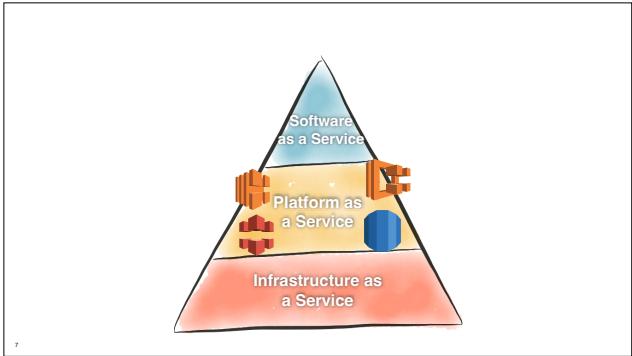
Managed Cloud Services

Virtual Servers vs Cloud Services



- All these AWS services are highly available, fault tolerant, and can be automatically deployed and backed up
- Only the RDS instance needs to be updated, and ~80% of that is automatic

6



AWS S3

Cloud Object Storage

- Amazon S3 is an object storage service that stores data as objects within buckets.
- An object is a file and any metadata that describes the file.
- A bucket is a container for objects.
- Not a File System
- Read/Write object data through AWS API

10

AWS S3

Cloud Object Storage

- Bucket names must be globally unique
- No size limits
- Objects can be public or private
- Public objects can have URLs for direct access
 - This makes S3 ideal for storing data on the internet you want other people to access.

11

AWS S3

S3 Public Website Bucket

- There are enough little things that need to be configured on an S3 bucket to allow for public web access that I built a CloudFormation template to codify it.
- AWS has a full tutorial for this:

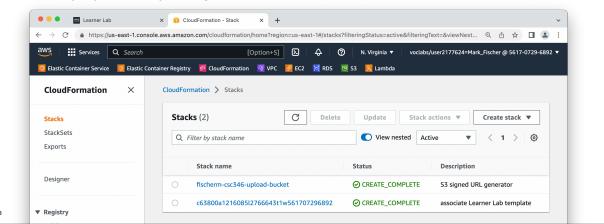
<https://docs.aws.amazon.com/AmazonS3/latest/userguide/HostingWebsiteOnS3Setup.html>

12

AWS S3

S3 Public Website Bucket

- To deploy the template, go to the CloudFormation console in the web UI.



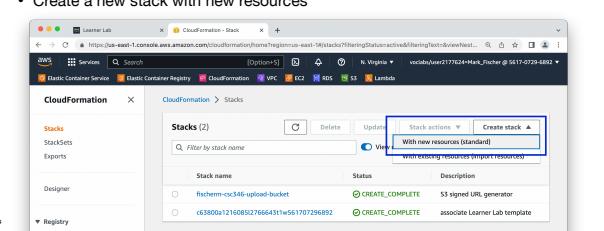
Stacks (2) | Create | Delete | Update | Stack actions | Create stack | View nested | Active | < | 1 | >

Stack name	Status	Description
fischem-csc346-upload-bucket	CREATE_COMPLETE	S3 signed URL generator
c5800a12160852766431w561707296892	CREATE_COMPLETE	associate Learner Lab template

AWS S3

S3 Public Website Bucket

- Create a new stack with new resources



Stack actions | Create stack | With new resources (standard)

Stack name	Status	Description
fischem-csc346-upload-bucket	CREATE_COMPLETE	S3 signed URL generator
c5800a12160852766431w561707296892	CREATE_COMPLETE	associate Learner Lab template

Create stack

Prerequisite - Prepare template

Prepare template
Every stack is based on a template. A template is a JSON or YAML file that contains configuration information about the AWS resources you want to include in the stack.

Template is ready Use a sample template Create template in Designer

Specify template

A template is a JSON or YAML file that describes your stack's resources and properties.

Template source
Selecting a template generates an Amazon S3 URL where it will be stored.

Amazon S3 URL Upload a template file

Amazon S3 URL
<https://fischem-csc346-upload-bucket.s3.amazonaws.com/templates/s3-website.yaml>

Cancel **Next**

AWS S3

S3 Public Website Bucket

- You can use my template directly from my class bucket.

© 2022, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookies performance

Specify stack details

Stack name
Stack name
fisherm-csc346-bucket
Stack name can include letters (a-z and A-Z), numbers (0-9), and dashes (-).

Parameters
Parameters are defined in your template and allow you to input custom values when you create or update a stack.

BucketName
The name of the S3 bucket.
fisherm-csc346-bucket

Next

AWS S3

S3 Public Website Bucket

- You need to specify a Stack name
- There's one parameter for this template, the bucket name
- I often have the stack name and bucket name be the same. Makes things simple
- Create a unique bucket name!

Events (9)

Timestamp	Logical ID	Status	Status reason
2022-10-30 20:37:58 UTC-0700	fisherm-csc346-upload-bucket	CREATE_COMPLETE	-
2022-10-30 20:37:57 UTC-0700	S3BucketPublicPolicy	CREATE_COMPLETE	-
2022-10-30 20:37:57 UTC-0700	S3BucketPublicPolicy	CREATE_IN_PROGRESS	Resource creation initiated
2022-10-30 20:37:56 UTC-0700	S3UploadBucket	CREATE_IN_PROGRESS	-
2022-10-30 20:37:54 UTC-0700	S3UploadBucket	CREATE_COMPLETE	-
2022-10-30 20:37:53 UTC-0700	S3UploadBucket	CREATE_IN_PROGRESS	Resource creation initiated
2022-10-30 20:37:52 UTC-0700	S3UploadBucket	CREATE_IN_PROGRESS	-

AWS S3

S3 Public Website Bucket

- Click through to deploy the stack
- Once the stack reaches CREATE_COMPLETE your S3 bucket should be created and configured correctly to host files able to be accessed publicly.
- We will use this in an upcoming homework to store images for our chat app.

AWS S3

Cloud Object Storage

- Clicking on a bucket shows its contents
- Can create “folders” and upload objects directly in the web UI

Objects (2) View

Name	Type	Last modified	Size	Storage class
exsample/	Folder	-	-	-
exsample.yaml	yaml	March 25, 2024, 08:52:07 (UTC-07:00)	1.4 KB	Standard

AWS S3 Cloud Object Storage

- “Folders” are just part of the object key
- It’s not a File System

arn:aws:s3:::fischerm-csc346-download/examples/Old-Main-Milky.jpg

Bucket Name

Object Key

AWS S3 Cloud Object Storage

- If configured as a public website, objects have publicly available URLs
- You can download this image from the URL

https://fischerm-csc346-download.s3.amazonaws.com/examples/Old-Main-Milky.jpg

AWS S3 Cloud Object Storage

- S3 underpins much of AWS
- Docker images in ECR are stored in S3 under the hood
- All CloudFormation templates you upload are stored in an S3 bucket
- All EC2 AMI images are stored in S3
- It is a really important service!

21

AWS S3

Too many features to go over in class

- Storage tiers - save money if you accept more risk
- Lifecycle Policies - Delete stuff after a while, or transition it to archive storage
- Integrates with many other Services - Event Based Triggers
- Cross-account access - Host files that others can use
- Requestor-pays - Host files that others have to pay to download (they don't pay you, they pay the AWS S3 network costs)
- Yes, you have to pay to read data out of S3, that's where they getcha!

22

AWS Lambda

Function as a Service?

23

AWS Lambda

Managed Code Execution

- Up to this point, if we had code we needed to execute, it had to run on a machine we managed.
 - Laptop
 - EC2
- AWS Lambda introduces another model

24

AWS Lambda

Managed Code Execution

"Lambda is a compute service that lets you run code without provisioning or managing servers. Lambda runs your code on a high-availability compute infrastructure and performs all of the administration of the compute resources, including server and operating system maintenance, capacity provisioning and automatic scaling, and logging."

<https://docs.aws.amazon.com/lambda/latest/dg/welcome.html>

25

AWS Lambda

Advantages

- Serverless - No infrastructure to manage
- Event-Driven - Nothing is "always running" (this can be good and bad)
- Pricing based only on what you use
- Scales automatically (can have limits placed)
- Can be massively parallelized
- Lets you focus on just your core application logic

26

AWS Lambda

Disadvantages

- Not for long-running processes. A given Lambda invocation cannot last longer than 15 minutes.
- Requires a different mental model for how you build an application.
 - Micro-services vs monolithic services.
- Vendor lock-in. Can't really take your AWS Lambda functions to Google App Engine.
- Memory and CPU limits are not as high as dedicated EC2 instances.
- Access to persistent file systems is not simple.

27

AWS Lambda

Image Resizing

- Let's add images to our app.
- Images are uploaded of all sorts of various sizes.
- In the posts list, we want the images to all be a uniform size.
- We want to normalize any uploaded image to be a set of standard sizes, a square thumbnail and a larger view, but still possibly smaller than the original image.

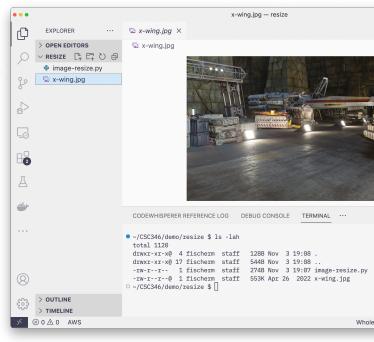
28

AWS Lambda

Image Resizing in Python

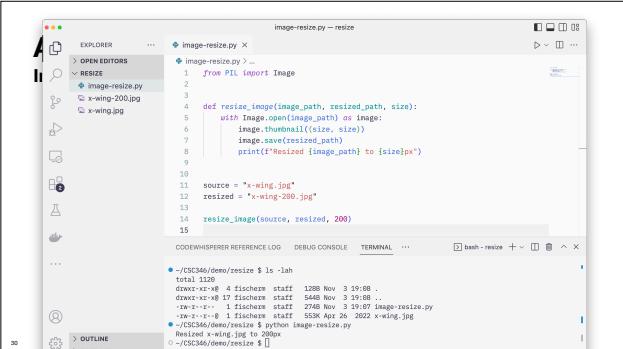
- How do we resize an image in Python?
- Use the Pillow / PIL module

29



The screenshot shows the AWS Lambda CodeWhisperer interface. On the left, the Explorer sidebar lists 'RESIZE' and 'image-resize.py'. The main area displays the code for 'image-resize.py' and a preview of the 'x-wing.jpg' image. The image is a photograph of a Star Wars X-wing fighter in a hangar. Below the code editor is a terminal window showing the command 'ls -l' and its output, which includes files like 'x-wing-200.jpg' and 'x-wing.jpg'.

```
ls -l
total 1120
drwxr-xr-x 8 4 fischen staff 1288 Nov 3 19:08 .
drwxr-xr-x 17 fischen staff 5448 Nov 3 19:08 ..
-rw-r--r-- 1 fischen staff 2748 Nov 3 19:07 image-resize.py
-rw-r--r-- 1 fischen staff 553K Apr 26 2022 x-wing.jpg
-rw-r--r-- 1 fischen staff 290K Apr 26 2022 x-wing-200.jpg
```



The screenshot shows the AWS Lambda CodeWhisperer interface after running the code. The terminal window now shows the command 'ls' and its output, indicating that the file 'x-wing-200.jpg' has been created. The code editor shows the same 'image-resize.py' file with the added line 'resized = image.resize(resized_path, 200)'.

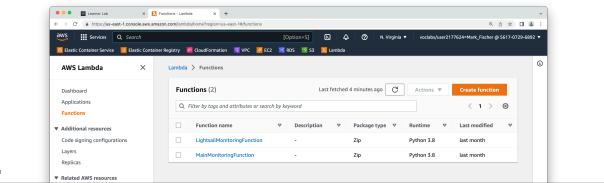
```
ls
total 1120
drwxr-xr-x 8 4 fischen staff 1288 Nov 3 19:08 .
drwxr-xr-x 17 fischen staff 5448 Nov 3 19:08 ..
-rw-r--r-- 1 fischen staff 2748 Nov 3 19:07 image-resize.py
-rw-r--r-- 1 fischen staff 553K Apr 26 2022 x-wing.jpg
-rw-r--r-- 1 fischen staff 290K Apr 26 2022 x-wing-200.jpg
```

AWS Lambda

Image Resizing in the Cloud

- That's all fine for a laptop, how do we do this in the cloud?

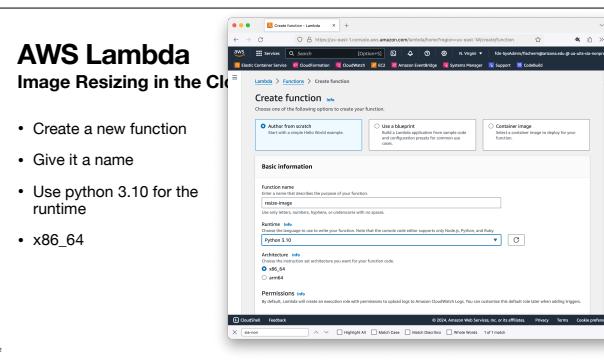
AWS Lambda Console - Search for Lambda



AWS Lambda

Image Resizing in the Cloud

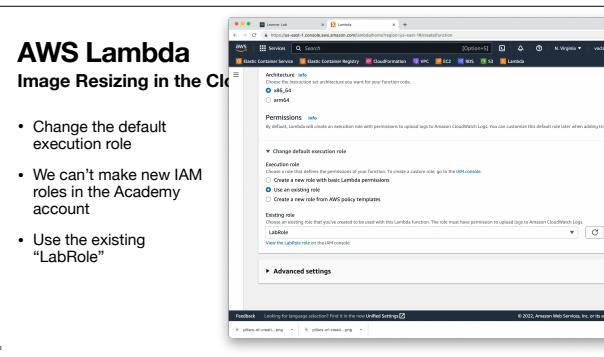
- Create a new function
- Give it a name
- Use python 3.10 for the runtime
- x86_64

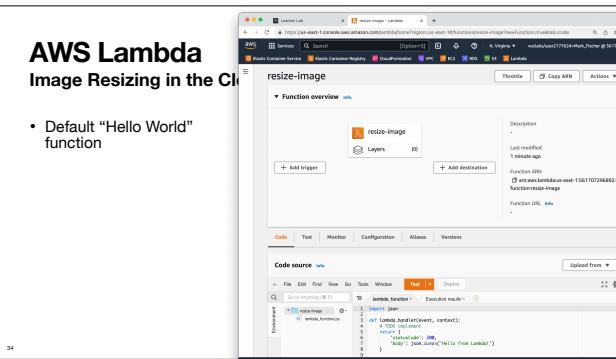


AWS Lambda

Image Resizing in the Cloud

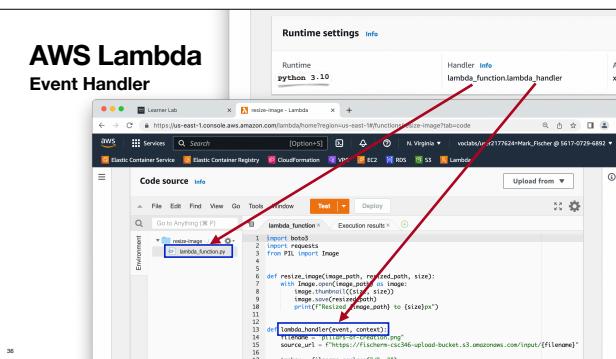
- Change the default execution role
- We can't make new IAM roles in the Academy account
- Use the existing "LabRole"





AWS Lambda Event Handler

- We mentioned that Lambda is event driven
- Your code runs inside of the Lambda Runtime
- The Lambda Runtime handles receipt of events, then calls your code and passes the event to it
- The entry point to your code is your event handler function



AWS Lambda

Event Triggers

- So what is in an event?
- It's largely dependent on what is triggering your Lambda Function
- So what can trigger Lambda?
 - In short, a lot of things!
- Most basic trigger is direct invocation. Either in the web console, or with the API

```
aws lambda invoke --function-name resize-image --payload '{"file": "x-wing.jpg"}'
```

37

AWS Lambda

Event Triggers

- Lambda integrates with more than 140 AWS services via direct integration and the Amazon EventBridge event bus.
- Commonly used Lambda event sources:

- | | |
|--|--|
| <ul style="list-style-type: none">• API Gateway• SNS• SQS• S3• CloudWatch Logs | <ul style="list-style-type: none">• CloudWatch Events• DynamoDB• EventBridge• Kinesis Data Streams• Step Functions |
|--|--|

38

AWS Lambda

Event Triggers

- Each event source will send different bits of data in the incoming `event` object.
- Here is a sample event coming from API Gateway
- Data relevant to an incoming HTTP REST call

```
{
  "resource": "/",
  "path": "/",
  "httpMethod": "GET",
  "requestContext": {
    "resourcePath": "/",
    "httpMethod": "GET",
    "path": "/Prod/",
    ...
  },
  "headers": {
    "accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8",
    "accept-encoding": "gzip, deflate, br",
    "host": "701impl41.execute-api.us-east-2.amazonaws.com",
    "user-agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3282.142 Safari/537.36"
  },
  "queryStringParameters": null,
  "multiValueQueryStringParameters": null,
  "pathParameters": null,
  "stageVariables": null,
  "body": null,
  "isBase64Encoded": false
}
```

39

AWS Lambda

Event Triggers

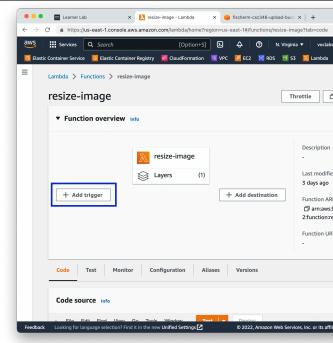
- Here's an example of an S3 ObjectCreated:Put event
 - Information about which bucket the object was created in as well as the object itself
 - Note that the Records key in the top level dictionary is an array. This event may contain multiple objects

```
| "Records": [
|   {
|     "eventVersion": "2.1",
|     "eventSource": "aws-lambda",
|     "awsRegion": "us-east-1",
|     "eventTime": "2022-06-06T20:17:18.352Z",
|     "eventName": "AWSLambdaFunctionInvoke",
|     "userIdentity": {
|       "principalId": "AMARQAYFSCSF86LKFWGIO0:user2177624",
|       "requestParameters": {"sourceIPAddress": "67.1.196.37"},
|       "responseElements": {
|         "x-amz-request-id": "V3W1V3KAKRPF74C",
|         "x-amz-id-2": "A+uVGXQD300w8c83BwHAbJ#RfMtDlnidgt5QfQ
|       },
|       "version": "1.0",
|       "configurationArn": "arn:aws:lambda:us-east-1:373tf1-399j-465-8778-7e4fd7a7c
|     },
|     "bucket": {
|       "name": "fischem-csm-244-uploaded-bucket",
|       "ownerIdentity": {
|         "principalId": "ANHNT1kR9RAGC"
|       },
|       "sequencer": "fischem-csm-244-uploaded-bucket",
|     },
|     "object": {
|       "key": "Input/wing.jpg",
|       "size": 56695,
|       "etag": "0$9b11919123dd9fefc3ebcd79e9c",
|       "sequencer": "00636164E4491ED05",
|     }
|   }
| ]
| }
```

AWS Lambda

Event Triggers

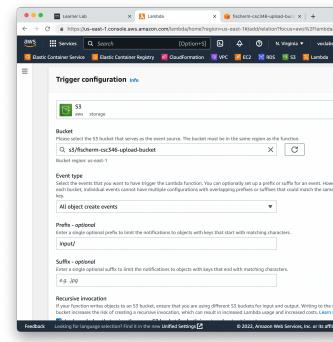
- In the Lambda console, click “Add trigger”



AWS Lambda

Event Triggers

- Choose S3 as the event source
 - Select the S3 bucket you want
 - We'll trigger on all the "CreateObject" events
 - I only want to trigger on objects with keys beginning with "input/"
 - Be careful about recursive triggering!!



AWS Lambda Event Triggers

- Once saved, you can see the trigger configuration in the "Configuration" tab of your function
- Now every time a new object is created in the input folder of that bucket, our Lambda function will run!

The screenshot shows the AWS Lambda Configuration page. In the left sidebar, 'Triggers' is selected under 'General Configuration'. A single trigger is listed: 'arn:aws:s3:us-east-1:269800669561:bucket:fischem-csc346-upload-bucket'. The details pane shows the event type is 'objectCreated' and the source is 'arn:aws:s3:::fischem-csc346-upload-bucket'. The notification name is 'e2753f1-3591-4465-877b-7edff7a75a'. The service principal is 'lambda.amazonaws.com', and the source account is '541907296092'. The statement ID is 'lambda-70cb8c7a-0ff0-4c11-acd4-4d239c0ef1f1'.

43

AWS Lambda Layers

- How do we import all the various python modules, such as the Pillow/PIL module?
- Lambda supports the idea of shared layers.
- I've created a layer which has all the dependencies built in.
- Layers aren't too hard to create, but we don't have enough time to go into that in class unfortunately.
- Only available in the same region, so use us-east-1

```
arn:aws:lambda:us-east-1:269800669561:layer:fischem-csc346-imagelayer:2
```

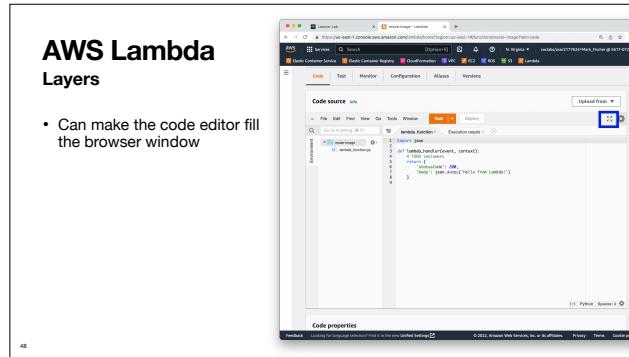
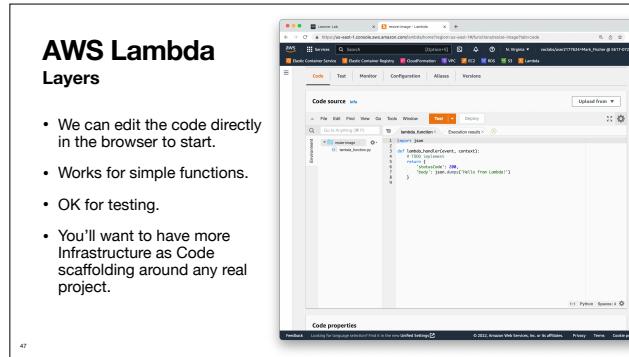
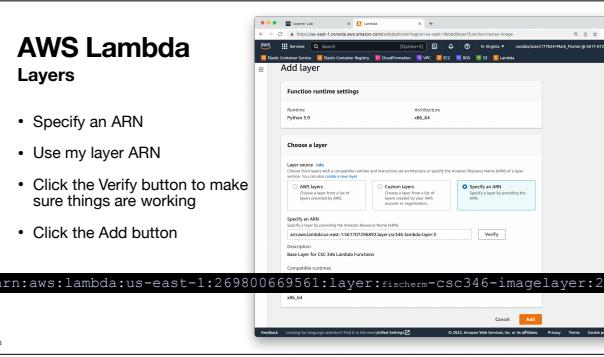
44

AWS Lambda Layers

- Scroll down
- Click the "Add a layer" button

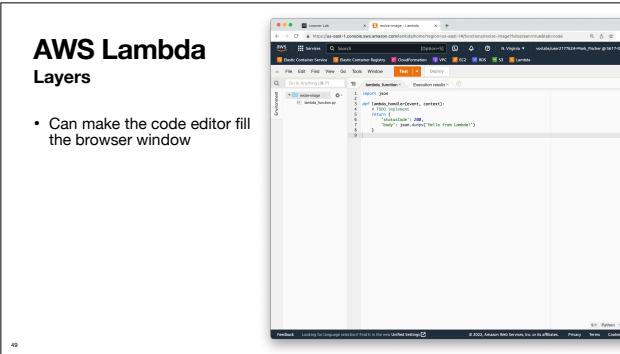
The screenshot shows the AWS Lambda function configuration page. In the 'Layers' section, there is a table with one row. The row contains 'Merge order: 1', 'Name: Python 3.9', 'Layer version: 1', 'Compatible runtime: Python 3.9', 'Compatible architectures: x86_64', and a blue 'Edit' button. Below the table is a note: 'There is no detail display.'

45



AWS Lambda Layers

- Can make the code editor fill the browser window



AWS Lambda

Image Resizing in the Cloud

- Where are our files?
 - The Lambda runtime has access to some temporary local storage
 - We need to get the file to resize from the event when a new object is added to the bucket

```
 1 // Lambda function
 2
 3 import AWS;
 4 import java.util.List;
 5 import java.util.Map;
 6 import java.util.stream.Collectors;
 7
 8 /**
 9  * This function processes incoming cold-receive events
10  * from S3 client to batch client.
11  */
12
13 public void handleEvent(ColdReceiveEvent event, Context context) {
14     List<Image> images = event.getImages();
15     String bucketName = event.getBucket();
16
17     images.forEach(image -> {
18         String key = image.getKey();
19         String extension = image.getExtension();
20
21         String fileUrl = "https://s3.amazonaws.com/" + bucketName + "/" + key;
22
23         String fileName = key.substring(key.lastIndexOf("/") + 1);
24         String baseName = fileName.substring(0, fileName.lastIndexOf("."));
25         String extensionLower = extension.toLowerCase();
26
27         String downloadUrl = print("Downloading (%s) from bucket (%s) to download path (%s)", fileUrl, bucketName, downloadPath);
28
29         S3Client s3Client = S3ClientBuilder.create().withRegion("us-east-1").build();
30
31         String itemKey = print("Uploading (%s) to bucket (%s)", fileName, downloadPath);
32
33         s3Client.uploadObject(itemKey, fileUrl, new ObjectMetadata());
34
35         print("Uploading (%s) to bucket (%s), object key (%s)", itemKey, downloadPath, fileUrl);
36
37         upload(itemKey, fileUrl, downloadPath, fileUrl);
38
39         delete(fileUrl);
40
41         print("Uploading (%s) to bucket (%s)", itemKey, downloadPath);
42
43     });
44 }
45
```

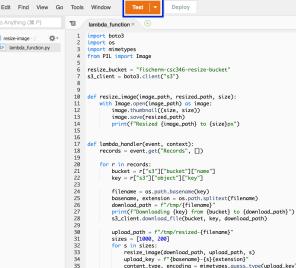
AWS Lambda

- Function needs to be Deployed before testing.

AWS Lambda

Image Resizing in the Cloud

- Once deployed, we can Test



The screenshot shows the AWS Lambda function editor with the following details:

- Region:** N. Virginia
- Name:** vendor/us/1778
- Description:** Lambda function to process CloudWatch Metrics
- Runtime:** Python 3.6
- Handler:** lambda.lambda_handler
- Code Editor:** The code is displayed in a monospaced font.

```
import json
import boto3
from botocore.exceptions import ClientError

def lambda_handler(event, context):
    # Create CloudWatch Metrics client
    metrics = boto3.client('metrics')

    # Extract event details
    event_name = event['awslogs']['log_group']
    log_stream = event['awslogs']['log_stream']
    timestamp = event['awslogs']['timestamp']

    # Define CloudWatch Metrics dimensions
    dimensions = [
        {"name": "LogStream", "value": log_stream},
        {"name": "LogGroup", "value": event_name}
    ]

    # Define CloudWatch Metrics metric data
    metric_data = [
        {
            "metricName": "Vendor Metrics",
            "dimensions": dimensions,
            "value": 100,
            "unit": "Count"
        }
    ]

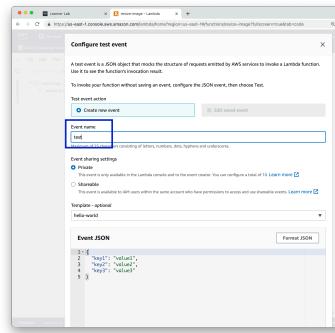
    # Put CloudWatch Metrics data
    try:
        response = metrics.put_metric_data(
            Namespace='CloudWatch Metrics Test',
            MetricData=metric_data,
            Timestamp=timestamp
        )
        print("CloudWatch Metrics data successfully sent to CloudWatch Metrics")
    except ClientError as e:
        print(f"An error occurred while putting CloudWatch Metrics data: {e}")

    return {
        'statusCode': 200,
        'body': json.dumps('CloudWatch Metrics data successfully sent')
    }
```

AWS Lambda

Image Resizing in the Cloud

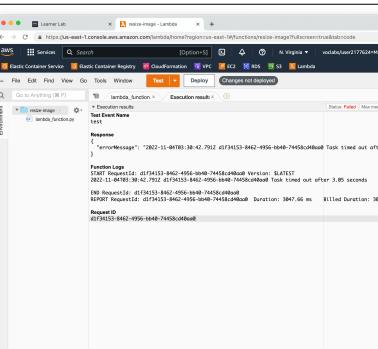
- The first time we hit Test, we're prompted to define a Test Event
 - Lambda is Event Driven
 - Our function currently doesn't use the event at all, so the default "hello-world" event is fine
 - Give it an Event name
 - Scroll down and Save

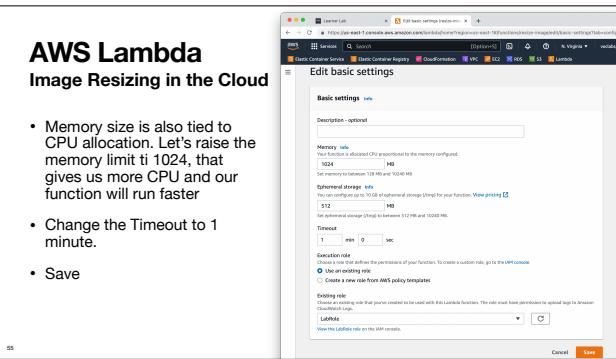


AWS Lambda

Image Resizing in the Cloud

- Try testing again
 - Error!
 - Task timed out after 3 seconds?
 - Lambda functions can last up to 15 minutes, but default to 3 seconds.





AWS Lambda
Image Resizing in the Cloud

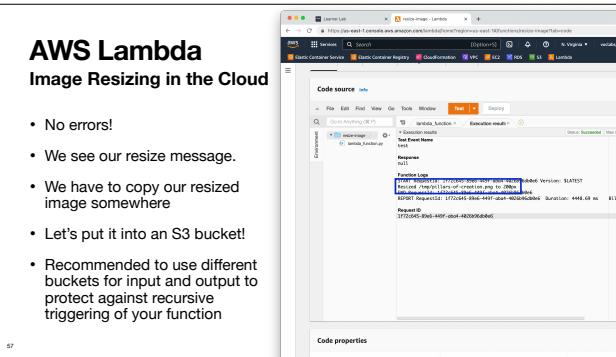
```

AWS Lambda
Image Resizing in the Cloud

# Go back to the code view and let's try our test again

def lambda_handler(event, context):
    records = event['Records']
    for record in records:
        bucket = record['s3']['bucket']['name']
        key = record['s3']['object']['key']
        file_name = os.path.join(bucket, key)
        file_size = os.path.getsize(file_name)
        if file_size > 1000000000:
            print(f"File {file_name} is too large, skipping")
            continue
        file_ext = os.path.splitext(key)[1]
        if file_ext not in ['.jpg', '.png']:
            print(f"File {file_name} has unsupported extension {file_ext}, skipping")
            continue
        file_size_kb = file_size / 1024
        print(f"Resizing file {file_name} from {file_size_kb} KB to {file_size_kb * 0.5} KB")
        resized_file_name = file_name.replace(key, f"resized{key}")
        with Image.open(file_name) as img:
            img.thumbnail((500, 500))
            img.save(resized_file_name)
        s3_client.upload_file(resized_file_name, bucket, key)
        print(f"Resized file {resized_file_name} uploaded to {bucket}/{key}")

```



AWS Lambda S3 Demo

58

AWS Lambda

Upload Images

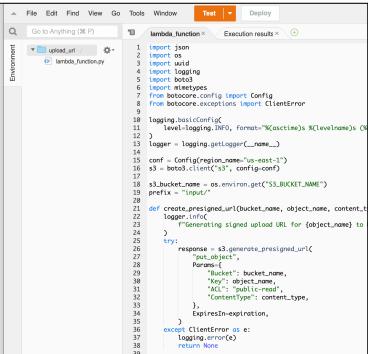
- How do we get our Chat client app to upload an image to our S3 bucket?
- AWS API calls!
- AWS S3 API provides a way to craft a 'signed' URL which we can use as the basis for a `PUT` or `POST` HTTP call to upload data directly to a bucket

59

AWS Lambda

Upload Images

- Using the `boto3` SDK we can create an `s3_client` object and use the `generate_presigned_url` method
- Get the bucket name from an Environment Variable



```
File Edit Find View Go Tools Window Test Deploy
Q Go to Anything (⌘+P) 1B lambda function Execution results
Environment
uploaded.jpg
lambda_function
1 import json
2 import os
3 import logging
4 import boto3
5 import botocore
6 import botocore.exceptions
7 from botocore.config import Config
8 from botocore.client import ClientError
9
10 logger = logging.getLogger('lambda_function')
11 level=logging.INFO, format='%(asctime)s %levelname)s (%(name)s) - %(message)s'
12
13 logger.setLevel(level)
14
15 config = Config(region_name='us-east-1')
16 s3 = boto3.client('s3', config=config)
17
18 s3.bucket_name = os.environ.get('S3_BUCKET_NAME')
19 prefix = "input/"
20
21 def create_presigned_url(bucket_name, object_name, content_type,
22                           logger=logger):
23     """Generating signed upload URL for (object_name) to
24     (bucket_name)."""
25     try:
26         response = s3.generate_presigned_url(
27             'put_object',
28             Params={
29                 'Bucket': bucket_name,
30                 'Key': object_name,
31                 'ACL': 'public-read',
32                 'ContentType': content_type,
33             },
34             ExpiresIn=expiration,
35         )
36     except ClientError as e:
37         logger.error(e)
38
39     return response
```

60

AWS Lambda Environment Variables

- Just like almost every other code execution method, Lambda provides a way to define Environment Variables

Environment variables (2)

Key	Value
\$S_DOWNLOAD_BUCKET_NAME	fischem-cs546-download
\$S_UPLOAD_BUCKET_NAME	fischem-cs546-upload

AWS Lambda Environment Variables

- These are accessible from your code using standard language functions for accessing environment variables
- Code can be used in multiple runtime environments without having to know the specifics of the runtime

```

10 generator = EnvironmentVar("generator")
11 import os
12 import logging
13 import boto3
14 from botocore.exceptions import ClientError
15 logging.basicConfig()
16 logger = logging.getLogger(__name__)
17 conf = ConfigParser(name="us-east-1")
18 conf.read("config.ini")
19 downloadBucket = os.environ.get("$S_DOWNLOAD_BUCKET_NAME")
20 downloadBucket = os.environ.get("$S_UPLOAD_BUCKET_NAME")
21 presignedUrl = None
22
23 def create_presigned_url(bucket_name, object_name, content_type):
24     """Generate signed upload URL for (object_name) to bucket
25     """
26     try:
27         response = s3.generate_presigned_url(
28             "put_object",
29             {
30                 "Bucket": bucket_name,
31                 "Key": object_name,
32                 "ACL": "public-read",
33                 "ContentType": content_type,
34             }
35         )
36     except ClientError as error:
37         logger.error(error)
38
39     return response
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54

```

AWS Lambda Function URLs

- For simple use cases, Lambda now provides a direct way to invoke the function through a URL
- Basic functionality
- API Gateway is a more robust and featured service for more production projects

No Function URL.

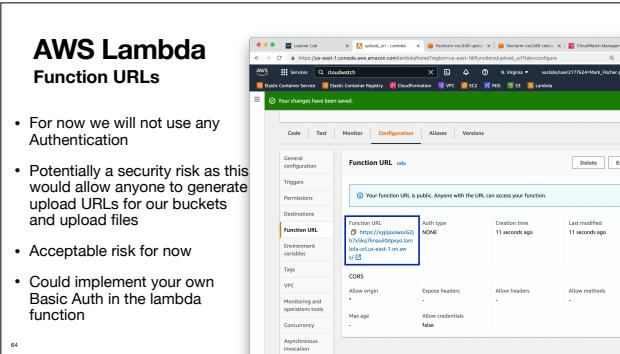
No Function URL is configured.

Create Function URL

AWS Lambda

Function URLs

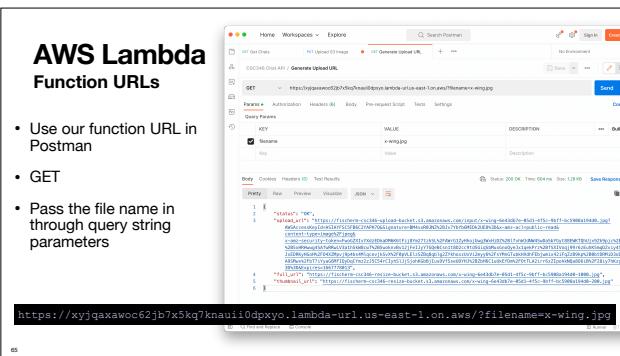
- For now we will not use any Authentication
 - Potentially a security risk as this would allow anyone to generate upload URLs for our buckets and upload files
 - Acceptable risk for now
 - Could implement your own Basic Auth in the lambda function



AWS Lambda

Function URLs

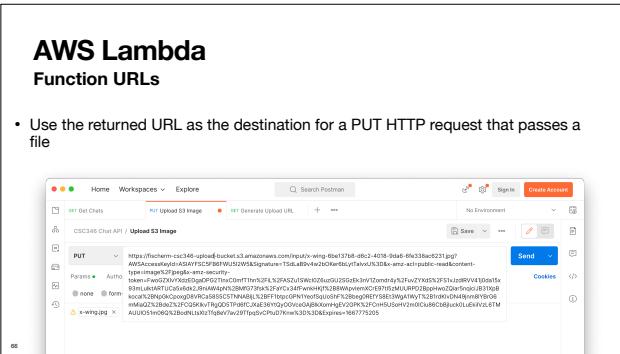
- Use our function URL in Postman
 - GET
 - Pass the file name in through query string parameters



AWS Lambda

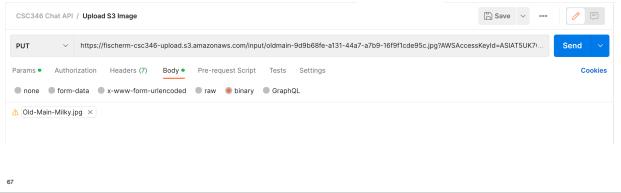
Function URLs

- Use the returned URL as the destination for a PUT HTTP request that passes a file



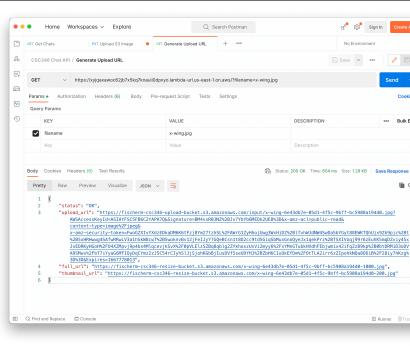
AWS Lambda Function URLs

- In Postman, select the image in the Body tab, and choose “binary”



AWS Lambda Function URLs

- Since we uploaded the image to the S3 bucket configured as the trigger for our resize function, the image should be resized automatically
- Our create upload URL call also returns the URLs of the resized objects
- We can view them directly



AWS Lambda Function URLs

- Since we uploaded the image to the S3 bucket configured as the trigger for our resize function, the image should be resized automatically
- Our create upload URL call also returns the URLs of the resized objects
- We can view them directly

