**AWS Content Manager POC**

|  |  |
| --- | --- |
| **SR** | **CONTENT** |
| 1 | Background |
| 2 | Requirement |
| 3 | General Pipeline |
| 4 | User Interface |
| 5 | Steps to replicate |
| 6 | AWS S3 |
| 7 | AWS DynamoDB |
| 8 | AWS IAM Users and group |
| 9 | AWS Lambda functions |
| 10 | API GATEWAY |
| 11 | Lambda function example |
| 12 | Useful links |

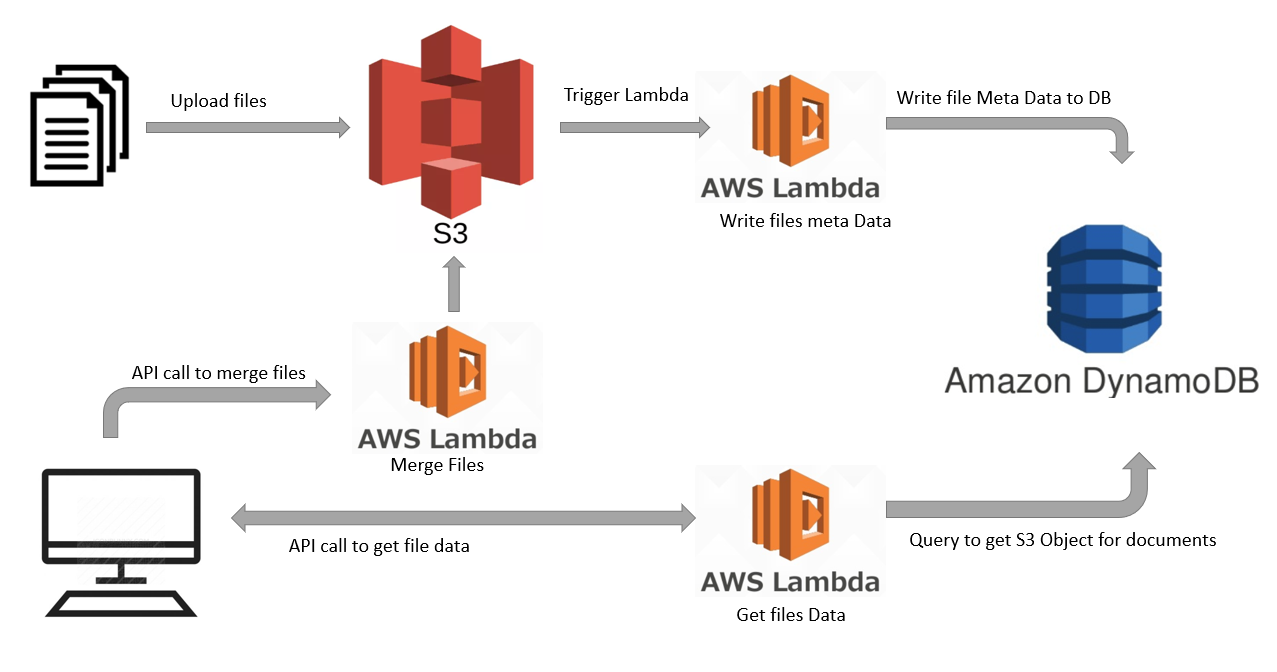
Background

A POC for storing the files along with their meta-data. The files can we searched and retrieved on the basis for querying the meta-data.

Requirement

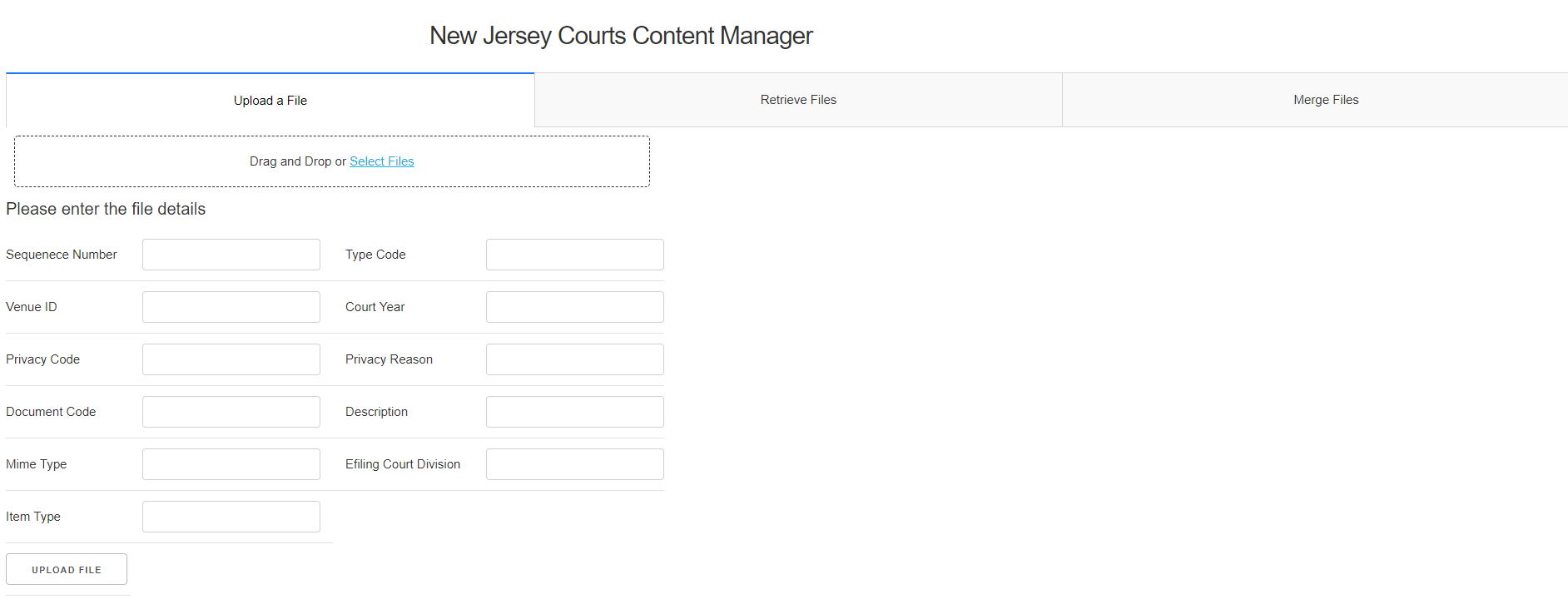
1. AWS S3
2. AWS dynamo DB
3. AWS IAM User groups and User
4. AWS Lambda functions
5. AWS API Gateway
6. Python
7. Boto3 (Python library for accessing AWS resources)
8. Dash (User Interface)

General Pipeline

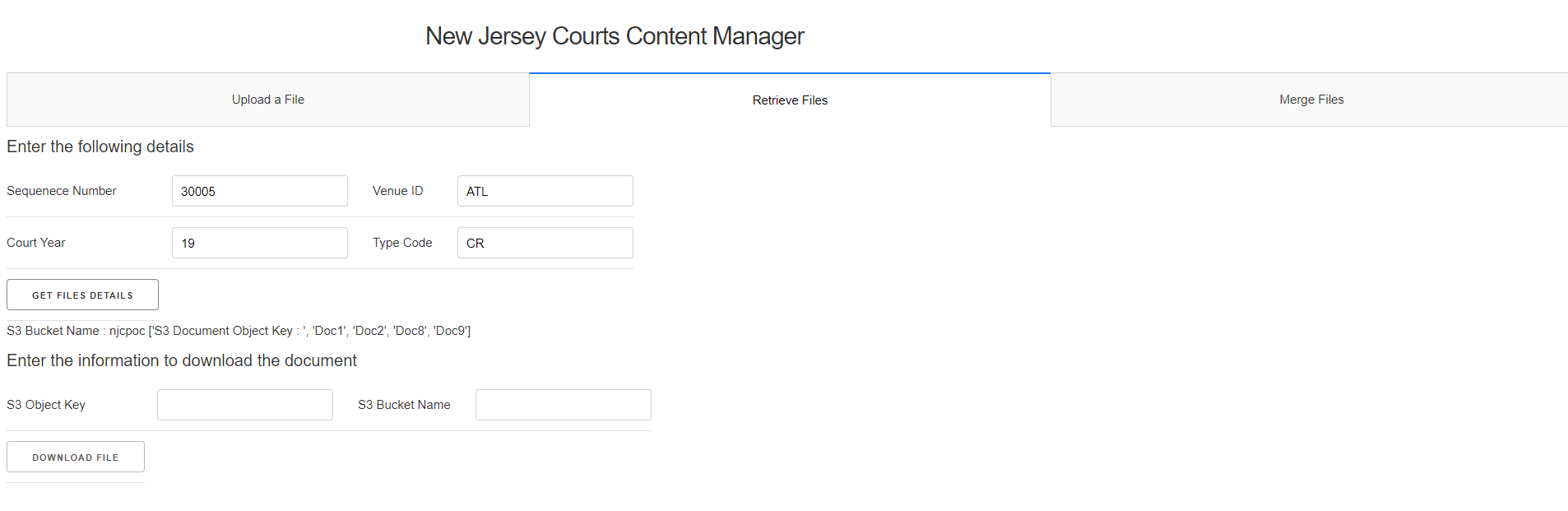


**User Interface Snapshots:**

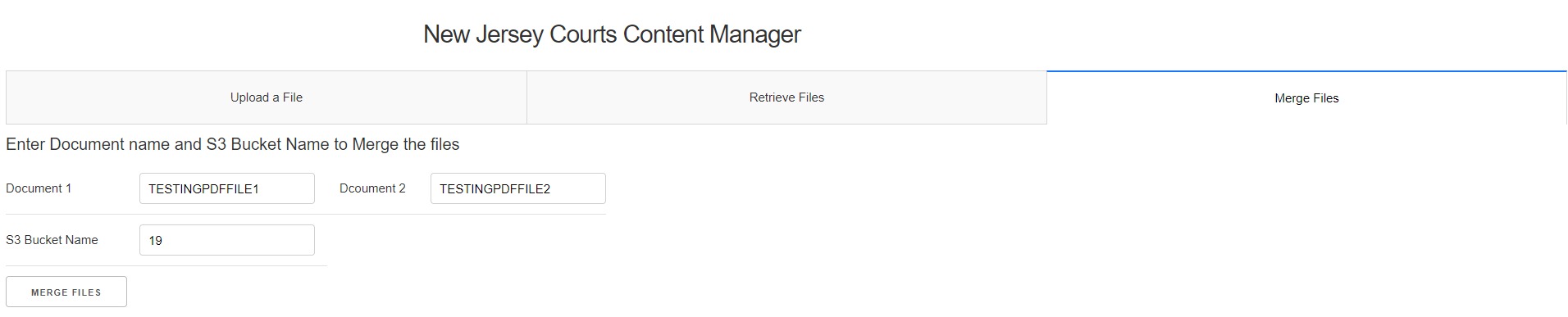
Upload File



Retrieve Files



Merge Files



**Steps to replicate the project:**

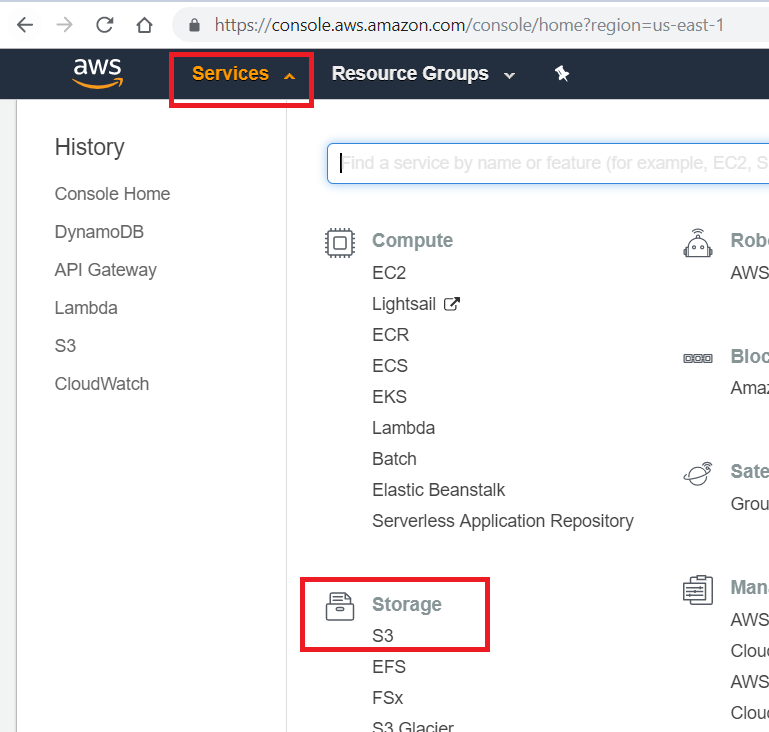
1. Create the AWS S3 bucket with name ‘njcpoc’
2. Create a DynamoDB table with name ‘CMPOC’
3. Create a Lambda function for writing file meta-data to DynamoDB and paste the code in file ‘S3MetaDataTrigger.py’
4. Add a lambda function trigger in the S3 bucket on file PUT
5. Create Lambda function to retrieve files data from DynamoDB and paste the code in file ‘RetriveFileFromDynamoDB.py’, also add API Gateway trigger for this lambda function
6. Create a Lambda function to Merge two files in S3 Bucket and paste the code in file ‘MergeFiles.py’. Create an API Gateway trigger for this function as well.
7. Under the Dash folder, update the retrieve files API in ‘getFilesAPI.py’ file and merge files API in ‘MergeFiles.py’ file
8. Maintain the files to be uploaded under the Dash folder
9. Run ‘AWSDash.py’ file to get the website

**Resources used general information:**

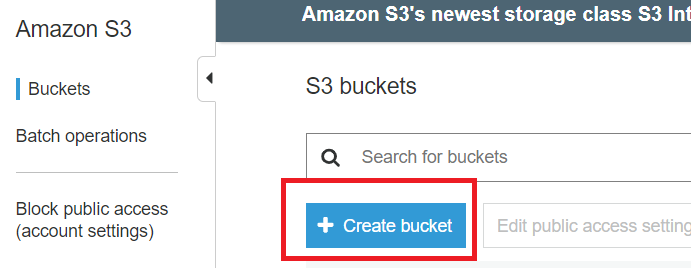
**AMAZON S3**

Login to Amazon AWS Console

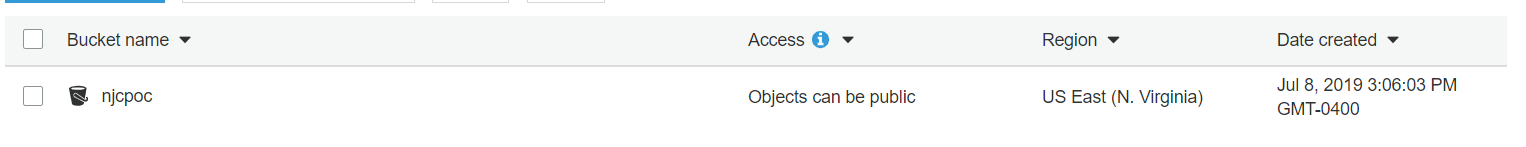
Go to Services and click on S3 under Storage



Create Bucket

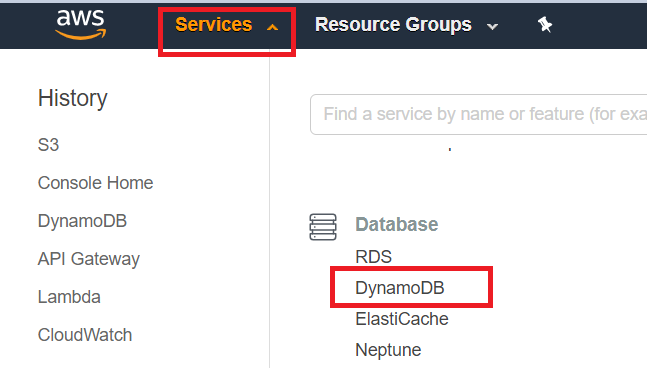


Set required Permissions and create



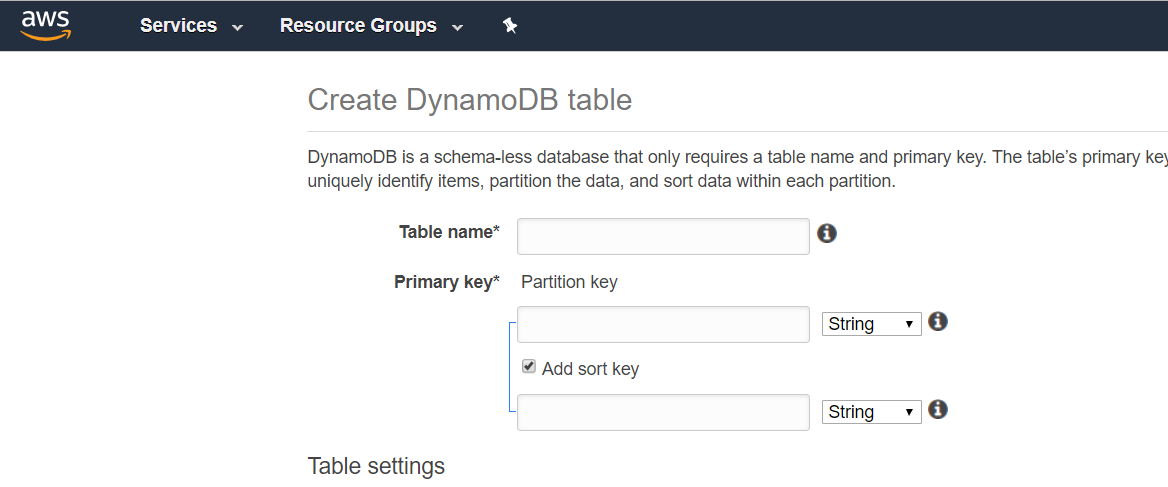
**AWS Dynamo DB Table**

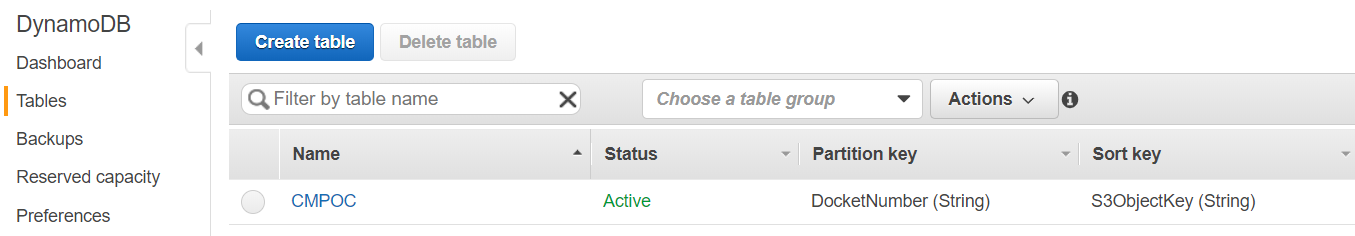
Under AWS Services Database resources select DynamoDB

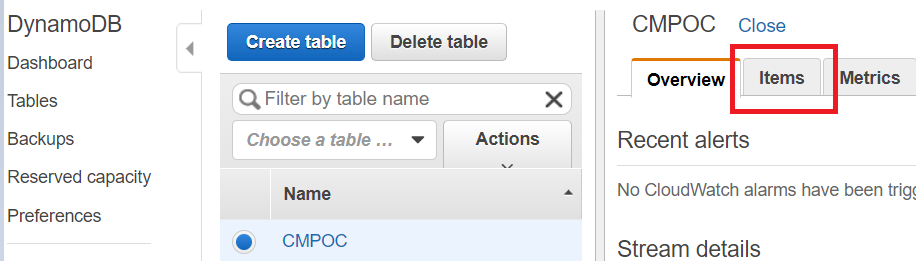


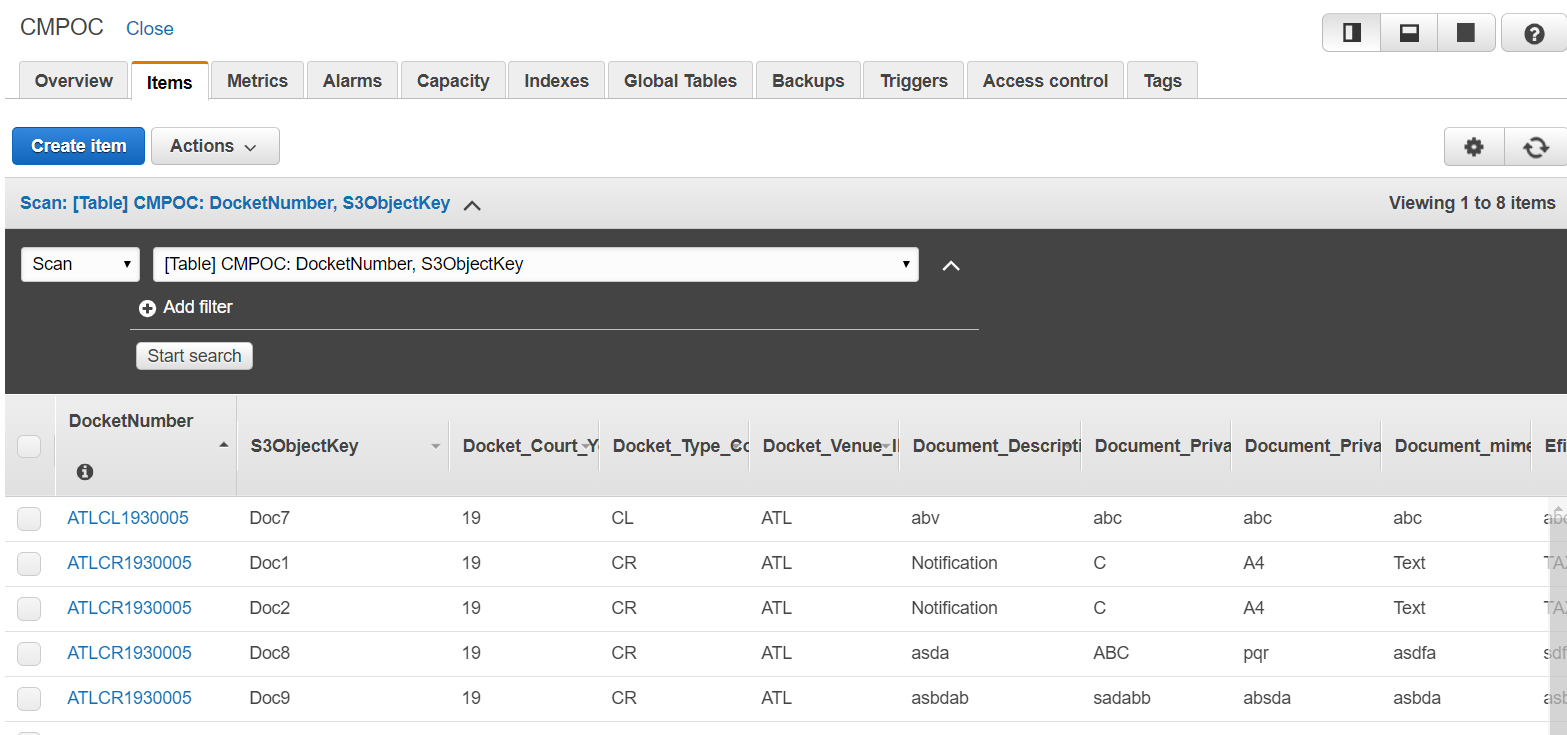
Click on create Table

1. Table Name
2. Partition Key – Acts as Primary key for the table (Unique and required)
3. Sort Key – Extra key for the table (If exist, acts as primary key along with combination of Partition Key)



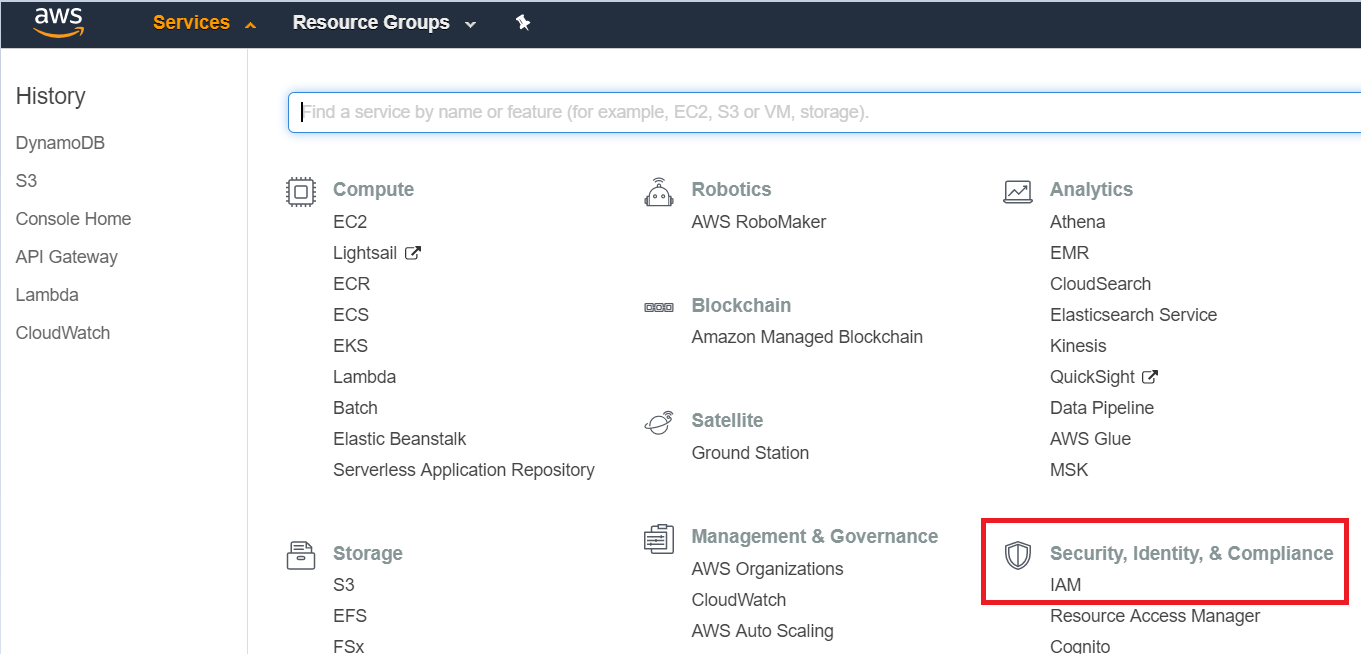




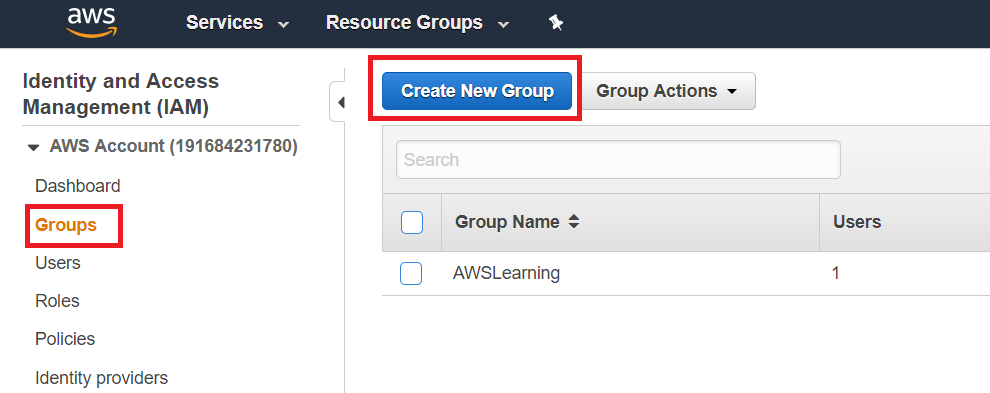


**IAM USER Group and Users**

Go to services and click IAM under security

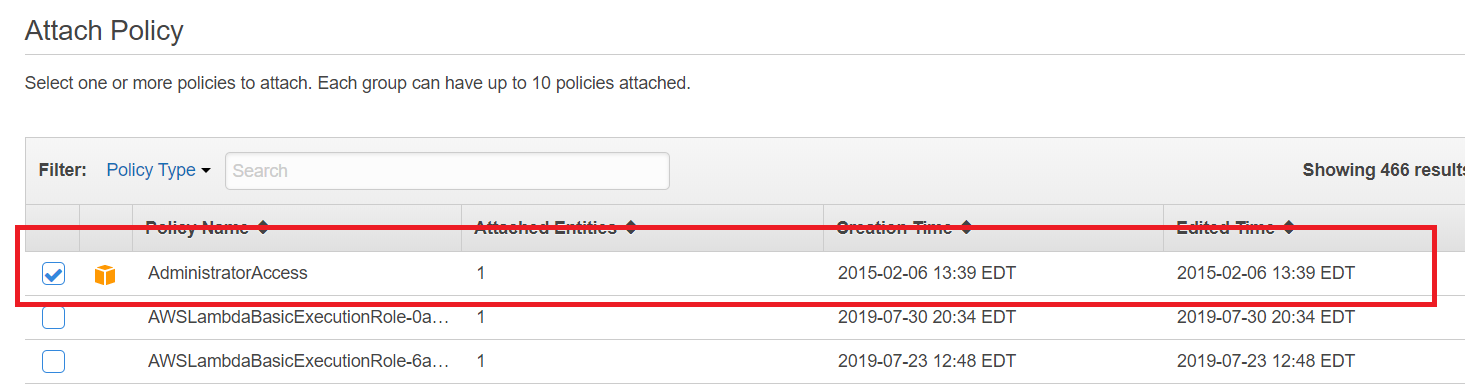


Go to groups and create a new group

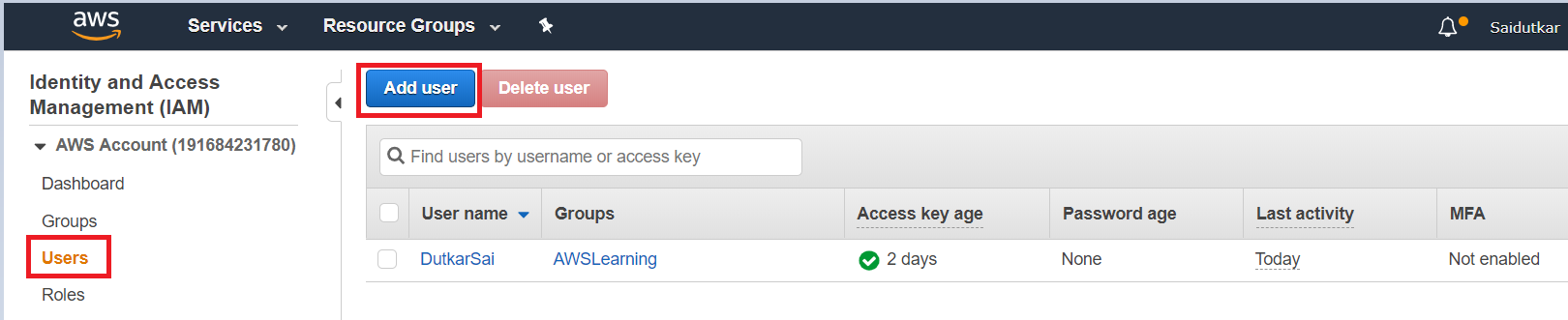


Determine the access for the group

* Administrator access allows access to all the resources for the user

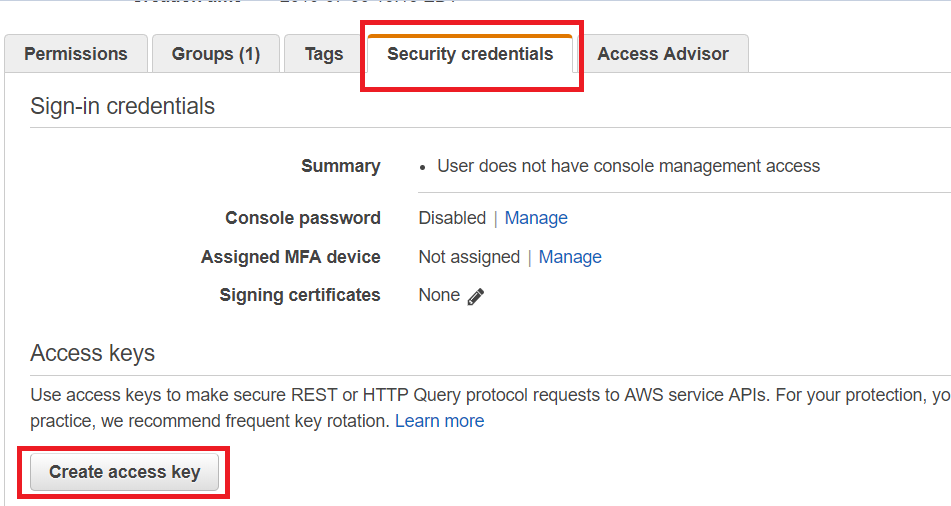


Go to Users and create a user under the Group created earlier

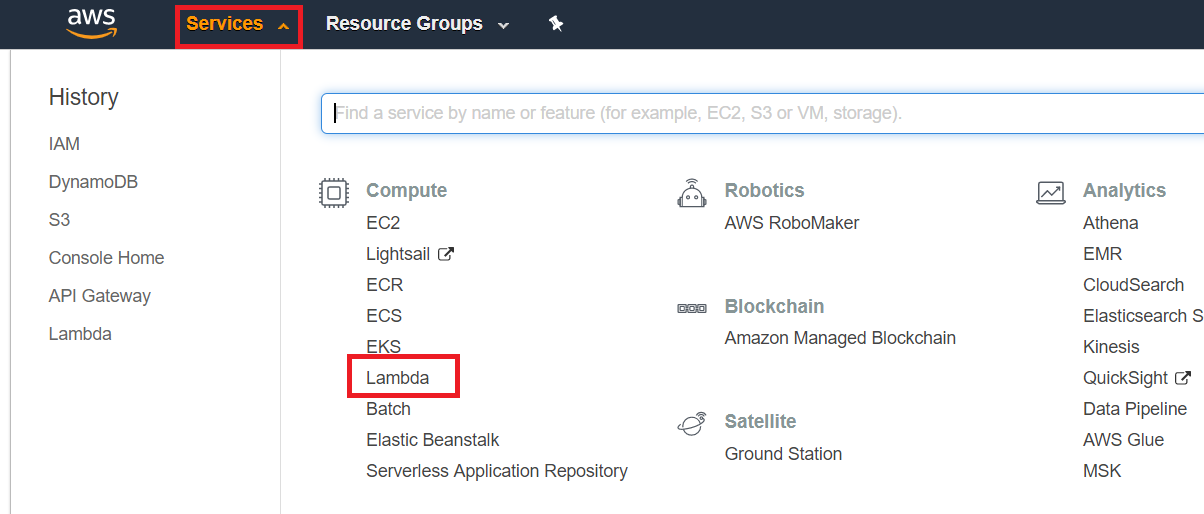


Go to the created user’s profile and under Security credentials, create the access key

Note down the **ACCESS KEY ID** and **ACCESS KEY**

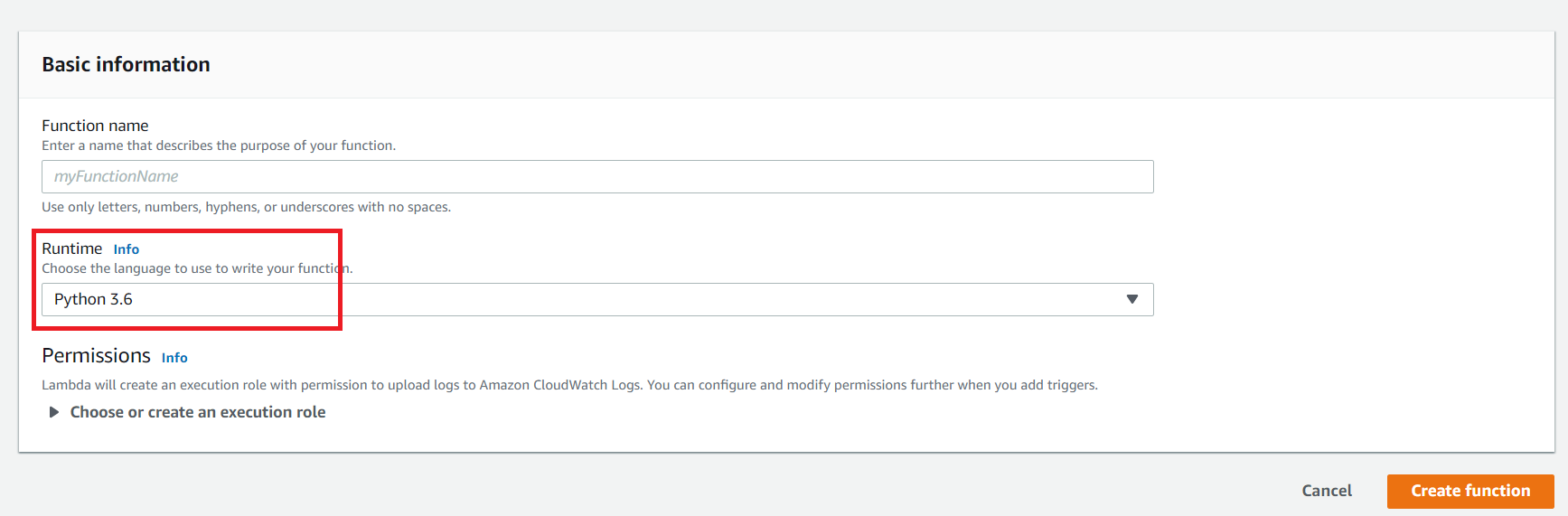


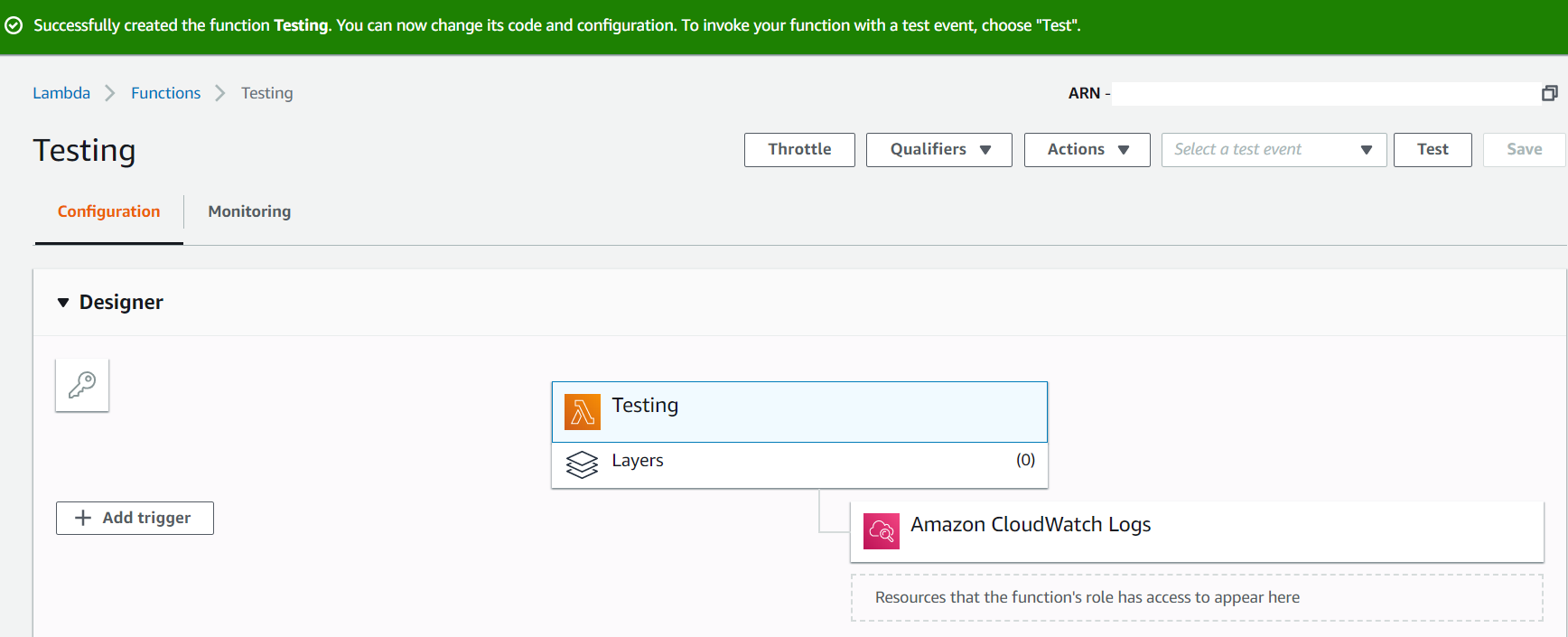
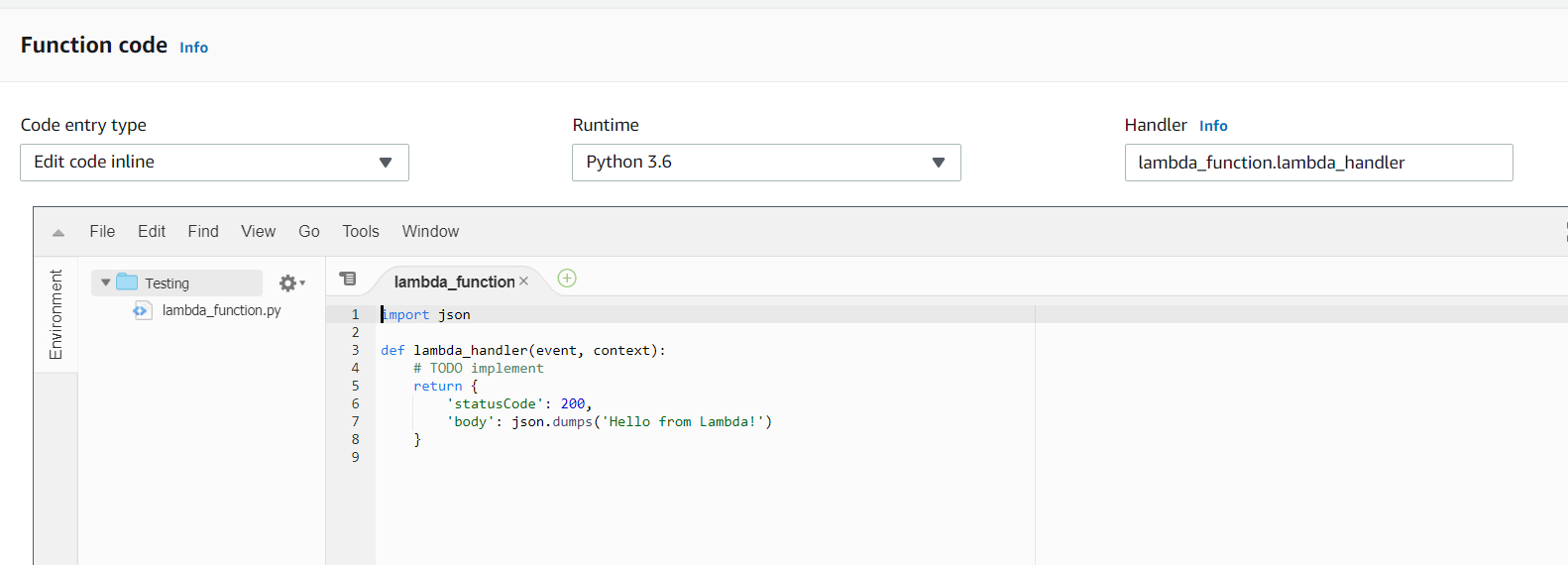
**AWS LAMBDA FUNCTION**



Click on create function

Give the name of function and select the runtime environment as Python 3.6 or Python 3.7

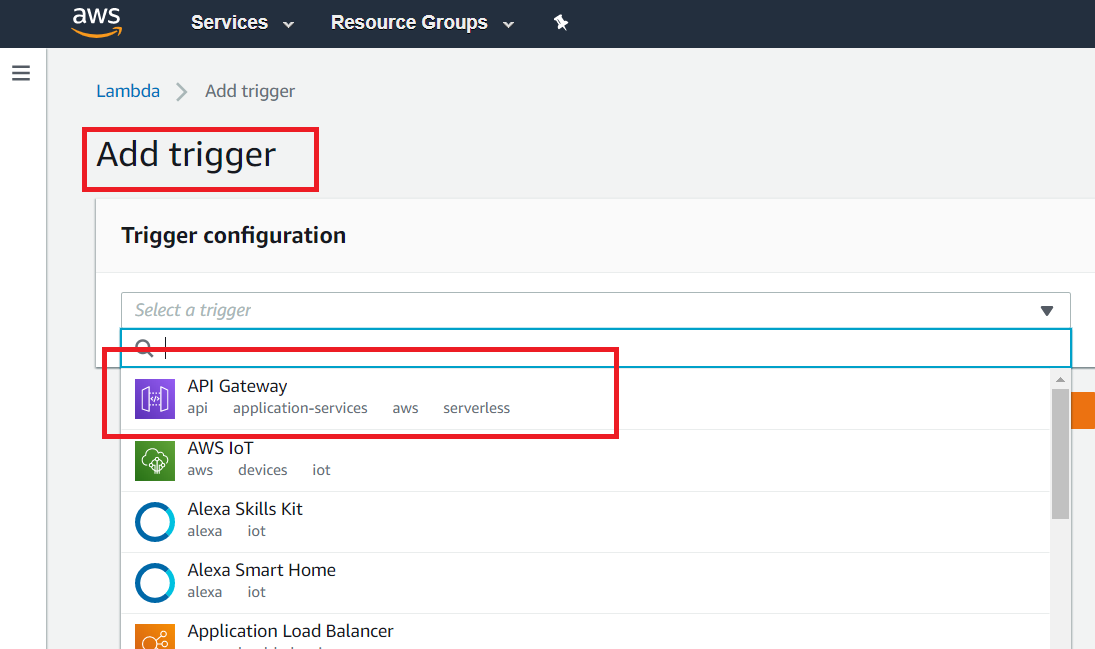


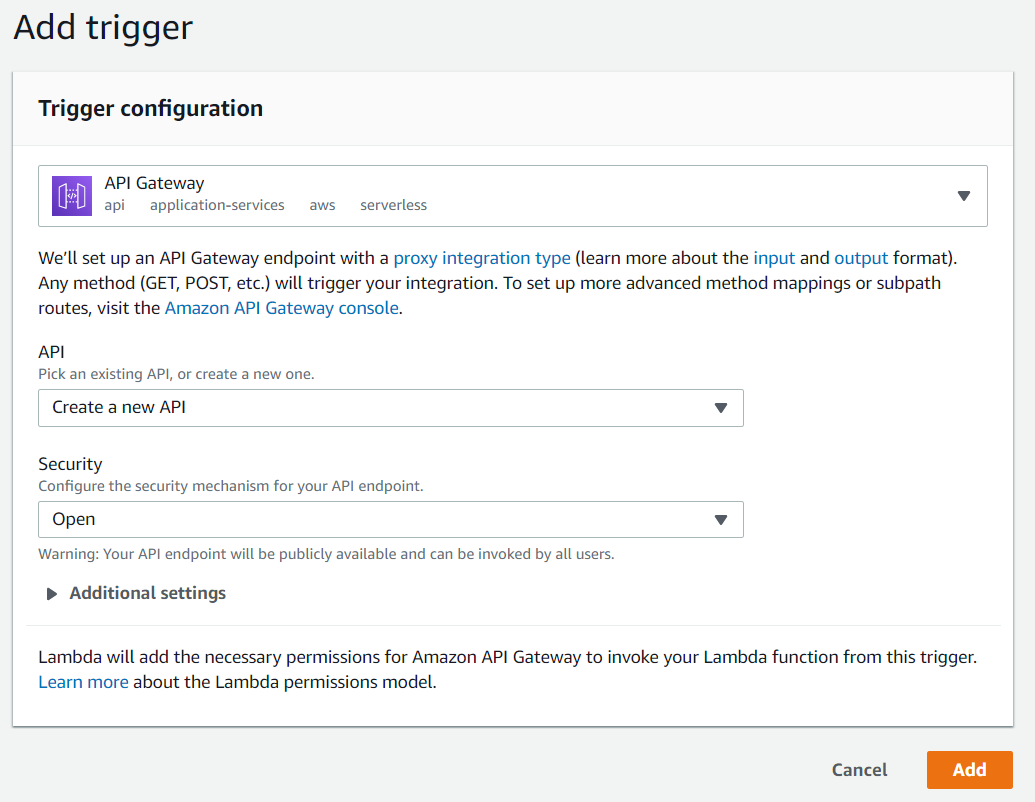
**AWS API GATEWAY**

In the Lambda function, click on Add trigger

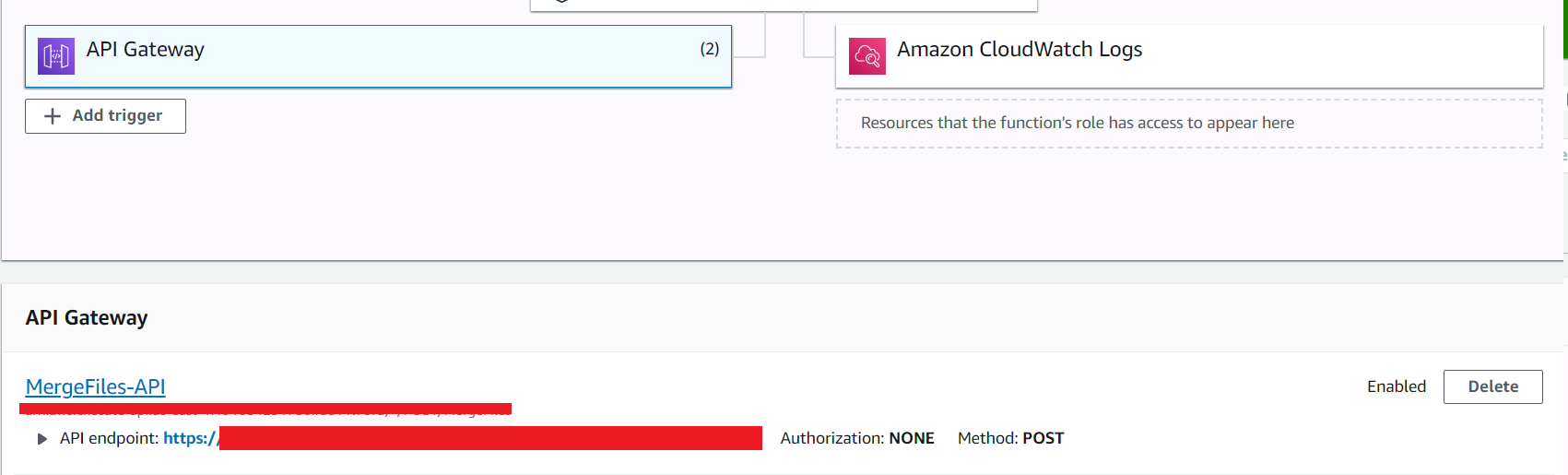
Under trigger select API Gateway



Select the security as required else keep it open

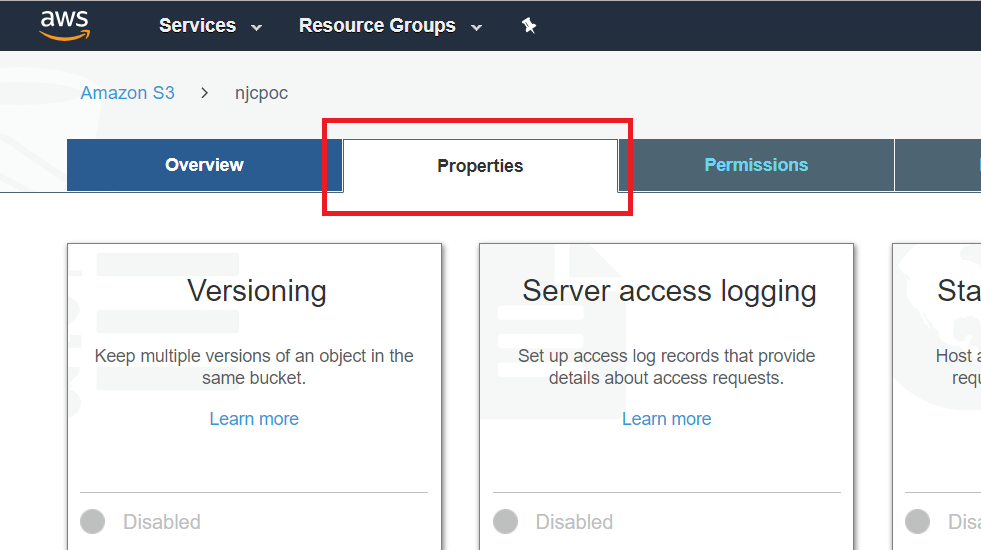


Copy the API endpoint:

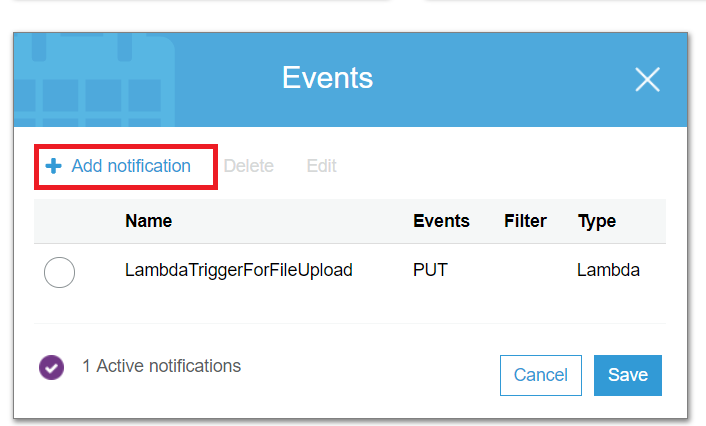


**Adding Lambda trigger on S3 bucket**

On your S3 bucket go to properties



**Click on Events and add trigger to Lambda function on PUT**



**LAMBDA FUNCTION – To add metadata of file to DynamoDB table**

from \_\_future\_\_ import print\_function

import boto3, logging

**// *User Access Key and ID***

**access\_key\_id="\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"**

**secret\_access\_key= "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"**

***// S3 resource using boto3***

**s3 = boto3.client('s3',aws\_access\_key\_id=access\_key\_id,**

**aws\_secret\_access\_key= secret\_access\_key)**

***// DynamoDB resource using boto3***

**dynamodb = boto3.resource('dynamodb',region\_name='us-east-1',aws\_access\_key\_id=access\_key\_id,aws\_secret\_access\_key=secret\_access\_key)**

**table = dynamodb.Table('CMPOC')**

logger = logging.getLogger()

logger.setLevel(logging.INFO)

***// Main Lambda function – The file meta data will be a part of event***

**def lambda\_handler(event, context):**

for record in event['Records']:

bucket = record['s3']['bucket']['name']

key = record['s3']['object']['key']

response = s3.head\_object(Bucket=bucket, Key=key)

// extracting the meta-data of file from event parameter

docket\_venue\_id = response['Metadata']['docket\_venue\_id']

docket\_court\_year= response['Metadata']['docket\_court\_year']

item\_type = response['Metadata']['item\_type']

document\_description = response['Metadata']['document\_description']

document\_privacy\_reason = response['Metadata']['document\_privacy\_reason']

docket\_type\_code = response['Metadata']['docket\_type\_code']

document\_mime\_type = response['Metadata']['document\_mime\_type']

document\_privacy\_code = response['Metadata']['document\_privacy\_code']

efiling\_court\_div = response['Metadata']['efiling\_court\_div']

document\_code = response['Metadata']['document\_code']

docket\_seq\_number = response['Metadata']['docket\_seq\_number']

DocketNumber = docket\_venue\_id+docket\_type\_code+docket\_court\_year+docket\_seq\_number

logger.info('Response: {}'.format(response))

***// Adding the meta data to DyanmoDB table***

table.put\_item(

Item={

'DocketNumber': DocketNumber,

'S3ObjectKey': key,

'S3BucketName': bucket,

'Docket\_Venue\_ID' : docket\_venue\_id,

'Docket\_Court\_Year': docket\_court\_year,

'Item\_Type' : item\_type,

'Document\_Description' : document\_description,

'Document\_Privacy\_Reason' : document\_privacy\_reason,

'Docket\_Type\_Code' : docket\_type\_code,

'Document\_mime\_Type' : document\_mime\_type,

'Document\_Privacy\_Code' : document\_privacy\_code,

'Efiling\_court\_Div' : efiling\_court\_div,

'document\_code' : document\_code,

'docket\_seq\_number' : docket\_seq\_number } )

**Useful resource Links:**

1. Upload-download file from S3 using boto3 <https://qiita.com/hengsokvisal/items/329924dd9e3f65dd48e7>
2. DynamoDB using Boto3

<https://boto3.amazonaws.com/v1/documentation/api/latest/guide/dynamodb.html>

1. Dash installation and examples

<https://dash.plot.ly/installation>