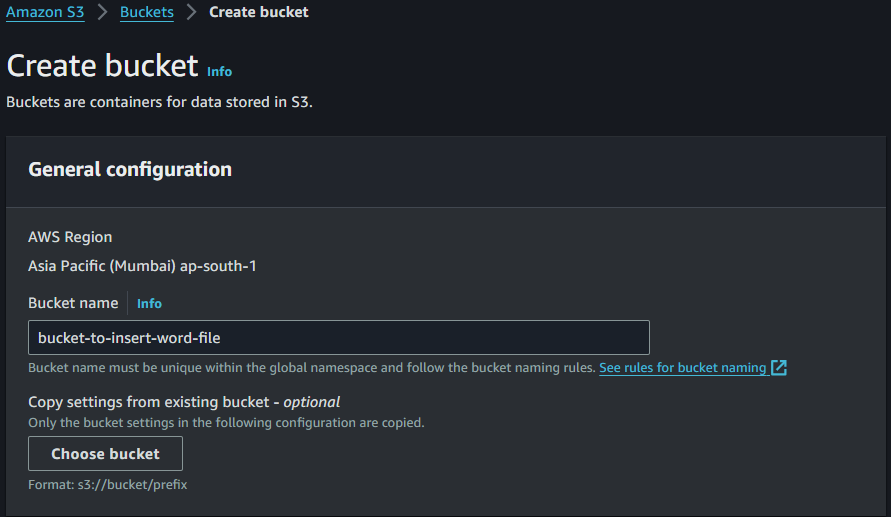
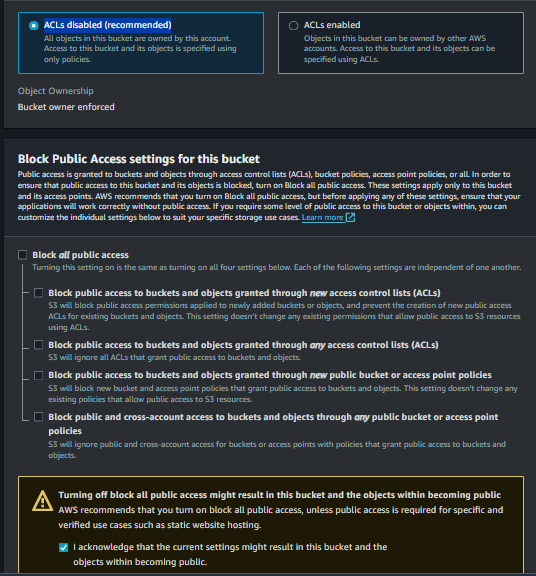
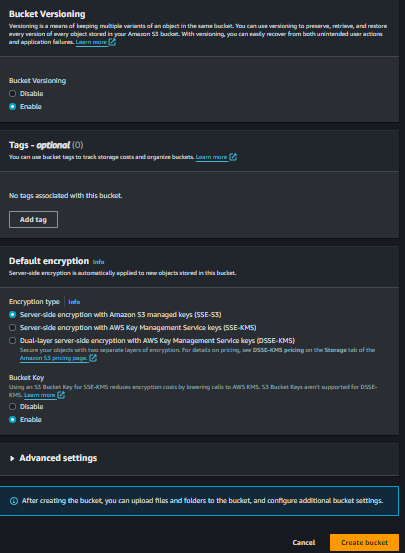
AIM: To convert WORD DOCUMENT into PDF file using AWS services.

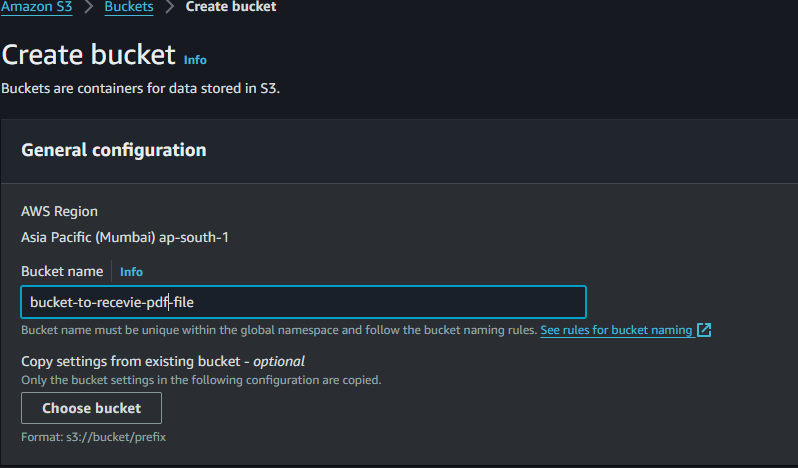
1. Create S3 Buckets:

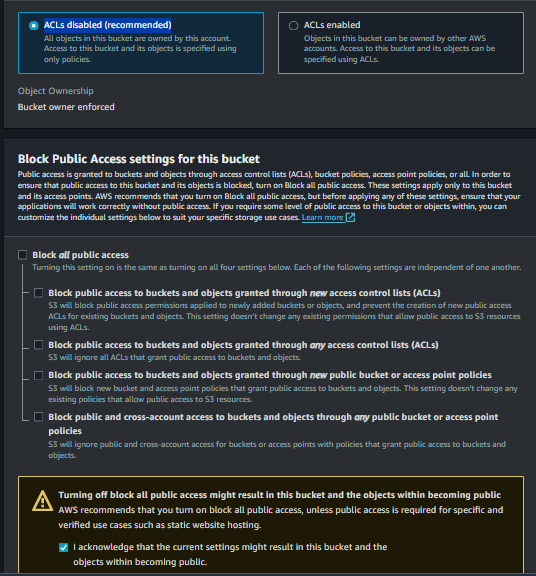
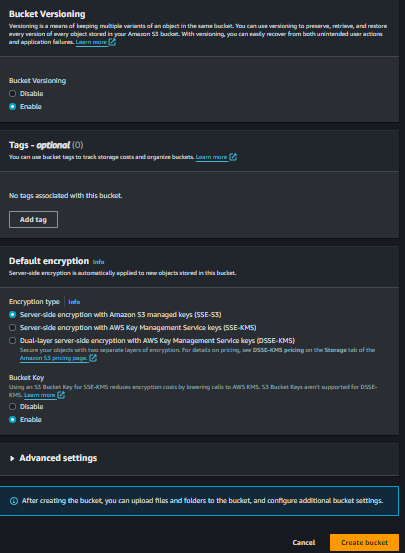
* Log in to your AWS Management Console.
* Navigate to the S3 service.
* Create two buckets: one for uploading DOC files and one for storing converted PDF files.
* BUCKET 1: Create bucket > bucket-to-insert-word-file > Allow all public access > acknowledge it > Bucket versioning (Enable) > create bucket.

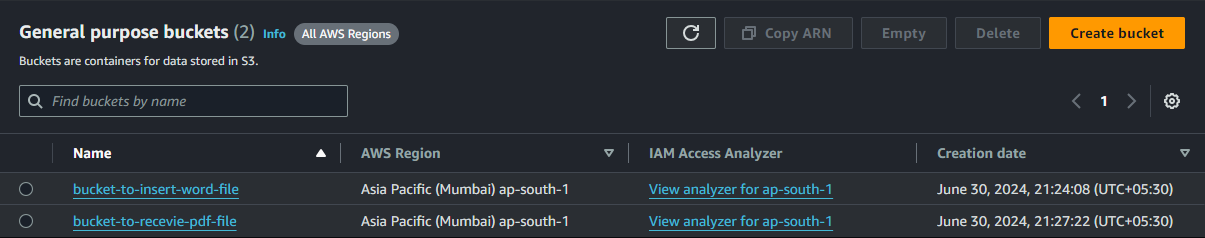


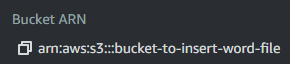
1. BUCKET 2 : Same as step 1 (BUCKET 2 : bucket-to-recevie-pdf-file).



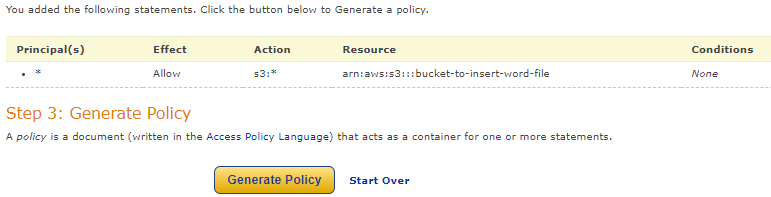
 



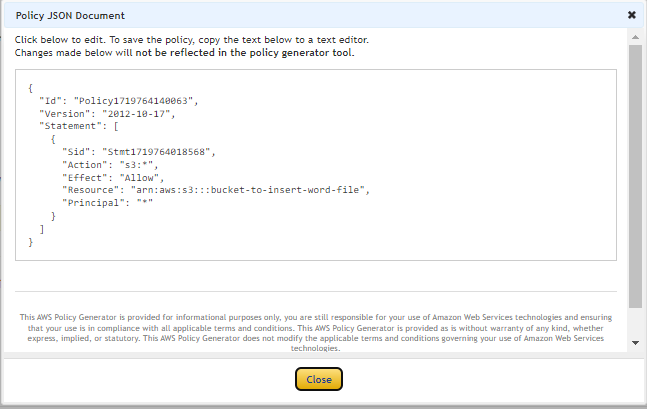
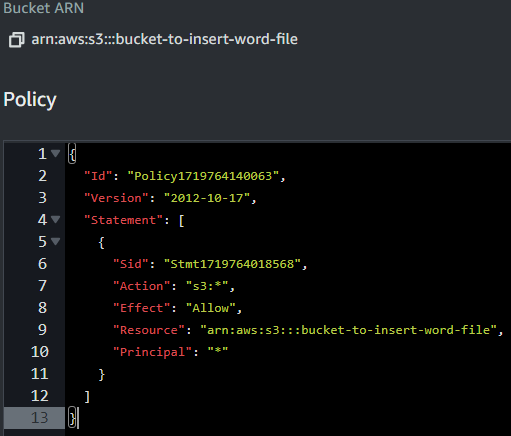
1. GIVE POLICY TO THE BUCKETS :

* BUCKET 1 : Click on bucket (bucket-to-insert-word-file ) > permissions > bucket policy > edit > policy generator > policy type ( S3 bucket policy ) > Add statement > principal (\*) > All Actions > Amazon Resource Name (ARN) >  Bucket ARN copy and paste > Add statement.

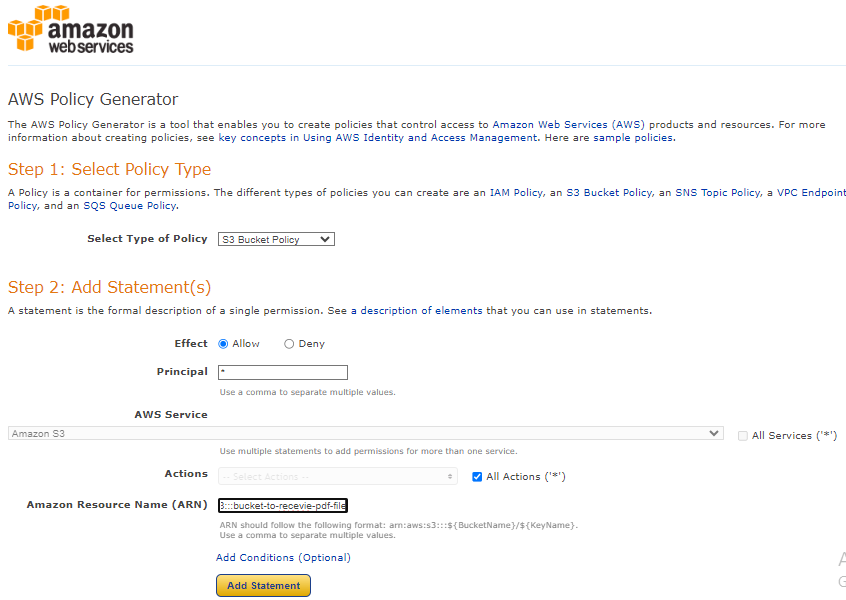


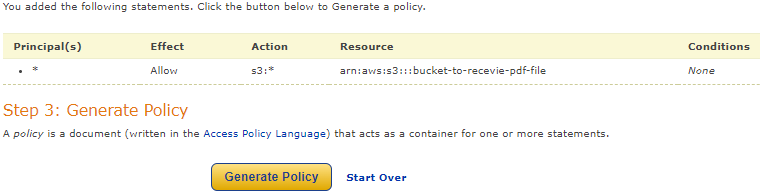


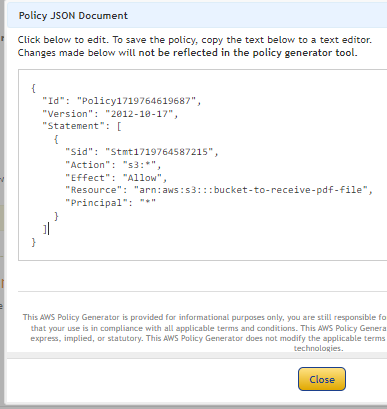
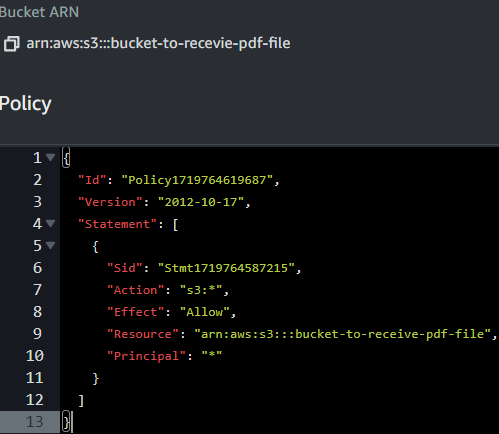
* Click on generate policy > copy the policy and paste it in the bucket policy > save changes.

* Do the same steps as the above for 2nd bucket policy.

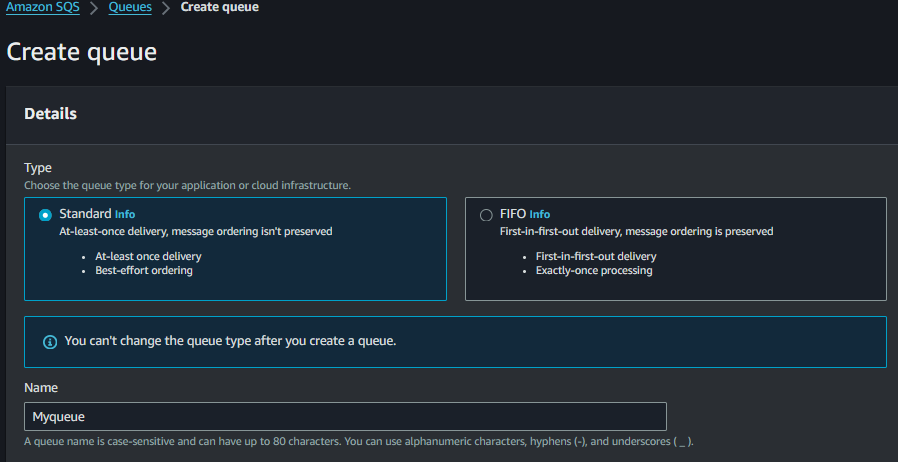
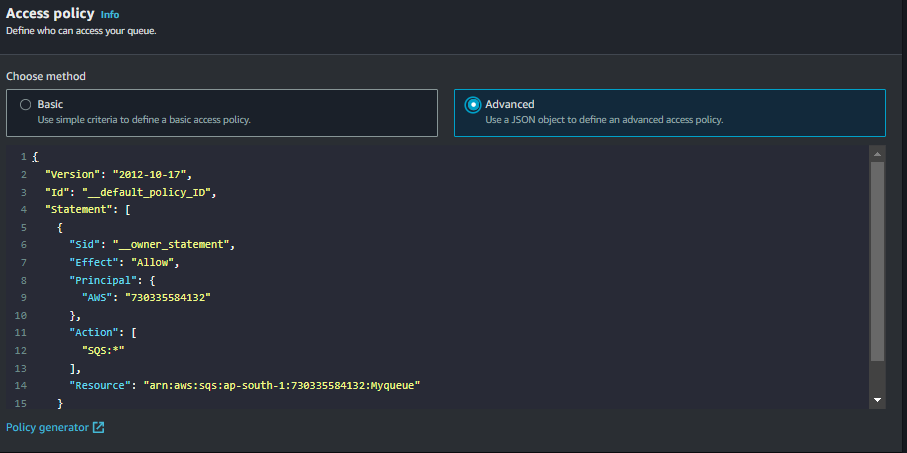




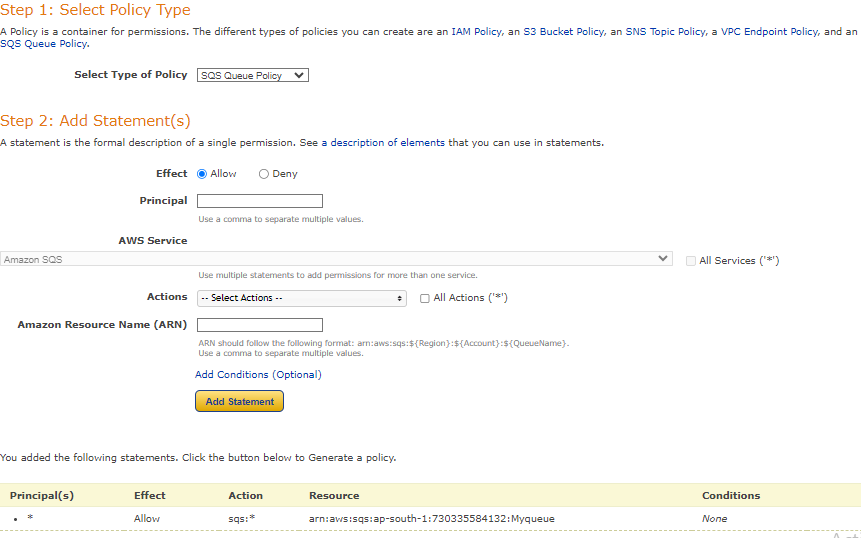
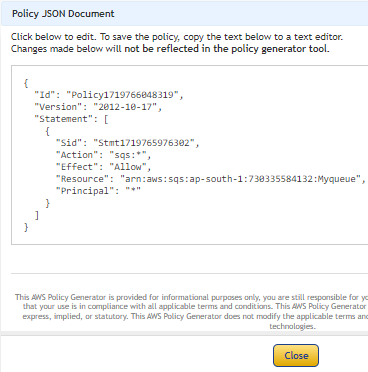
 

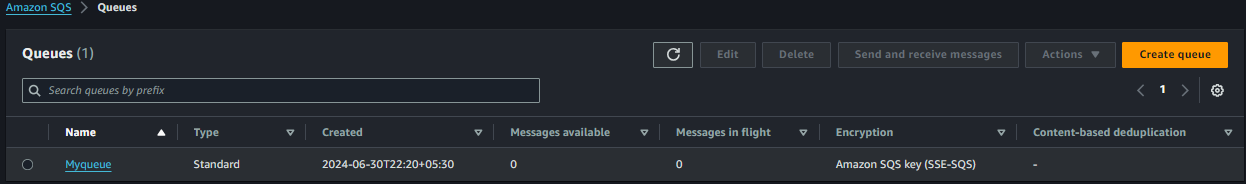
1. Now let’s create SQS queue :

* Duplicate the tab > search for SQS > create queue > type (standard) > name (Myqueue) > Access policy > advanced > copy the resource “arn:aws:sqs:ap-south-1:730335584132:Myqueue” > policy generator .

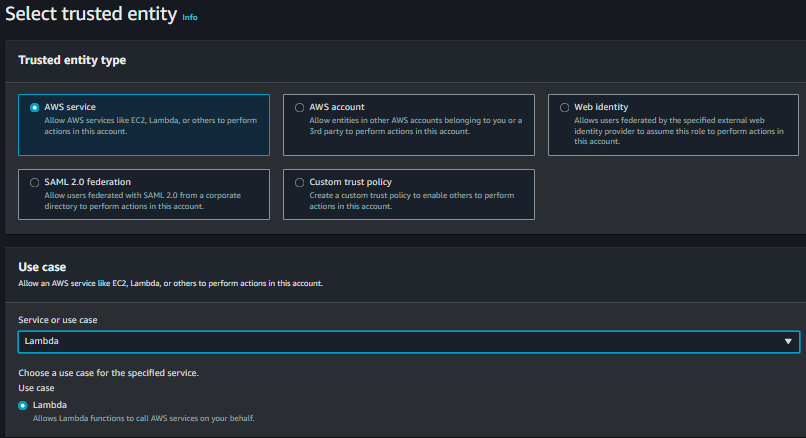
* Policy type ( SQS queue policy ) > Add statements (principal (\*)) > ARN ( paste the arn which we have copied in the above steps > Add statement > generate policy > copy the policy and paste it in the advanced JSON section > create queue.



1. Now let’s create ROLE :

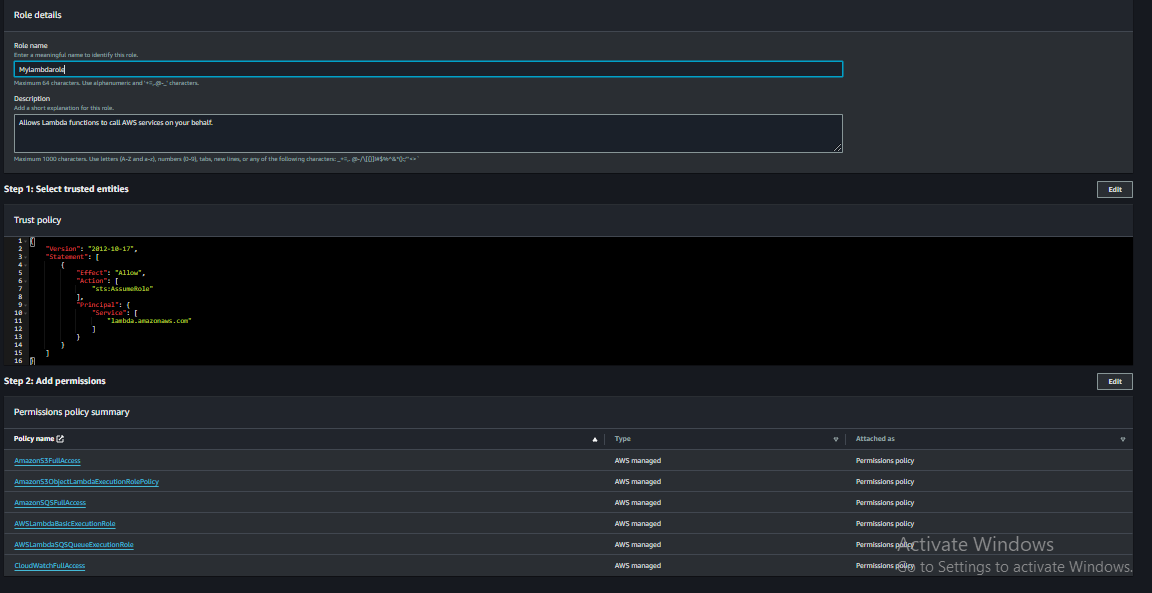
* Duplicate the tab > search IAM > Roles > create role > trusted entity type (AWS services) > use case >lambda.

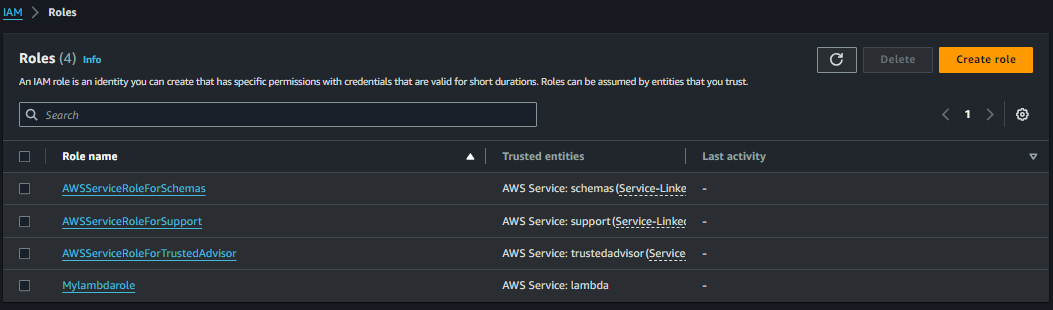


* Now let’s add permissions to the role :

1. Full access :- (S3, SQS, cloud watch).
2. AmazonS3ObjectLambdaExecutionRolePolicy.
3. AWSLambdaBasicExecutionRole.
4. AWSLambdaSQSQueueExecutionRole.

* Next > name it (Mylambdarole) > create role.





* Click on Mylambdarole > Add permissions > create inline policy > specify permissions > JSON > in policy editor paste the following code :

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Action": [

"s3:GetObject",

"s3:PutObject"

],

"Resource": [

" arn:aws:s3:::bucket-to-insert-word-file",

" arn:aws:s3:::bucket-to-recevie-pdf-file "

]

},

{

"Effect": "Allow",

"Action": [

"sqs:ReceiveMessage",

"sqs:DeleteMessage"

],

"Resource": " arn:aws:sqs:ap-south-1:730335584132:Myqueue "

}

]

}

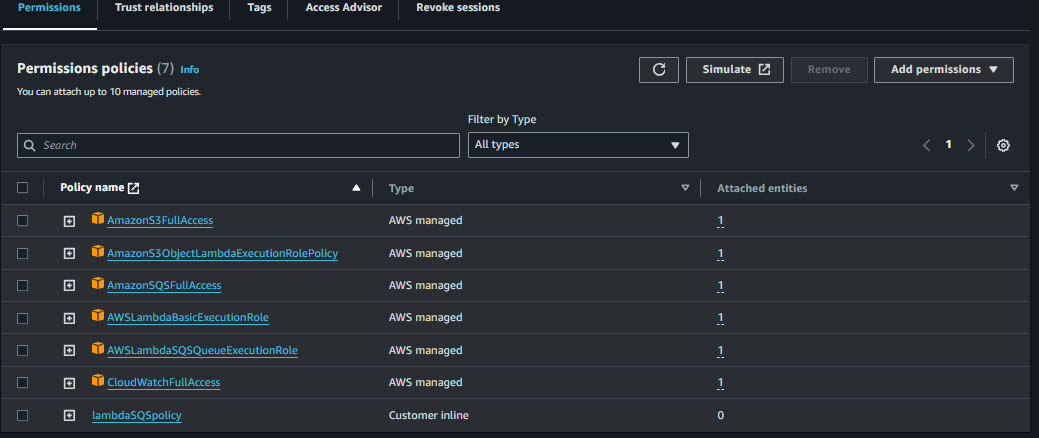
* In the above code make sure you put your both buckets resource ARN

1. “arn:aws:s3:::bucket-to-insert-word-file”
2. “arn:aws:s3:::bucket-to-recevie-pdf-file”

* Also make sure to change the SQS queue ARN >give your SQS queue ARN

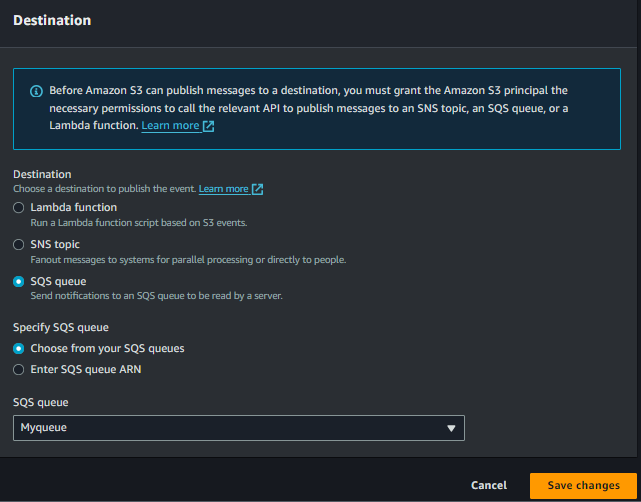
1. arn:aws:sqs:ap-south-1:730335584132:Myqueue.

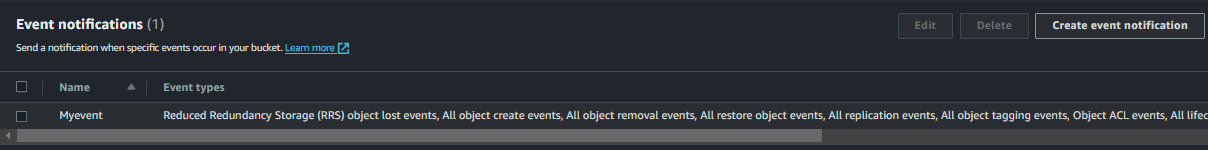
* Next > policy name > lambdaSQSpolicy > create policy > our policy has been attached to our role.



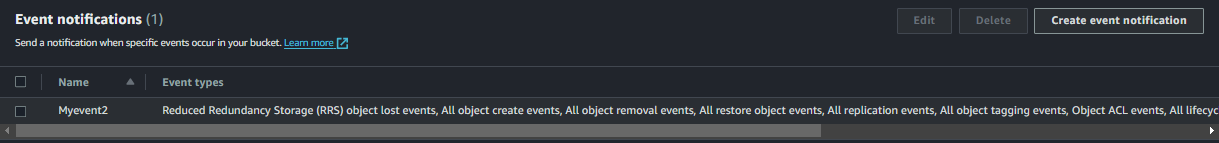
1. Now let’s create Event notification for our S3 buckets.

* Go to S3 bucket > click on > bucket-to-insert-word-file > properties > Event notification > create event notification > name it > (Myevent) > event types > All access > Destination SQS queue > specify SQS queue > Myqueue > save.



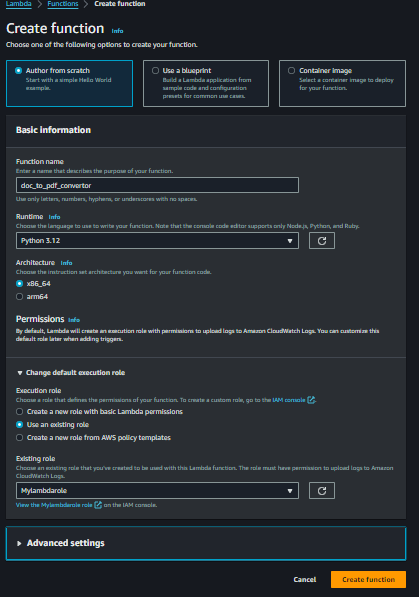


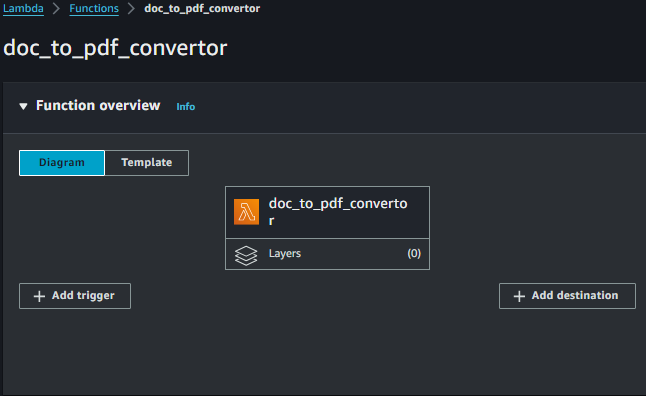
* Go to S3 bucket > click on > bucket-to-recevie-pdf-file > Same as above.



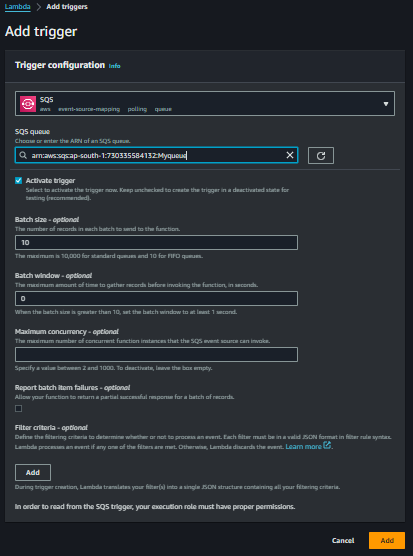
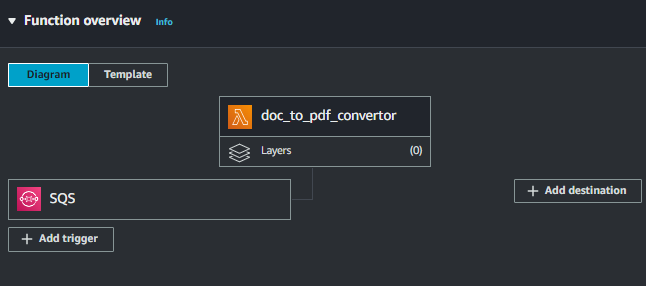
1. Now let’s create Lambda function:

* Duplicate the tab > search lambda > create function > function name > doc\_to\_pdf\_convertor > Runtime (python 3.12) > change default execution role > use an existing role > Mylambdarole > create function.





1. Now let’s add trigger > add trigger > select source > SQS > Myqueue > Add.

* Code > paste the following code there:

import boto3

import os

import uuid

import json

from botocore.exceptions import NoCredentialsError, PartialCredentialsError

s3 = boto3.client('s3')

sqs = boto3.client('sqs')

ORIGINAL\_BUCKET = ' bucket-to-insert-word-file'

CONVERTED\_BUCKET = ' bucket-to-recevie-pdf-file'

QUEUE\_URL = ' https://sqs.ap-south1.amazonaws.com/730335584132/Myqueue'

def lambda\_handler(event, context):

for record in event['Records']:

receipt\_handle = record.get('receiptHandle')

s3\_object\_key = None # Initialize the variable outside the try-except block

try:

# Debug log for incoming event record

print(f"Processing record: {record}")

# Parse the SQS message body

body = json.loads(record.get('body', '{}'))

s3\_info = body['Records'][0]['s3']

s3\_object\_key = s3\_info['object']['key']

if not s3\_object\_key:

raise ValueError("S3 object key not found in the event record")

download\_path = f'/tmp/{uuid.uuid4()}\_{os.path.basename(s3\_object\_key)}'

s3.download\_file(ORIGINAL\_BUCKET, s3\_object\_key, download\_path)

print(f"Downloaded {s3\_object\_key} to {download\_path}")

# Perform the document conversion (example: converting .docx to .pdf)

converted\_path = convert\_document(download\_path)

print(f"Converted document: {download\_path} to {converted\_path}")

# Upload the converted file back to S3

converted\_key = f'converted/{os.path.basename(converted\_path)}'

s3.upload\_file(converted\_path, CONVERTED\_BUCKET, converted\_key)

print(f"Uploaded converted file to {CONVERTED\_BUCKET}/{converted\_key}")

# Delete the message from the queue

if receipt\_handle:

sqs.delete\_message(QueueUrl=QUEUE\_URL, ReceiptHandle=receipt\_handle)

print(f"Deleted message from queue: {QUEUE\_URL}")

except NoCredentialsError:

print("Error: Credentials not available")

except PartialCredentialsError:

print("Error: Incomplete credentials")

except Exception as e:

print(f"Error processing {s3\_object\_key if s3\_object\_key else 'unknown'}: {str(e)}")

def convert\_document(input\_path):

# Example conversion logic

output\_path = input\_path.replace('.docx', '.pdf')

# Use a library like python-docx or other to perform actual conversion

# Here we simply rename the file for demonstration

os.rename(input\_path, output\_path)

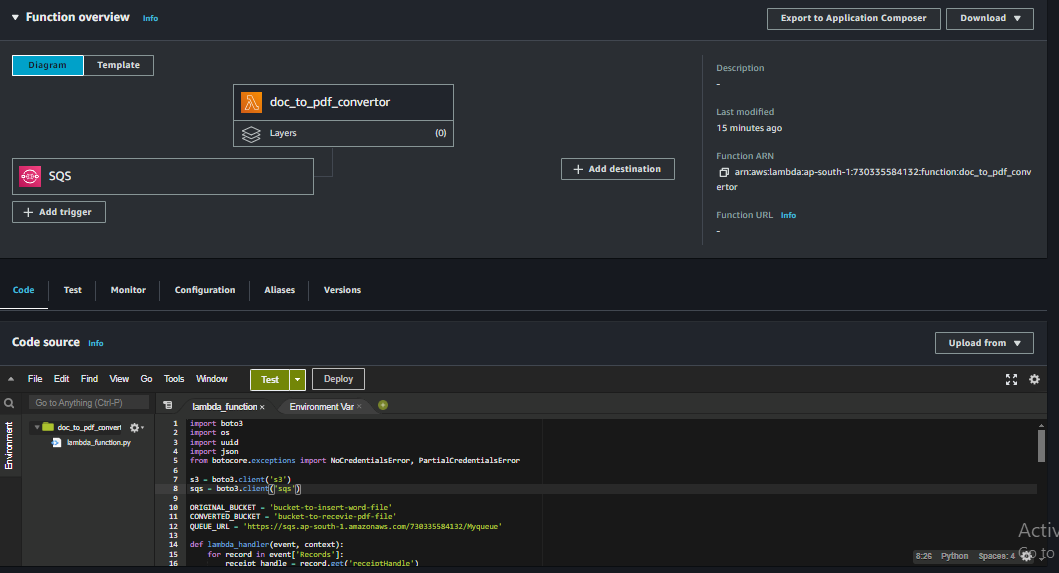
return output\_path.

* Make sure in the above code in the ORIGINAL\_BUCKET paste your buckets name and your Queue URL. As below.

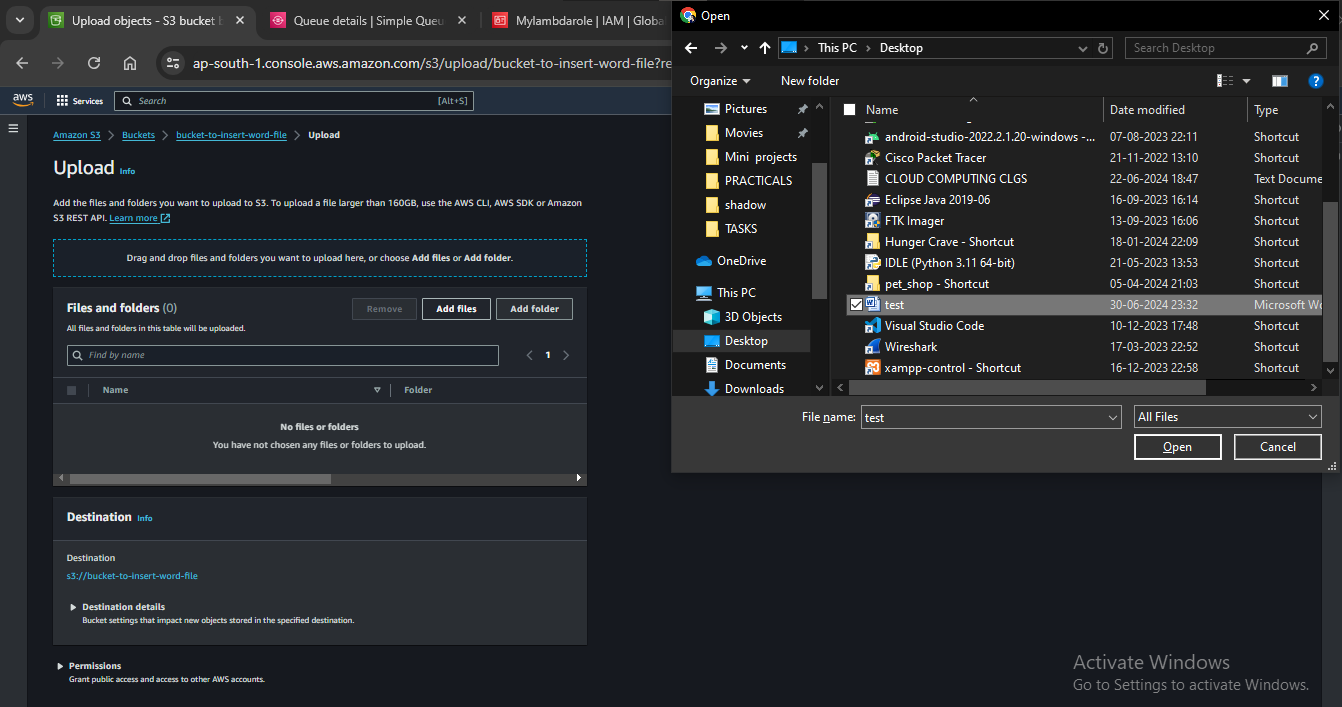
ORIGINAL\_BUCKET = ' bucket-to-insert-word-file'

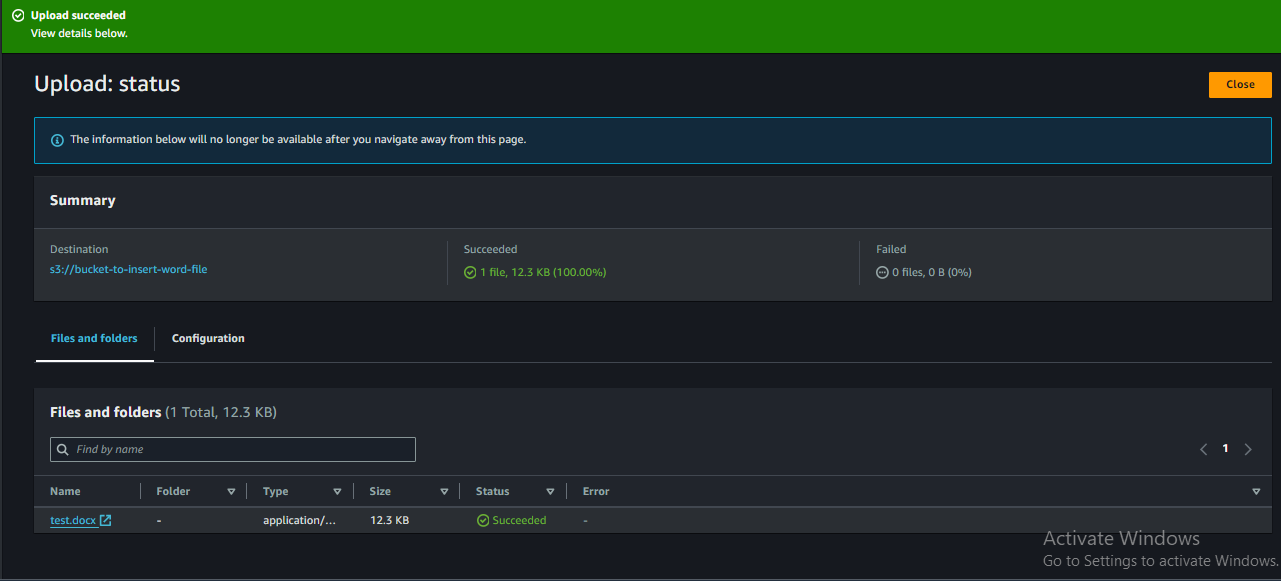
CONVERTED\_BUCKET = ' bucket-to-recevie-pdf-file'

QUEUE\_URL = ‘https://sqs.apsouth1.amazonaws.com/730335584132/Myqueue'



1. Now our services are ready to execute lets test them.
2. Now let’s upload a file in our bucket.





1. Now we can see in our receive bucket we have got our WORD file successfully converted into PDF file.

