**Task 1: How to move file from one s3 bucket to another s3 bucket using aws lambda.**

**AWS Account:** You need an AWS account to access AWS services.

**IAM Role:** Create an IAM role for your Lambda function with permissions to read from the source S3 bucket and write to the destination S3 bucket.

**To create IAM Role:**

**Navigate to the IAM Console:**

Sign in to the AWS Management Console and open the IAM console.

**Create a New Role:**

In the left navigation pane, click on "Roles", then click "Create role".

**Choose Lambda Service:**

Under "Select type of trusted entity", choose "AWS service", then select "Lambda" from the list of services.

**Attach Permissions Policies:**

Click "Next: Permissions".

* Here, you can attach policies that grant the necessary permissions to your Lambda function.
* To allow read access to the source S3 bucket, attach a policy like AmazonS3ReadOnlyAccess.
* To allow write access to the destination S3 bucket, you can create a custom policy or use AmazonS3FullAccess (Note: This policy grants full access to all S3 resources, so use it with caution).
* To create a custom policy, click on "Create policy", then choose the S3 service and select the permissions required (e.g., s3:GetObject for read access, s3:PutObject for write access). After defining the policy, attach it to the role.

**Review and Name the Role:**

Click "Next: Tags" to add any tags if necessary. Tags are optional metadata that you can assign to AWS resources.

* Click "Next: Review".
* Enter a name for your role (e.g., "LambdaS3AccessRole").
* Optionally, add a description to help identify the role's purpose.
* Click "Create role".

**Update Lambda Function with the IAM Role:**

Now that the IAM role is created, go back to the Lambda function configuration.

Under "Permissions", click on the existing role.

Select the IAM role you just created from the list.

Save the changes.

With this IAM role attached to your Lambda function, it will have the necessary permissions to read from the source S3 bucket and write to the destination S3 bucket.

**Steps:**

1. **Create Lambda Function**:
   * Go to the AWS Lambda console.
   * Click on "Create function" and choose "Author from scratch".
   * Enter a name for your function, choose the runtime (Python, Node.js, etc.), and select the IAM role you created earlier.
   * Click "Create function".
2. **Write Lambda Code**:
   * Inside the Lambda function code editor, you can use any supported language to write code.
   * Here's a Python code:

**import boto3 # official aws sdk library for pythonn**

**s3 = boto3.client('s3') # here the client is s3**

**def lambda\_handler(event, context): # two parameters, event: it contains the event that triggered #lambda function , context: runtime information**

**source\_bucket = event['Records'][0]['s3']['bucket']['name']**

**key = event['Records'][0]['s3']['object']['key']**

**destination\_bucket = "shyam-dest-bucket"**

**# Copy the object to the destination bucket**

**s3.copy\_object(CopySource={'Bucket': source\_bucket, 'Key': key},**

**Bucket=destination\_bucket,**

**Key=key)**

**return {**

**'statusCode': 200,**

**'body': 'File moved successfully!'**

**}**

This code retrieves the source bucket name and object key from the event triggered by an S3 PUT event, then copies the object to the destination bucket.

1. **Set Lambda Trigger**:
   * Go to the "Designer" tab of your Lambda function.
   * Click on "Add trigger" and select "S3".
   * Choose the source bucket as the trigger bucket and select the event type (e.g., "All object create events").
   * Click "Add".
2. **Configure Destination Bucket**:
   * Make sure the destination bucket exists and you have the necessary permissions to write to it.
   * If the destination bucket is in a different AWS account, ensure that appropriate permissions are set up for cross-account access.
3. **Testing**:
   * Upload a file to the source bucket.
   * Monitor the Lambda function execution in the AWS Lambda console. You should see logs indicating successful execution.
   * The testing code to run the event in json format:

**{**

**"Records": [**

**{**

**"eventVersion": "2.1",**

**"eventSource": "aws:s3",**

**"awsRegion": "us-east-1",**

**"eventName": "ObjectCreated:Put",**

**"s3": {**

**"bucket": {**

**"name": "shyam-source-bucket"**

**},**

**"object": {**

**"key": "beach.jpg"**

**}**

**}**

**}**

**]**

**}**

This setup allows you to automatically transfer files between S3 buckets using AWS Lambda.