**Lambda Function**

import json

import boto3

import csv

def lambda\_handler(event, context):

**# Retrieve bucket and key from the event**

bucket = event['Records'][0]['s3']['bucket']['name']

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

**# Download CSV file from S3**

s3 = boto3.client('s3')

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

lines = response['Body'].read().decode('utf-8').split('\n')

**# Parse CSV file and generate recommendations**

recommendations = []

migration\_strategies = {'lift-and-shift': [], 're-platforming': [], 're-architecting': []}

csv\_data = csv.DictReader(lines)

for row in csv\_data:

**# Analyze server configurations and generate recommendations**

recommendation, migration\_strategy = generate\_recommendation(row)

recommendations.append(recommendation)

if migration\_strategy:

migration\_strategies[migration\_strategy].append(row['ExternalId'])

**# Return recommendations and migration strategies**

return {

'statusCode': 200,

'body': {

'recommendations': recommendations,

'migration\_strategies': migration\_strategies

}

}

def generate\_recommendation(server\_config):

**# Analyze server configurations and generate recommendations based on different attributes**

recommendation = "Migration Plan Recommendation for server with ExternalId {}: \n".format(server\_config['ExternalId'])

migration\_strategy = None

**# Example recommendation logic based on CPU utilization**

if float(server\_config['CPU.UsagePct.Avg']) > 80:

migration\_strategy = 're-platforming'

hint = "High CPU utilization may require re-platforming"

recommendation += "- Analyze current workload and performance requirements.\n"

recommendation += "- Identify EC2 instances with higher CPU capacity.\n"

recommendation += "- Perform load testing to ensure new instance meets performance needs.\n"

recommendation += "- Plan for downtime or rolling updates during migration. (Hint: {})\n".format(hint)

**# Example recommendation logic based on RAM utilization**

if float(server\_config['RAM.UsedSizeInMB.Avg']) / float(server\_config['RAM.TotalSizeInMB']) > 0.75:

migration\_strategy = 're-platforming'

hint = "High RAM utilization may require re-platforming"

recommendation += "- Assess memory requirements of current applications.\n"

recommendation += "- Choose EC2 instance types with more memory.\n"

recommendation += "- Optimize memory usage to minimize costs.\n"

recommendation += "- {}.\n".format(hint)

**# Example recommendation logic based on disk I/O**

if float(server\_config['DiskReadsPerSecondInKB.Avg']) > 1000 or float(server\_config['DiskWritesPerSecondInKB.Avg']) > 1000:

migration\_strategy = 're-architecting'

hint = "High disk I/O may require re-architecting"

recommendation += "- Evaluate disk I/O patterns and requirements.\n"

recommendation += "- Select EC2 instances with higher I/O performance or provisioned IOPS.\n"

recommendation += "- Consider optimizing disk usage or implementing caching mechanisms.\n"

recommendation += "- {}.\n".format(hint)

**# Example recommendation logic based on network throughput**

if float(server\_config['NetworkReadsPerSecondInKB.Avg']) > 1000 or float(server\_config['NetworkWritesPerSecondInKB.Avg']) > 1000:

migration\_strategy = 're-platforming'

hint = "High network throughput may require re-platforming"

recommendation += "- Analyze network traffic patterns and bandwidth requirements.\n"

recommendation += "- Choose EC2 instances with higher network performance.\n"

recommendation += "- Implement content delivery networks (CDNs) for static assets.\n"

recommendation += "- {}.\n".format(hint)

**# Example recommendation logic based on operating system**

if server\_config['OS.Name'] == 'Windows' and server\_config['OS.Version'] == '2012':

migration\_strategy = 're-platforming'

hint = "Windows Server 2012 may require re-platforming"

recommendation += "- Check application compatibility with newer Windows Server versions.\n"

recommendation += "- Plan for OS upgrade and compatibility testing.\n"

recommendation += "- {}.\n".format(hint)

**# Example recommendation logic based on application dependencies**

if server\_config['Applications'] != '':

migration\_strategy = 're-platforming'

hint = "Applications may require re-platforming"

recommendation += "- Create an inventory of installed applications and dependencies.\n"

recommendation += "- Validate compatibility of applications with target environment.\n"

recommendation += "- Plan for application migration or reconfiguration.\n"

recommendation += "- {}.\n".format(hint)

**# Example recommendation logic based on security requirements**

if 'PCI' in server\_config['Tags']:

recommendation += "- Ensure compliance with PCI DSS requirements during migration.\n"

**# Example recommendation logic based on cost optimization**

if float(server\_config['CPU.UsagePct.Avg']) < 10 and float(server\_config['RAM.UsedSizeInMB.Avg']) / float(server\_config['RAM.TotalSizeInMB']) < 0.25:

migration\_strategy = 'lift-and-shift'

hint = "Underutilized servers may be candidates for lift-and-shift"

recommendation += "- Rightsize or consolidate resources for cost optimization.\n"

recommendation += "- {}.\n".format(hint)

return recommendation, migration\_strategy

**Execution Result**

Test Event Name

test1

Response

{

"statusCode": 200,

"body": {

"recommendations": [

"Migration Plan Recommendation for server with ExternalId SampleInventoryId1: \n- Create an inventory of installed applications and dependencies.\n- Validate compatibility of applications with target environment.\n- Plan for application migration or reconfiguration.\n- Applications may require re-platforming.\n"

],

"migration\_strategies": {

"lift-and-shift": [],

"re-platforming": [

"SampleInventoryId1"

],

"re-architecting": []

}

}

}

Function Logs

START RequestId: 4b122042-8689-46ce-af0c-038cadf9b3e2 Version: $LATEST

END RequestId: 4b122042-8689-46ce-af0c-038cadf9b3e2

REPORT RequestId: 4b122042-8689-46ce-af0c-038cadf9b3e2 Duration: 2876.22 ms Billed Duration: 2877 ms Memory Size: 128 MB Max Memory Used: 83 MB Init Duration: 292.79 ms

Request ID

4b122042-8689-46ce-af0c-038cadf9b3e2