**Dyanmo Code :**

import json

import boto3

import random

dynamodb = boto3.resource('dynamodb')

table\_name = 'tester'

accounts\_table\_name = 'Accounts'

table = dynamodb.Table(table\_name)

accounts\_table = dynamodb.Table(accounts\_table\_name)

def lambda\_handler(event, context):

StatusCode = int(event['queryStringParameters']['StatusCode'])

if StatusCode == 100:

uid = event['queryStringParameters']['uid']

response = table.get\_item(Key={'uid': uid})

item = response.get('Item', {})

name = item.get('name')

password = item.get('password')

isvalid = item.get('isvalid')

transactionResponse = {

'StatusCode': StatusCode,

'uid': uid,

'name': name,

'password': password,

'isvalid': isvalid

}

elif StatusCode == 101:

uid = event['queryStringParameters']['uid']

name = event['queryStringParameters']['name']

password = event['queryStringParameters']['password']

isvalid = 'yes'

item = {

'uid': uid,

'isvalid': isvalid,

'name': name,

'password': password

}

try:

table.put\_item(Item=item)

request = 'success'

except:

request = 'failure'

transactionResponse = {

'StatusCode': StatusCode,

'request': request,

'uid': uid,

'password': password

}

elif StatusCode == 111:

uid = event['queryStringParameters']['uid']

source\_acc = event['queryStringParameters']['source\_acc']

destination\_acc = event['queryStringParameters']['destination\_acc']

amount = int(event['queryStringParameters']['amount'])

atmno = event['queryStringParameters']['atmno']

response = accounts\_table.get\_item(Key={'Acc\_no': source\_acc})

source\_account = response.get('Item', {})

fetched\_atmno = source\_account.get('atmno')

fetched\_balance = source\_account.get('balance', 0)

response = accounts\_table.get\_item(Key={'Acc\_no': destination\_acc})

destination\_account = response.get('Item', {})

if atmno == fetched\_atmno:

if amount <= fetched\_balance:

fetched\_balance -= amount

fetched\_balance\_dest = destination\_account.get('balance', 0) + amount

accounts\_table.update\_item(

Key={'Acc\_no': source\_acc},

UpdateExpression='SET balance = :value',

ExpressionAttributeValues={

':value': fetched\_balance

}

)

accounts\_table.update\_item(

Key={'Acc\_no': destination\_acc},

UpdateExpression='SET balance = :value',

ExpressionAttributeValues={

':value': fetched\_balance\_dest

}

)

transactionResponse = {

'StatusCode': StatusCode,

'request': 'successful',

'uid': uid,

'trans\_status': 'success'

}

else:

transactionResponse = {

'StatusCode': StatusCode,

'request': 'failed',

'uid': uid,

'trans\_status': 'insufficient\_balance'

}

else:

transactionResponse = {

'StatusCode': StatusCode,

'request': "no",

'uid': uid,

'trans\_status': 'invalid\_atmno'

}

elif StatusCode == 136:

uid = event['queryStringParameters']['uid']

bank = event['queryStringParameters']['bank']

branch = event['queryStringParameters']['branch']

accno = event['queryStringParameters']['accno']

ifsc = event['queryStringParameters']['ifsc']

phno = event['queryStringParameters']['phno']

atmno = event['queryStringParameters']['atmno']

random\_number = random.randint(10000, 50000)

item = {

'uid': uid,

'bank': bank,

'branch': branch,

'Acc\_no': accno,

'ifsc': ifsc,

'phno': phno,

'atmno': atmno,

'balance': random\_number

}

try:

accounts\_table.put\_item(Item=item)

request = 'success'

except:

request = 'failure'

transactionResponse = {

'StatusCode': StatusCode,

'request': request,

'uid': uid

}

elif(StatusCode == 102):

'''

StatusCode = 102

queryStringParameters = {

status code = 102

uid = 'abc@abc.com'

accno ="asdb234"

atmno = 8456

}

responce params = {

status code = 102

request status = 'successfull'

uid = 'abc@abc.com'

invoke url : https://e5hbne06n1.execute-api.ap-south-1.amazonaws.com/test/crud?StatusCode=102&accno=526698

}

'''

# step 1 : parse the query string params

accno = event['queryStringParameters']['accno']

# step 2 : Contrruct the body of the response object

transactionResponse = {} #this is an empty object in which we fill the values for the json response object

#fill the values

response = accounts\_table.get\_item(Key={'Acc\_no': accno})

item = response.get('Item', {})

fetched\_atmno = item.get('atmno')

fetched\_uid = item.get('uid')

transactionResponse['StatusCode'] = StatusCode

transactionResponse['uid'] = fetched\_uid

transactionResponse['atmno'] = fetched\_atmno

transactionResponse['accno'] = accno

elif(StatusCode == 103):

'''

StatusCode = 103

queryStringParameters = {

status code = 103

accno ="asdb234"

}

responce params = {

status code = 103

transactionResponse['StatusCode'] = StatusCode

transactionResponse['uid'] = fetched\_uid

transactionResponse['atmno'] = fetched\_atmno

transactionResponse['accno'] = accno

transactionResponse['bank'] = fetched\_bank

transactionResponse['branch'] = fetched\_branch

transactionResponse['ifsc'] = fetched\_ifsc

transactionResponse['phno'] = fetched\_phno

invoke url : https://e5hbne06n1.execute-api.ap-south-1.amazonaws.com/test/crud?StatusCode=102&accno=526698

}

'''

# step 1 : parse the query string params

accno = event['queryStringParameters']['accno']

# step 2 : Contrruct the body of the response object

transactionResponse = {} #this is an empty object in which we fill the values for the json response object

#fill the values

response = accounts\_table.get\_item(Key={'Acc\_no': accno})

item = response.get('Item', {})

fetched\_atmno = item.get('atmno')

fetched\_uid = item.get('uid')

fetched\_bank = item.get('bank')

fetched\_branch = item.get('branch')

fetched\_ifsc = item.get('ifsc')

fetched\_phno = item.get('phno')

transactionResponse['StatusCode'] = StatusCode

transactionResponse['uid'] = fetched\_uid

transactionResponse['atmno'] = fetched\_atmno

transactionResponse['accno'] = accno

transactionResponse['bank'] = fetched\_bank

transactionResponse['branch'] = fetched\_branch

transactionResponse['ifsc'] = fetched\_ifsc

transactionResponse['phno'] = fetched\_phno

elif(StatusCode == 118):

'''

StatusCode = 118

queryStringParameters = {

status code = 118

accno ="asdb234"

p

}

responce params = {

status code = 118

request status = 'successfull'

uid = 'abc@abc.com'

invoke url : https://e5hbne06n1.execute-api.ap-south-1.amazonaws.com/test/crud?StatusCode=102&accno=526698

}

'''

# step 1 : parse the query string params

accno = event['queryStringParameters']['accno']

atmno = event['queryStringParameters']['atmno']

# step 2 : Contrruct the body of the response object

transactionResponse = {} #this is an empty object in which we fill the values for the json response object

#fill the values

response = accounts\_table.get\_item(Key={'Acc\_no': accno})

item = response.get('Item', {})

fetched\_atmno = item.get('atmno')

fetched\_uid = item.get('uid')

request = "Deletion Failed"

if(atmno == fetched\_atmno):

accounts\_table.delete\_item(Key={'Acc\_no': accno})

request = 'success'

request = "successful"

transactionResponse['StatusCode'] = StatusCode

transactionResponse['uid'] = fetched\_uid

transactionResponse['atmno'] = fetched\_atmno

transactionResponse['accno'] = request

elif(StatusCode == 160):

'''

StatusCode = 102

queryStringParameters = {

status code = 102

uid = 'abc@abc.com'

accno ="asdb234"

atmno = 8456

}

responce params = {

status code = 102

request status = 'successfull'

uid = 'abc@abc.com'

invoke url : https://e5hbne06n1.execute-api.ap-south-1.amazonaws.com/test/crud?StatusCode=102&accno=526698

}

'''

# step 1: parse the query string params

accno = event['queryStringParameters']['accno']

# step 2: Construct the body of the response object

transactionResponse = {} # empty object to store the response

# Fetch the account balance from the DynamoDB table

response = accounts\_table.get\_item(Key={'Acc\_no': accno})

item = response.get('Item', {})

fetched\_balance = str(item.get('balance'))

fetched\_atmno = item.get('atmno') # Fetch the ATM number

# Fill the values in the response object

transactionResponse['StatusCode'] = StatusCode

transactionResponse['uid'] = fetched\_balance # Store the balance instead of UID

transactionResponse['atmno'] = fetched\_atmno

transactionResponse['accno'] = accno

elif(StatusCode == 165):

'''

StatusCode = 102

queryStringParameters = {

status code = 102

uid = 'abc@abc.com'

accno ="asdb234"

atmno = 8456

}

responce params = {

status code = 102

request status = 'successfull'

uid = 'abc@abc.com'

invoke url : https://e5hbne06n1.execute-api.ap-south-1.amazonaws.com/test/crud?StatusCode=102&accno=526698

}

'''

# step 1: parse the query string params

accno = event['queryStringParameters']['accno']

# step 2: Construct the body of the response object

transactionResponse = {} # empty object to store the response

# Fetch the account balance from the DynamoDB table

response = accounts\_table.get\_item(Key={'Acc\_no': accno})

item = response.get('Item', {})

fetched\_bank = item.get('bank')

fetched\_branch = item.get('branch')

fetched\_ifsc = item.get('ifsc')

fetched\_phone = item.get('phno')

fetched\_atmno = item.get('atmno') # Fetch the ATM number

# Fill the values in the response object

transactionResponse['StatusCode'] = StatusCode

transactionResponse['bank'] = fetched\_bank

transactionResponse['branch'] = fetched\_branch

transactionResponse['ifsc'] = fetched\_ifsc

transactionResponse['phone'] = fetched\_phone

transactionResponse['atmno'] = fetched\_atmno

transactionResponse['accno'] = accno

responseObject = {

'statusCode': 200,

'headers': {

'Content-Type': 'application/json'

},

'body': json.dumps(transactionResponse)

}

return responseObject