**LAMBDA FUNCTION CODE**

**Hello World Function**

import json

def lambda\_handler(event, context):

    # TODO implement

    return {

        'statusCode': 200,

        'body': json.dumps('Hello, World from AWS Lambda!')

    }

**Get Heart Rate**

1. import json
2. import requests
3. def lambda\_handler(event, context):
4. # Parse access token and dates from event
5. \_access\_token = event['access\_token']
6. \_startdate = event['startdate']
7. \_enddate = event['enddate']
8. # Set headers
9. headers = {'Authorization': 'Bearer ' + \_access\_token}
10. # Set API URL for heart rate data
11. FITBIT\_API\_URL = 'https://api.fitbit.com/1/user/-/activities/heart/date/' + \_startdate + '/' + \_enddate + '.json'
12. # Make request [GET] to Fitbit API
13. response = requests.get(FITBIT\_API\_URL, headers=headers)
14. # If the request was successfully executed, return heart rate data
15. if response.status\_code == 200:
16. heart\_rate\_data = response.json()['activities-heart']
17. return {
18. 'statusCode': 200,
19. 'body': json.dumps(heart\_rate\_data)
20. }
21. else:
22. return {
23. 'statusCode': response.status\_code,
24. 'body': 'Could not fetch heart rate data.'
25. }

**3.Get Heart Rate 2**

import json

import requests

def lambda\_handler(event, context):

    # Extract parameters from event

    access\_token = event.get('access\_token')

    start\_date = event.get('start\_date')

    end\_date = event.get('end\_date')

    # Validate inputs

    if not access\_token or not start\_date or not end\_date:

        return {

            'statusCode': 400,

            'body': 'Missing required parameters: access\_token, startdate, or enddate'

        }

    # Set request headers

    headers = {

        'Authorization': f'Bearer {access\_token}',

        'Content-Type': 'application/json'

    }

    # Fitbit API URL for step count

    fitbit\_api\_url = f"https://api.fitbit.com/1/user/-/activities/steps/date/{start\_date}/{end\_date}.json"

    try:

        # Make request to Fitbit API

        response = requests.get(fitbit\_api\_url, headers=headers)

        # Check for success response

        if response.status\_code == 200:

            step\_data = response.json()

            return {

                'statusCode': 200,

                'body': json.dumps(step\_data)

            }

        else:

            return {

                'statusCode': response.status\_code,

                'body': f"Error fetching data: {response.text}"

            }

    except Exception as e:

        return {

            'statusCode': 500,

            'body': f"Internal Server Error: {str(e)}"

        }

**4. fitbit\_auth**

import json

import requests

import sys

from datetime import datetime

import boto3

def lambda\_handler(event, context):

    url="https://api.fitbit.com/oauth2/token"

    # Parse access token from event

    auth\_code = event['authorization\_code']

    headers={

        "Authorization": "Basic MjNQWEo2OjllMTc3NTk3ZTA0NzdhMzBlMGRhNzdmZjIxMGEzM2E0",

        "Content-Type": "application/x-www-form-urlencoded"

    }

    data={

        "grant\_type":"authorization\_code",

        "redirect\_uri":"https://localhost:8080/",

        "code":auth\_code

    }

    response=requests.post(url,headers=headers,data=data)

    # If the request was successfully executed, return heart rate data

    if(response.status\_code == 200):

        ret\_auth = response.json()

        # create dynamodb object instance

        mydb = boto3.resource('dynamodb')

        # new a table object given a table name

        mytable = mydb.Table('fitbitToken')

        # Get current date and time

        now = datetime.now()

        # Format the output of timesamp

        formatted\_date\_time = now.strftime("%Y-%m-%d %H:%M:%S")

        # build data items from response

        item = {

            'access\_token': ret\_auth['access\_token'],

            'refresh\_token': ret\_auth['refresh\_token'],

            'user\_id': ret\_auth['user\_id'],

            'expires\_in': ret\_auth['expires\_in'],

            'token\_type': ret\_auth['token\_type'],

            'time\_stamp': formatted\_date\_time

        }

        # save access token information to database, put new authrization data to Table-fitbitToken

        mytable.put\_item(Item=item)

        # return client with correct code.

        return {

            'statusCode': 200,

            # 'body': json.dumps(ret\_auth)

            'body': f"Successfully get access token for user id:{ret\_auth['user\_id']}."

        }

    else:

        return {

            'statusCode': response.status\_code,

            'body': 'Could not complete request for access token.'

        }

**5. Save Heart rate**

import json

import requests

import boto3

# load access token data from database

def get\_access\_token():

    # create dynamodb client instance

    myclient = boto3.client('dynamodb')

    # get the item from the table

    GetItem = myclient.get\_item(

        TableName='fitbitToken',

        Key={

            'user\_id': {

                'S': 'BN4WML'

            },

        }

    )

    return GetItem['Item']['access\_token']['S']

def save\_heart\_rate\_data(heart\_rate\_json):

    # create dynamodb object instance

    mydb = boto3.resource('dynamodb')

    # new a table object given a table name

    mytable = mydb.Table('HeartRate')

    # put item into the table

    mytable.put\_item(

        Item={

            'collect\_date': heart\_rate\_json['dateTime'],

            'restingHeartRate': heart\_rate\_json['value']['restingHeartRate'],

            'heartRateZones': json.dumps(heart\_rate\_json['value']['heartRateZones'])

        }

    )

def lambda\_handler(event, context):

    # TODO implement

    \_access\_token = get\_access\_token()

    \_collectdate = event['collect\_date']

    # Set headers

    headers = {'Authorization': 'Bearer ' + \_access\_token}

    # Set API URL

    FIIBIT\_API\_URL = 'https://api.fitbit.com/1/user/-/activities/heart/date/'+ \_collectdate + '/1d.json'

    # make request [GET] to Fitbit API

    response = requests.get(FIIBIT\_API\_URL, headers=headers)

    # If the request was successfully executed, return heart rate data

    if(response.status\_code == 200):

        heart\_rate\_data = response.json()['activities-heart'][0]

        # save 1d heart rate data to database

        save\_heart\_rate\_data(heart\_rate\_data)

        return {

            'statusCode': 200,

            'body': 'Successfully collected heart rate data and save to database.'

        }

    else:

        return {

            'statusCode': response.status\_code,

            'body': 'Could not collect heart rate data and save to database.'

        }

6.**getHeartRateTest**

import json

import requests

import boto3

def get\_access\_token():

    # create dynamodb client instance

    myclient = boto3.client('dynamodb')

    # get the item from the table

    GetItem = myclient.get\_item(

        TableName='fitbitToken',

        Key={

            'user\_id': {

                'S': 'BN4WML'

            },

        }

    )

    return GetItem['Item']['access\_token']['S']

def lambda\_handler(event, context):

    # TODO implement

    \_access\_token = get\_access\_token()

    \_startdate = event['startdate']

    \_enddate = event['enddate']

    # Set headers

    headers = {'Authorization': 'Bearer ' + \_access\_token}

    # Set API URL

    FIIBIT\_API\_URL = 'https://api.fitbit.com/1/user/-/activities/heart/date/'+ \_startdate + '/' + \_enddate + '.json'

    # make request [GET] to Fitbit API

    response = requests.get(FIIBIT\_API\_URL, headers=headers)

    # If the request was successfully executed, return heart rate data

    if(response.status\_code == 200):

        heart\_rate\_data = response.json()['activities-heart']

        return {

            'statusCode': 200,

            'body': json.dumps(heart\_rate\_data)

        }

    else:

        return {

            'statusCode': response.status\_code,

            'body': 'Could not fetch heart data'

        }

7**.fitbitHeartrate**

import json

import requests

import boto3

def get\_access\_token():

    # create dynamodb client instance

    myclient = boto3.client('dynamodb')

    # get the item from the table

    GetItem = myclient.get\_item(

        TableName='fitbitToken',

        Key={

            'user\_id': {

                'S': 'BN4WML'

            },

        }

    )

    return GetItem['Item']['access\_token']['S']

def lambda\_handler(event, context):

    # TODO implement

    \_access\_token = get\_access\_token()

    \_startdate = event['startdate']

    \_enddate = event['enddate']

    # Set headers

    headers = {'Authorization': 'Bearer ' + \_access\_token}

    # Set API URL

    FIIBIT\_API\_URL = 'https://api.fitbit.com/1/user/-/activities/heart/date/'+ \_startdate + '/' + \_enddate + '.json'

    # make request [GET] to Fitbit API

    response = requests.get(FIIBIT\_API\_URL, headers=headers)

    # If the request was successfully executed, return heart rate data

    if(response.status\_code == 200):

        heart\_rate\_data = response.json()['activities-heart']

        return {

            'statusCode': 200,

            'body': json.dumps(heart\_rate\_data)

        }

    else:

        return {

            'statusCode': response.status\_code,

            'body': 'Could not fetch heart data'

        }

8. **auto\_tokenRefresh**

import json

import requests

import sys

from datetime import datetime, timedelta

import boto3

def get\_fitbit\_auth():

    # create dynamodb client instance

    myclient = boto3.client('dynamodb')

    # get the item from the table

    GetItem = myclient.get\_item(

        TableName='fitbitToken',

        Key={

            'user\_id': {

                'S': 'BN4WML'

            },

        }

    )

    return GetItem['Item']

def lambda\_handler(event, context):

    ## retrive fitbit authorization data from db

    fitbit\_auth = get\_fitbit\_auth()

    ## set access token and refresh token

    access\_token = fitbit\_auth['access\_token']['S']

    refresh\_token = fitbit\_auth['refresh\_token']['S']

    time\_stamp = fitbit\_auth['time\_stamp']['S']

    expires\_in = fitbit\_auth['expires\_in']['N']

    ## set refresh url

    url="https://api.fitbit.com/oauth2/token"

    headers={

        "Authorization": "Basic MjNQWEo2OjllMTc3NTk3ZTA0NzdhMzBlMGRhNzdmZjIxMGEzM2E0",

        "Content-Type": "application/x-www-form-urlencoded"

    }

    data={

        "grant\_type":"refresh\_token",

        "refresh\_token":refresh\_token,

        "client\_id":"[23PXJ6]"

    }

    ## send refresh token request.

    response=requests.post(url,headers=headers,data=data)

    # If the request was successfully executed, put new authrization data to Table-fitbitToken

    if(response.status\_code == 200):

        ret\_auth = response.json()

        # create dynamodb object instance

        mydb = boto3.resource('dynamodb')

        # new a table object given a table name

        mytable = mydb.Table('fitbitToken')

        # Get current date and time

        now = datetime.now()

        # Format the output of timesamp

        formatted\_date\_time = now.strftime("%Y-%m-%d %H:%M:%S")

        # build data items from response

        item = {

            'access\_token': ret\_auth['access\_token'],

            'refresh\_token': ret\_auth['refresh\_token'],

            'user\_id': ret\_auth['user\_id'],

            'expires\_in': ret\_auth['expires\_in'],

            'token\_type': ret\_auth['token\_type'],

            'time\_stamp': formatted\_date\_time

        }

        # save access token information to database

        mytable.put\_item(Item=item)

        # return client with correct code.

        json\_return = {

            'statusCode': 200,

            # 'body': json.dumps(ret\_auth)

            'body': f"Successfully refreshed access token for user id:{ret\_auth['user\_id']}."

        }

        print(json\_return)

    else:

        json\_return = {

            'statusCode': response.status\_code,

            'body': 'Could not complete request for access token.'

        }

        print(json\_return)

9. refreshtokenname