"""

This sample demonstrates an implementation of the Lex Code Hook Interface

in order to serve a sample bot which manages reservations for hotel rooms and car rentals.

Bot, Intent, and Slot models which are compatible with this sample can be found in the Lex Console

as part of the 'BookTrip' template.

For instructions on how to set up and test this bot, as well as additional samples,

visit the Lex Getting Started documentation http://docs.aws.amazon.com/lex/latest/dg/getting-started.html.

"""

import json

import datetime

import time

import os

import dateutil.parser

import logging

import pymysql

import sqlalchemy

from sqlalchemy import create\_engine

import pandas as pd

logger = logging.getLogger()

logger.setLevel(logging.DEBUG)

# --- Helpers that build all of the responses ---

def elicit\_slot(session\_attributes, intent\_name, slots, slot\_to\_elicit, message):

return {

'sessionAttributes': session\_attributes,

'dialogAction': {

'type': 'ElicitSlot',

'intentName': intent\_name,

'slots': slots,

'slotToElicit': slot\_to\_elicit,

'message': message

}

}

def confirm\_intent(session\_attributes, intent\_name, slots, message):

return {

'sessionAttributes': session\_attributes,

'dialogAction': {

'type': 'ConfirmIntent',

'intentName': intent\_name,

'slots': slots,

'message': message

}

}

def close(session\_attributes, fulfillment\_state, message):

response = {

'sessionAttributes': session\_attributes,

'dialogAction': {

'type': 'Close',

'fulfillmentState': fulfillment\_state,

'message': message

}

}

return response

def delegate(session\_attributes, slots):

return {

'sessionAttributes': session\_attributes,

'dialogAction': {

'type': 'Delegate',

'slots': slots

}

}

# --- Helper Functions ---

def rec\_restaurant(intent\_request):

user = intent\_request['currentIntent']['slots']['user\_id']

#user = 'ET8n-r7glWYqZhuR6GcdNw'

session\_attributes = intent\_request['sessionAttributes'] if intent\_request['sessionAttributes'] is not None else {}

# Create database connection

db\_uri = 'mysql+pymysql://admin:bLeY2vQFGD5XFR4@brobsoaw-mysql.cskt1xevcmmb.us-east-1.rds.amazonaws.com/yelp'

engine = create\_engine(db\_uri)

# connect to database engine

conn = engine.connect()

# Query MySQL database

query = "SELECT \* FROM user\_recs\_clean WHERE user\_id = '%s'" % user

df = pd.read\_sql(query, engine)

# Format results as response

response = f"Here are our recomendations. Choice 1: {df['business\_name1'][0]}. \n " + f" Choice 2: {df['business\_name2'][0]}. \n " + f" Choice 3: {df['business\_name3'][0]}. \n "

conn.close()

return close(

session\_attributes,

'Fulfilled',

{

'contentType': 'PlainText',

'content': response

}

)

# --- Intents ---

def dispatch(intent\_request):

"""

Called when the user specifies an intent for this bot.

"""

logger.debug('dispatch userId={}, intentName={}'.format(intent\_request['userId'], intent\_request['currentIntent']['name']))

intent\_name = intent\_request['currentIntent']['name']

# Dispatch to your bot's intent handlers=

if intent\_name == 'RecRestaurant':

return rec\_restaurant(intent\_request)

raise Exception('Intent with name ' + intent\_name + ' not supported')

# --- Main handler ---

def lambda\_handler(event, context):

"""

Route the incoming request based on intent.

The JSON body of the request is provided in the event slot.

"""

# By default, treat the user request as coming from the America/New\_York time zone.

os.environ['TZ'] = 'America/New\_York'

time.tzset()

logger.debug('event.bot.name={}'.format(event['bot']['name']))

return dispatch(event)