import boto3

from botocore.exceptions import ClientError

import bcrypt

from datetime import datetime, timedelta

import jwt # PyJWT library for token management

import os

import logging

# Initialize DynamoDB client

dynamodb = boto3.client('dynamodb')

dynamodb\_resource = boto3.resource("dynamodb")

# Constants for token generation

SECRET\_KEY = os.getenv('JWT\_SECRET', 'your-secret-key') # Store this securely!

TOKEN\_EXPIRATION\_MINUTES = 15 # Shorter expiration for added security

REFRESH\_TOKEN\_EXPIRATION\_DAYS = 7 # Refresh token expiration

# Constants for login

MAX\_LOGIN\_ATTEMPTS = 5

LOCKOUT\_DURATION\_MINUTES = 15

RATE\_LIMIT\_WINDOW\_SECONDS = 60 # 1 minute

RATE\_LIMIT\_MAX\_REQUESTS = 10 # Max 10 requests in 1 minute

# Configure logging

logging.basicConfig(level=logging.INFO)

logger = logging.getLogger()

def generate\_client\_table\_name(company\_id):

"""Generate the Table name string based on the id."""

return f"{company\_id}Employee"

def generate\_auth\_table\_name(company\_id):

"""Generate the Table name string based on the id."""

return f"{company\_id}Employee\_Auth"

def generate\_token(email, guid, expiration\_minutes):

"""Generate a JWT token for the user."""

expiration = datetime.utcnow() + timedelta(minutes=expiration\_minutes)

payload = {

"email": email,

"guid": guid,

"exp": expiration

}

token = jwt.encode(payload, SECRET\_KEY, algorithm="HS256")

return token

def generate\_refresh\_token(email, guid):

"""Generate a JWT refresh token."""

expiration = datetime.utcnow() + timedelta(days=REFRESH\_TOKEN\_EXPIRATION\_DAYS)

payload = {

"email": email,

"guid": guid,

"exp": expiration

}

refresh\_token = jwt.encode(payload, SECRET\_KEY, algorithm="HS256")

return refresh\_token

def lambda\_handler(event, context):

# Extract email, password, and company\_id (using the correct JSON key "CompanyID")

email = event.get('email')

password = event.get('password')