from functools import wraps

import jwt

from flask import request

from backend import conn, config

import ibm\_db

# Middleware function that checks for JWT token in header

# All routes that have the @token\_required decorator will be protected

def token\_required(f):

@wraps(f)

def decorated(\*args, \*\*kwargs):

token = None

if "Authorization" in request.headers:

token = request.headers["Authorization"].split(" ")[1]

if not token:

return {

"error": "Unauthorized"

}, 401

try:

# Get the user's email from the decoded token

data = jwt.decode(

token, config["APP\_SECRET"], algorithms=["HS256"])

# Retreive user's info from the database

sql = f"select \* from users where email='{data['email']}'"

stmt = ibm\_db.prepare(conn, sql)

ibm\_db.execute(stmt)

current\_user = ibm\_db.fetch\_assoc(stmt)

# If user does not exist throw error.

if current\_user is None:

return {

"error": "Unauthorized"

}, 401

except Exception as e:

return {

"error": str(e)

}, 500

# Pass the authorized user in function args.

return f(current\_user, \*args, \*\*kwargs)

return decorated