from sqlalchemy.orm.session import sessionmaker  
from app.models import QuestionMaster, QuizInstance, QuizMaster, QuizQuestions, UserMaster, UserResponses, UserSession  
from app import db  
import uuid  
from flask import session  
import datetime  
from typing import List  
  
"""  
[Services Module] Implement various helper functions here as a part of api  
 implementation using MVC Template  
create\_user= Done   
check\_user\_session\_is\_active  
check\_if\_session\_is\_active  
login\_user= Done   
logout\_user  
check\_if\_admin  
add\_question  
list\_questions  
create\_quiz  
assign\_quiz  
check\_quiz\_access  
view\_quiz  
get\_assigned\_quiz\_info  
get\_all\_quiz\_info  
attempt\_quiz  
quiz\_results  
"""  
  
  
def create\_user(\*\*kwargs):  
 try:  
 user = UserMaster(  
 uuid.uuid4(),  
 kwargs['name'],  
 kwargs['username'],  
 kwargs['password'],  
 kwargs['is\_admin']  
 )  
  
 db.session.add(user)  
 db.session.commit()  
 except Exception as err:  
 raise err  
  
  
# This method will check if the session for the mentioned user\_id is active or not.  
def check\_user\_session\_is\_active(user\_id):  
 try:  
 user\_session = UserSession.query.filter\_by(user\_id=user\_id, is\_active=1).first()  
 if user\_session:  
  
 return True, user\_session  
 else:  
 return False, None  
  
 except Exception as err:  
 raise err  
  
  
# This method will check the credentials for the user and perform login activity  
def login\_user(\*\*kwargs):  
 try:  
 user = UserMaster.query.filter\_by(username=kwargs['username'], password=kwargs['password']).first()  
 if user:  
 print('user logged in')  
  
 is\_active, session\_id = check\_user\_session\_is\_active(user.id)  
  
 if not is\_active:  
 session\_id = uuid.uuid4()  
 user\_session = UserSession(uuid.uuid4(), user.id, session\_id)  
 db.session.add(user\_session)  
 db.session.commit()  
  
 else:  
 session['session\_id'] = session\_id  
  
 session['session\_id'] = session\_id  
 return True, session\_id  
  
 else:  
 return False, None  
  
 except Exception as err:  
 raise err