from aiogram.utils.helper import Helper, HelperMode, ListItem  
from aiogram import Dispatcher, types  
from aiogram.dispatcher import FSMContext  
from aiogram.dispatcher.filters import state  
from aiogram.types.base import String  
from aiogram.utils import executor, callback\_data  
from create\_bot import dp, bot  
from aiogram.types import ReplyKeyboardMarkup, KeyboardButton, ContentType, LabeledPrice  
import time  
from aiogram.dispatcher.filters.state import State, StatesGroup  
import asyncio  
from handlers import client  
from apscheduler.schedulers.asyncio import AsyncIOScheduler  
from handlers import apsched, rules, kick\_user, price\_checker, user\_input  
from database import database\_conn  
  
PAYMENTS\_TOKEN = '401643678:TEST:d009db99-1e8f-454e-ae3a-dba629e886d5'  
b1 = KeyboardButton('/Оплатить', *callback\_data*='user\_id')  
b2 = KeyboardButton('/Меню')  
kb\_client = ReplyKeyboardMarkup(*resize\_keyboard*=True, *one\_time\_keyboard*=True)  
kb\_client.add(b1).add(b2)  
schedule = AsyncIOScheduler(*timezone*='Europe/Moscow')  
  
  
class FSMAdmin3(StatesGroup):  
 price = State()  
 type = State()  
 solve = State()  
 input = State()  
 mraz = State()  
  
*# start*async def on\_startup():  
 schedule.start()  
 *print*('стартануло')  
  
  
*# buy*@dp.message\_handler(*commands*=['Оплатить'])  
async def buy(*message*: types.Message, *state*: FSMContext):  
 if PAYMENTS\_TOKEN.split(':')[1] == 'TEST':  
 async with *state*.proxy() as data:  
 *# read =  
 # ret = read  
 # rit = list(ret)  
 # rat = ','.join(''.join(tup) for tup in rit)* await *message*.answer(*text*='Выберите тип игры, который хотите оплатить:')  
 await database\_conn.sql\_select\_command6(*message*)  
 await FSMAdmin3.type.set()  
  
  
@dp.message\_handler(*state*=FSMAdmin3.type)  
async def state1(*message*: types.Message, *state*: FSMContext):  
 async with *state*.proxy() as data:  
 data['type'] = *message*.text  
 sex = data['type']  
  
 *print*(data['type'])  
 await FSMAdmin3.next()  
  
 *# data['price'] = await database\_conn.sql\_select\_command3(data)  
  
 # await bot.send\_message(message.chat.id, "Тестовый платеж")  
 # await database\_conn.sql\_select\_command2(x['tipe'])  
 # print(x['price'])* async with *state*.proxy() as data:  
 read = await database\_conn.sql\_select\_command3(*state*)  
 *print*(read)  
 for ret in read:  
 for x in ret:  
 x = *str*(ret[0])  
 await FSMAdmin3.price.set()  
 rat = ','.join(''.join(tup) for tup in data)  
 *# print(int(x), 'ssc')  
 #PRICE = types.LabeledPrice(label='test', amount=int(x) \* 100)* '''await bot.send\_invoice(chat\_id=message.chat.id, title="Участие в одной игре",  
 description="Оплата на участие в одной игре",  
 provider\_token=PAYMENTS\_TOKEN,  
 currency="rub",  
 is\_flexible=False,  
 prices=[PRICE],  
 start\_parameter="one-month-subscription",  
 payload="test-invoice-payload",  
 ) # keyError 'price'''  
  
 await rules(*state*, *message*)  
 '''if True:  
 # в шедулер будет подставляться время из бд  
 schedule.add\_job(apsched.timer\_message\_cron, trigger='interval', seconds=9, kwargs={'message': message})  
 print('запустилось')  
 schedule.start()  
 await asyncio.sleep(10)  
 await bot.unban\_chat\_member(chat\_id='-1001781599983', user\_id='5870143609', only\_if\_banned=True)  
 await rules(state, message)'''  
  
  
async def rules(*state*: FSMContext, *message*: types.Message):  
 async with *state*.proxy() as data:  
 sex = data['type']  
 *print*(*str*(sex), 'sexim 11')  
 read = await database\_conn.sql\_select\_command7(*state*)  
 for ret in read:  
 for a in ret:  
 a = *str*(ret[0])  
 await bot.send\_message(*chat\_id*='-1001781599983', *text*=a)  
 await bot.send\_message(*chat\_id*='-1001781599983', *text*='на ответ у вас есть 30 секунд')  
 await *message*.answer("Введите ваш ответ: ")  
 *print*(*state*)  
 await FSMAdmin3.mraz.set()  
 kex = await database\_conn.sql\_select\_command7(*state*)  
 for ket in kex:  
 for b in ket:  
 c = *str*(ret[1])  
 d = *str*(ret[0])  
 data['mraz'] = c  
 *print*(data['mraz'], 'mraz')  
 *print*(c, 'c')  
 *print*(d, 'd')  
 await FSMAdmin3.solve.set()  
  
  
@dp.message\_handler(*state*=FSMAdmin3.solve)  
async def solution(*state*: FSMContext, *message*: types.Message):  
 async with *state*.proxy() as data:  
 zoz = *message*.text  
 *print*(zoz, 'solve')  
 '''riad = await database\_conn.sql\_select\_command8(state)  
 for rut in riad:  
 for y in rut:  
 y = str(rut[0])  
 if y == data['mraz']:  
 print('yes')  
 await bot.send\_message(chat\_id='-1001781599983', text='правильный ответ')  
 await message.reply(text='вы победили, вы можете вывести деньги по кнопке /Вывести')  
 else:  
 print('no')  
 await bot.send\_message(chat\_id='-1001781599983', text='неверно')  
 if True:  
 schedule.add\_job(kick\_user.kick\_user, trigger='interval', seconds=60, kwargs={'bot': bot})  
 print('запустилось, но 3')  
 await asyncio.sleep(1)  
 # pre checkout  
  
 '''  
@dp.pre\_checkout\_query\_handler(lambda *query*: True)  
async def process\_pre\_checkout\_query(*pre\_checkout\_query*: types.PreCheckoutQuery):  
 await bot.answer\_pre\_checkout\_query(*pre\_checkout\_query*.id, *ok*=True)  
 *print*('хуйня')  
  
*# successful payment* '''  
 @dp.message\_handler(content\_types=ContentType.SUCCESSFUL\_PAYMENT)  
 async def successful\_payment(message: types.Message):  
 print("SUCCESSFUL PAYMENT")  
 payment\_info = message.successful\_payment.to\_python()  
 for k, v in payment\_info.items():  
 print(f"{k}= {v}")  
 await bot.send\_message(message.chat.id,  
 f"Платеж на сумму {message.successful\_payment.total\_amount // 100}{message.successful\_payment.currency} провел успешно ")'''  
  
  
  
  
  
  
def register\_handlers\_ckb(*dp*: Dispatcher):  
 *dp*.pre\_checkout\_query\_handler(lambda *query*: True)  
 *dp*.message\_handler(*content\_types*=ContentType.SUCCESSFUL\_PAYMENT)  
  
 if \_\_name\_\_ == '\_\_main\_\_':  
 executor.start\_polling(*dp*, *skip\_updates*=True, *on\_startup*=on\_startup)