D:\ApiTest\testcase\test\_case\_orientationModule\test\_case\_oriModMain.py：

import allure  
import pytest  
import random  
from api.oriMod\_api import OriModApi  
from testcase.test\_case\_optimize.conftest import \*  
from utils.response\_handler import handle\_response  
from utils.read\_oriModMain import base\_data  
  
# ANSI转义序列用于控制台颜色输出  
RED = "\033[91m"  
GREEN = "\033[92m"  
ENDC = "\033[0m"  
YELLOW = "\033[93m"  
  
def generate\_random\_model\_name(prefix="Py\_", action=None):  
 random\_number = random.randint(10, 99) # 生成两位随机数  
 name = f"{prefix}{random\_number}"  
 if action:  
 TestAdmax\_oriModMain.generated\_names[action] = name  
 return name  
  
@allure.epic("ADMax-定向模块")  
@allure.feature("定向模块主流程测试")  
@allure.description("定向模块主流程测试流程：新增-编辑-复制-删除，并在每个操作后进行查询")  
class TestAdmax\_oriModMain:  
 # 定义一个字典来保存生成的模板名称  
 generated\_names = {}  
  
 @allure.story("定向模块新增")  
 @allure.title("新增定向模块：pytest\_add")  
 @allure.testcase("https://test-admax.yostar.net/api/orientationModule/addOrEdit", "定向模块新增地址")  
 @allure.severity("blocker") # 用例等级  
 @pytest.mark.run(order=1)  
 def test\_add\_oriMod(self):  
 data = base\_data.read\_yaml()['add\_oriMod']  
 data['model\_name'] = generate\_random\_model\_name("Py\_add\_", "add")  
 result = OriModApi.add\_orientationModule(data)  
 handle\_response(result)  
  
 @allure.story("定向模块查询")  
 @allure.title("查询上一步中新增的模块：pytest\_add")  
 @allure.testcase("https://test-admax.yostar.net/api/orientationModule/list", "定向模块查询地址")  
 @allure.severity("blocker") # 用例等级  
 @pytest.mark.run(order=2)  
 def test\_add\_oriMod\_Search(self):  
 data = base\_data.read\_yaml()['add\_oriMod\_Search']  
 data['model\_name'] = self.generated\_names["add"]  
 result = OriModApi.oriMod\_Search(data)  
 assert result['data']['list'][0]['model\_name'] == self.generated\_names["add"]  
 handle\_response(result)  
  
 @allure.story("定向模块编辑")  
 @allure.title("编辑新增后的模板")  
 @allure.testcase("https://test-admax.yostar.net/api/orientationModule/addOrEdit", "定向模块编辑地址")  
 @allure.severity("blocker") # 用例等级  
 @pytest.mark.run(order=3)  
 def test\_add\_oriMod\_Edit(self):  
 add\_id = add\_oriMod\_id()  
 data = base\_data.read\_yaml()['add\_oriMod\_Edit']  
 data['id'] = add\_id  
 data['model\_name'] = generate\_random\_model\_name("Py\_edit\_", "edit")  
 result = OriModApi.add\_orientationModule(data)  
 handle\_response(result)  
  
 @allure.story("定向模块查询")  
 @allure.title("查询编辑后的模板：py\_Edit")  
 @allure.testcase("https://test-admax.yostar.net/api/orientationModule/list", "定向模块查询地址")  
 @allure.severity("blocker") # 用例等级  
 @pytest.mark.run(order=4)  
 def test\_oriMod\_Edit\_Search(self):  
 data = base\_data.read\_yaml()['oriMod\_Edit\_Search']  
 result = OriModApi.oriMod\_Search(data)  
 assert result['data']['list'][0]['model\_name'] == self.generated\_names["edit"]  
 handle\_response(result)  
  
 @allure.story("定向模块-复制")  
 @allure.title("复制编辑后的模板：py\_Edit——>copy\_pyEdit")  
 @allure.testcase("https://test-admax.yostar.net/api/orientationModule/copy", "定向模块复制地址")  
 @allure.severity("blocker") # 用例等级  
 @pytest.mark.run(order=5)  
 def test\_copy\_oriMod(self):  
 copy\_id = edit\_oriMod\_id()  
 data = base\_data.read\_yaml()['copy\_oriMod']  
 data['copy\_id'] = copy\_id  
 data['model\_name'] = generate\_random\_model\_name("Py\_copy\_", "copy")  
 result = OriModApi.copy\_oriMod(data)  
 handle\_response(result)  
  
 @allure.story("定向模块-查询")  
 @allure.title("查询复制后的模板：copy\_pyEdit")  
 @allure.testcase("https://test-admax.yostar.net/api/orientationModule/list", "定向模块查询地址")  
 @allure.severity("blocker") # 用例等级  
 @pytest.mark.run(order=6)  
 def test\_copy\_oriMod\_Search(self):  
 data = base\_data.read\_yaml()['copy\_oriMod\_Search']  
 data['model\_name'] = self.generated\_names["copy"]  
 result = OriModApi.oriMod\_Search(data)  
 assert result['data']['list'][0]['model\_name'] == self.generated\_names["copy"]  
 handle\_response(result)  
  
 @allure.story("定向模块删除")  
 @allure.title("删除新增/编辑/复制的模板模板数据，防止下次测试冲突")  
 @allure.testcase("https://test-admax.yostar.net/api/orientationModule/delete", "定向模块删除地址")  
 @allure.severity("blocker") # 用例等级  
 @pytest.mark.run(order=7)  
 def test\_del\_oriMod(self):  
 # 获取需要删除的模板 ID 列表  
 del\_ids = [edit\_oriMod\_id(), copy\_oriMod\_id()]  
  
 for del\_id in del\_ids:  
 data = base\_data.read\_yaml()['del\_oriMod']  
 data['id'] = del\_id  
 result = OriModApi.del\_orientationModule(data)  
 handle\_response(result)

D:\ApiTest\utils\read\_oriModMain.py:

import configparser  
import os  
  
import yaml  
  
yaml\_path = os.path.join(os.path.dirname(os.path.dirname(\_\_file\_\_)), 'data', 'oriModMain.yaml')  
ini\_path = os.path.join(os.path.dirname(os.path.dirname(\_\_file\_\_)), 'config', 'settings.ini')  
  
  
class ReadOriModMain:  
 def \_\_init\_\_(self):  
 self.yaml\_path = yaml\_path  
 self.ini\_path = ini\_path  
  
 def read\_yaml(self):  
 f = open(self.yaml\_path, encoding="utf-8")  
 data = yaml.safe\_load(f)  
 return data  
  
 def read\_ini(self):  
 config = configparser.ConfigParser()  
 config.read(self.ini\_path, encoding='utf-8')  
 return config  
  
  
base\_data = ReadOriModMain()

D:\ApiTest\data\oriModMain.yaml:

add\_oriMod:  
 id: 0  
 channel\_id: OceanEngine  
 model\_name: pytest\_add  
 model\_desc: pytest新建  
 content:  
 channel\_id: OceanEngine  
 district: NONE  
 region\_version: 1.0.0  
 city: [ ]  
 location\_type: ALL  
 district\_select: ''  
 gender: NONE  
 age: [ ]  
 retargeting\_tags\_include: [ ]  
 retargeting\_tags\_exclude: [ ]  
 action\_categories\_select: ''  
 interest\_categories\_select: ''  
 crowd: '1'  
 interest\_action\_mode: UNLIMITED  
 action\_scene: [ ]  
 action\_days: 30  
 action\_categories: [ ]  
 action\_words: [ ]  
 interest\_categories: [ ]  
 interest\_words: [ ]  
 superior\_popularity\_type: NONE  
 device\_type:  
 - MOBILE  
 ac:  
 - WIFI  
 hide\_if\_exists: UNLIMITED  
 hide\_if\_converted: APP  
 converted\_time\_duration: TODAY  
 android\_osv: '0.0'  
 ios\_osv: '0.0'  
 carrier: [ ]  
 device\_brand: [ ]  
 auto\_extend\_enabled: 0  
 auto\_extend\_targets: [ ]  
 model\_name: pytest\_add  
 model\_desc: pytest新建  
  
add\_oriMod\_Search:  
 channel\_id: OceanEngine  
 model\_name: 'pytest\_add'  
  
add\_oriMod\_Edit:  
 id: 179  
 channel\_id: OceanEngine  
 model\_name: py\_Edit  
 model\_desc: 新建后编辑校验  
 content:  
 channel\_id: OceanEngine  
 district: NONE  
 region\_version: 1.0.0  
 city: [ ]  
 location\_type: ALL  
 district\_select: ''  
 gender: NONE  
 age: [ ]  
 retargeting\_tags\_include: [ ]  
 retargeting\_tags\_exclude: [ ]  
 action\_categories\_select: ''  
 interest\_categories\_select: ''  
 crowd: '1'  
 interest\_action\_mode: UNLIMITED  
 action\_scene: [ ]  
 action\_days: 30  
 action\_categories: [ ]  
 action\_words: [ ]  
 interest\_categories: [ ]  
 interest\_words: [ ]  
 aweme\_fan\_behaviors: [ ]  
 aweme\_fan\_time\_scope: ''  
 aweme\_fan\_categories: [ ]  
 aweme\_fan\_accounts: [ ]  
 superior\_popularity\_type: NONE  
 device\_type:  
 - MOBILE  
 ac:  
 - WIFI  
 hide\_if\_exists: UNLIMITED  
 hide\_if\_converted: APP  
 converted\_time\_duration: TODAY  
 android\_osv: '0.0'  
 ios\_osv: '0.0'  
 carrier: [ ]  
 device\_brand: [ ]  
 auto\_extend\_enabled: 0  
 auto\_extend\_targets: [ ]  
 model\_name: py\_Edit  
 model\_desc: 新建后编辑校验  
 id: 179  
  
  
oriMod\_Edit\_Search:  
 channel\_id: OceanEngine  
 model\_name: 'py\_Edit'  
copy\_oriMod\_Search:  
 channel\_id: OceanEngine  
 model\_name: 'copy\_pyEdit'  
  
copy\_oriMod:  
 copy\_id: 204  
 model\_name: copy\_pyEdit  
 model\_desc: 从py\_Edit复制的模板  
  
  
del\_oriMod:  
 channel\_id: OceanEngine  
 id: 174

D:\ApiTest\testcase\test\_case\_optimize\conftest.py:

from utils.log\_util import logger  
from utils.mysql\_util import db  
  
  
def add\_oriMod\_id():  
 *"""  
 查询并返回用例中新建的定向模板的ID  
 :return: 定向模板的ID  
 """* # SQL查询语句，查找model\_name包含'pytest\_sql'且未被删除的记录  
 sql = """  
 SELECT id  
 FROM admax\_test.channel\_orientation\_model  
 WHERE model\_name LIKE '%pytest\_add%'  
 AND deleted\_at IS NULL  
 LIMIT 1  
 """  
  
 # 执行SQL查询  
 result = db.select\_db(sql)  
  
 # 记录查询结果  
 logger.info(f'SQL执行结果: {result}')  
  
 # 检查查询结果是否为空  
 if not result:  
 logger.warning('未找到符合条件的定向模板')  
 return None  
  
 # 返回查询到的第一个记录的ID  
 return result[0]['id']  
  
def edit\_oriMod\_id():  
 *"""  
 查询并返回用例中新建的定向模板的ID  
 :return: 定向模板的ID  
 """* # SQL查询语句，查找model\_name包含'pytest\_sql'且未被删除的记录  
 sql = """  
 SELECT id  
 FROM admax\_test.channel\_orientation\_model  
 WHERE model\_name LIKE '%py\_Edit%'  
 AND deleted\_at IS NULL  
 LIMIT 1  
 """  
  
 # 执行SQL查询  
 result = db.select\_db(sql)  
  
 # 记录查询结果  
 logger.info(f'SQL执行结果: {result}')  
  
 # 检查查询结果是否为空  
 if not result:  
 logger.warning('未找到符合条件的定向模板')  
 return None  
  
 # 返回查询到的第一个记录的ID  
 return result[0]['id']

D:\ApiTest\api\oriMod\_api.py:

*"""  
这是一个API接口封装文件，提供了多个与广告投放和定向模块相关的功能。  
"""*from core.oriMod\_apiUtil import OriModApiUtil  
  
  
class OriModApi(object):  
  
 def add\_orientationModule(params):  
 *"""  
 新增定向模块  
 :param params: 新增参数  
 :return: 新增结果的JSON响应  
 """* response = OriModApiUtil.add\_orientationModule(json=params)  
 return response.json()  
  
 def oriMod\_Search(params):  
 *"""  
 获取定向查询数据  
 :param params: 查询参数  
 :return: 查询结果的JSON响应  
 """* response = OriModApiUtil.oriMod\_Search(params=params)  
 return response.json()  
  
 def add\_oriMod\_Edit(params):  
 *"""  
 编辑定向模块  
 :param params: 编辑参数  
 :return: 编辑结果的JSON响应  
 """* response = OriModApiUtil.add\_oriMod\_Edit(json=params)  
 return response.json()  
  
 def copy\_oriMod(params):  
 *"""  
 获取定向查询数据  
 :param params: 查询参数  
 :return: 查询结果的JSON响应  
 """* response = OriModApiUtil.copy\_oriMod(json=params)  
 return response.json()  
  
 def del\_orientationModule(params):  
 *"""  
 删除定向模块  
 :param params: 删除参数  
 :return: 删除结果的JSON响应  
 """* response = OriModApiUtil.del\_orientationModule(json=params)  
 return response.json()  
  
  
api = OriModApi()

D:\ApiTest\core\oriMod\_apiUtil.py：

from core.rest\_client import RestClient  
  
  
class OriModApiUtil(RestClient):  
 *"""  
 ApiUtil 类封装了与广告投放和定向模块相关的API请求。  
 继承自 RestClient，提供了常用的HTTP请求方法。  
 """* def \_\_init\_\_(self):  
 *"""  
 初始化 ApiUtil 实例，调用父类的初始化方法。  
 """* super().\_\_init\_\_()  
  
 def add\_orientationModule(self, \*\*kwargs):  
 *"""  
 新增定向模块。  
  
 :param kwargs: 请求参数  
 :return: 响应对象  
 """* return self.post('/api/orientationModule/addOrEdit', \*\*kwargs)  
  
 def oriMod\_Search(self, \*\*kwargs):  
 *"""  
 获取定向查询数据。  
  
 :param kwargs: 请求参数  
 :return: 响应对象  
 """* return self.get('/api/orientationModule/list', \*\*kwargs)  
  
 def add\_oriMod\_Edit(self, \*\*kwargs):  
 *"""  
 编辑定向模块。  
  
 :param kwargs: 请求参数  
 :return: 响应对象  
 """* return self.post('/api/orientationModule/addOrEdit', \*\*kwargs)  
  
 def copy\_oriMod(self, \*\*kwargs):  
 *"""  
 编辑定向模块。  
  
 :param kwargs: 请求参数  
 :return: 响应对象  
 """* return self.post('/api/orientationModule/copy', \*\*kwargs)  
  
 def del\_orientationModule(self, \*\*kwargs):  
 *"""  
 删除定向模块。  
  
 :param kwargs: 请求参数  
 :return: 响应对象  
 """* return self.post('/api/orientationModule/delete', \*\*kwargs)  
  
  
# 创建 ApiUtil 的实例，供其他模块使用  
OriModApiUtil = OriModApiUtil()

D:\ApiTest\core\rest\_client.py：

import json  
import time  
  
import requests  
  
from utils.log\_util import logger  
  
# 测试环境的URL  
test\_url = "https://test-admax.yostar.net"  
time = int(time.time())  
# 测试环境的授权信息  
Authorization\_test = {  
 "Head": {  
 "token": "7d752b00c14e7657a17469467cfe2e86ad2b9c7a",  
 "region": "CN",  
 "version": "1.0",  
 "time": time  
 }  
}  
  
# 将授权信息转换为请求头格式  
token\_test = {"Authorization": json.dumps(Authorization\_test)}  
  
# print(token\_test)  
class RestClient:  
 *"""  
 RestClient 类用于封装HTTP请求，提供GET和POST方法。  
 """* def \_\_init\_\_(self):  
 *"""  
 初始化 RestClient 实例，设置日志记录器、授权信息和测试URL。  
 """* self.logger = logger  
 self.token\_test = token\_test  
 self.test\_url = test\_url  
  
 def get(self, url, \*\*kwargs):  
 *"""  
 发送GET请求。  
  
 :param url: 请求的URL路径  
 :param kwargs: 其他请求参数  
 :return: 响应对象  
 """* self.logger.debug("请求头：{}".format(self.token\_test))  
 self.logger.debug("请求参数：{}".format(json.dumps(kwargs)))  
 self.logger.debug("请求地址：{}".format(self.test\_url + url))  
  
 response = requests.get(self.test\_url + url, headers=self.token\_test, \*\*kwargs)  
 self.logger.debug("返回结果：{}".format(response.json()))  
  
 return response  
  
 def post(self, url, \*\*kwargs):  
 *"""  
 发送POST请求。  
  
 :param url: 请求的URL路径  
 :param kwargs: 其他请求参数  
 :return: 响应对象  
 """* self.logger.debug("请求头：{}".format(self.token\_test))  
 self.logger.debug("请求参数：{}".format(json.dumps(kwargs)))  
 self.logger.debug("请求地址：{}".format(self.test\_url + url))  
  
 response = requests.post(self.test\_url + url, headers=self.token\_test, \*\*kwargs)  
 self.logger.debug("返回结果：{}".format(response.json()))  
  
 return response  
  
  
  
# import json  
# from encodings.utf\_7 import encode  
#  
# import requests  
# from Tools.scripts.generate\_opcode\_h import header  
#  
# from utils.log\_util import logger  
#  
# test\_url = "https://test-admax.yostar.net"  
# Authorization\_test = {  
# "Head": {"token": "9f6ed40924488358990084e699901b35476de082", "region": "CN", "version": "1.0",  
# "time": 1691486926}}  
# token\_test = {"Authorization": json.dumps(Authorization\_test)}  
#  
# class RestClient:  
# def \_\_init\_\_(self):  
# self.token\_test = token\_test  
# self.test\_url = test\_url  
# def get(self,url, \*\*kwargs):  
# # return requests.get(self.test\_url + url,headers = self.token\_test, \*\*kwargs)  
# return self.request(url,"GET", \*\*kwargs)  
#  
# def post(self,url, \*\*kwargs):  
# return self.request(url,"POST", \*\*kwargs)  
#  
# def request(self,url,method, \*\*kwargs):  
#  
# if method == "GET":  
# return requests.get(self.test\_url + url,headers = self.token\_test, \*\*kwargs)  
#  
# if method == "POST":  
# return requests.get(self.test\_url + url,headers = self.token\_test, \*\*kwargs)  
# logger.info("请求方式：{}".format(method))  
# logger.info("请求头：{}".format(self.token\_test))  
# logger.info("请求参数：\n{}".format(json.dumps(kwargs, indent=2)))  
# logger.info("请求地址：{}".format(self.test\_url + url))  
# logger.info("返回结果：{}".format(  
# requests.request(method, self.test\_url + url, headers=self.token\_test, \*\*kwargs).json()))

D:\ApiTest\utils\mysql\_util.py：

import pymysql  
  
from utils.log\_util import logger  
from utils.read\_ini import read\_ini  
  
data = read\_ini()['mysql']  
# print(data)  
Db\_CONF = {  
 'host': data['MYSQL\_HOST'],  
 'port': int(data['MYSQL\_PORT']),  
 'user': data['MYSQL\_USER'],  
 'password': data['MYSQL\_PASSWORD']  
}  
  
class MySQL:  
 def \_\_init\_\_(self):  
 self.conn = pymysql.connect(\*\*Db\_CONF)  
 self.cur = self.conn.cursor(cursor=pymysql.cursors.DictCursor)  
  
  
 def \_\_del\_\_(self):  
 self.cur.close()  
 self.conn.close()  
  
 def select\_db(self, sql):  
 *"""  
 执行sql语句，返回查询结果  
 :param sql:  
 :return:  
 """* logger.info(f"开始执行sql:" + sql)  
 self.cur.execute(sql)  
 return self.cur.fetchall()  
  
 def execute\_db(self, sql):  
 *"""  
 执行sql语句，返回受影响行数  
 :param sql:  
 :return:  
 """* try:  
 logger.info(f"开始执行sql:"+ sql)  
 self.cur.execute(sql)  
 self.conn.commit()  
 except Exception as e:  
 logger.error("执行sql时报错：".format(e))  
  
db = MySQL()  
if \_\_name\_\_ == '\_\_main\_\_':  
 db = MySQL()  
 # sql = "SELECT channel\_id FROM admax\_test.advertiser\_report where id = 5"  
 # result = db.select\_db(sql)  
 # # print(type(result))  
 # print(result[0]['channel\_id'])

D:\ApiTest\utils\log\_util.py：

import logging  
import os  
import time  
  
# 获取项目的根目录  
root\_path = os.path.dirname(os.path.abspath(os.path.dirname(\_\_file\_\_)))  
log\_path = os.path.join(root\_path, 'log')  
  
# 打印日志路径  
print(log\_path)  
  
# 如果日志目录不存在，则创建  
if not os.path.exists(log\_path):  
 os.mkdir(log\_path)  
  
class Logger:  
 def \_\_init\_\_(self):  
 # 获取当前日期  
 current\_date = time.strftime("%Y-%m-%d")  
 # 创建日期文件夹  
 date\_folder = os.path.join(log\_path, current\_date)  
 if not os.path.exists(date\_folder):  
 os.mkdir(date\_folder)  
  
 # 定义日志文件名  
 log\_filename = "{}.log".format(time.strftime("%H点%M分%S秒"))  
 self.logname = os.path.join(date\_folder, log\_filename)  
  
 # 定义一个日志容器  
 self.logger = logging.getLogger("log")  
 # 设置日志打印的级别  
 self.logger.setLevel(logging.DEBUG)  
 # 创建日志输入的格式  
 self.formater = logging.Formatter(  
 '[%(asctime)s][%(filename)s %(lineno)d][%(levelname)s]: %(message)s')  
 # 创建日志处理器，用来存放日志文件  
 self.filelogger = logging.FileHandler(self.logname, mode='a', encoding="UTF-8")  
 # 创建日志处理器，在控制台打印  
 self.console = logging.StreamHandler()  
 # 设置控制台打印日志级别  
 self.console.setLevel(logging.DEBUG)  
 # 文件存放日志级别  
 self.filelogger.setLevel(logging.DEBUG)  
 # 文件存放日志格式  
 self.filelogger.setFormatter(self.formater)  
 # 控制台打印日志格式  
 self.console.setFormatter(self.formater)  
 # 将日志输出渠道添加到日志收集器中  
 self.logger.addHandler(self.filelogger)  
 self.logger.addHandler(self.console)  
  
logger = Logger().logger  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 logger.info("---INFO日志---")  
 logger.debug("---DEBUG日志---")  
 logger.warning("---WARNING日志---")  
 logger.error("---ERROR日志---")  
 logger.critical("---CRITICAL日志---")

D:\ApiTest\utils\read\_ini.py：

import configparser  
import os  
path = os.path.join(os.path.dirname(os.path.dirname(\_\_file\_\_)), 'config', 'settings.ini')  
  
def read\_ini():  
 config = configparser.ConfigParser()  
 config.read(path,encoding='utf-8')  
 return config  
  
# get\_ini = read\_ini()  
# print(read\_ini()['host']['test\_url'])

D:\ApiTest\pytest.ini：

[pytest]  
testpath=./testcase/test\_case\_optimize/test\_post\_get\_optimize\_class.py  
  
markers =  
 p0: 主流程测试  
 p1: 主要功能测试  
 p2: 全量测试  
  
addopts = -vs --alluredir=./report

D:\ApiTest\config\settings.ini：

[host]  
test\_url = https://test-advertising.yostar.net  
test\_url\_ssl = https://staging-advertising.yostar.net  
  
[mysql]  
MYSQL\_HOST = rm-uf6kf5r2g781f0fg0mo.mysql.rds.aliyuncs.com  
MYSQL\_PORT = 3306  
MYSQL\_USER = admax\_test  
MYSQL\_PASSWORD = 4znK9SR0ymYW3UwG  
  
[redis]  
REDIS\_HOST = r-uf6xrv207own00ws6vpd.redis.rds.aliyuncs.com  
REDIS\_PASSWORD = ASRYSf0o6mxMhiDP

D:\ApiTest\log\2024-11-01\19点07分38秒.log：

[2024-11-01 19:07:38,278][*rest\_client.py 63*][DEBUG]: 请求头：{'Authorization': '{"Head": {"token": "f11b556a0a15636392b54e395831bb5576874f76", "region": "CN", "version": "1.0", "time": 1730459258}}'}  
[2024-11-01 19:07:38,278][*rest\_client.py 64*][DEBUG]: 请求参数：{"json": {"id": *0*, "channel\_id": "OceanEngine", "model\_name": "Py\_add\_63", "model\_desc": "pytest\u65b0\u5efa", "content": {"channel\_id": "OceanEngine", "district": "NONE", "region\_version": "1.0.0", "city": [], "location\_type": "ALL", "district\_select": "", "gender": "NONE", "age": [], "retargeting\_tags\_include": [], "retargeting\_tags\_exclude": [], "action\_categories\_select": "", "interest\_categories\_select": "", "crowd": "1", "interest\_action\_mode": "UNLIMITED", "action\_scene": [], "action\_days": *30*, "action\_categories": [], "action\_words": [], "interest\_categories": [], "interest\_words": [], "superior\_popularity\_type": "NONE", "device\_type": ["MOBILE"], "ac": ["WIFI"], "hide\_if\_exists": "UNLIMITED", "hide\_if\_converted": "APP", "converted\_time\_duration": "TODAY", "android\_osv": "0.0", "ios\_osv": "0.0", "carrier": [], "device\_brand": [], "auto\_extend\_enabled": *0*, "auto\_extend\_targets": [], "model\_name": "pytest\_add", "model\_desc": "pytest\u65b0\u5efa"}}}  
[2024-11-01 19:07:38,278][*rest\_client.py 65*][DEBUG]: 请求地址：*https://test-admax.yostar.net/api/orientationModule/addOrEdit*[2024-11-01 19:07:38,457][*rest\_client.py 68*][DEBUG]: 返回结果：{'code': *200*, 'data': {}, 'msg': 'OK'}  
[2024-11-01 19:07:38,469][*rest\_client.py 46*][DEBUG]: 请求头：{'Authorization': '{"Head": {"token": "f11b556a0a15636392b54e395831bb5576874f76", "region": "CN", "version": "1.0", "time": 1730459258}}'}  
[2024-11-01 19:07:38,469][*rest\_client.py 47*][DEBUG]: 请求参数：{"params": {"channel\_id": "OceanEngine", "model\_name": "Py\_add\_63"}}  
[2024-11-01 19:07:38,470][*rest\_client.py 48*][DEBUG]: 请求地址：*https://test-admax.yostar.net/api/orientationModule/list*[2024-11-01 19:07:38,601][*rest\_client.py 51*][DEBUG]: 返回结果：{'code': *200*, 'data': {'list': [{'id': *262*, 'channel\_id': 'OceanEngine', 'user\_name': '张顺', 'model\_name': 'Py\_add\_63', 'model\_desc': 'pytest新建'}], 'paginate': {'page': *1*, 'size': *20*, 'total\_number': *1*, 'total\_page': *1*}}, 'msg': 'OK'}  
[2024-11-01 19:07:38,605][*mysql\_util.py 31*][INFO]: 开始执行sql:  
 SELECT id  
 FROM *admax\_test.channel\_orientation\_model* WHERE model\_name LIKE 'Py\_add\_%'  
 AND deleted\_at IS NULL  
 LIMIT *1*[2024-11-01 19:07:38,610][*conftest.py 23*][INFO]: SQL执行结果: [{'id': *262*}]  
[2024-11-01 19:07:38,619][*rest\_client.py 63*][DEBUG]: 请求头：{'Authorization': '{"Head": {"token": "f11b556a0a15636392b54e395831bb5576874f76", "region": "CN", "version": "1.0", "time": 1730459258}}'}  
[2024-11-01 19:07:38,620][*rest\_client.py 64*][DEBUG]: 请求参数：{"json": {"id": *262*, "channel\_id": "OceanEngine", "model\_name": "Py\_edit\_24", "model\_desc": "\u65b0\u5efa\u540e\u7f16\u8f91\u6821\u9a8c", "content": {"channel\_id": "OceanEngine", "district": "NONE", "region\_version": "1.0.0", "city": [], "location\_type": "ALL", "district\_select": "", "gender": "NONE", "age": [], "retargeting\_tags\_include": [], "retargeting\_tags\_exclude": [], "action\_categories\_select": "", "interest\_categories\_select": "", "crowd": "1", "interest\_action\_mode": "UNLIMITED", "action\_scene": [], "action\_days": *30*, "action\_categories": [], "action\_words": [], "interest\_categories": [], "interest\_words": [], "aweme\_fan\_behaviors": [], "aweme\_fan\_time\_scope": "", "aweme\_fan\_categories": [], "aweme\_fan\_accounts": [], "superior\_popularity\_type": "NONE", "device\_type": ["MOBILE"], "ac": ["WIFI"], "hide\_if\_exists": "UNLIMITED", "hide\_if\_converted": "APP", "converted\_time\_duration": "TODAY", "android\_osv": "0.0", "ios\_osv": "0.0", "carrier": [], "device\_brand": [], "auto\_extend\_enabled": *0*, "auto\_extend\_targets": [], "model\_name": "py\_Edit", "model\_desc": "\u65b0\u5efa\u540e\u7f16\u8f91\u6821\u9a8c", "id": *179*}}}  
[2024-11-01 19:07:38,620][*rest\_client.py 65*][DEBUG]: 请求地址：*https://test-admax.yostar.net/api/orientationModule/addOrEdit*[2024-11-01 19:07:38,779][*rest\_client.py 68*][DEBUG]: 返回结果：{'code': *200*, 'data': {}, 'msg': 'OK'}  
[2024-11-01 19:07:38,791][*rest\_client.py 46*][DEBUG]: 请求头：{'Authorization': '{"Head": {"token": "f11b556a0a15636392b54e395831bb5576874f76", "region": "CN", "version": "1.0", "time": 1730459258}}'}  
[2024-11-01 19:07:38,791][*rest\_client.py 47*][DEBUG]: 请求参数：{"params": {"channel\_id": "OceanEngine", "model\_name": "py\_Edit"}}  
[2024-11-01 19:07:38,791][*rest\_client.py 48*][DEBUG]: 请求地址：*https://test-admax.yostar.net/api/orientationModule/list*[2024-11-01 19:07:38,955][*rest\_client.py 51*][DEBUG]: 返回结果：{'code': *200*, 'data': {'list': [{'id': *262*, 'channel\_id': 'OceanEngine', 'user\_name': '张顺', 'model\_name': 'Py\_edit\_24', 'model\_desc': '新建后编辑校验'}], 'paginate': {'page': *1*, 'size': *20*, 'total\_number': *1*, 'total\_page': *1*}}, 'msg': 'OK'}  
[2024-11-01 19:07:38,958][*mysql\_util.py 31*][INFO]: 开始执行sql:  
 SELECT id  
 FROM *admax\_test.channel\_orientation\_model* WHERE model\_name LIKE 'Py\_edit\_%'  
 AND deleted\_at IS NULL  
 LIMIT *1*[2024-11-01 19:07:38,964][*conftest.py 51*][INFO]: SQL执行结果: [{'id': *262*}]  
[2024-11-01 19:07:38,973][*rest\_client.py 63*][DEBUG]: 请求头：{'Authorization': '{"Head": {"token": "f11b556a0a15636392b54e395831bb5576874f76", "region": "CN", "version": "1.0", "time": 1730459258}}'}  
[2024-11-01 19:07:38,973][*rest\_client.py 64*][DEBUG]: 请求参数：{"json": {"copy\_id": *262*, "model\_name": "Py\_copy\_14", "model\_desc": "\u4ecepy\_Edit\u590d\u5236\u7684\u6a21\u677f"}}  
[2024-11-01 19:07:38,973][*rest\_client.py 65*][DEBUG]: 请求地址：*https://test-admax.yostar.net/api/orientationModule/copy*[2024-11-01 19:07:39,141][*rest\_client.py 68*][DEBUG]: 返回结果：{'code': *200*, 'data': {}, 'msg': 'OK'}  
[2024-11-01 19:07:39,153][*rest\_client.py 46*][DEBUG]: 请求头：{'Authorization': '{"Head": {"token": "f11b556a0a15636392b54e395831bb5576874f76", "region": "CN", "version": "1.0", "time": 1730459258}}'}  
[2024-11-01 19:07:39,153][*rest\_client.py 47*][DEBUG]: 请求参数：{"params": {"channel\_id": "OceanEngine", "model\_name": "Py\_copy\_14"}}  
[2024-11-01 19:07:39,153][*rest\_client.py 48*][DEBUG]: 请求地址：*https://test-admax.yostar.net/api/orientationModule/list*[2024-11-01 19:07:39,290][*rest\_client.py 51*][DEBUG]: 返回结果：{'code': *200*, 'data': {'list': [{'id': *263*, 'channel\_id': 'OceanEngine', 'user\_name': '张顺', 'model\_name': 'Py\_copy\_14', 'model\_desc': '从py\_Edit复制的模板'}], 'paginate': {'page': *1*, 'size': *20*, 'total\_number': *1*, 'total\_page': *1*}}, 'msg': 'OK'}  
[2024-11-01 19:07:39,293][*mysql\_util.py 31*][INFO]: 开始执行sql:  
 SELECT id  
 FROM *admax\_test.channel\_orientation\_model* WHERE model\_name LIKE 'Py\_edit\_%'  
 AND deleted\_at IS NULL  
 LIMIT *1*[2024-11-01 19:07:39,306][*conftest.py 51*][INFO]: SQL执行结果: [{'id': *262*}]  
[2024-11-01 19:07:39,306][*mysql\_util.py 31*][INFO]: 开始执行sql:  
 SELECT id  
 FROM *admax\_test.channel\_orientation\_model* WHERE model\_name LIKE 'Py\_copy\_%'  
 AND deleted\_at IS NULL  
 LIMIT *1*[2024-11-01 19:07:39,312][*conftest.py 78*][INFO]: SQL执行结果: [{'id': *263*}]  
[2024-11-01 19:07:39,320][*rest\_client.py 63*][DEBUG]: 请求头：{'Authorization': '{"Head": {"token": "f11b556a0a15636392b54e395831bb5576874f76", "region": "CN", "version": "1.0", "time": 1730459258}}'}  
[2024-11-01 19:07:39,321][*rest\_client.py 64*][DEBUG]: 请求参数：{"json": {"channel\_id": "OceanEngine", "id": *262*}}  
[2024-11-01 19:07:39,321][*rest\_client.py 65*][DEBUG]: 请求地址：*https://test-admax.yostar.net/api/orientationModule/delete*[2024-11-01 19:07:39,467][*rest\_client.py 68*][DEBUG]: 返回结果：{'code': *200*, 'data': {}, 'msg': 'OK'}  
[2024-11-01 19:07:39,477][*rest\_client.py 63*][DEBUG]: 请求头：{'Authorization': '{"Head": {"token": "f11b556a0a15636392b54e395831bb5576874f76", "region": "CN", "version": "1.0", "time": 1730459258}}'}  
[2024-11-01 19:07:39,477][*rest\_client.py 64*][DEBUG]: 请求参数：{"json": {"channel\_id": "OceanEngine", "id": *263*}}  
[2024-11-01 19:07:39,477][*rest\_client.py 65*][DEBUG]: 请求地址：*https://test-admax.yostar.net/api/orientationModule/delete*[2024-11-01 19:07:39,624][*rest\_client.py 68*][DEBUG]: 返回结果：{'code': *200*, 'data': {}, 'msg': 'OK'}