vimport socket

import threading

import json

# 存储用户信息：用户名:密码

users = {

"user1": "pass1",

"user2": "pass2"

}

# 存储已登录用户：用户名:客户端连接

logged\_in\_users = {}

def handle\_client(client\_socket, addr):

print(f"新客户端连接: {addr}")

try:

# 接收登录信息

login\_data = client\_socket.recv(1024).decode()

if not login\_data:

return

# 解析登录信息

try:

login\_info = json.loads(login\_data)

username = login\_info.get("username")

password = login\_info.get("password")

# 验证用户

if username in users and users[username] == password:

# 登录成功

logged\_in\_users[username] = client\_socket

client\_socket.send(json.dumps({"status": "success", "message": "登录成功"}).encode())

print(f"用户 {username} 登录成功")

# 处理消息接收

while True:

data = client\_socket.recv(1024).decode()

if not data:

break

try:

message = json.loads(data)

target = message.get("target")

content = message.get("content")

if target in logged\_in\_users:

# 转发消息给目标用户

target\_socket = logged\_in\_users[target]

target\_socket.send(json.dumps({

"from": username,

"content": content

}).encode())

print(f"转发消息: {username} -> {target}: {content}")

else:

client\_socket.send(json.dumps({

"status": "error",

"message": f"用户 {target} 未在线"

}).encode())

except json.JSONDecodeError:

print("接收到无效的JSON数据")

else:

# 登录失败

client\_socket.send(json.dumps({"status": "error", "message": "用户名或密码错误"}).encode())

print(f"用户登录失败: {username}")

except json.JSONDecodeError:

print("接收到无效的登录JSON数据")

except Exception as e:

print(f"客户端处理错误: {e}")

finally:

# 客户端断开连接，移除登录状态

for username, socket in logged\_in\_users.items():

if socket == client\_socket:

del logged\_in\_users[username]

print(f"用户 {username} 断开连接")

break

client\_socket.close()

def start\_server():

# 创建TCP socket

server\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

# 绑定地址和端口

server\_address = ('localhost', 8888)

server\_socket.bind(server\_address)

# 开始监听

server\_socket.listen(5)

print(f"服务器启动，监听地址: {server\_address[0]}:{server\_address[1]}")

try:

while True:

# 接受客户端连接

client\_socket, addr = server\_socket.accept()

# 为每个客户端创建新线程处理

client\_thread = threading.Thread(target=handle\_client, args=(client\_socket, addr))

client\_thread.daemon = True

client\_thread.start()

except KeyboardInterrupt:

print("服务器正在关闭...")

finally:

# 关闭服务器socket

server\_socket.close()

if \_\_name\_\_ == "\_\_main\_\_":

start\_server()

import socket

import threading

import json

import time

class QQClient:

def \_\_init\_\_(self, host='localhost', port=8888):

self.host = host

self.port = port

self.socket = None

self.username = None

self.running = False

def connect(self):

try:

# 创建socket并连接到服务器

self.socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

self.socket.connect((self.host, self.port))

print(f"已连接到服务器: {self.host}:{self.port}")

return True

except Exception as e:

print(f"连接服务器失败: {e}")

return False

def login(self, username, password):

if not self.socket:

if not self.connect():

return False

self.username = username

# 发送登录信息

login\_info = {"username": username, "password": password}

self.socket.send(json.dumps(login\_info).encode())

# 接收登录结果

try:

response = self.socket.recv(1024).decode()

result = json.loads(response)

if result["status"] == "success":

print(result["message"])

# 启动消息接收线程

self.running = True

receive\_thread = threading.Thread(target=self.receive\_messages)

receive\_thread.daemon = True

receive\_thread.start()

return True

else:

print(result["message"])

return False

except Exception as e:

print(f"登录处理错误: {e}")

return False

def send\_message(self, target, content):

if not self.socket or not self.username:

print("请先登录")

return

# 构建消息

message = {"target": target, "content": content}

try:

self.socket.send(json.dumps(message).encode())

print(f"[我] -> {target}: {content}")

except Exception as e:

print(f"发送消息失败: {e}")

def receive\_messages(self):

while self.running:

try:

data = self.socket.recv(1024).decode()

if not data:

break

try:

message = json.loads(data)

sender = message.get("from", "未知用户")

content = message.get("content", "")

print(f"\n[{sender}] -> 我: {content}")

print(f"\n请输入消息(目标用户, 消息内容，用空格分隔): ")

except json.JSONDecodeError:

print("接收到无效的消息格式")

except Exception as e:

print(f"接收消息错误: {e}")

break

# 断开连接

self.running = False

self.socket.close()

self.socket = None

print("与服务器的连接已断开")

def close(self):

self.running = False

if self.socket:

self.socket.close()

self.socket = None

def main():

client = QQClient()

# 登录流程

while True:

username = input("请输入用户名(user1或user2): ")

password = input("请输入密码: ")

if client.login(username, password):

break

else:

choice = input("登录失败，是否重试?(y/n): ")

if choice.lower() != 'y':

return

# 消息发送流程

print("\n聊天开始，输入格式: 目标用户 消息内容")

print("输入'exit'退出程序")

while client.running:

try:

input\_str = input("\n请输入消息(目标用户, 消息内容，用空格分隔): ")

if input\_str.lower() == 'exit':

client.running = False

break

# 分割目标用户和消息内容

parts = input\_str.split(' ', 1)

if len(parts) < 2:

print("输入格式错误，请使用'目标用户 消息内容'格式")

continue

target, content = parts

client.send\_message(target, content)

except KeyboardInterrupt:

print("\n正在退出...")

client.running = False

break

client.close()

if \_\_name\_\_ == "\_\_main\_\_":

main()