1. server.py

```
7. 附录
7.1 clien.py
import ftplib
import pymysql
from datetime import datetime

class client():
    def __init__(self):
        self.c_connect_sql()#连接数据库
        self.host='localhost'
        self.port=22
        self.username='yeye'#初始用户名
        self.password='123456'

        # 客户端连接数据库
        self.conn_client=self.conn_client

        #创建游标对象
        self.cursor = self.cursor


        #标识客户端是否已经登陆
        self.log=False

        self.ftp=ftplib.FTP()#ftp对象

        # self.remote_file_path='./remote_file_path'

    """
    1.连接数据库+创建游标对象
    """
    def c_connect_sql(self):
        self.conn_client = pymysql.connect(
            host='localhost',
            port=3306,
            user='root',
            password='lx030312',
            database='ftp_info',
            charset='utf8mb4'
        )
        # 创建游标对象
        self.cursor = self.conn_client.cursor()

    """
    # 先启动服务器之后再连接FTP服务器
    # 2.连接FTP服务器
    """
    def connect_ftp(self,host,port,username,password):
        self.ftp.connect(host, port)
        self.ftp.login(username, password)
        # 返回FTP连接对象
        return self.ftp



    """
    3.一个注册功能
    """
    def sign_up(self,username,password):
        self.c_connect_sql()#连接数据库
        self.username=username
        self.password=password
        if self.username=='' or self.password=='':
            return '用户名为空或密码可能为空'

        #查询数据库中是否有该用户名
        """
        fetchall函数
        返回一个由查询结果组成的列表，其中每一项都是一个元组，代表一条记录，会把查到的那一行的
        数据都返回，例如(‘yeyelcy’,'123456')，如果存在他会把账号加密码均放到一个元组中并返回
        查询不到则返回空元组
        """
        self.cursor.execute('SELECT * FROM users WHERE username=%s',(username))
        tmp_is=self.cursor.fetchall()#fetchall用来一次性查询结果

        # 如果用户名存在则返回True，如果不存在则返回false
        if bool(tmp_is):
            print('用户名已经存在，请勿重复注册')
            return '用户名已经存在，请勿重复注册'
        else:
            self.cursor.execute('INSERT INTO users(username,password) VALUES(%s,%s)',(self.username,self.password))

        self.conn_client.commit()#提交
        # self.conn_client.close()#关闭

    """
    4.一个登陆功能
    """
    def sign_in(self,HOST,PORT,username,password):
        self.c_connect_sql()  # 连接数据库

        """
        登陆成功后就将用户名和密码赋值给了client对象的属性
        """
        self.username = username
        self.password = password
        if self.username == '' or self.password == '':
            print('用户名为空或密码可能为空')
        else:
            self.cursor.execute('SELECT * FROM users WHERE username=%s',(username))
        tmp_is = self.cursor.fetchall()  # fetchall用来一次性查询结果

            # 如果用户名存在则返回True，如果不存在则返回false
            if bool(tmp_is):
                #如果存在账户，则进行连接FTP请求
                #账号密码正确且均为字符串类型
                self.connect_ftp(HOST,PORT,self.username,self.password)
                self.log=True#标识已经登陆
                self.ftp.set_pasv(True)
                return self.ftp
            else:
                print('查无此用户')


    """
    5. 上传文件功能，remote_path指的是目标文件夹中的文件，local_path指的是下载到本地的文件夹
    """

    def ftp_upload(self,ftp,local_file_path):
        if self.log:
            # 打开本地文件并上传到FTP服务器
            a = local_file_path.split('/')  # a是要上传的文件名
            print(f'要上传的文件名为{a}')
            with open(local_file_path, 'rb') as fp:
                #这个ftp默认如果已经登陆的话就会被拥有
                ftp.storbinary('STOR ' + a[-1], fp)  # 往服务器中上传文件
        else:
            print('请您登陆之后再操作')



        # 记录上传信息到数据库
        op='up_load'
        now = datetime.now().strftime("%Y-%m-%d %H:%M:%S") #获取当前时间
        now=str(now)
        # print(type(now))
        print([self.username,a[-1], op, now])#注意a[-1]才是文件名
        a[-1]=str(a[-1])
        self.conn_client.autocommit(True)
        self.cursor.execute('INSERT INTO downloads (username,filename,op,time) VALUES (%s, %s, %s, %s)', (self.username, a[-1], op, now))

        print("upload  successfully")
        self.conn_client.commit()
        self.conn_client.close()  # 关闭

        # 关闭FTP连接
        # self.ftp.quit()
    """
    6. 下载文件功能
    """

    #LOCAL_FILE_PATH是文件要下载到的文件夹
```

```
        def ftp_download(self,ftp,local_file_path, remote_file_path):

            if self.log:

                # 打开本地文件并从FTP服务器下载
                a = remote_file_path.split('/')
                # print(local_file_path + '/' + a[-1])
                # print(remote_file_path)

                with open(local_file_path + '/' + a[-1], 'wb') as fp:
                    # print('what?')
                    ftp.retrbinary('RETR ' + remote_file_path, fp.write)
                # print('hello?')
            else:
                print('请你登陆后再操作')



            #记录下载/上传文件信息
            op='down_load'
            now = datetime.now()
            formatted = now.strftime("%Y-%m-%d %H:%M:%S")
            # print(type(formatted))
            self.cursor.execute('INSERT INTO downloads(username, filename, op,time) VALUES (%s, %s, %s, %s)',
                                 (self.username, a[-1], op,formatted))

            print(f"Downlload file {remote_file_path}: successfully")

            # print("download  successfully")
            self.conn_client.commit()

            # 关闭FTP连接
            # self.ftp.quit()

    if __name__=='__main__':

          test=client()
          ftp=test.sign_in('172.19.96.1',22,'yeyelcy','123456')#登陆
          test.ftp_upload(ftp,'./local_file_path/周杰伦-告白气球.mp3')#上传文件

          # # test.ftp_download(ftp,'./local_file_path','周杰伦-告白气球.mp3')#下载文件
```

7.2 server.py

```
import pymysql
import os
from pyftpdlib.authorizers import DummyAuthorizer
from pyftpdlib.handlers import FTPHandler
from pyftpdlib.servers import FTPServer


class MyHandler(FTPHandler):

    def on_file_received(self, file):
        # 当有文件上传时，会触发该函数
        print('File %s has been received.' % file)

    def on_file_sent(self, file):
        # 当有文件下载时，会触发该函数
        print('File %s has been sent.' % file)
class server():
    def __init__(self):
        #首先连接数数据：
        self.s_connect_sql()
        # 设置FTP服务器的根目录
        self.FTP_ROOT = './ftp_root'
        # 如果不存在就创建一个文件夹
        if not os.path.exists(self.FTP_ROOT):
            os.makedirs(self.FTP_ROOT)

        # 设置FTP服务器的IP和端口
        self.HOST = 'localhost'
        self.PORT = 22

        # 客户端连接数据库
        self.conn_client = self.conn_client

        # 创建游标对象
        self.cursor = self.cursor

        #使用DummyAuthorizer类来管理用户账号和权限
        self.authorizer=DummyAuthorizer()

        # 创建FTP服务器的处理器
        self.handler=FTPHandler

    """2.连接数据库"""
    def s_connect_sql(self):
        self.conn_client = pymysql.connect(
            host='localhost',
            port=3306,
            user='root',
            password='lx030312',
            database='ftp_info',
            charset='utf8mb4'
        )
        # 创建游标对象
        self.cursor = self.conn_client.cursor()

    #向FTP添加可以登陆的用户信息
    """3.添加用户信息"""
    def add_author(self):
        self.s_connect_sql()  # 连接数据库
        self.cursor.execute('SELECT * FROM users')#查询用户表中所有数据
        tmp_is = self.cursor.fetchall()#获取用户表中所有数据

        """
        FTP_ROOT：表示要添加的用户的根目录路径，类型为字符串
        第四个参数表示要添加的用户的权限，类型为字符串。默认值为 'elradfmwMT'，
        表示该用户拥有所有权限，包括读取、删除、创建、修改、移动和重命名文件等。
        """
        ## 设置FTP服务器的登录用户和密码
        for i in tmp_is:
            self.authorizer.add_user(i[1], i[2], self.FTP_ROOT, perm='elradfmwMT')
        self.handler.authorizer = self.authorizer

    """4.查看所有有权限的用户"""
    def check_users(self):
        for user in self.authorizer.user_table:
            print(user)

    """5. 运行FTP服务器"""
    def login(self):
        load_handler = MyHandler#监听文件的上传与否
        load_handler.authorizer = self.authorizer
        # 创建FTP服务器实例并启动
        server = FTPServer((self.HOST, self.PORT), self.handler)
        print('Starting FTP server on %s:%s' % (self.HOST, self.PORT))
        server.serve_forever()


if __name__=='__main__':
    s_test=server()
    s_test.add_author()
    s_test.login()
```

7.3 win_login.py

```
from PyQt5.QtWidgets import QApplication, QWidget, QLabel, QLineEdit, QPushButton, \
    QHBoxLayout, QVBoxLayout, QGroupBox,QSizePolicy
from PyQt5.QtGui import QIcon, QPixmap
from PyQt5.QtCore import Qt, QSize

from client import client
from win_client import MainWindow

# from win_main import MainWindow
import sys


class LoginWindow(QWidget):
    def __init__(self):
        super().__init__()

        # 设置窗口标题和大小
        self.setWindowTitle('Login')
        self.resize(400, 600)

        # 设定背景图和整体布局占比
        background = QLabel(self)
        pixmap = QPixmap('background.jpg')
        background.setPixmap(pixmap)
        self.resize(pixmap.width(), pixmap.height())

        # 创建控件
        logo_label = QLabel(self)
        logo_label.setPixmap(QPixmap('./WIN_ico/logo.png'))
        logo_label.setAlignment(Qt.AlignCenter)

        title_label = QLabel(self)
        title_label.setText('Sign in to FTP')
        title_label.setAlignment(Qt.AlignCenter)


        #用户名输入框
        self.username_edit = QLineEdit(self)
        self.username_edit.setPlaceholderText('Enter your username')
        self.username_edit.setText('yeyelcy')

        #密码输入框
        self.password_edit = QLineEdit(self)
        self.password_edit.setEchoMode(QLineEdit.Password)#设置密码不可见
        self.password_edit.setPlaceholderText('Enter your password')
        self.password_edit.setText('123456')

        # 添加服务器 IP 地址输入框
        self.ip_label = QLabel(self)
        self.ip_label.setText('Server IP:')
        self.ip_edit = QLineEdit(self)
        self.ip_edit.setPlaceholderText('Enter the server IP address')
        self.ip_edit.setText('localhost')  # 设置默认值

        # 添加端口号输入框
        self.port_label = QLabel(self)
        self.port_label.setText('Port:')
        self.port_edit = QLineEdit(self)
        self.port_edit.setPlaceholderText('Enter the port number')
        self.port_edit.setText('22')



        #登录按钮
        login_btn = QPushButton('Login', self)
        login_btn.clicked.connect(self.login_clint)

        #注册按钮
        register_btn = QPushButton('Register', self)
        register_btn.clicked.connect(self.login_up)

        btn_group = QGroupBox(self)
        btn_layout = QHBoxLayout(btn_group)
        btn_layout.addWidget(login_btn)
        btn_layout.addWidget(register_btn)

        # 构建主布局，并设置左右两边的间距
        main_layout = QVBoxLayout(self)
        main_layout.addStretch(3)
        main_layout.addWidget(logo_label, stretch=5)
        main_layout.addWidget(title_label, stretch=1)
        main_layout.addStretch(3)
```

```
    main_layout.addWidget(self.username_edit, stretch=1)
    main_layout.addWidget(self.password_edit, stretch=1)
    main_layout.addWidget(btn_group, stretch=1)
    main_layout.addStretch(4)

    ip_port_layout = QHBoxLayout()
    ip_port_layout.addWidget(self.ip_label)
    ip_port_layout.addWidget(self.ip_edit)
    ip_port_layout.addWidget(self.port_label)
    ip_port_layout.addWidget(self.port_edit)
    main_layout.addLayout(ip_port_layout)

    main_layout.addWidget(btn_group, stretch=1)
    main_layout.addStretch(4)


    # 绑定提示文本
        self.username_edit.setToolTip('Please enter your username')
        self.password_edit.setToolTip('Please enter your password')




    def show_login(self):
        self.show()


    #登陆客户端代码
    def login_clint(self):
        self.sign_in = client()  # 会自动指定端口号
        self.username = self.username_edit.text()
        self.password = self.password_edit.text()
        self.host = self.ip_edit.text()
        self.port = int(self.port_edit.text())
        print('账号密码为：')
        print(self.username, self.password)
        print("ip和端口号为")
        print(self.host,self.port,self)
        a=self.sign_in.sign_in(self.host,self.port,self.username,self.password)#进行登陆

        #self.sign_in是一个client对象
        if self.sign_in.log:
            self.log_client=MainWindow()
            MainWindow.ftp = a
            MainWindow.user_log = self.sign_in
            #self.log_client.ftp=a # 为 下一个界面添加一个具有权限的服务器端ftp
            self.log_client.show()
        else:
            print('请登录重试')

    #注册客户端代码
    def login_up(self):
        self.sign_in=client()
        self.username = self.username_edit.text()
        self.password = self.password_edit.text()
        # self.host=self.ip_edit.text()
        # self.port=self.port_edit.text()
        print('账号密码为：')
        print(self.username, self.password)
        self.sign_in.sign_up(self.username, self.password)  # 进行登陆

if __name__ == '__main__':
    app = QApplication([])
    login_window = LoginWindow()
    login_window.show_login()
    app.exec_()
```

7.4 win_client.py

```
# -*- coding: utf-8 -*-


from PyQt5.QtWidgets import QPushButton
from PyQt5.QtWidgets import  QWidget, QVBoxLayout, QPushButton, QLabel,QFileDialog,QMainWindow,QApplication
from PyQt5 import QtCore, QtWidgets
import sys
from client import client
import ftplib




class UploadWindow(QWidget):
    def __init__(self):
        super().__init__()


        # 创建控件
        label = QLabel('This is the upload window.')
        upload_button = QPushButton('Upload')
        layout = QVBoxLayout(self)
        layout.addWidget(label)
        layout.addWidget(upload_button)

class DownloadWindow(QWidget):
    def __init__(self):
        super().__init__()

        # 创建控件
        label = QLabel('This is the download window.')
        download_button = QPushButton('Download')
        layout = QVBoxLayout(self)
        layout.addWidget(label)
        layout.addWidget(download_button)

class AccountWindow(QWidget):
    def __init__(self):
        super().__init__()

        # 创建控件
        label = QLabel('This is the account management window.')
        manage_button = QPushButton('Manage')
        layout = QVBoxLayout(self)
        layout.addWidget(label)
        layout.addWidget(manage_button)


class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName("MainWindow")
        MainWindow.resize(582, 384)

        # 创建主控件
        self.centralwidget = QtWidgets.QWidget(MainWindow)
        self.centralwidget.setMinimumSize(QtCore.QSize(568, 379))
        self.centralwidget.setObjectName("centralwidget")

        # 创建水平布局器
        self.horizontalLayout_3 = QtWidgets.QHBoxLayout(self.centralwidget)
        self.horizontalLayout_3.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_3.setSpacing(0)
        self.horizontalLayout_3.setObjectName("horizontalLayout_3")

        # 创建分组框
        self.groupBox = QtWidgets.QGroupBox(self.centralwidget)
        self.groupBox.setMinimumSize(QtCore.QSize(568, 379))
        self.groupBox.setTitle("")
        self.groupBox.setObjectName("groupBox")

        # 创建垂直布局器
        self.verticalLayout = QtWidgets.QVBoxLayout(self.groupBox)
        self.verticalLayout.setContentsMargins(0, 0, 0, 0)
        self.verticalLayout.setSpacing(0)
        self.verticalLayout.setObjectName("verticalLayout")

        # 创建框架
        self.frame = QtWidgets.QFrame(self.groupBox)
        self.frame.setMinimumSize(QtCore.QSize(568, 45))
        self.frame.setFrameShape(QtWidgets.QFrame.StyledPanel)
        self.frame.setFrameShadow(QtWidgets.QFrame.Raised)
        self.frame.setObjectName("frame")

        # 创建水平布局器
        self.horizontalLayout_2 = QtWidgets.QHBoxLayout(self.frame)
        self.horizontalLayout_2.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_2.setSpacing(0)
        self.horizontalLayout_2.setObjectName("horizontalLayout_2")

        # 创建水平布局器
        self.horizontalLayout = QtWidgets.QHBoxLayout()
        self.horizontalLayout.setSpacing(0)
        self.horizontalLayout.setObjectName("horizontalLayout")

        # 创建按钮1
        self.pushButton_1 = QtWidgets.QPushButton(self.frame)
        self.pushButton_1.setMinimumSize(QtCore.QSize(75, 23))
        self.pushButton_1.setObjectName("pushButton_1")
        self.horizontalLayout.addWidget(self.pushButton_1)

            # 创建按钮2
            self.pushButton_2 = QtWidgets.QPushButton(self.frame)
            self.pushButton_2.setMinimumSize(QtCore.QSize(75, 23))
            self.pushButton_2.setObjectName("pushButton_2")
            self.horizontalLayout.addWidget(self.pushButton_2)

            # 创建按钮3
            self.pushButton_3 = QtWidgets.QPushButton(self.frame)
            self.pushButton_3.setMinimumSize(QtCore.QSize(75, 23))
            self.pushButton_3.setObjectName("pushButton_3")
            self.horizontalLayout.addWidget(self.pushButton_3)


            self.horizontalLayout_2.addLayout(self.horizontalLayout)

            # 创建布局器项
            spacerItem = QtWidgets.QSpacerItem(348, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
            self.horizontalLayout_2.addItem(spacerItem)
            self.verticalLayout.addWidget(self.frame)

            #创建了一个堆叠窗口对象 stacked_widget
            self.stackedWidget = QtWidgets.QStackedWidget(self.groupBox)
            self.stackedWidget.setMinimumSize(QtCore.QSize(568, 334))
            self.stackedWidget.setObjectName("stackedWidget")

            # 添加页面1控件
            self.page = QtWidgets.QWidget()
            self.page.setObjectName("page")

            self.horizontalLayout_4 = QtWidgets.QHBoxLayout(self.page)
            self.horizontalLayout_4.setContentsMargins(0, 0, 0, 0)
            self.horizontalLayout_4.setSpacing(0)
            self.horizontalLayout_4.setObjectName("horizontalLayout_4")

            #添加一个选择文件的功能
            self.upload_button = QtWidgets.QPushButton(self.page)
            self.upload_button.clicked.connect(self.upload_file)
            self.horizontalLayout_4.addWidget(self.upload_button)
            self.stackedWidget.addWidget(self.page)

            # 添加页面2控件
            self.page_2 = QtWidgets.QWidget()
            self.page_2.setObjectName("page_2")
            self.horizontalLayout_5 = QtWidgets.QHBoxLayout(self.page_2)
            self.horizontalLayout_5.setContentsMargins(0, 0, 0, 0)
            self.horizontalLayout_5.setSpacing(0)
            self.horizontalLayout_5.setObjectName("horizontalLayout_5")


            self.download_button = QtWidgets.QPushButton(self.page_2)

            self.download_button.clicked.connect(self.download_file)
            self.horizontalLayout_5.addWidget(self.download_button)

            self.stackedWidget.addWidget(self.page_2)



            # 添加页面3控件
            self.page_3 = QtWidgets.QWidget()
            self.page_3.setObjectName("page_3")
            self.horizontalLayout_6 = QtWidgets.QHBoxLayout(self.page_2)
            self.horizontalLayout_6.setContentsMargins(0, 0, 0, 0)
            self.horizontalLayout_6.setSpacing(0)
            self.horizontalLayout_6.setObjectName("horizontalLayout_5")

            self.user_info=QtWidgets.QPushButton(self.page_3)
            self.user_info.clicked.connect(self.user_acount)
            self.user_info.setText('check on user info')
            self.horizontalLayout_6.addWidget(self.user_info)

            self.stackedWidget.addWidget(self.page_3)


            #添加整体的
            self.verticalLayout.addWidget(self.stackedWidget)
            self.horizontalLayout_3.addWidget(self.groupBox)
            MainWindow.setCentralWidget(self.centralwidget)

            self.retranslateUi(MainWindow)
            self.stackedWidget.setCurrentIndex(1)
            QtCore.QMetaObject.connectSlotsByName(MainWindow)

        def upload_file(self):
            # 打开文件选择对话框
            file_path, _ = QFileDialog.getOpenFileName(self, "Upload File", "", "All Files (*);;Text Files (*.txt)")
            print(file_path)

            # 如果有选择文件，执行上传操作
            if file_path:
                #self.up_o=client()
                # TODO self.up_o.ftp_upload(self,self.ftp,file_path)
                if MainWindow.ftp:
                    #self.up_o.ftp_upload(MainWindow.ftp,file_path)
                    MainWindow.user_log.ftp_upload(MainWindow.ftp,file_path)

                    # 执行上传操作
                    print("Upload file:", file_path)
                else:
                    print("报错")

        def download_file(self):
            # TODO: 执行下载操作
            down_path, _ = QFileDialog.getOpenFileName(self, "Save File", "", "All Files (*);;Text Files (*.txt)")

            print("Download file path:", down_path)
            down_path=down_path.split('/')[-1]


            # 如果选择了要下载的文件路径，执行保存文件的操作
            if down_path:
                if MainWindow.ftp:
                    MainWindow.user_log.ftp_download(MainWindow.ftp,'./local_file_path',down_path)

            else:
                print('报错')

        def user_acount(self):
            print("用户信息为",MainWindow.user_log.username)

        def retranslateUi(self, MainWindow):
            _translate = QtCore.QCoreApplication.translate
            MainWindow.setWindowTitle(_translate("MainWindow", "win_client"))
            self.pushButton_1.setText(_translate("MainWindow", "upload"))
            self.pushButton_2.setText(_translate("MainWindow", "download"))
            self.pushButton_3.setText(_translate("MainWindow", "account"))
            # self.pushButton_4.setText(_translate("MainWindow", "页面1")
            self.upload_button.setText(_translate("MainWindow", "upload_button"))
            # self.pushButton_5.setText(_translate("MainWindow", "页面2"))
            self.download_button.setText(_translate("MainWindow","Download File"))
    class MainWindow(QMainWindow, Ui_MainWindow):
        ftp = None
        user_log = None
        def __init__(self, parent=None):
            super(MainWindow, self).__init__(parent)
            self.setupUi(self)


            # self.ftp=ftplib.FTP()#ftp对象
            self.pushButton_1.clicked.connect(lambda: self.stackedWidget.setCurrentIndex(0))
            self.pushButton_2.clicked.connect(lambda: self.stackedWidget.setCurrentIndex(1))
            self.pushButton_3.clicked.connect(lambda: self.stackedWidget.setCurrentIndex(2))
    if __name__=='__main__':
        app = QApplication(sys.argv)
        myWin = MainWindow()
        myWin.show()
        sys.exit(app.exec_())
```

7.5 server_mysql.sql

```
--查看当前数据库
select database();
```

```

--创建一个数据库
create database if not exists ftp_info;

--选择一个数据库
use ftp_info;
-- 创建用户表
CREATE TABLE users (
    id INT(11) NOT NULL AUTO_INCREMENT,
    username VARCHAR(50) NOT NULL,
    password VARCHAR(50) NOT NULL,
    PRIMARY KEY (id)
);

-- 创建文件表
CREATE TABLE files (
    id INT(11) NOT NULL AUTO_INCREMENT,
    filename VARCHAR(50) NOT NULL,
    filepath VARCHAR(100) NOT NULL,
    PRIMARY KEY (id)
);

-- 创建下载日志表
CREATE TABLE downloads (
    id INT(11) NOT NULL AUTO_INCREMENT,
    username VARCHAR(50) NOT NULL,
    filename VARCHAR(50) NOT NULL,
    op VARCHAR(50) NOT NULL,
    time VARCHAR(50) NOT NULL,
    PRIMARY KEY (id)
);


--设置最大连接数
SET GLOBAL max_connections =100;
```

2. client.py

```
import ftplib
import pymysql
from datetime import datetime

class client():
    def __init__(self):
        self.c_connect_sql()#连接数据库
        self.host='localhost'
        self.port=22
        self.username='yeye'#初始用户名
        self.password='123456'

        # 客户端连接数据库
        self.conn_client=self.conn_client
        #创建游标对象
        self.cursor = self.cursor


        #标识客户端是否已经登陆
        self.log=False

        self.ftp=ftplib.FTP()#ftp对象

        # self.remote_file_path='./remote_file_path'

    """
    1.连接数据库+创建游标对象
    """
    def c_connect_sql(self):
        self.conn_client = pymysql.connect(
            host='localhost',
            port=3306,
            user='root',
            password='lx030312',
            database='ftp_info',
            charset='utf8mb4'
        )
        # 创建游标对象
        self.cursor = self.conn_client.cursor()

    """
    # 先启动服务器之后再连接FTP服务器
    # 2.连接FTP服务器
    """
    def connect_ftp(self,host,port,username,password):
        self.ftp.connect(host, port)
        self.ftp.login(username, password)
        # 返回FTP连接对象
        return self.ftp




    """
    3.一个注册功能
    """
    def sign_up(self,username,password):
        self.c_connect_sql()#连接数据库
        self.username=username
        self.password=password
        if self.username=='' or self.password=='':
            return '用户名为空或密码可能为空'

        #查询数据库中是否有该用户名
        """
        fetchall函数
        返回一个由查询结果组成的列表，其中每一项都是一个元组，代表一条记录，会把查到的那一行的
        数据都返回，例如(‘yeyelcy’,'123456')，如果存在他会把账号加密码均放到一个元组中并返回
        查询不到则返回空元组
        """
        self.cursor.execute('SELECT * FROM users WHERE username=%s',(username))
        tmp_is=self.cursor.fetchall()#fetchall用来一次性查询结果

        # 如果用户名存在则返回True，如果不存在则返回false
        if bool(tmp_is):
            print('用户名已经存在，请勿重复注册')
            return '用户名已经存在，请勿重复注册'
        else:
            self.cursor.execute('INSERT INTO users(username,password)
VALUES(%s,%s)',(self.username,self.password))

        self.conn_client.commit()#提交
        # self.conn_client.close()#关闭

    """
    4.一个登陆功能
    """
    def sign_in(self,HOST,PORT,username,password):
        self.c_connect_sql()  # 连接数据库

        """
        登陆成功后就将用户名和密码赋值给了client对象的属性
        """
        self.username = username
        self.password = password
        if self.username == '' or self.password == '':
            print('用户名为空或密码可能为空')
        else:
            self.cursor.execute('SELECT * FROM users WHERE username=%s',
(username))
            tmp_is = self.cursor.fetchall()  # fetchall用来一次性查询结果

            # 如果用户名存在则返回True，如果不存在则返回false
            if bool(tmp_is):
                #如果存在账户，则进行连接FTP请求
                #账号密码正确且均为字符串类型
                self.connect_ftp(HOST,PORT,self.username,self.password)
                self.log=True#标识已经登陆
                self.ftp.set_pasv(True)
                return self.ftp
            else:
                print('查无此用户')


    """
    5. 上传文件功能，remote_path指的是目标文件夹中的文件，local_path指的是下载到本地的文件
夹
    """

    def ftp_upload(self,ftp,local_file_path):
        if self.log:
            # 打开本地文件并上传到FTP服务器
            a = local_file_path.split('/')  # a是要上传的文件名
            print(f'要上传的文件名为{a}')
            with open(local_file_path, 'rb') as fp:
                #这个ftp默认如果已经登陆的话就会被拥有
                ftp.storbinary('STOR ' + a[-1], fp)  # 往服务器中上传文件
        else:
            print('请您登陆之后再操作')




        # 记录上传信息到数据库
        op='up_load'
        now = datetime.now().strftime("%Y-%m-%d %H:%M:%S") #获取当前时间
        now=str(now)
        # print(type(now))
        print([self.username,a[-1], op, now])#注意a[-1]才是文件名
        a[-1]=str(a[-1])
        self.conn_client.autocommit(True)
        self.cursor.execute('INSERT INTO downloads (username,filename,op,time)
VALUES (%s, %s, %s, %s)', (self.username, a[-1], op, now))

        print("upload  successfully")
        self.conn_client.commit()
        self.conn_client.close()  # 关闭

        # 关闭FTP连接
        # self.ftp.quit()
    """
    6. 下载文件功能
    """

    #LOCAL_FILE_PATH是文件要下载到的文件夹
    def ftp_download(self,ftp,local_file_path, remote_file_path):

        if self.log:

            # 打开本地文件并从FTP服务器下载
            a = remote_file_path.split('/')
            # print(local_file_path + '/' + a[-1])
            # print(remote_file_path)

            with open(local_file_path + '/' + a[-1], 'wb') as fp:
                # print('what?')
                ftp.retrbinary('RETR ' + remote_file_path, fp.write)
            # print('hello?')
        else:
            print('请你登陆后再操作')




        #记录下载/上传文件信息
        op='down_load'
        now = datetime.now()
        formatted = now.strftime("%Y-%m-%d %H:%M:%S")
        # print(type(formatted))
        self.cursor.execute('INSERT INTO downloads(username, filename, op,time) 
VALUES (%s, %s, %s, %s)',
                             (self.username, a[-1], op,formatted))

        print(f"Downlload file {remote_file_path}: successfully")

        # print("download  successfully")
        self.conn_client.commit()

        # 关闭FTP连接
        # self.ftp.quit()

if __name__=='__main__':

      test=client()
      ftp=test.sign_in('172.19.96.1',22,'yeyelcy','123456')#登陆
      test.ftp_upload(ftp,'./local_file_path/周杰伦-告白气球.mp3')#上传文件
      # # test.ftp_download(ftp,'./local_file_path','周杰伦-告白气球.mp3')#下载文件
```

3. win_login.py

```
from PyQt5.QtWidgets import QApplication, QWidget, QLabel, QLineEdit, 
QPushButton, \
    QHBoxLayout, QVBoxLayout, QGroupBox,QSizePolicy
from PyQt5.QtGui import QIcon, QPixmap
from PyQt5.QtCore import Qt, QSize

from client import client
from win_client import MainWindow

# from win_main import MainWindow
import sys


class LoginWindow(QWidget):
    def __init__(self):
        super().__init__()

        # 设置窗口标题和大小
        self.setWindowTitle('Login')
        self.resize(400, 600)

        # 设定背景图和整体布局占比
        background = QLabel(self)
        pixmap = QPixmap('background.jpg')
        background.setPixmap(pixmap)
        self.resize(pixmap.width(), pixmap.height())

        # 创建控件
        logo_label = QLabel(self)
        logo_label.setPixmap(QPixmap('./WIN_ico/logo.png'))
        logo_label.setAlignment(Qt.AlignCenter)

        title_label = QLabel(self)
        title_label.setText('Sign in to FTP')
        title_label.setAlignment(Qt.AlignCenter)


        #用户名输入框
        self.username_edit = QLineEdit(self)
        self.username_edit.setPlaceholderText('Enter your username')
        self.username_edit.setText('yeyelcy')

        #密码输入框
        self.password_edit = QLineEdit(self)
        self.password_edit.setEchoMode(QLineEdit.Password)#设置密码不可见
        self.password_edit.setPlaceholderText('Enter your password')
        self.password_edit.setText('123456')

        # 添加服务器 IP 地址输入框
        self.ip_label = QLabel(self)
        self.ip_label.setText('Server IP:')
        self.ip_edit = QLineEdit(self)
        self.ip_edit.setPlaceholderText('Enter the server IP address')
        self.ip_edit.setText('localhost')  # 设置默认值

        # 添加端口号输入框
        self.port_label = QLabel(self)
        self.port_label.setText('Port:')
        self.port_edit = QLineEdit(self)
        self.port_edit.setPlaceholderText('Enter the port number')
        self.port_edit.setText('22')



        #登录按钮
        login_btn = QPushButton('Login', self)
        login_btn.clicked.connect(self.login_clint)

        #注册按钮
        register_btn = QPushButton('Register', self)
        register_btn.clicked.connect(self.login_up)

        btn_group = QGroupBox(self)
        btn_layout = QHBoxLayout(btn_group)
        btn_layout.addWidget(login_btn)
        btn_layout.addWidget(register_btn)

        # 构建主布局，并设置左右两边的间距
        main_layout = QVBoxLayout(self)
        main_layout.addStretch(3)
        main_layout.addWidget(logo_label, stretch=5)
        main_layout.addWidget(title_label, stretch=1)
        main_layout.addStretch(3)
        main_layout.addWidget(self.username_edit, stretch=1)
        main_layout.addWidget(self.password_edit, stretch=1)
        main_layout.addWidget(btn_group, stretch=1)
        main_layout.addStretch(4)
        ip_port_layout = QHBoxLayout()
        ip_port_layout.addWidget(self.ip_label)
        ip_port_layout.addWidget(self.ip_edit)
        ip_port_layout.addWidget(self.port_label)
        ip_port_layout.addWidget(self.port_edit)
        main_layout.addLayout(ip_port_layout)

        main_layout.addWidget(btn_group, stretch=1)
        main_layout.addStretch(4)


        # 绑定提示文本
        self.username_edit.setToolTip('Please enter your username')
        self.password_edit.setToolTip('Please enter your password')




    def show_login(self):
        self.show()


    #登陆客户端代码
    def login_clint(self):
        self.sign_in = client()  # 会自动指定端口号
        self.username = self.username_edit.text()
        self.password = self.password_edit.text()
        self.host = self.ip_edit.text()
        self.port = int(self.port_edit.text())
        print('账号密码为：')
        print(self.username, self.password)
        print("ip和端口号为")
        print(self.host,self.port,self)
        a=self.sign_in.sign_in(self.host,self.port,self.username, self.password)#进行登陆

        #self.sign_in是一个client对象
        if self.sign_in.log:
            self.log_client=MainWindow()
            MainWindow.ftp = a
            MainWindow.user_log = self.sign_in
            #self.log_client.ftp=a # 为 下一个界面添加一个具有权限的服务器端ftp
            self.log_client.show()
        else:
            print('请登录重试')

    #注册客户端代码
    def login_up(self):
        self.sign_in=client()
        self.username = self.username_edit.text()
        self.password = self.password_edit.text()
        # self.host=self.ip_edit.text()
        # self.port=self.port_edit.text()
        print('账号密码为：')
        print(self.username, self.password)
        self.sign_in.sign_up(self.username, self.password)  # 进行登陆

if __name__ == '__main__':
    app = QApplication([])
    login_window = LoginWindow()
    login_window.show_login()
    app.exec_()
```

4. win_client.py

```
# -*- coding: utf-8 -*-

from PyQt5.QtWidgets import QPushButton
from PyQt5.QtWidgets import  QWidget, QVBoxLayout, QPushButton,
QLabel,QFileDialog,QMainWindow,QApplication
from PyQt5 import QtCore, QtWidgets
import sys
from client import client
import ftplib




class UploadWindow(QWidget):
    def __init__(self):
        super().__init__()


        # 创建控件
        label = QLabel('This is the upload window.')
        upload_button = QPushButton('Upload')
        layout = QVBoxLayout(self)
        layout.addWidget(label)
        layout.addWidget(upload_button)

class DownloadWindow(QWidget):
    def __init__(self):
        super().__init__()

        # 创建控件
        label = QLabel('This is the download window.')
        download_button = QPushButton('Download')
        layout = QVBoxLayout(self)
        layout.addWidget(label)
        layout.addWidget(download_button)

class AccountWindow(QWidget):
    def __init__(self):
        super().__init__()

        # 创建控件
        label = QLabel('This is the account management window.')
        manage_button = QPushButton('Manage')
        layout = QVBoxLayout(self)
        layout.addWidget(label)
        layout.addWidget(manage_button)


class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName("MainWindow")
        MainWindow.resize(582, 384)

        # 创建主控件
        self.centralwidget = QtWidgets.QWidget(MainWindow)
        self.centralwidget.setMinimumSize(QtCore.QSize(568, 379))
        self.centralwidget.setObjectName("centralwidget")

        # 创建水平布局器
        self.horizontalLayout_3 = QtWidgets.QHBoxLayout(self.centralwidget)
        self.horizontalLayout_3.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_3.setSpacing(0)
        self.horizontalLayout_3.setObjectName("horizontalLayout_3")

        # 创建分组框
        self.groupBox = QtWidgets.QGroupBox(self.centralwidget)
        self.groupBox.setMinimumSize(QtCore.QSize(568, 379))
        self.groupBox.setTitle("")
        self.groupBox.setObjectName("groupBox")

        # 创建垂直布局器
        self.verticalLayout = QtWidgets.QVBoxLayout(self.groupBox)
        self.verticalLayout.setContentsMargins(0, 0, 0, 0)
        self.verticalLayout.setSpacing(0)
        self.verticalLayout.setObjectName("verticalLayout")

        # 创建框架
        self.frame = QtWidgets.QFrame(self.groupBox)
        self.frame.setMinimumSize(QtCore.QSize(568, 45))
        self.frame.setFrameShape(QtWidgets.QFrame.StyledPanel)
        self.frame.setFrameShadow(QtWidgets.QFrame.Raised)
        self.frame.setObjectName("frame")

        # 创建水平布局器
        self.horizontalLayout_2 = QtWidgets.QHBoxLayout(self.frame)
        self.horizontalLayout_2.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_2.setSpacing(0)
        self.horizontalLayout_2.setObjectName("horizontalLayout_2")

        # 创建水平布局器
        self.horizontalLayout = QtWidgets.QHBoxLayout()
        self.horizontalLayout.setSpacing(0)
        self.horizontalLayout.setObjectName("horizontalLayout")

        # 创建按钮1
        self.pushButton_1 = QtWidgets.QPushButton(self.frame)
        self.pushButton_1.setMinimumSize(QtCore.QSize(75, 23))
        self.pushButton_1.setObjectName("pushButton_1")
        self.horizontalLayout.addWidget(self.pushButton_1)

        # 创建按钮2
        self.pushButton_2 = QtWidgets.QPushButton(self.frame)
        self.pushButton_2.setMinimumSize(QtCore.QSize(75, 23))
        self.pushButton_2.setObjectName("pushButton_2")
        self.horizontalLayout.addWidget(self.pushButton_2)

        # 创建按钮3
        self.pushButton_3 = QtWidgets.QPushButton(self.frame)
        self.pushButton_3.setMinimumSize(QtCore.QSize(75, 23))
        self.pushButton_3.setObjectName("pushButton_3")
        self.horizontalLayout.addWidget(self.pushButton_3)


        self.horizontalLayout_2.addLayout(self.horizontalLayout)

        # 创建布局器项
        spacerItem = QtWidgets.QSpacerItem(348, 20,
QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_2.addItem(spacerItem)
        self.verticalLayout.addWidget(self.frame)

        #创建了一个堆叠窗口对象 stacked_widget
        self.stackedWidget = QtWidgets.QStackedWidget(self.groupBox)
        self.stackedWidget.setMinimumSize(QtCore.QSize(568, 334))
        self.stackedWidget.setObjectName("stackedWidget")

        # 添加页面1控件
        self.page = QtWidgets.QWidget()
        self.page.setObjectName("page")

        self.horizontalLayout_4 = QtWidgets.QHBoxLayout(self.page)
        self.horizontalLayout_4.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_4.setSpacing(0)
        self.horizontalLayout_4.setObjectName("horizontalLayout_4")

        #添加一个选择文件的功能
        self.upload_button = QtWidgets.QPushButton(self.page)
        self.upload_button.clicked.connect(self.upload_file)
        self.horizontalLayout_4.addWidget(self.upload_button)
        self.stackedWidget.addWidget(self.page)

        # 添加页面2控件
        self.page_2 = QtWidgets.QWidget()
        self.page_2.setObjectName("page_2")
        self.horizontalLayout_5 = QtWidgets.QHBoxLayout(self.page_2)
        self.horizontalLayout_5.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_5.setSpacing(0)
        self.horizontalLayout_5.setObjectName("horizontalLayout_5")


        self.download_button = QtWidgets.QPushButton(self.page_2)

        self.download_button.clicked.connect(self.download_file)
        self.horizontalLayout_5.addWidget(self.download_button)

        self.stackedWidget.addWidget(self.page_2)



        # 添加页面3控件
        self.page_3 = QtWidgets.QWidget()
        self.page_3.setObjectName("page_3")
        self.horizontalLayout_6 = QtWidgets.QHBoxLayout(self.page_2)
        self.horizontalLayout_6.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_6.setSpacing(0)
        self.horizontalLayout_6.setObjectName("horizontalLayout_5")

        self.user_info=QtWidgets.QPushButton(self.page_3)
        self.user_info.clicked.connect(self.user_acount)
        self.user_info.setText('check on user info')
        self.horizontalLayout_6.addWidget(self.user_info)

        self.stackedWidget.addWidget(self.page_3)


        #添加整体的
        self.verticalLayout.addWidget(self.stackedWidget)
        self.horizontalLayout_3.addWidget(self.groupBox)
        MainWindow.setCentralWidget(self.centralwidget)

        self.retranslateUi(MainWindow)
        self.stackedWidget.setCurrentIndex(1)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)

    def upload_file(self):
        # 打开文件选择对话框
        file_path, _ = QFileDialog.getOpenFileName(self, "Upload File", "", "All Files (*);;Text Files (*.txt)")
        print(file_path)

        # 如果有选择文件，执行上传操作
        if file_path:
            #self.up_o=client()
            # TODO self.up_o.ftp_upload(self,self.ftp,file_path)
            if MainWindow.ftp:
                #self.up_o.ftp_upload(MainWindow.ftp,file_path)
                MainWindow.user_log.ftp_upload(MainWindow.ftp,file_path)
                # 执行上传操作
                print("Upload file:", file_path)
            else:
                print("报错")

    def download_file(self):
        # TODO: 执行下载操作
        down_path, _ = QFileDialog.getOpenFileName(self, "Save File", "", "All Files (*);;Text Files (*.txt)")

        print("Download file path:", down_path)
        down_path=down_path.split('/')[-1]


        # 如果选择了要下载的文件路径，执行保存文件的操作
        if down_path:
            if MainWindow.ftp:

 MainWindow.user_log.ftp_download(MainWindow.ftp,'./local_file_path',down_path)

        else:
            print('报错')

    def user_acount(self):
        print("用户信息为",MainWindow.user_log.username)

    def retranslateUi(self, MainWindow):
        _translate = QtCore.QCoreApplication.translate
        MainWindow.setWindowTitle(_translate("MainWindow", "win_client"))
        self.pushButton_1.setText(_translate("MainWindow", "upload"))
        self.pushButton_2.setText(_translate("MainWindow", "download"))
        self.pushButton_3.setText(_translate("MainWindow", "account"))
        # self.pushButton_4.setText(_translate("MainWindow", "页面1")
        self.upload_button.setText(_translate("MainWindow", "upload_button"))
        # self.pushButton_5.setText(_translate("MainWindow", "页面2"))
        self.download_button.setText(_translate("MainWindow","Download File"))
class MainWindow(QMainWindow, Ui_MainWindow):
    ftp = None
    user_log = None
    def __init__(self, parent=None):
        super(MainWindow, self).__init__(parent)
        self.setupUi(self)


        # self.ftp=ftplib.FTP()#ftp对象
        self.pushButton_1.clicked.connect(lambda:
self.stackedWidget.setCurrentIndex(0))
        self.pushButton_2.clicked.connect(lambda:
self.stackedWidget.setCurrentIndex(1))
        self.pushButton_3.clicked.connect(lambda:
self.stackedWidget.setCurrentIndex(2))
if __name__=='__main__':
    app = QApplication(sys.argv)
    myWin = MainWindow()
    myWin.show()
    sys.exit(app.exec_())
```

5. server_mysql.sql

```
--查看当前数据库
select database();

--创建一个数据库
create database if not exists ftp_info;
```

```sql
--选择一个数据库
use ftp_info;
-- 创建用户表
CREATE TABLE users (
    id INT(11) NOT NULL AUTO_INCREMENT,
    username VARCHAR(50) NOT NULL,
    password VARCHAR(50) NOT NULL,
    PRIMARY KEY (id)
);

-- 创建文件表
CREATE TABLE files (
    id INT(11) NOT NULL AUTO_INCREMENT,
    filename VARCHAR(50) NOT NULL,
    filepath VARCHAR(100) NOT NULL,
    PRIMARY KEY (id)
);

-- 创建下载日志表
CREATE TABLE downloads (
    id INT(11) NOT NULL AUTO_INCREMENT,
    username VARCHAR(50) NOT NULL,
    filename VARCHAR(50) NOT NULL,
    op VARCHAR(50) NOT NULL,
    time VARCHAR(50) NOT NULL,
    PRIMARY KEY (id)
);


--设置最大连接数
SET GLOBAL max_connections =100;
```