**《计算机网络》**

**课程设计报告**

（2022 —2023 学年第二学期）

题 目 客户端FTP软件的设计与实现

学生姓名 刘晨阳

专业班级 计算机2106

学生学号 312105010207

教师姓名 孟 慧

2023年6月23日

|  |
| --- |
| **河南理工大学计算机学院**  **计算机网络课程设计报告评价分值标准** |
| |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 序号 | 评价指标 | 分值 | 评价等级及参考分值 | | | | | 评价分 | | 优 | 良 | 中 | 合格 | 差 | | 1 | 报告能够体现学生熟练运用计算机网络的基本理论，结合客观实际情况，设计满足特定需求的计算机网络软硬件系统方案及相关模块、算法，并能够准确描述完整的实现原理和方法。 | 50 | ≥45 | ≥40 | ≥35 | ≥30 | ＜30 |  | | 2 | 报告能够体现学生结合计算机网络领域工程及应用问题，采用专业工具分析，选择相应开发技术实现计算机网络中的功能设计。 | 30 | ≥27 | ≥24 | ≥21 | ≥18 | ＜18 |  | | 3 | 报告书写规范，内容完整充实，按照课程设计报告的要求规范排版，特别是图表的规范设计。 | 20 | ≥18 | ≥16 | ≥14 | ≥12 | ＜12 |  | | 总得分 | | | | | | | |  |   签名（签章）：  日期： 年 月 日 |
|  |

目录

[1. FTP基本知识 6](#_Toc138478651)

[1.1 FTP的基本概念 6](#_Toc138478652)

[1.2 FTP的工作原理 6](#_Toc138478653)

[1.3 FTP的应用 6](#_Toc138478654)

[2. FTP基本原理 7](#_Toc138478655)

[2.1 FTP软件设计流程图 7](#_Toc138478656)

[2.2 FTP 程序设计原理 7](#_Toc138478657)

[2.3 程序设计文件部署与说明 8](#_Toc138478658)

[3. 基于Python的FTP软件核心代码 9](#_Toc138478659)

[3.1 FTP 核心功能实现 9](#_Toc138478660)

[3.2 MySQL数据库实现 13](#_Toc138478661)

[3.3 QT前端界面实现 14](#_Toc138478662)

[4. 运行与测试 15](#_Toc138478663)

[4.1 登陆与注册 15](#_Toc138478664)

[4.2文件的上传与下载 16](#_Toc138478665)

[4.3 用户操作的数据库记录 18](#_Toc138478666)

[5. 课程设计心得与体会 19](#_Toc138478667)

[6. 参考文献 20](#_Toc138478668)

[7. 附录 21](#_Toc138478669)

[7.1 clien.py 21](#_Toc138478670)

[7.2 server.py 26](#_Toc138478671)

[7.3 win\_login.py 28](#_Toc138478672)

[7.4 win\_client.py 32](#_Toc138478673)

[7.5 server\_mysql.sql 39](#_Toc138478674)

摘要

FTP作为最早应用于互联网的文件传输协议之一，在数据传输领域发挥着重要作用。很多企业、机构和个人通过FTP的方式传输和备份重要数据，以确保数据的安全性和可靠性。FTP也被用作软件和媒体文件的分发渠道，随着技术的发展FTP的安全性和性能也在不断提升，作为一种广泛应用的文件传输协议，具有实际的应用价值。因此本次课程设计选择开发实现一个客户端FTP软件。

该客户端FTP软件基于Python 、MySQL语言，基于Python3.10的环境，使用Pycharm环境进行设计开发，能够实现以下基本功能和任务： 下载功能。连接用户指定的FTP服务器，获取服务器目录下的文件列表，点击下载用户所需的文件，存储在用户想存储的地方；上传功能。用户可以自行选择本机上存储的文件，验证用户名和密码，在FTP服务器上进行注册。

最终实现了基于多线程的客户端FTP软件，其界面简洁友好，能够让用户根据提示输入指定的FTP服务器IP地址和端口号进行登陆，完成文件的上传和下载功能，很好地实现了目标功能。同时用户在服务器上的相关操作如下载和上传等行为及其时间会被记录在该服务器连接的对应数据库中，以便查询，很好的完成了目标功能

**关键词：**FTP客户端软件设计；ftplib库；pyqt5库；数据传输；MySQL数据库；

# 1. FTP基本知识

1.1 FTP的基本概念

FTP（File Transfer Protocol）是一种用于在计算机之间传输文件的标准网络协议。它定义了客户端和服务器之间进行文件传输和交互的规则和方法。

它采用的是一种C/S的客户端-服务器方式。FTP提供交互式的访问，允许客户指明文件的类型与格式，如指明是否使用ASCII码，并允许文件具有存取权限，如访问文件的用户必须经过授权，并输入有效的口令等，它屏蔽了各计算机系统的细节，因而适合于在异构网络中任意计算机之间传送文件。

FTP是一种简单而有效的协议，用于在计算机之间进行文件传输。它提供了一组命令和规则，使用户能够方便地管理和传输文件。通过FTP，用户可以将文件从一个计算机上传到另一个计算机，或者从远程服务器下载文件到本地计算机

1.2 FTP的工作原理

FTP通过客户端对向服务端发送FTP命令，并接受来自服务端的相应来完成文件的上传与下载，具体实现如下。客户端首先通过TCP/IP协议建立与服务器的连接，默认情况下，FTP使用标准端口号21进行控制连接，当有数据连接时，客户端通过命令通道与FTP服务端建立数据连接，这条TCP数据连接用于客户端与服务端之间的文件传输。

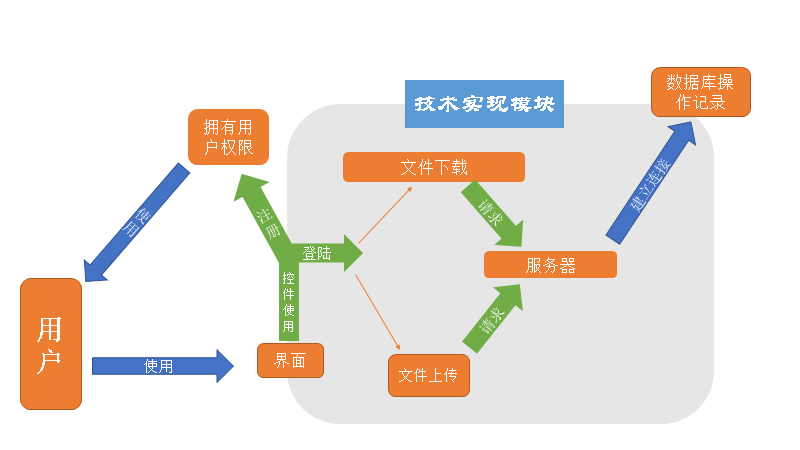
控制连接在整个会话期间一直保持打开，用于传送FTP相关控制命令，数据连接用于文件传输，在每次文件传输时才建立，传输结束就关闭。FTP通过控制连接和数据连接的方式完成客户端与服务端之间的文件传输，其工作原理简单而高效，使用户能够方便地管理和传输文件

1.3 FTP的应用

FTP的常见用途是在计算机之间传输文件，尤其是用于批量传输文件，此外还可以让网站设计者将构成网站内容的大量文件批量上传到他们的Web服务器，其应用范围广泛，涵盖了文件共享、网站维护、软件更新、大文件传输、数据备份等多个领域。它提供了一种简单、高效的方式来进行文件传输和访问

# 2. FTP基本原理

2.1 FTP软件设计流程图



2.2 FTP 程序设计原理

该FTP程序设计软件是基于Python语言实现的，整体从后端功能，前端界面以及MySQL数据库记录操作等3个方面来完成设计

首先是FTP程序设计的后端设计部分，后端部分又分为客户端和服务端两部分，主要采用的是Python中的ftplib库来实现。

在服务器的实现中，我们规定了FTP服务器的IP地址和端口号，同时设置了服务器的文件根目录，然后为服务端添加可访问的用户权限信息，在类里面编写允许客户端上传和下载文件的函数。

在客户端的实现中，首先客户端使用ftplib类创建一个ftp对象，并使用connect()方法连接到FTP服务器。然后，使用login()方法进行身份验证，提供用户名和密码以访问FTP服务器。接下来，客户端可以使用storbinary()方法将本地文件上传到FTP服务器指定的路径，或使用retrbinary()方法从FTP服务器下载文件到本地。最后，使用quit()方法断开与FTP服务器的连接

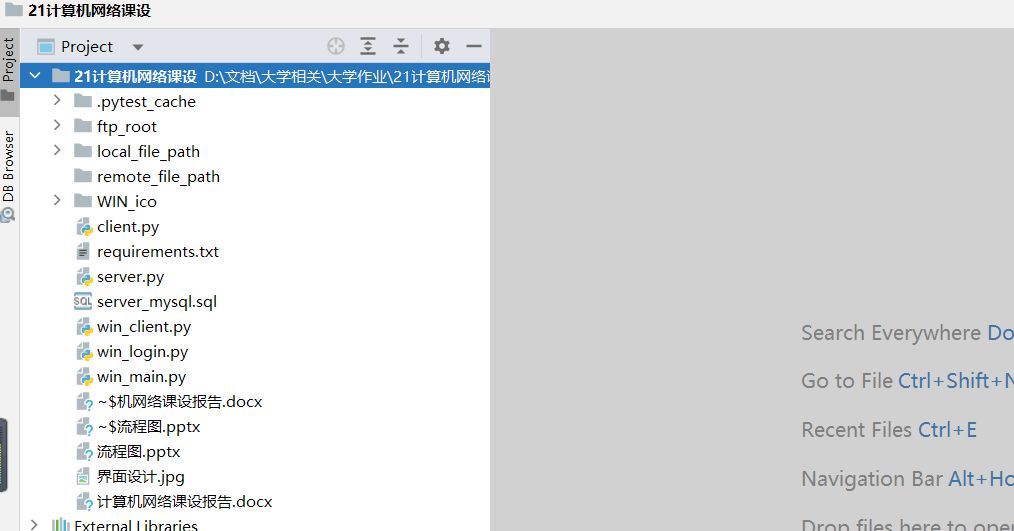
在数据库的设计中，我们同时将客户端和服务端连接到MySQL的数据库中，一方面是为了验证和管理FTP的用户信息，另一方面是为了记录FTP用户在服务器上所进行的操作，如上传文件或下载文件等

2.3 程序设计文件部署与说明

1. 编译环境及语言

该程序的功能是基于Python语言来实现的，采用的编译器是Pycharm，使用的解释器是python 3.10版本，主要使用到的库为pyftplib、PyMySQL、PyQt5。

1. 项目文件说明



如图所示

server.py文件是服务端程序，要运行FTP软件，首先应该将该py文件运行起来

client.py文件是客户端的后端程序，关于FTP软件的实现类及函数都在该文件中

ftp\_root文件夹是该FTP软件的服务器根目录，服务器的文件存储在该文件夹中

local\_file\_path是用于存放本地文件的文件夹

WIN\_ico文件夹是用于存放QT前端界面编写中要用到的一些素材的文件

requirements.txt文件是该项目中使用Python语言编写时要用到和导入的库，附带有详细的库的版本号

server\_mysql.sql是MySQL语言编写的sql数据库文件，用于创建和连接数据库

win\_client.py文件是客户端的主要操作界面，主要提供有文件上传和文件下载功能

win\_login.py文件是客户端的登陆界面，也是本项目的入口文件

# 3. 基于Python的FTP软件核心代码

3.1 FTP 核心功能实现

3.1.1 服务端实现

FTP软件的核心功能实现部分主要分为server.py和client.py两个文件来实现，这也是该FTP软件的后端实现部分。

首先是server.py文件，在该文件中，采用了DummyAuthorizer，该模块用于管理用户账号和权限，采用FTPHandler模块，用于FTP服务器的处理器使用。

在服务器中，我们自定义了两个类，首先是一个处理器类MyHandler，继承自FTPHandler，并重写了两个方法，在该类中，on\_file\_received函数是用于监听文件上传的，当文件上传时，该函数即被出发，打印出文件名，on\_file\_sent函数是用于监听文件下载的，当有文件下载时会触发该函数，打印出文件名。

第二个类则是该服务器后端功能的核心，定义一个server类，用于实际运行FTP服务器。

该封装类中，首先我们对该类进行初始化方法的配置：

1. **def** \_\_init\_\_(self):
2. #首先连接数数据：
3. self.s\_connect\_sql()
4. # 设置FTP服务器的根目录
5. self.FTP\_ROOT = './ftp\_root'
6. # 如果不存在就创建一个文件夹
7. **if** **not** os.path.exists(self.FTP\_ROOT):
8. os.makedirs(self.FTP\_ROOT)
9. # 设置FTP服务器的IP和端口
10. self.HOST = 'localhost'
11. self.PORT = 22
12. # 客户端连接数据库
13. self.conn\_client = self.conn\_client
14. # 创建游标对象
15. self.cursor = self.cursor
16. #使用DummyAuthorizer类来管理用户账号和权限
17. self.authorizer=DummyAuthorizer()
18. # 创建FTP服务器的处理器
19. self.handler=FTPHandler

该初始化方法中包含建立服务端与数据库的连接，调用自定义的连接数据库的函数self.s\_connect\_sql()。其次是对FTP服务器的根目录进行设置，self.FTP\_ROOT = './ftp\_root'，将其设置在当前目录下的ftp\_root文件夹下，这里需要写一个判断语句，如果根目录不存在，则创建一个文件夹，if not os.path.exists(self.FTP\_ROOT): os.makedirs(self.FTP\_ROOT)，使用的是os库中的makedirs函数。然后对FTP服务器的IP地址和端口号进行设置，一般FTP端口号采用的是21，这里我的服务器IP地址设置为本地IP，本地IP既可以在cmd窗口命令中采用ipconfig语句查看，这里也可以直接写为一个字符串”localhost”，代表的就是本地的主机IP。在该类的初始化方法中我们还创建了一个DummyAuthorizer对象来管理用户账号和权限，创建了FTPHandler对象作为FTP服务器的处理器。

其次是add\_author函数：

1. **def** add\_author(self):
2. self.s\_connect\_sql()  # 连接数据库
3. self.cursor.execute('SELECT \* FROM users')#查询用户表中所有数据
4. tmp\_is = self.cursor.fetchall()#获取用户表中所有数据
5. ## 设置FTP服务器的登录用户和密码
6. **for** i **in** tmp\_is:
7. self.authorizer.add\_user(i[1], i[2], self.FTP\_ROOT, perm='elradfmwMT')
8. self.handler.authorizer = self.authorizer

该函数用于向数据库中添加已经注册过的用户信息，同时令数据库中已经注册过的这些信息拥有在FTP服务器上的拥有所有权限，包括读取、删除、创建、修改、移动和重命名文件等

最后是login函数：

1. **def** login(self):
2. load\_handler = MyHandler#监听文件的上传与否
3. load\_handler.authorizer = self.authorizer
4. # 创建FTP服务器实例并启动
5. server = FTPServer((self.HOST, self.PORT), self.handler)
6. **print**('Starting FTP server on %s:%s' % (self.HOST, self.PORT))
7. server.serve\_forever()

该函数用于运行FTP的服务器，FTP服务器开始监听，等待客户端的连接与请求

3.1.2 客户端实现

在客户端的实现中，采用编写了一个自定义类client的形式，包含c\_connect\_sql()、connect\_ftp()、sign\_up()、sign\_in()、ftp\_upload()、ftp\_download()等函数。

首先是c\_connect\_sql()函数：

1. **def** c\_connect\_sql(self):
2. self.conn\_client = pymysql.connect(
3. host='localhost',
4. port=3306,
5. user='root',
6. password='\*\*\*\*\*\*',
7. database='ftp\_info',
8. charset='utf8mb4'
9. )
10. # 创建游标对象
11. self.cursor = self.conn\_client.cursor()

该函数用来连接我们的数据库，同时建立一个游标对象，以便我们在python语言中直接操作数据库中的内容

然后是connect\_ftp()函数，该函数用来连接FTP服务器，要链接到服务器，必须确保服务器已经处于监听状态且建立连接的时候，输入的服务端IP和端口号均正确才能正常与服务器建立连接。

为了对FTP服务器上的用户进行更好的管理，本软件实现了注册和登陆功能，用户可以先进行注册然后获取相应的FTP用户权限，从而对FTP服务器进行文件上传和文件下载的功能。

注册函数如下：

1. **def** sign\_up(self,username,password):
2. self.c\_connect\_sql()#连接数据库
3. self.username=username
4. self.password=password
5. **if** self.username=='' **or** self.password=='':
6. **return** '用户名为空或密码可能为空'
8. #查询数据库中是否有该用户名
9. self.cursor.execute('SELECT \* FROM users WHERE username=%s',(username))
10. tmp\_is=self.cursor.fetchall()#fetchall用来一次性查询结果
12. # 如果用户名存在则返回True，如果不存在则返回false
13. **if** bool(tmp\_is):
14. **print**('用户名已经存在，请勿重复注册')
15. **return** '用户名已经存在，请勿重复注册'
16. **else**:
17. self.cursor.execute('INSERT INTO users(username,password) VALUES(%s,%s)',(self.username,self.password))
19. self.conn\_client.commit()#提交

该函数对用户要注册输入的用户名和密码进行注册，首先对用户名进行检验，如果该用户名已经存在在数据库中，就不再对该账号进行注册，会提醒用户该账号已经注册，请勿重复注册。如果该用户名并不存在在数据库中，则该函数就会将相应的账号和密码写入到该服务器连接的数据库中，并赋予其对应的FTP服务端权限。

登陆函数如下：

1. **def** sign\_in(self,HOST,PORT,username,password):
2. self.c\_connect\_sql()  # 连接数据库
3. self.username = username
4. self.password = password
5. **if** self.username == '' **or** self.password == '':
6. **print**('用户名为空或密码可能为空')
7. **else**:
8. self.cursor.execute('SELECT \* FROM users WHERE username=%s', (username))
9. tmp\_is = self.cursor.fetchall()  # fetchall用来一次性查询结果
11. # 如果用户名存在则返回True，如果不存在则返回false
12. **if** bool(tmp\_is):
13. #如果存在账户，则进行连接FTP请求
14. #账号密码正确且均为字符串类型
15. self.connect\_ftp(HOST,PORT,self.username,self.password)
16. self.log=True#标识已经登陆
17. self.ftp.set\_pasv(True)
18. **return** self.ftp
19. **else**:
20. **print**('查无此用户')

该函数是客户端的登陆函数，用户在输入账号和密码之后，该客户端就会与FTP服务端建立连接，服务端界面也会显示某用户已经登入，如果该用户并不存在，则本软件设计会提示查无此用户，此时用户就可以先进行注册再进行登陆即可。

3.2 MySQL数据库实现

本软件设计为了方便FTP服务端的用户信息管理和行为操作记录，使用MySQL语言创建了一个数据库，该数据库中创建了两个表，其中，users表用来记录拥有FTP服务端权限的用户账号密码信息，downloads表用来记录用户的上传下载操作

该数据库的实现比较容易，首先创建一个数据库，使用语句

1. **create** **database** if not exists ftp\_info;

然后创建一个用户表和一个操作记录表，分别使用以下语句：

1. **CREATE** **TABLE** users (
2. id **INT**(11) NOT NULL AUTO\_INCREMENT,
3. username **VARCHAR**(50) NOT NULL,
4. **password** **VARCHAR**(50) NOT NULL,
5. **PRIMARY** **KEY** (id)
6. );
8. **CREATE** **TABLE** downloads (
9. id **INT**(11) NOT NULL AUTO\_INCREMENT,
10. username **VARCHAR**(50) NOT NULL,
11. filename **VARCHAR**(50) NOT NULL,
12. op **VARCHAR**(50) NOT NULL,
13. **time** **VARCHAR**(50) NOT NULL,
14. **PRIMARY** **KEY** (id)
15. );
16. *--设置最大连接数*SET GLOBAL max\_connections =100;

同时，该数据库设置了一个最大连接数为100。

3.3 QT前端界面实现

为了方便用户操作，本软件设计采用Python语言中的pyqt5库编写了一个简单的客户端登陆界面和客户端操作界面。

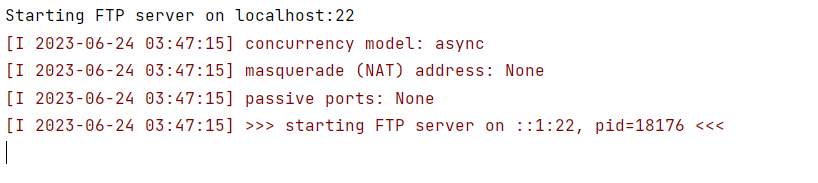
登陆界面是win\_login.py文件，在该文件中，封装了一个LoginWindow类，继承了QWidget类，该LoginWindow类包含以下功能，该界面使用了PyQt5中的水平布局(QHBoxLayout)和垂直布局(QVBoxLayout)来创建主布局和子布局。它提供了用户名和密码输入框，用户可以在输入框中输入用户名和密码。提供服务器IP地址和端口号输入框，用户可以在输入框中输入服务器的IP地址和端口号。此外还提供了登陆按钮和注册按钮，用户可以点击登陆按钮进行登陆，若没有账号，则可以选择先进行注册，登陆成功之后，服务器端则会显示用户已登陆，同时与客户端建立连接。

客户端的功能操作界面是win\_client.py文件，该文件封装了UploadWindow类、DownloadWindow类、AccountWindow类、AccountWindow类、Ui\_MainWindow类以及MainWindow。该界面采用切换页面，其中切换界面分别为上传文件界面、下载文件界面以及查看当前用户信息界面，也就是UploadWindow类、DownloadWindow类、AccountWindow类，然后Ui\_MainWindow类定义了用户界面的布局和控件的设置，而最后的MainWindow类是一个主窗口类，继承自QMainWindow和Ui\_MainWindow。该类定义了主窗口的事件处理函数和程序的入口点，通过这些类和函数的组合，实现了一个具有登陆、注册、上传、下载和账号管理功能的简单用户界面

# 4. 运行与测试

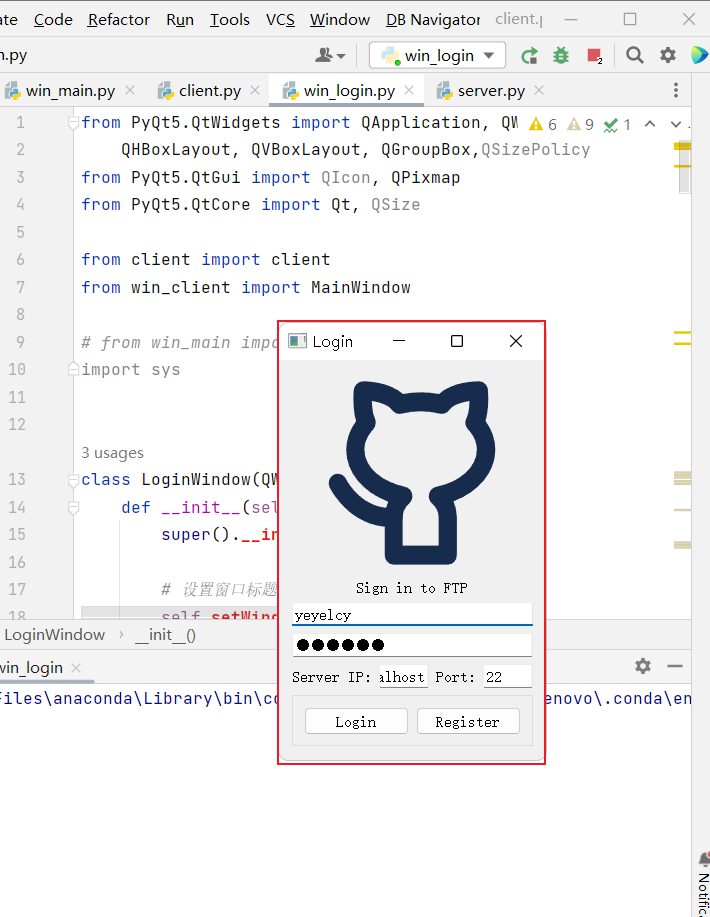
4.1 登陆与注册

1. 首先运行FTP的服务端进程，使得服务器进入监听状态，等待客户端的监听，也就是server.py文件，运行结果如下图：



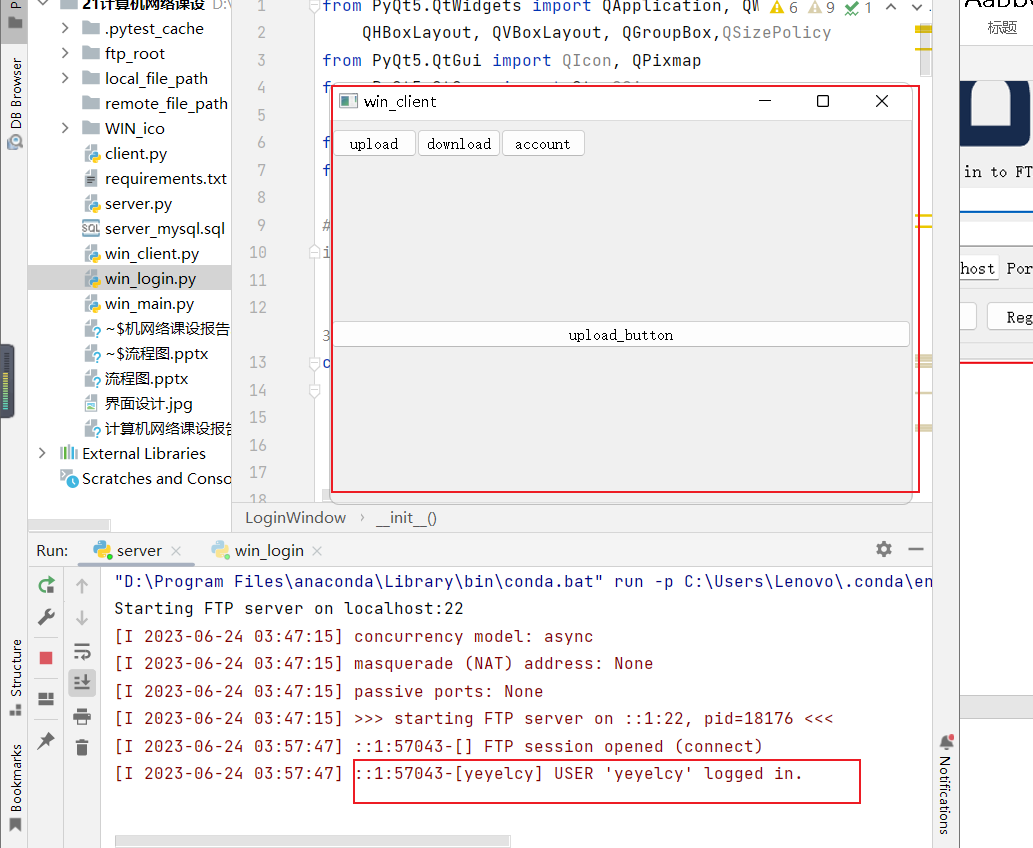
可以看到我们的FTP服务器已经运行起来了

1. 然后进入客户端登陆界面，也就是win\_login.py文件，如图，用户输入账号密码以及FTP服务端的IP地址和端口号之后，即可登陆进入FTP服务端，由于我这里设置的服务端端口号为22，故这里也输入端口号为22，点击login即可

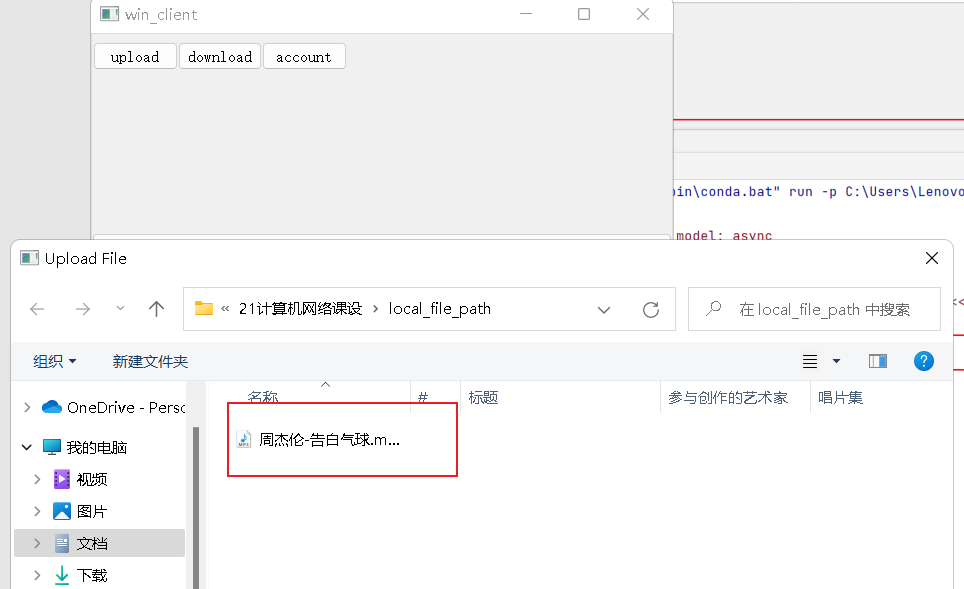


4.2文件的上传与下载

登陆进入之后，可以看到服务端已经显示登入，同时客户端的功能界面显示出来了



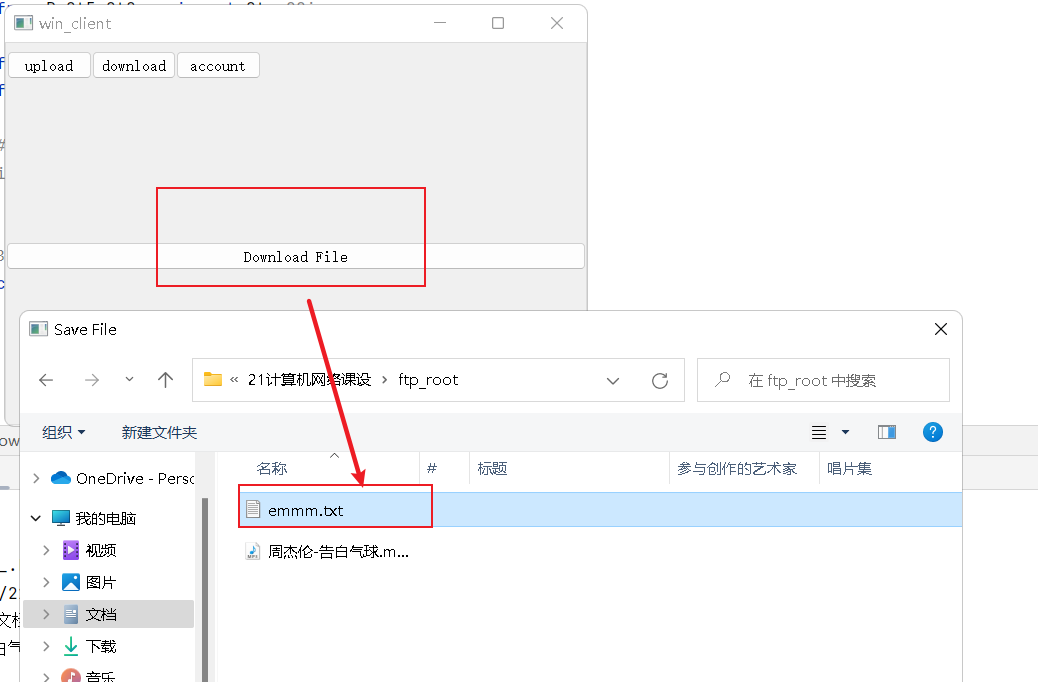
然后点击uploaded按钮即可上传文件，如图，我们选择这首音乐，点击确认即可



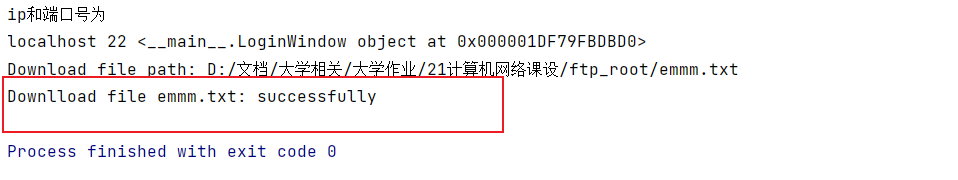
可以看到该文件已经被成功上传了，就在该服务器的根目录下ftp\_root文件夹中，同时客户端这边也显示已经上传成功



同样的，当我们点击下载按钮，如图从服务器根目录下选择一个txt文件夹，点击确认

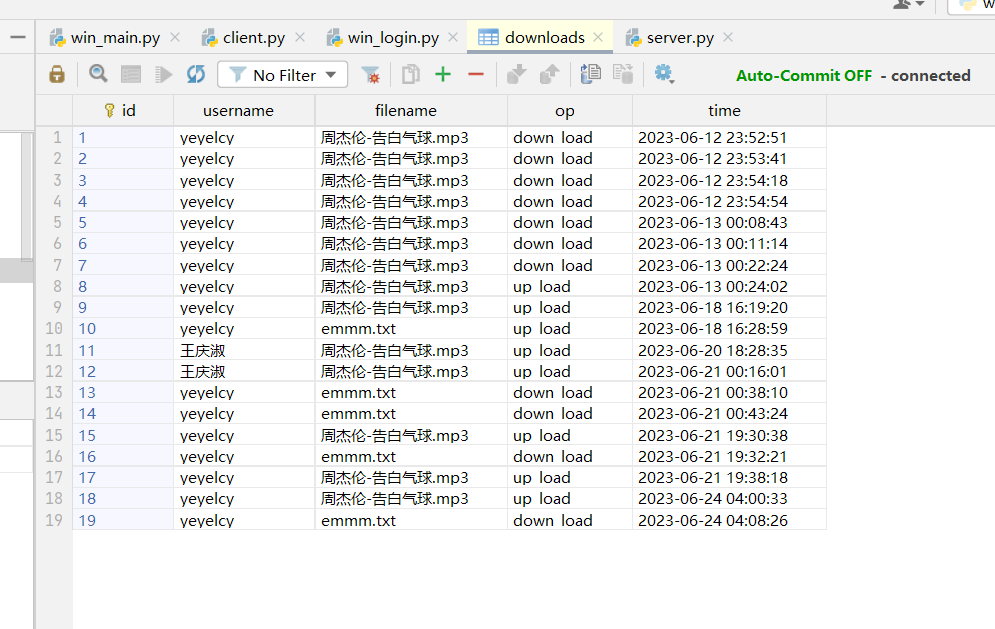


就可以看到文件已经被下载成功了，如图所示



4.3 用户操作的数据库记录

我们打开数据库的downloads 表即可查询用户的上传和下载记录，如图所示



可以看到用户的每一个操作及其操作时间都被记录在了数据库的downloads表中

# 5. 课程设计心得与体会

在进行基于Python的FTP软件的课程设计中，我学会了很多东西，首先，我比较深入了了解了Python的基础知识、Python面向对象的写法和特性以及Python在网络编程中的应用。通过网络中对数据进行传输技术的学习，我了解了FTP协议在生活中的重要应用及其原理，我认识到FTP是一种客户端-服务器的C/S方式，服务器对客户端的请求连接处于监听状态，当收到请求建立连接后，对其发送响应的响应信息进行连接的建立。在此基础上我采用了Python中的ftblib库，它是Python的标准库之一，其提供了一种方便、高效的方式来实现FTP客户端功能，使开发者能够轻松地与远程FTP服务器进行文件传输操作，提高了开发效率和代码的可维护性，它是实现整个整个程序的关键。在本次FTP课程设计中，我意识到，在实际的开发中，理论与实际结合是非常重要的，要将学习的理论知识合理应用到实际的生活应用开发中去，提高自己的实践能力。

在代码的实现过程中，我也遇到了很多的问题，比如在FTP客户端登陆的时候，登陆界面无法与功能操部分结合起来，一开始我是用的是实例属性，当界面进行跳转和信号连接之后，实例属性被刷新，无法继续衔接上一个实例对象的登陆事实，最后通过类属性解决了这个问题，使得各模块进行了一个连接。实际上，也可以通过pyqt5中的一个自定义信号来解决这个问题。此外还有一点需要注意，一开始在登陆的时候，用户权限不足是无法访问FTP服务器的，所以一定要为FTP用户添加用户权限才能正常完成软件的功能操作。在开发过程中，要不仅要学习计算机网络和Python的编程知识，也要不断探索新技术和解决方案，不断优化软件的性能和提升用户体验。本次课程设计让我更深入地理解了Python的网络编程知识以及计算机网络中FTP协议及其工作原理，虽然过程中遇到了一定的困难，但这些都终将成为我不断提升自我的见证，收益颇多

# 6. 参考文献

[1]谢希仁.《计算机网络第七版》[M].电子工业出版社:北京市,2017.

[2] 郝浩． FTP原理解析［J］.计算机与网络，2016(14) :40－41

[3] 葛伟伦． FTP协议两种工作方式下消息交互的分析［J］.电脑知识与技术，2016，12( 8) :30－31，35．

[4] Postel J，Ｒeynolds J． File Transfer Protocol( FTP) ［S］． Network Working Group，1985．

# 7. 附录

7.1 clien.py

1. **import** ftplib
2. **import** pymysql
3. **from** datetime **import** datetime
5. **class** client():
6. **def** \_\_init\_\_(self):
7. self.c\_connect\_sql()#连接数据库
8. self.host='localhost'
9. self.port=22
10. self.username='yeye'#初始用户名
11. self.password='123456'
13. # 客户端连接数据库
14. self.conn\_client=self.conn\_client
16. #创建游标对象
17. self.cursor = self.cursor

20. #标识客户端是否已经登陆
21. self.log=False
23. self.ftp=ftplib.FTP()#ftp对象
25. # self.remote\_file\_path='./remote\_file\_path'
27. """
28. 1.连接数据库+创建游标对象
29. """
30. **def** c\_connect\_sql(self):
31. self.conn\_client = pymysql.connect(
32. host='localhost',
33. port=3306,
34. user='root',
35. password='lx030312',
36. database='ftp\_info',
37. charset='utf8mb4'
38. )
39. # 创建游标对象
40. self.cursor = self.conn\_client.cursor()
42. """
43. # 先启动服务器之后再连接FTP服务器
44. # 2.连接FTP服务器
45. """
46. **def** connect\_ftp(self,host,port,username,password):
47. self.ftp.connect(host, port)
48. self.ftp.login(username, password)
49. # 返回FTP连接对象
50. **return** self.ftp


54. """
55. 3.一个注册功能
56. """
57. **def** sign\_up(self,username,password):
58. self.c\_connect\_sql()#连接数据库
59. self.username=username
60. self.password=password
61. **if** self.username=='' **or** self.password=='':
62. **return** '用户名为空或密码可能为空'
64. #查询数据库中是否有该用户名
65. """
66. fetchall函数
67. 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
68. 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
69. 查询不到则返回空元组
70. """
71. self.cursor.execute('SELECT \* FROM users WHERE username=%s',(username))
72. tmp\_is=self.cursor.fetchall()#fetchall用来一次性查询结果
74. # 如果用户名存在则返回True，如果不存在则返回false
75. **if** bool(tmp\_is):
76. **print**('用户名已经存在，请勿重复注册')
77. **return** '用户名已经存在，请勿重复注册'
78. **else**:
79. self.cursor.execute('INSERT INTO users(username,password) VALUES(%s,%s)',(self.username,self.password))
81. self.conn\_client.commit()#提交
82. # self.conn\_client.close()#关闭
84. """
85. 4.一个登陆功能
86. """
87. **def** sign\_in(self,HOST,PORT,username,password):
88. self.c\_connect\_sql()  # 连接数据库
90. """
91. 登陆成功后就将用户名和密码赋值给了client对象的属性
92. """
93. self.username = username
94. self.password = password
95. **if** self.username == '' **or** self.password == '':
96. **print**('用户名为空或密码可能为空')
97. **else**:
98. self.cursor.execute('SELECT \* FROM users WHERE username=%s', (username))
99. tmp\_is = self.cursor.fetchall()  # fetchall用来一次性查询结果
101. # 如果用户名存在则返回True，如果不存在则返回false
102. **if** bool(tmp\_is):
103. #如果存在账户，则进行连接FTP请求
104. #账号密码正确且均为字符串类型
105. self.connect\_ftp(HOST,PORT,self.username,self.password)
106. self.log=True#标识已经登陆
107. self.ftp.set\_pasv(True)
108. **return** self.ftp
109. **else**:
110. **print**('查无此用户')

113. """
114. 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
115. """
117. **def** ftp\_upload(self,ftp,local\_file\_path):
118. **if** self.log:
119. # 打开本地文件并上传到FTP服务器
120. a = local\_file\_path.split('/')  # a是要上传的文件名
121. **print**(f'要上传的文件名为{a}')
122. with open(local\_file\_path, 'rb') as fp:
123. #这个ftp默认如果已经登陆的话就会被拥有
124. ftp.storbinary('STOR ' + a[-1], fp)  # 往服务器中上传文件
125. **else**:
126. **print**('请您登陆之后再操作')


130. # 记录上传信息到数据库
131. op='up\_load'
132. now = datetime.now().strftime("%Y-%m-%d %H:%M:%S") #获取当前时间
133. now=str(now)
134. # print(type(now))
135. **print**([self.username,a[-1], op, now])#注意a[-1]才是文件名
136. a[-1]=str(a[-1])
137. self.conn\_client.autocommit(True)
138. self.cursor.execute('INSERT INTO downloads (username,filename,op,time) VALUES (%s, %s, %s, %s)', (self.username, a[-1], op, now))
140. **print**("upload  successfully")
141. self.conn\_client.commit()
142. self.conn\_client.close()  # 关闭
144. # 关闭FTP连接
145. # self.ftp.quit()
146. """
147. 6. 下载文件功能
148. """
150. #LOCAL\_FILE\_PATH是文件要下载到的文件夹
151. **def** ftp\_download(self,ftp,local\_file\_path, remote\_file\_path):
153. **if** self.log:
155. # 打开本地文件并从FTP服务器下载
156. a = remote\_file\_path.split('/')
157. # print(local\_file\_path + '/' + a[-1])
158. # print(remote\_file\_path)
160. with open(local\_file\_path + '/' + a[-1], 'wb') as fp:
161. # print('what?')
162. ftp.retrbinary('RETR ' + remote\_file\_path, fp.write)
163. # print('hello?')
164. **else**:
165. **print**('请你登陆后再操作')


169. #记录下载/上传文件信息
170. op='down\_load'
171. now = datetime.now()
172. formatted = now.strftime("%Y-%m-%d %H:%M:%S")
173. # print(type(formatted))
174. self.cursor.execute('INSERT INTO downloads(username, filename, op,time) VALUES (%s, %s, %s, %s)',
175. (self.username, a[-1], op,formatted))
177. **print**(f"Downlload file {remote\_file\_path}: successfully")
179. # print("download  successfully")
180. self.conn\_client.commit()
182. # 关闭FTP连接
183. # self.ftp.quit()
185. **if** \_\_name\_\_=='\_\_main\_\_':
187. test=client()
188. ftp=test.sign\_in('172.19.96.1',22,'yeyelcy','123456')#登陆
189. test.ftp\_upload(ftp,'./local\_file\_path/周杰伦-告白气球.mp3')#上传文件
190. # # test.ftp\_download(ftp,'./local\_file\_path','周杰伦-告白气球.mp3')#下载文件

7.2 server.py

1. **import** pymysql
2. **import** os
3. **from** pyftpdlib.authorizers **import** DummyAuthorizer
4. **from** pyftpdlib.handlers **import** FTPHandler
5. **from** pyftpdlib.servers **import** FTPServer

8. **class** MyHandler(FTPHandler):
10. **def** on\_file\_received(self, file):
11. # 当有文件上传时，会触发该函数
12. **print**('File %s has been received.' % file)
14. **def** on\_file\_sent(self, file):
15. # 当有文件下载时，会触发该函数
16. **print**('File %s has been sent.' % file)
17. **class** server():
18. **def** \_\_init\_\_(self):
19. #首先连接数数据：
20. self.s\_connect\_sql()
21. # 设置FTP服务器的根目录
22. self.FTP\_ROOT = './ftp\_root'
23. # 如果不存在就创建一个文件夹
24. **if** **not** os.path.exists(self.FTP\_ROOT):
25. os.makedirs(self.FTP\_ROOT)
27. # 设置FTP服务器的IP和端口
28. self.HOST = 'localhost'
29. self.PORT = 22
31. # 客户端连接数据库
32. self.conn\_client = self.conn\_client
34. # 创建游标对象
35. self.cursor = self.cursor
37. #使用DummyAuthorizer类来管理用户账号和权限
38. self.authorizer=DummyAuthorizer()
40. # 创建FTP服务器的处理器
41. self.handler=FTPHandler
43. """2.连接数据库"""
44. **def** s\_connect\_sql(self):
45. self.conn\_client = pymysql.connect(
46. host='localhost',
47. port=3306,
48. user='root',
49. password='lx030312',
50. database='ftp\_info',
51. charset='utf8mb4'
52. )
53. # 创建游标对象
54. self.cursor = self.conn\_client.cursor()
56. #向FTP添加可以登陆的用户信息
57. """3.添加用户信息"""
58. **def** add\_author(self):
59. self.s\_connect\_sql()  # 连接数据库
60. self.cursor.execute('SELECT \* FROM users')#查询用户表中所有数据
61. tmp\_is = self.cursor.fetchall()#获取用户表中所有数据
63. """
64. FTP\_ROOT：表示要添加的用户的根目录路径，类型为字符串
65. 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'，
66. 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。
67. """
68. ## 设置FTP服务器的登录用户和密码
69. **for** i **in** tmp\_is:
70. self.authorizer.add\_user(i[1], i[2], self.FTP\_ROOT, perm='elradfmwMT')
71. self.handler.authorizer = self.authorizer
73. """4.查看所有有权限的用户"""
74. **def** check\_users(self):
75. **for** user **in** self.authorizer.user\_table:
76. **print**(user)
78. """5. 运行FTP服务器"""
79. **def** login(self):
80. load\_handler = MyHandler#监听文件的上传与否
81. load\_handler.authorizer = self.authorizer
82. # 创建FTP服务器实例并启动
83. server = FTPServer((self.HOST, self.PORT), self.handler)
84. **print**('Starting FTP server on %s:%s' % (self.HOST, self.PORT))
85. server.serve\_forever()

88. **if** \_\_name\_\_=='\_\_main\_\_':
89. s\_test=server()
90. s\_test.add\_author()
91. s\_test.login()

7.3 win\_login.py

1. **from** PyQt5.QtWidgets **import** QApplication, QWidget, QLabel, QLineEdit, QPushButton, \
2. QHBoxLayout, QVBoxLayout, QGroupBox,QSizePolicy
3. **from** PyQt5.QtGui **import** QIcon, QPixmap
4. **from** PyQt5.QtCore **import** Qt, QSize
6. **from** client **import** client
7. **from** win\_client **import** MainWindow
9. # from win\_main import MainWindow
10. **import** sys

13. **class** LoginWindow(QWidget):
14. **def** \_\_init\_\_(self):
15. super().\_\_init\_\_()
17. # 设置窗口标题和大小
18. self.setWindowTitle('Login')
19. self.resize(400, 600)
21. # 设定背景图和整体布局占比
22. background = QLabel(self)
23. pixmap = QPixmap('background.jpg')
24. background.setPixmap(pixmap)
25. self.resize(pixmap.width(), pixmap.height())
27. # 创建控件
28. logo\_label = QLabel(self)
29. logo\_label.setPixmap(QPixmap('./WIN\_ico/logo.png'))
30. logo\_label.setAlignment(Qt.AlignCenter)
32. title\_label = QLabel(self)
33. title\_label.setText('Sign in to FTP')
34. title\_label.setAlignment(Qt.AlignCenter)

37. #用户名输入框
38. self.username\_edit = QLineEdit(self)
39. self.username\_edit.setPlaceholderText('Enter your username')
40. self.username\_edit.setText('yeyelcy')
42. #密码输入框
43. self.password\_edit = QLineEdit(self)
44. self.password\_edit.setEchoMode(QLineEdit.Password)#设置密码不可见
45. self.password\_edit.setPlaceholderText('Enter your password')
46. self.password\_edit.setText('123456')
48. # 添加服务器 IP 地址输入框
49. self.ip\_label = QLabel(self)
50. self.ip\_label.setText('Server IP:')
51. self.ip\_edit = QLineEdit(self)
52. self.ip\_edit.setPlaceholderText('Enter the server IP address')
53. self.ip\_edit.setText('localhost')  # 设置默认值
55. # 添加端口号输入框
56. self.port\_label = QLabel(self)
57. self.port\_label.setText('Port:')
58. self.port\_edit = QLineEdit(self)
59. self.port\_edit.setPlaceholderText('Enter the port number')
60. self.port\_edit.setText('22')


64. #登录按钮
65. login\_btn = QPushButton('Login', self)
66. login\_btn.clicked.connect(self.login\_clint)
68. #注册按钮
69. register\_btn = QPushButton('Register', self)
70. register\_btn.clicked.connect(self.login\_up)
72. btn\_group = QGroupBox(self)
73. btn\_layout = QHBoxLayout(btn\_group)
74. btn\_layout.addWidget(login\_btn)
75. btn\_layout.addWidget(register\_btn)
77. # 构建主布局，并设置左右两边的间距
78. main\_layout = QVBoxLayout(self)
79. main\_layout.addStretch(3)
80. main\_layout.addWidget(logo\_label, stretch=5)
81. main\_layout.addWidget(title\_label, stretch=1)
82. main\_layout.addStretch(3)
83. main\_layout.addWidget(self.username\_edit, stretch=1)
84. main\_layout.addWidget(self.password\_edit, stretch=1)
85. main\_layout.addWidget(btn\_group, stretch=1)
86. main\_layout.addStretch(4)
88. ip\_port\_layout = QHBoxLayout()
89. ip\_port\_layout.addWidget(self.ip\_label)
90. ip\_port\_layout.addWidget(self.ip\_edit)
91. ip\_port\_layout.addWidget(self.port\_label)
92. ip\_port\_layout.addWidget(self.port\_edit)
93. main\_layout.addLayout(ip\_port\_layout)
95. main\_layout.addWidget(btn\_group, stretch=1)
96. main\_layout.addStretch(4)

99. # 绑定提示文本
100. self.username\_edit.setToolTip('Please enter your username')
101. self.password\_edit.setToolTip('Please enter your password')



106. **def** show\_login(self):
107. self.show()

110. #登陆客户端代码
111. **def** login\_clint(self):
112. self.sign\_in = client()  # 会自动指定端口号
113. self.username = self.username\_edit.text()
114. self.password = self.password\_edit.text()
115. self.host = self.ip\_edit.text()
116. self.port = int(self.port\_edit.text())
117. **print**('账号密码为：')
118. **print**(self.username, self.password)
119. **print**("ip和端口号为")
120. **print**(self.host,self.port,self)
121. a=self.sign\_in.sign\_in(self.host,self.port,self.username, self.password)#进行登陆
123. #self.sign\_in是一个client对象
124. **if** self.sign\_in.log:
125. self.log\_client=MainWindow()
126. MainWindow.ftp = a
127. MainWindow.user\_log = self.sign\_in
128. #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
129. self.log\_client.show()
130. **else**:
131. **print**('请登录重试')
133. #注册客户端代码
134. **def** login\_up(self):
135. self.sign\_in=client()
136. self.username = self.username\_edit.text()
137. self.password = self.password\_edit.text()
138. # self.host=self.ip\_edit.text()
139. # self.port=self.port\_edit.text()
140. **print**('账号密码为：')
141. **print**(self.username, self.password)
142. self.sign\_in.sign\_up(self.username, self.password)  # 进行登陆
144. **if** \_\_name\_\_ == '\_\_main\_\_':
145. app = QApplication([])
146. login\_window = LoginWindow()
147. login\_window.show\_login()
148. app.exec\_()

7.4 win\_client.py

1. # -\*- coding: utf-8 -\*-

4. **from** PyQt5.QtWidgets **import** QPushButton
5. **from** PyQt5.QtWidgets **import**  QWidget, QVBoxLayout, QPushButton, QLabel,QFileDialog,QMainWindow,QApplication
6. **from** PyQt5 **import** QtCore, QtWidgets
7. **import** sys
8. **from** client **import** client
9. **import** ftplib



14. **class** UploadWindow(QWidget):
15. **def** \_\_init\_\_(self):
16. super().\_\_init\_\_()

19. # 创建控件
20. label = QLabel('This is the upload window.')
21. upload\_button = QPushButton('Upload')
22. layout = QVBoxLayout(self)
23. layout.addWidget(label)
24. layout.addWidget(upload\_button)
26. **class** DownloadWindow(QWidget):
27. **def** \_\_init\_\_(self):
28. super().\_\_init\_\_()
30. # 创建控件
31. label = QLabel('This is the download window.')
32. download\_button = QPushButton('Download')
33. layout = QVBoxLayout(self)
34. layout.addWidget(label)
35. layout.addWidget(download\_button)
37. **class** AccountWindow(QWidget):
38. **def** \_\_init\_\_(self):
39. super().\_\_init\_\_()
41. # 创建控件
42. label = QLabel('This is the account management window.')
43. manage\_button = QPushButton('Manage')
44. layout = QVBoxLayout(self)
45. layout.addWidget(label)
46. layout.addWidget(manage\_button)

49. **class** Ui\_MainWindow(object):
50. **def** setupUi(self, MainWindow):
51. MainWindow.setObjectName("MainWindow")
52. MainWindow.resize(582, 384)
54. # 创建主控件
55. self.centralwidget = QtWidgets.QWidget(MainWindow)
56. self.centralwidget.setMinimumSize(QtCore.QSize(568, 379))
57. self.centralwidget.setObjectName("centralwidget")
59. # 创建水平布局器
60. self.horizontalLayout\_3 = QtWidgets.QHBoxLayout(self.centralwidget)
61. self.horizontalLayout\_3.setContentsMargins(0, 0, 0, 0)
62. self.horizontalLayout\_3.setSpacing(0)
63. self.horizontalLayout\_3.setObjectName("horizontalLayout\_3")
65. # 创建分组框
66. self.groupBox = QtWidgets.QGroupBox(self.centralwidget)
67. self.groupBox.setMinimumSize(QtCore.QSize(568, 379))
68. self.groupBox.setTitle("")
69. self.groupBox.setObjectName("groupBox")
71. # 创建垂直布局器
72. self.verticalLayout = QtWidgets.QVBoxLayout(self.groupBox)
73. self.verticalLayout.setContentsMargins(0, 0, 0, 0)
74. self.verticalLayout.setSpacing(0)
75. self.verticalLayout.setObjectName("verticalLayout")
77. # 创建框架
78. self.frame = QtWidgets.QFrame(self.groupBox)
79. self.frame.setMinimumSize(QtCore.QSize(568, 45))
80. self.frame.setFrameShape(QtWidgets.QFrame.StyledPanel)
81. self.frame.setFrameShadow(QtWidgets.QFrame.Raised)
82. self.frame.setObjectName("frame")
84. # 创建水平布局器
85. self.horizontalLayout\_2 = QtWidgets.QHBoxLayout(self.frame)
86. self.horizontalLayout\_2.setContentsMargins(0, 0, 0, 0)
87. self.horizontalLayout\_2.setSpacing(0)
88. self.horizontalLayout\_2.setObjectName("horizontalLayout\_2")
90. # 创建水平布局器
91. self.horizontalLayout = QtWidgets.QHBoxLayout()
92. self.horizontalLayout.setSpacing(0)
93. self.horizontalLayout.setObjectName("horizontalLayout")
95. # 创建按钮1
96. self.pushButton\_1 = QtWidgets.QPushButton(self.frame)
97. self.pushButton\_1.setMinimumSize(QtCore.QSize(75, 23))
98. self.pushButton\_1.setObjectName("pushButton\_1")
99. self.horizontalLayout.addWidget(self.pushButton\_1)
101. # 创建按钮2
102. self.pushButton\_2 = QtWidgets.QPushButton(self.frame)
103. self.pushButton\_2.setMinimumSize(QtCore.QSize(75, 23))
104. self.pushButton\_2.setObjectName("pushButton\_2")
105. self.horizontalLayout.addWidget(self.pushButton\_2)
107. # 创建按钮3
108. self.pushButton\_3 = QtWidgets.QPushButton(self.frame)
109. self.pushButton\_3.setMinimumSize(QtCore.QSize(75, 23))
110. self.pushButton\_3.setObjectName("pushButton\_3")
111. self.horizontalLayout.addWidget(self.pushButton\_3)

114. self.horizontalLayout\_2.addLayout(self.horizontalLayout)
116. # 创建布局器项
117. spacerItem = QtWidgets.QSpacerItem(348, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
118. self.horizontalLayout\_2.addItem(spacerItem)
119. self.verticalLayout.addWidget(self.frame)
121. #创建了一个堆叠窗口对象 stacked\_widget
122. self.stackedWidget = QtWidgets.QStackedWidget(self.groupBox)
123. self.stackedWidget.setMinimumSize(QtCore.QSize(568, 334))
124. self.stackedWidget.setObjectName("stackedWidget")
126. # 添加页面1控件
127. self.page = QtWidgets.QWidget()
128. self.page.setObjectName("page")
130. self.horizontalLayout\_4 = QtWidgets.QHBoxLayout(self.page)
131. self.horizontalLayout\_4.setContentsMargins(0, 0, 0, 0)
132. self.horizontalLayout\_4.setSpacing(0)
133. self.horizontalLayout\_4.setObjectName("horizontalLayout\_4")
135. #添加一个选择文件的功能
136. self.upload\_button = QtWidgets.QPushButton(self.page)
137. self.upload\_button.clicked.connect(self.upload\_file)
138. self.horizontalLayout\_4.addWidget(self.upload\_button)
139. self.stackedWidget.addWidget(self.page)
141. # 添加页面2控件
142. self.page\_2 = QtWidgets.QWidget()
143. self.page\_2.setObjectName("page\_2")
144. self.horizontalLayout\_5 = QtWidgets.QHBoxLayout(self.page\_2)
145. self.horizontalLayout\_5.setContentsMargins(0, 0, 0, 0)
146. self.horizontalLayout\_5.setSpacing(0)
147. self.horizontalLayout\_5.setObjectName("horizontalLayout\_5")

150. self.download\_button = QtWidgets.QPushButton(self.page\_2)
152. self.download\_button.clicked.connect(self.download\_file)
153. self.horizontalLayout\_5.addWidget(self.download\_button)
155. self.stackedWidget.addWidget(self.page\_2)


159. # 添加页面3控件
160. self.page\_3 = QtWidgets.QWidget()
161. self.page\_3.setObjectName("page\_3")
162. self.horizontalLayout\_6 = QtWidgets.QHBoxLayout(self.page\_2)
163. self.horizontalLayout\_6.setContentsMargins(0, 0, 0, 0)
164. self.horizontalLayout\_6.setSpacing(0)
165. self.horizontalLayout\_6.setObjectName("horizontalLayout\_5")
167. self.user\_info=QtWidgets.QPushButton(self.page\_3)
168. self.user\_info.clicked.connect(self.user\_acount)
169. self.user\_info.setText('check on user info')
170. self.horizontalLayout\_6.addWidget(self.user\_info)
172. self.stackedWidget.addWidget(self.page\_3)

175. #添加整体的
176. self.verticalLayout.addWidget(self.stackedWidget)
177. self.horizontalLayout\_3.addWidget(self.groupBox)
178. MainWindow.setCentralWidget(self.centralwidget)
180. self.retranslateUi(MainWindow)
181. self.stackedWidget.setCurrentIndex(1)
182. QtCore.QMetaObject.connectSlotsByName(MainWindow)
184. **def** upload\_file(self):
185. # 打开文件选择对话框
186. file\_path, \_ = QFileDialog.getOpenFileName(self, "Upload File", "", "All Files (\*);;Text Files (\*.txt)")
187. **print**(file\_path)
189. # 如果有选择文件，执行上传操作
190. **if** file\_path:
191. #self.up\_o=client()
192. # TODO self.up\_o.ftp\_upload(self,self.ftp,file\_path)
193. **if** MainWindow.ftp:
194. #self.up\_o.ftp\_upload(MainWindow.ftp,file\_path)
195. MainWindow.user\_log.ftp\_upload(MainWindow.ftp,file\_path)
196. # 执行上传操作
197. **print**("Upload file:", file\_path)
198. **else**:
199. **print**("报错")
201. **def** download\_file(self):
202. # TODO: 执行下载操作
203. down\_path, \_ = QFileDialog.getOpenFileName(self, "Save File", "", "All Files (\*);;Text Files (\*.txt)")
205. **print**("Download file path:", down\_path)
206. down\_path=down\_path.split('/')[-1]

209. # 如果选择了要下载的文件路径，执行保存文件的操作
210. **if** down\_path:
211. **if** MainWindow.ftp:
212. MainWindow.user\_log.ftp\_download(MainWindow.ftp,'./local\_file\_path',down\_path)
214. **else**:
215. **print**('报错')
217. **def** user\_acount(self):
218. **print**("用户信息为",MainWindow.user\_log.username)
220. **def** retranslateUi(self, MainWindow):
221. \_translate = QtCore.QCoreApplication.translate
222. MainWindow.setWindowTitle(\_translate("MainWindow", "win\_client"))
223. self.pushButton\_1.setText(\_translate("MainWindow", "upload"))
224. self.pushButton\_2.setText(\_translate("MainWindow", "download"))
225. self.pushButton\_3.setText(\_translate("MainWindow", "account"))
226. # self.pushButton\_4.setText(\_translate("MainWindow", "页面1")
227. self.upload\_button.setText(\_translate("MainWindow", "upload\_button"))
228. # self.pushButton\_5.setText(\_translate("MainWindow", "页面2"))
229. self.download\_button.setText(\_translate("MainWindow","Download File"))
230. **class** MainWindow(QMainWindow, Ui\_MainWindow):
231. ftp = None
232. user\_log = None
233. **def** \_\_init\_\_(self, parent=None):
234. super(MainWindow, self).\_\_init\_\_(parent)
235. self.setupUi(self)

238. # self.ftp=ftplib.FTP()#ftp对象
239. self.pushButton\_1.clicked.connect(**lambda**: self.stackedWidget.setCurrentIndex(0))
240. self.pushButton\_2.clicked.connect(**lambda**: self.stackedWidget.setCurrentIndex(1))
241. self.pushButton\_3.clicked.connect(**lambda**: self.stackedWidget.setCurrentIndex(2))
242. **if** \_\_name\_\_=='\_\_main\_\_':
243. app = QApplication(sys.argv)
244. myWin = MainWindow()
245. myWin.show()
246. sys.exit(app.exec\_())

7.5 server\_mysql.sql

1. --查看当前数据库
2. **select** **database**();
4. --创建一个数据库
5. **create** **database** if not exists ftp\_info;
7. --选择一个数据库
8. use ftp\_info;
9. -- 创建用户表
10. **CREATE** **TABLE** users (
11. id **INT**(11) NOT NULL AUTO\_INCREMENT,
12. username **VARCHAR**(50) NOT NULL,
13. **password** **VARCHAR**(50) NOT NULL,
14. **PRIMARY** **KEY** (id)
15. );
17. -- 创建文件表
18. **CREATE** **TABLE** files (
19. id **INT**(11) NOT NULL AUTO\_INCREMENT,
20. filename **VARCHAR**(50) NOT NULL,
21. filepath **VARCHAR**(100) NOT NULL,
22. **PRIMARY** **KEY** (id)
23. );
25. -- 创建下载日志表
26. **CREATE** **TABLE** downloads (
27. id **INT**(11) NOT NULL AUTO\_INCREMENT,
28. username **VARCHAR**(50) NOT NULL,
29. filename **VARCHAR**(50) NOT NULL,
30. op **VARCHAR**(50) NOT NULL,
31. **time** **VARCHAR**(50) NOT NULL,
32. **PRIMARY** **KEY** (id)
33. );

36. --设置最大连接数
37. **SET** **GLOBAL** max\_connections =100;