



河南理工大学

《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

| 号 | 评价指标 | 分 值 | 评价等级及参考分值 | | | | | 评 价分 |
|---|--|--------|-----------|----|----|----|-----|---------|
| | | | 优 | 良 | 中 | 及格 | 不及格 | |
| | 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 | |
| | 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 | |
| | 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 | | | | | | | | |
| <div style="text-align: right; margin-top: 20px;">签名（签章）:</div> | | | | | | | | |

日期： 年 月 日

目录

| | |
|---------------------------------|----|
| 1. FTP 基本知识..... | 6 |
| 1.1 FTP 的基本概念 | 6 |
| 1.2 FTP 的工作原理 | 6 |
| 1.3 FTP 的应用 | 6 |
| 2. FTP 基本原理..... | 7 |
| 2.1 FTP 软件设计流程图 | 7 |
| 2.2 FTP 程序设计原理 | 7 |
| 2.3 程序设计文件部署与说明 | 8 |
| 3. 基于 Python 的 FTP 软件核心代码 | 9 |
| 3.1 FTP 核心功能实现 | 9 |
| 3.2 MySQL 数据库实现 | 13 |
| 3.3 QT 前端界面实现..... | 14 |
| 4. 运行与测试 | 15 |
| 4.1 登陆与注册 | 15 |
| 4.2 文件的上传与下载..... | 16 |
| 4.3 用户操作的数据库记录 | 18 |
| 5. 课程设计心得与体会 | 19 |
| 6. 参考文献..... | 20 |
| 7. 附录 | 21 |
| 7.1 clien.py | 21 |
| 7.2 server.py..... | 26 |
| 7.3 win_login.py | 28 |
| 7.4 win_client.py | 32 |
| 7.5 server_mysql.sql | 39 |

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端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

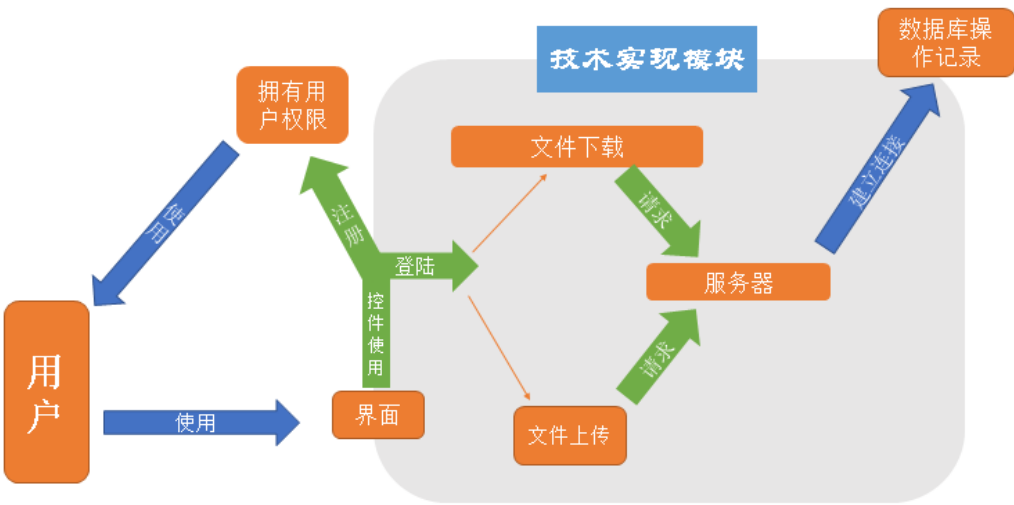
1.3 FTP 的应用

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

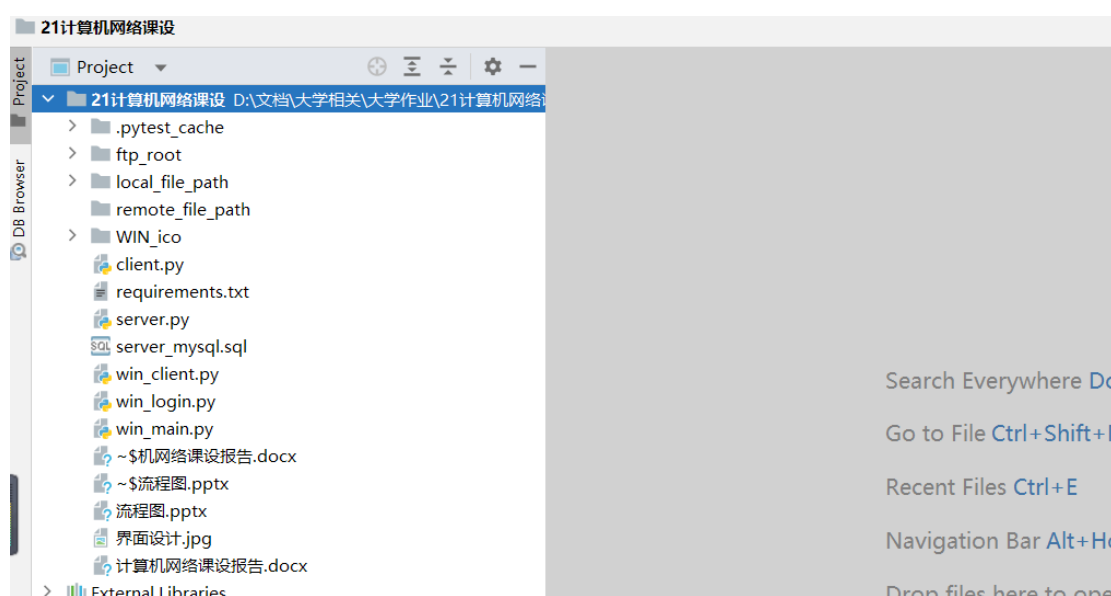
在数据库的设计中，我们同时将客户端和服务端连接到 MySQL 的数据库中，一方面是为了验证和管理 FTP 的用户信息，另一方面是为了记录 FTP 用户在服务器上所进行的操作，如上传文件或下载文件等

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`。

2. 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, pe
rm='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'
7.
8.     #查询数据库中是否有该用户名
9.     self.cursor.execute('SELECT * FROM users WHERE username=%s',
10. (username))
11.
12.     tmp_is=self.cursor.fetchall()#fetchall 用来一次性查询结果
13.
14.     # 如果用户名存在则返回 True, 如果不存在则返回 false
15.     if bool(tmp_is):
16.         print('用户名已经存在, 请勿重复注册')
17.         return '用户名已经存在, 请勿重复注册'
18.     else:
19.         self.cursor.execute('INSERT INTO users(username,password)
VALUES(%s,%s)',(self.username,self.password))
20.
21.         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',
9. (username))
10.
11.         tmp_is = self.cursor.fetchall() # fetchall 用来一次性
12. 查询结果
```

```

10.
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. );
7.
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

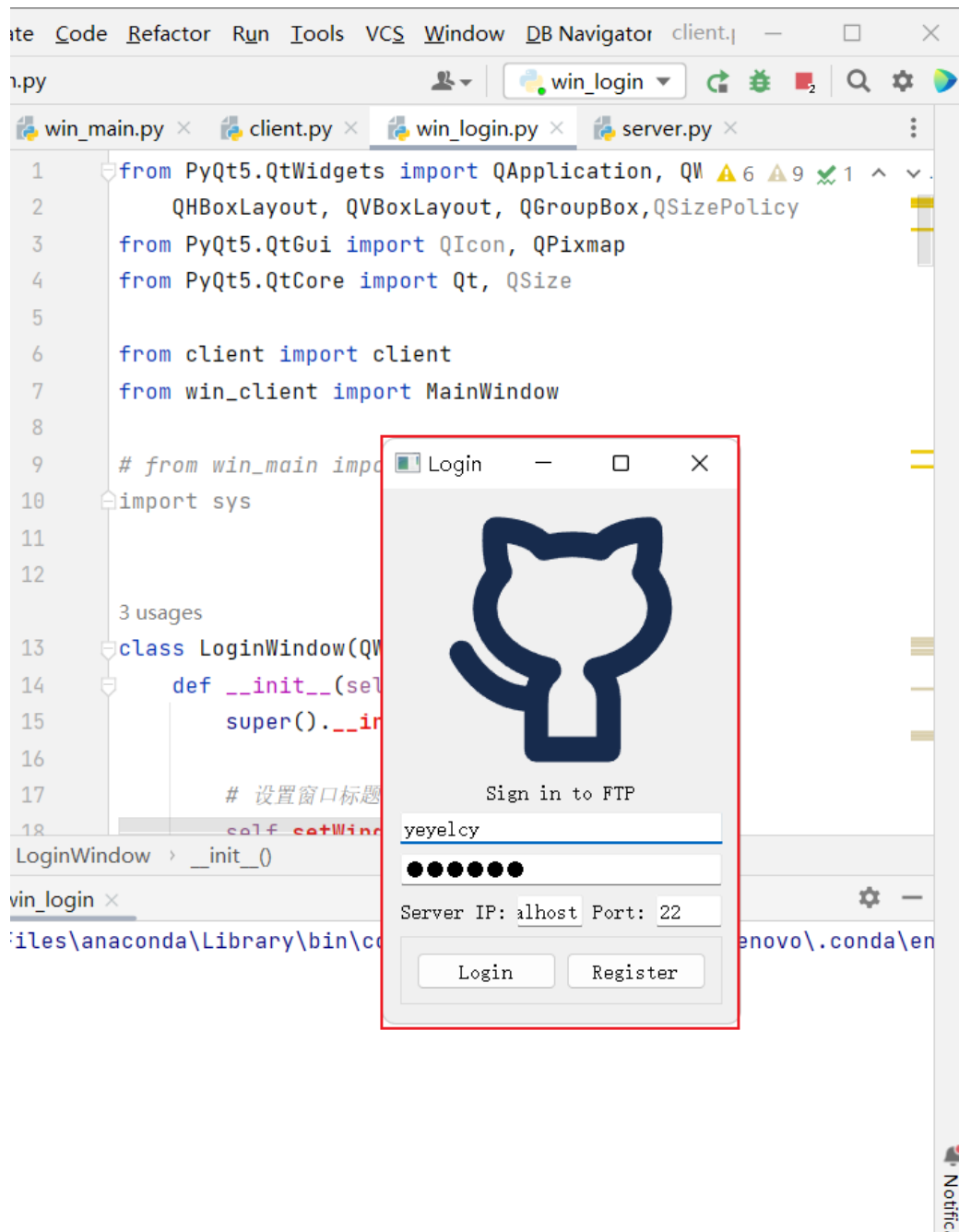
4.1 登陆与注册

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:

```
Starting FTP server on localhost:22
[I 2023-06-24 03:47:15] concurrency model: async
[I 2023-06-24 03:47:15] masquerade (NAT) address: None
[I 2023-06-24 03:47:15] passive ports: None
[I 2023-06-24 03:47:15] >>> starting FTP server on ::1:22, pid=18176 <<<
|
```

可以看到我们的 FTP 服务器已经运行起来了

2. 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

夹中，同时客户端这边也显示已经上传成功

```
~$流程图.pptx
流程图.pptx
界面设计.jpg
计算机网络课设报
cternal Libraries
cratches and Conso

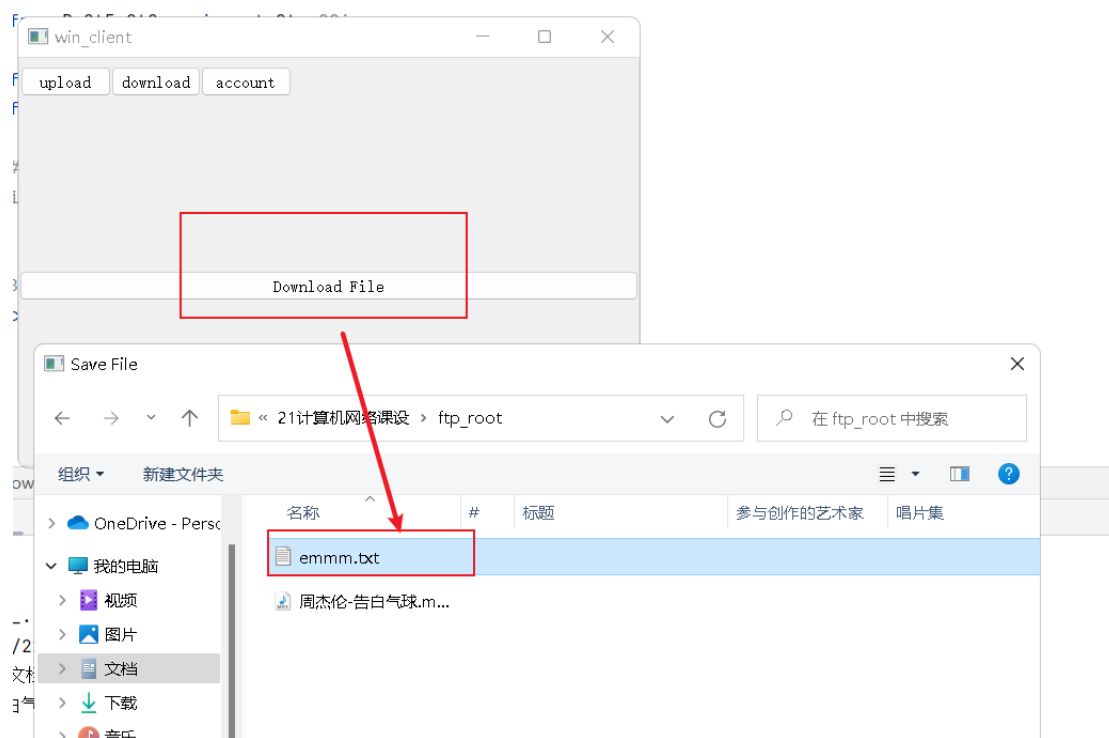
class LoginWindow(QWidget):
    def __init__(self):
        super().__init__()

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

LoginWindow > __init_0

server x win_login x
yeyelcy 123456
ip和端口号为
localhost 22 <__main__.LoginWindow object at 0x000002994F7C9BD0>
D:/文档/大学相关/大学作业/21计算机网络课设/local_file_path/周杰伦-告白气球.mp3
要上传的文件名为['D:', '文档', '大学相关', '大学作业', '21计算机网络课设', 'local_file_path', '周杰伦-告白气球.mp3']
['yeyelcy', '周杰伦-告白气球.mp3', 'up_load', '2023-06-24 04:00:33']
upload successfully
Upload file: D:/文档/大学相关/大学作业/21计算机网络课设/local_file_path/周杰伦-告白气球.mp3
```

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

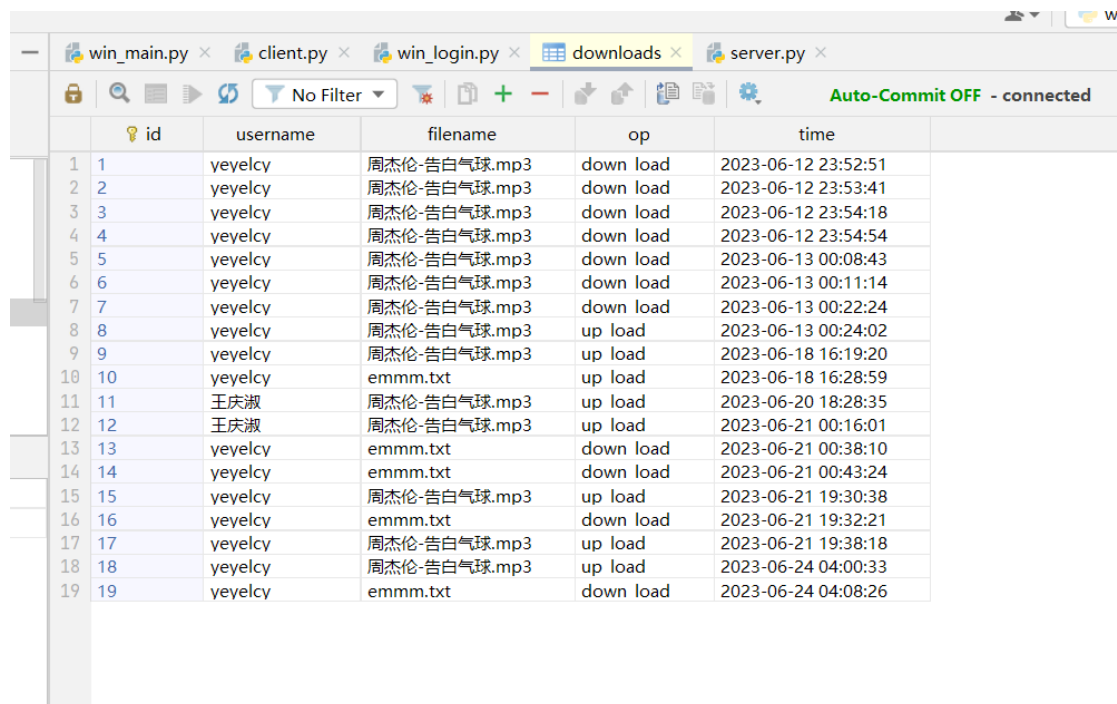


就可以看到文件已经被下载成功了，如图所示

```
ip和端口号为
localhost 22 <__main__.LoginWindow object at 0x000001DF79FBDBD0>
Download file path: D:/文档/大学相关/大学作业/21计算机网络课设/ftp_root/emmm.txt
Download file emmm.txt: successfully
Process finished with exit code 0
```

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



| | id | username | filename | op | time |
|----|----|----------|--------------|-----------|---------------------|
| 1 | 1 | yeyelcy | 周杰伦-告白气球.mp3 | down load | 2023-06-12 23:52:51 |
| 2 | 2 | yeyelcy | 周杰伦-告白气球.mp3 | down load | 2023-06-12 23:53:41 |
| 3 | 3 | yeyelcy | 周杰伦-告白气球.mp3 | down load | 2023-06-12 23:54:18 |
| 4 | 4 | yeyelcy | 周杰伦-告白气球.mp3 | down load | 2023-06-12 23:54:54 |
| 5 | 5 | yeyelcy | 周杰伦-告白气球.mp3 | down load | 2023-06-13 00:08:43 |
| 6 | 6 | yeyelcy | 周杰伦-告白气球.mp3 | down load | 2023-06-13 00:11:14 |
| 7 | 7 | yeyelcy | 周杰伦-告白气球.mp3 | down load | 2023-06-13 00:22:24 |
| 8 | 8 | yeyelcy | 周杰伦-告白气球.mp3 | up load | 2023-06-13 00:24:02 |
| 9 | 9 | yeyelcy | 周杰伦-告白气球.mp3 | up load | 2023-06-18 16:19:20 |
| 10 | 10 | yeyelcy | emmm.txt | up load | 2023-06-18 16:28:59 |
| 11 | 11 | 王庆淑 | 周杰伦-告白气球.mp3 | up load | 2023-06-20 18:28:35 |
| 12 | 12 | 王庆淑 | 周杰伦-告白气球.mp3 | up load | 2023-06-21 00:16:01 |
| 13 | 13 | yeyelcy | emmm.txt | down load | 2023-06-21 00:38:10 |
| 14 | 14 | yeyelcy | emmm.txt | down load | 2023-06-21 00:43:24 |
| 15 | 15 | yeyelcy | 周杰伦-告白气球.mp3 | up load | 2023-06-21 19:30:38 |
| 16 | 16 | yeyelcy | emmm.txt | down load | 2023-06-21 19:32:21 |
| 17 | 17 | yeyelcy | 周杰伦-告白气球.mp3 | up load | 2023-06-21 19:38:18 |
| 18 | 18 | yeyelcy | 周杰伦-告白气球.mp3 | up load | 2023-06-24 04:00:33 |
| 19 | 19 | yeyelcy | emmm.txt | down load | 2023-06-24 04:08:26 |

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 `ftplib` 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，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
4.
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'
12.
13.        # 客户端连接数据库
14.        self.conn_client=self.conn_client
15.
16.        #创建游标对象
17.        self.cursor = self.cursor
18.
19.
20.        #标识客户端是否已经登陆
21.        self.log=False
22.
23.        self.ftp=ftplib.FTP()#ftp 对象
24.
25.        # self.remote_file_path='./remote_file_path'
26.
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()
41.
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
51.
52.
53.
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'
63.
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 usernam
e=%s',(username))
72.             tmp_is=self.cursor.fetchall()#fetchall 用来一次性查询结
果
73.
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,pa
ssword) VALUES(%s,%s)',(self.username,self.password))
80.
81.         self.conn_client.commit()#提交
82.         # self.conn_client.close()#关闭
83.
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
89.
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 use
rname=%s', (username))
99.                 tmp_is = self.cursor.fetchall() # fetchall 用来一
次性查询结果
100.
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,s
elf.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')
111.
112.
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.         """
116.

```

```

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')
127.
128.
129.
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")
133.             #获取当前时间
134.             now=str(now)
135.             # print(type(now))
136.             print([self.username,a[-1], op, now])#注意 a[-1]才是文件名
137.             a[-1]=str(a[-1])
138.             self.conn_client.autocommit(True)
139.             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
143.
144.             # 关闭 FTP 连接
145.             # self.ftp.quit()
146.             """
147.             6. 下载文件功能
148.             """
149.
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):
152.
153.                 if self.log:

```



```

154.
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)
159.
160.         with open(local_file_path + '/' + a[-
161. 1], 'wb') as fp:
162.             # print('what?')
163.             ftp.retrbinary('RETR ' + remote_file_path,
164. fp.write)
165.             # print('hello?')
166.         else:
167.             print('请你登陆后再操作')
168.
169.
170.         #记录下载/上传文件信息
171.         op='down_load'
172.         now = datetime.now()
173.         formatted = now.strftime("%Y-%m-%d %H:%M:%S")
174.         # print(type(formatted))
175.         self.cursor.execute('INSERT INTO downloads(username
176. e, filename, op,time) VALUES (%s, %s, %s, %s)',
177. (self.username, a[-
178. 1], op,formatted))
179.
180.         print(f"Downlload file {remote_file_path}: success
181. fully")
182.
183.         # print("download successfully")
184.         self.conn_client.commit()
185.
186.         # 关闭 FTP 连接
187.         # self.ftp.quit()
188.
189.
190. if __name__=='__main__':
191.
192.     test=client()
193.     ftp=test.sign_in('172.19.96.1',22,'yeyelcy','123456'
194. )#登陆
195.     test.ftp_upload(ftp,'./local_file_path/周杰伦-告白气
196. 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 pyftplib.authorizers import DummyAuthorizer
4. from pyftplib.handlers import FTPHandler
5. from pyftplib.servers import FTPServer
6.
7.
8. class MyHandler(FTPHandler):
9.
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)
13.
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.         #首先连接数据库:
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)
26.
27.         # 设置 FTP 服务器的 IP 和端口
28.         self.HOST = 'localhost'
29.         self.PORT = 22
30.
31.         # 客户端连接数据库
32.         self.conn_client = self.conn_client
33.
34.         # 创建游标对象
35.         self.cursor = self.cursor
36.
37.         #使用 DummyAuthorizer 类来管理用户账号和权限
38.         self.authorizer=DummyAuthorizer()
39.
```

```

40.         # 创建 FTP 服务器的处理器
41.         self.handler=FTPHandler
42.
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()
55.
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
62.
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
72.
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)
77.
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.handle
r)
84.         print('Starting FTP server on %s:%s' % (self.HOST, sel
f.PORT))
85.         server.serve_forever()
86.
87.
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, QLi
neEdit, QPushButton, \
2.     QHBoxLayout, QVBoxLayout, QGroupBox, QSizePolicy
3. from PyQt5.QtGui import QIcon, QPixmap
4. from PyQt5.QtCore import Qt, QSize
5.
6. from client import client
7. from win_client import MainWindow
8.
9. # from win_main import MainWindow
10. import sys
11.
12.
13. class LoginWindow(QWidget):
14.     def __init__(self):
15.         super().__init__()
16.
17.         # 设置窗口标题和大小
18.         self.setWindowTitle('Login')
19.         self.resize(400, 600)
20.
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())
26.

```

```

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)
31.
32.         title_label = QLabel(self)
33.         title_label.setText('Sign in to FTP')
34.         title_label.setAlignment(Qt.AlignCenter)
35.
36.
37.         #用户名输入框
38.         self.username_edit = QLineEdit(self)
39.         self.username_edit.setPlaceholderText('Enter your user
name')
40.         self.username_edit.setText('yeyelcy')
41.
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 pass
word')
46.         self.password_edit.setText('123456')
47.
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 a
ddress')
53.         self.ip_edit.setText('localhost') # 设置默认值
54.
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 numb
er')
60.         self.port_edit.setText('22')
61.
62.
63.
64.         #登录按钮
65.         login_btn = QPushButton('Login', self)

```

```

66.         login_btn.clicked.connect(self.login_clint)
67.
68.         #注册按钮
69.         register_btn = QPushButton('Register', self)
70.         register_btn.clicked.connect(self.login_up)
71.
72.         btn_group = QGroupBox(self)
73.         btn_layout = QHBoxLayout(btn_group)
74.         btn_layout.addWidget(login_btn)
75.         btn_layout.addWidget(register_btn)
76.
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)
87.
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)
94.
95.         main_layout.addWidget(btn_group, stretch=1)
96.         main_layout.addStretch(4)
97.
98.
99.         # 绑定提示文本
100.        self.username_edit.setToolTip('Please enter your u
username')
101.        self.password_edit.setToolTip('Please enter your p
assword')
102.
103.
104.
105.
106.        def show_login(self):
107.            self.show()

```

```

108.
109.
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.us
122.         ername, 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
129.             有权限的服务器端 ftp
130.             self.log_client.show()
131.         else:
132.             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)
143.         # 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 -*-
2.
3.
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
10.
11.
12.
13.
14. class UploadWindow(QWidget):
15.     def __init__(self):
16.         super().__init__()
17.
18.
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)
25.
26. class DownloadWindow(QWidget):
27.     def __init__(self):
28.         super().__init__()
29.
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)
36.
37. class AccountWindow(QWidget):
38.     def __init__(self):
39.         super().__init__()
40.
```



```

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)
47.
48.
49. class Ui_MainWindow(object):
50.     def setupUi(self, MainWindow):
51.         MainWindow.setObjectName("MainWindow")
52.         MainWindow.resize(582, 384)
53.
54.         # 创建主控件
55.         self.centralwidget = QtWidgets.QWidget(MainWindow)
56.         self.centralwidget.setMinimumSize(QtCore.QSize(568, 37
9))
57.         self.centralwidget.setObjectName("centralwidget")
58.
59.         # 创建水平布局器
60.         self.horizontalLayout_3 = QtWidgets.QHBoxLayout(self.c
entralwidget)
61.         self.horizontalLayout_3.setContentsMargins(0, 0, 0, 0)
62.
63.         self.horizontalLayout_3.setSpacing(0)
64.         self.horizontalLayout_3.setObjectName("horizontalLayou
t_3")
65.
66.         # 创建分组框
67.         self.groupBox = QtWidgets.QGroupBox(self.centralwidget
)
68.         self.groupBox.setMinimumSize(QtCore.QSize(568, 379))
69.         self.groupBox.setTitle("")
70.         self.groupBox.setObjectName("groupBox")
71.
72.         # 创建垂直布局器
73.         self.verticalLayout = QtWidgets.QVBoxLayout(self.group
Box)
74.         self.verticalLayout.setContentsMargins(0, 0, 0, 0)
75.         self.verticalLayout.setSpacing(0)
76.         self.verticalLayout.setObjectName("verticalLayout")
77.
78.         # 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")
83.
84.         # 创建水平布局器
85.         self.horizontalLayout_2 = QtWidgets.QHBoxLayout(self.f
rame)
86.         self.horizontalLayout_2.setContentsMargins(0, 0, 0, 0)

87.         self.horizontalLayout_2.setSpacing(0)
88.         self.horizontalLayout_2.setObjectName("horizontalLayu
t_2")
89.
90.         # 创建水平布局器
91.         self.horizontalLayout = QtWidgets.QHBoxLayout()
92.         self.horizontalLayout.setSpacing(0)
93.         self.horizontalLayout.setObjectName("horizontalLayout"
)
94.
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)
100.
101.         # 创建按钮 2
102.         self.pushButton_2 = QtWidgets.QPushButton(self.fra
me)
103.         self.pushButton_2.setMinimumSize(QtCore.QSize(75,
23))
104.         self.pushButton_2.setObjectName("pushButton_2")
105.         self.horizontalLayout.addWidget(self.pushButton_2)

106.
107.         # 创建按钮 3
108.         self.pushButton_3 = QtWidgets.QPushButton(self.fra
me)
109.         self.pushButton_3.setMinimumSize(QtCore.QSize(75,
23))

```

```

110.         self.pushButton_3.setObjectName("pushButton_3")
111.         self.horizontalLayout.addWidget(self.pushButton_3)

112.
113.
114.         self.horizontalLayout_2.addLayout(self.horizontalL
ayout)
115.
116.         # 创建布局器项
117.         spacerItem = QtWidgets.QSpacerItem(348, 20, QtWidg
ets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
118.         self.horizontalLayout_2.addItem(spacerItem)
119.         self.verticalLayout.addWidget(self.frame)
120.
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")

125.
126.         # 添加页面 1 控件
127.         self.page = QtWidgets.QWidget()
128.         self.page.setObjectName("page")
129.
130.         self.horizontalLayout_4 = QtWidgets.QHBoxLayout(se
lf.page)
131.         self.horizontalLayout_4.setContentsMargins(0, 0, 0
, 0)
132.         self.horizontalLayout_4.setSpacing(0)
133.         self.horizontalLayout_4.setObjectName("horizontalL
ayout_4")
134.
135.         #添加一个选择文件的功能
136.         self.upload_button = QtWidgets.QPushButton(self.pa
ge)
137.         self.upload_button.clicked.connect(self.upload_fil
e)
138.         self.horizontalLayout_4.addWidget(self.upload_but
ton)
139.         self.stackedWidget.addWidget(self.page)
140.
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(se
self.page_2)
145.         self.horizontalLayout_5.setContentsMargins(0, 0, 0
, 0)
146.         self.horizontalLayout_5.setSpacing(0)
147.         self.horizontalLayout_5.setObjectName("horizontalL
ayout_5")
148.
149.
150.         self.download_button = QtWidgets.QPushButton(self.
page_2)
151.
152.         self.download_button.clicked.connect(self.download
_file)
153.         self.horizontalLayout_5.addWidget(self.download_bu
tton)
154.
155.         self.stackedWidget.addWidget(self.page_2)
156.
157.
158.
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(se
lf.page_2)
163.         self.horizontalLayout_6.setContentsMargins(0, 0, 0
, 0)
164.         self.horizontalLayout_6.setSpacing(0)
165.         self.horizontalLayout_6.setObjectName("horizontalL
ayout_5")
166.
167.         self.user_info=QtWidgets.QPushButton(self.page_3)
168.
169.         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)
171.
172.         self.stackedWidget.addWidget(self.page_3)
173.
174.

```

```

175.         #添加整体的
176.         self.verticalLayout.addWidget(self.stackedWidget)

177.         self.horizontalLayout_3.addWidget(self.groupBox)
178.         MainWindow.setCentralWidget(self.centralwidget)
179.
180.         self.retranslateUi(MainWindow)
181.         self.stackedWidget.setCurrentIndex(1)
182.         QtCore.QMetaObject.connectSlotsByName(MainWindow)

183.
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)
188.
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")
200.
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)")
204.
205.             print("Download file path:", down_path)
206.             down_path=down_path.split('/')[ -1]
207.
208.
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(MainWindo
w.ftp, './local_file_path', down_path)
213.
214.         else:
215.             print('报错')
216.
217.     def user_acount(self):
218.         print("用户信息为",MainWindow.user_log.username)
219.
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("MainWindo
w", "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)
236.
237.
238.             # self.ftp=ftplib.FTP()#ftp 对象
239.             self.pushButton_1.clicked.connect(lambda: self.sta
ckedWidget.setCurrentIndex(0))
240.             self.pushButton_2.clicked.connect(lambda: self.sta
ckedWidget.setCurrentIndex(1))
241.             self.pushButton_3.clicked.connect(lambda: self.sta
ckedWidget.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();
3.
4.  --创建一个数据库
5.  create database if not exists ftp_info;
6.
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. );
16.
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. );
24.
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. );
34.
35.
36. --设置最大连接数
37. SET GLOBAL max_connections =100;
```

