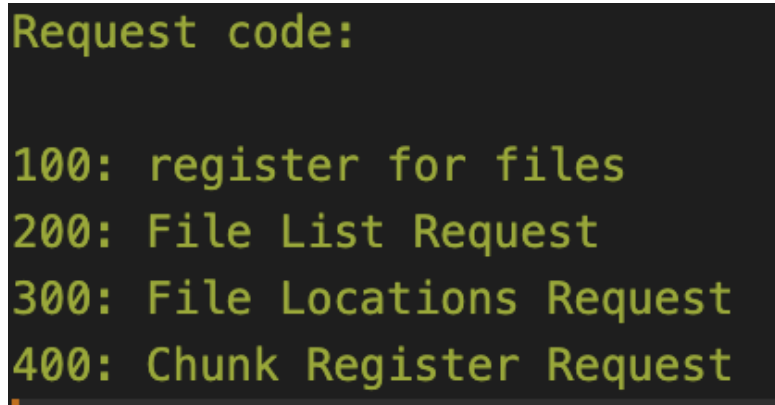


Detailed Description:

My P2P file system has two main components: Server, peer. Server serves as the central control for all the information in the network, and peers will call on server to register or request different kind of information like chunk request and file location. Peers will initialize different request on server through status code. The status codes along with their protocol are shown below.



```
Request code:

100: register for files
200: File List Request
300: File Locations Request
400: Chunk Register Request
```

If the peer wish to request file from other peers, it will obtain peer's ip address and port, then it will directly connect with that peer with multithreading mechanism.

Structure Description:

The main files responsible for peer and file structure are node.py and file.py. Each of them contains many helper function to let other services update or retrieve information contained in each peer and each file.

server.py contains most of the logic for the server side, includes all the operation request from peers. Server can accept multiple connection at the same time, so no peer will need to wait for long time to connect.

Client.py contains most of the information for peer client side logic, includes send request to server, manage local file information, and request file chunk from other peers.

client_runner.py is what user will use to request/update information from peer and server. Detailed command operations will be shown later.

Client_reciever.py is where peer handle request from other peer. This class verify the information, and send corresponding chunk information back to other peers. If the sending is successful, it will register chunk on the server, so other peer can request chunk from this new peer.

Config.py this is the file responsible for all the global variable like server ip address, and server port.

Notice: since this is a multi file program, defining a global variable to store all local file information is not easy to achieve, so I added 2 pickle file "server_data.pkl" and "peer_local_storage.pkl" as the local mini database for all information updated.

Project Status Report:

As far as now, every component works correctly. Since I'm doing this project with Yang Xu, I also added rarest first request. Everything has been tested. However, we encountered some problem when running our system on window's computer, and this is being further tested.

Output:

```

(P2P File Sharing) paradox@Jianxiangs-MacBook-Pro P2P File Sharing % python client_runner.py --reg
104.38.105.225
Please enter all file path you wish to register: (Seperate by a single space)
Please enter the dirctory you want to upload: (Seperate by a single space)Files
INVALID FILE TYPE
8841312
39509576
395976
395976
4420656
All file registered successfully
['Files/.DS_Store', 'Files/out.pdf', 'Files/test.mp4', 'Files/out.jpg', 'Files/test.png', 'Files/test.pdf']
(P2P File Sharing) paradox@Jianxiangs-MacBook-Pro P2P File Sharing % python client_runner.py --reg
104.38.105.225
Please enter all file path you wish to register: (Seperate by a single space)
Please enter the dirctory you want to upload: (Seperate by a single space)Files
INVALID FILE TYPE
8841312
39509576
(P2P File Sharing) paradox@Jianxiangs-MacBook-Pro P2P File Sharing % python client_runner.py --reg
104.39.65.25
Please enter all file path you wish to register: (Seperate by a single space)
Please enter the dirctory you want to upload: (Seperate by a single space)Files
INVALID FILE TYPE
8841312
39509576
395976
395976
4420656
All file registered successfully
['Files/.DS_Store', 'Files/out.pdf', 'Files/test.mp4', 'Files/out.jpg', 'Files/test.png', 'Files/test.pdf']
(P2P File Sharing) paradox@Jianxiangs-MacBook-Pro P2P File Sharing % python client_runner.py --file_list_request
104.39.65.25
Files/.DS_Store : 0
Files/out.pdf : 8841312
Files/test.mp4 : 39509576
Files/out.jpg : 395976
Files/test.png : 395976
Files/test.pdf : 4420656
total_file : 6
(P2P File Sharing) paradox@Jianxiangs-MacBook-Pro P2P File Sharing % python client_runner.py --file_location
104.39.65.25
input the filename to search for locations: Files/out.jpg
IP Addr: 104.39.65.25 Port: 61057 Available Chunks: [0, 1, 2, 3]
(P2P File Sharing) paradox@Jianxiangs-MacBook-Pro P2P File Sharing % python client_runner.py --download
104.39.65.25
Enter the file you wish to download: (Select extractly like file list): Files/out.jpg
File already exsited locally!

```

Program Compile Instruction:

First set up the environment:

- Install pipenv in your system first
- Cd to program root folder
- Run: pipenv install
- Run: pipenv shell

Open 3 separate terminal window:

First window: run: “python server.py” (this is to start the server)

Second window: run “python client_reciever.py” (this is to accept incoming peer connection)

Third window: in this window, you can enter any of following command to request service from this program

To register local file:

```
python client_runner.py --reg
```

To request all file list:

```
python client_runner.py --file_list_request
```

To request file location of some file

```
python client_runner.py --file_location
```

To request a download:

```
python client_runner.py --download
```