P2P File Sharing System Report

沈泓立（12311016）

Overview

This project implements a simplified BitTorrent-like P2P file sharing system where seeders share files through a tracker-coordinated network, and requesters download files by querying the tracker for available seeders.

- **Tracker.py**: The tracker acts as a central coordination server that records and manages file information (e.g., `info_hash` and file lists) registered by seeders, along with their addresses/ports, and returns the corresponding seeder list upon requester queries.
- **Seeder.py**: The seeder is a file provider responsible for generating and sharing torrent files, computing `info_hash`, registering file details with the tracker, and serving file downloads to requesters.
- **Requester.py**: The requester is a file downloader that parses torrent files to obtain `info_hash`, requests the seeder list from the tracker, and downloads the target file from the selected seeder(s).

Initial Project Stucture

```
~/Desktop/Assignment_1/
├── Assignment_1.pdf        # Project documentation/report
├── example.txt             # Example data/configuration file
├── requester.py            # Downloader (client) implementation
├── seeder.py               # Resource provider (server) implementation
└── Tracker.py              # Central coordination server (Tracker) implementation
```

Final Project Structure

```
~/Desktop/Ass/Assignment_1/
├── peer_6881/                      # seeder folder
│   ├── example.txt
│   └── example.txt.torrent
│
├── peer_6882/                      # requester folder
│   ├── example.txt (ro82)
│   └── example.txt.torrent
│
├── Assignment_1.pdf                # Project documentation (root only)
├── example.txt                     # Root-level shared file (optional)
├── example.txt.torrent             # Root-level torrent file (optional)
├── requester.py                    # Downloader client implementation
├── seeder.py                       # File provider implementation
└── Tracker.py                      # Central tracker server implementation
```

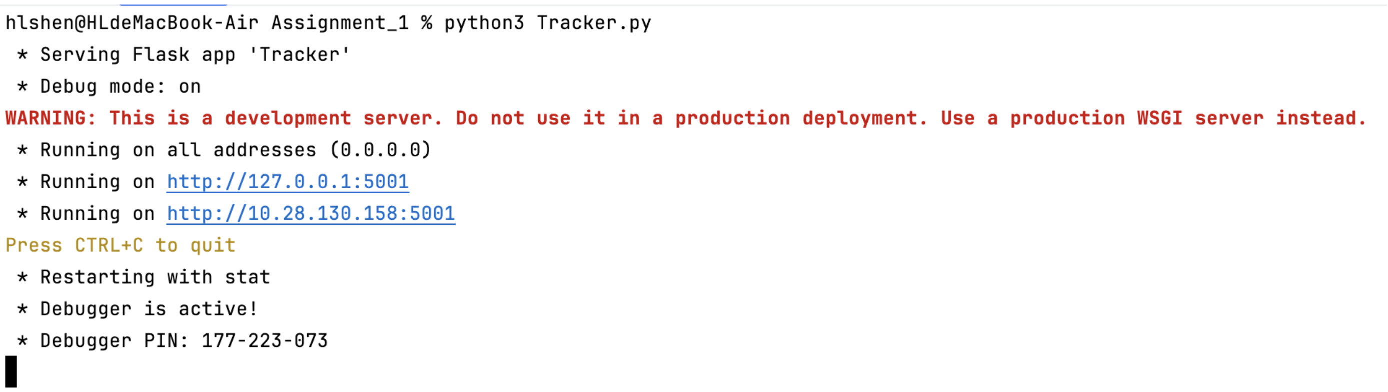
Execution

This project is executed under `python` environment.

Execute Tracker.py

```
python3 Tracker.py
#python Tracker.py
```

Expected output:

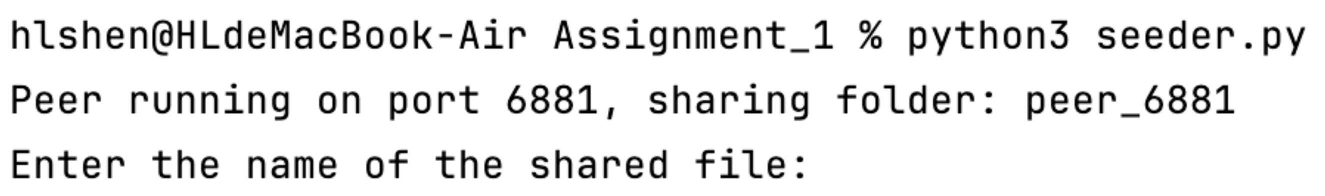


```
hlshen@HLdeMacBook-Air Assignment_1 % python3 Tracker.py
 * Serving Flask app 'Tracker'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5001
 * Running on http://10.28.130.158:5001
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 177-223-073
```

Execute seeder.py

```
python3 seeder.py
#python seeder.py
```

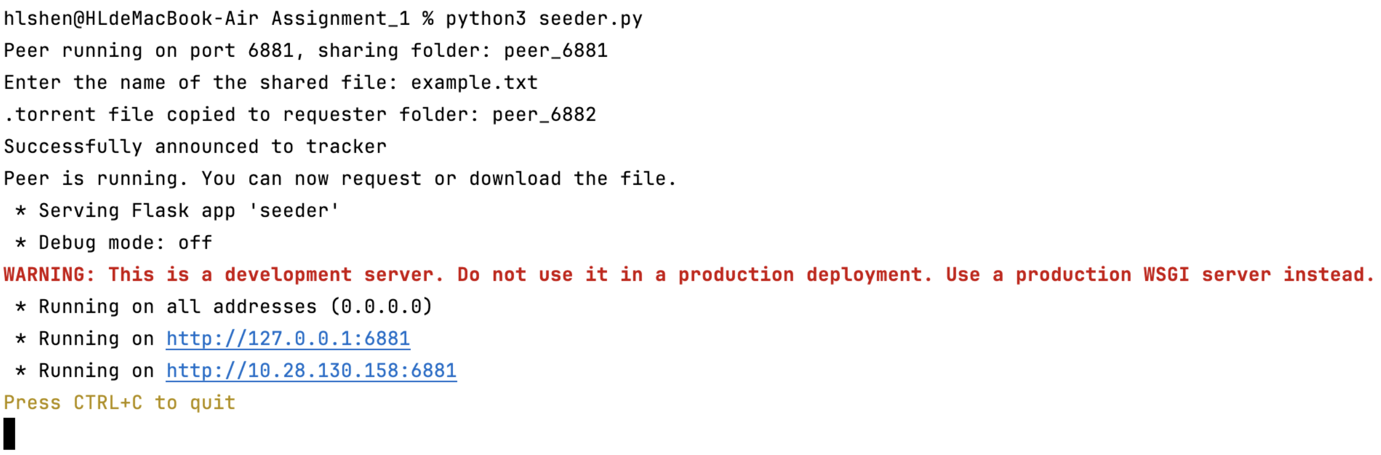
Expected output:



```
hlshen@HLdeMacBook-Air Assignment_1 % python3 seeder.py
Peer running on port 6881, sharing folder: peer_6881
Enter the name of the shared file:
```

Then we need to enter the name of the file. (e.g example.txt)

Expected output:



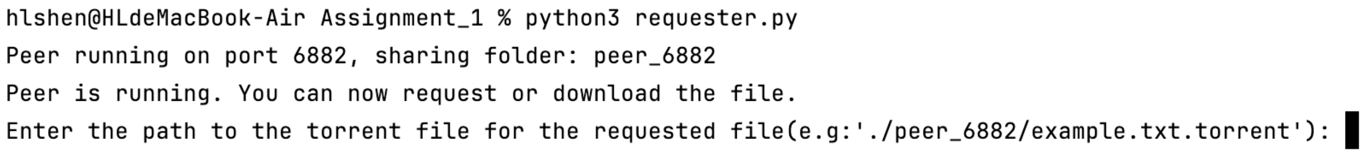
```
hlshen@HLdeMacBook-Air Assignment_1 % python3 seeder.py
Peer running on port 6881, sharing folder: peer_6881
Enter the name of the shared file: example.txt
.torrent file copied to requester folder: peer_6882
Successfully announced to tracker
Peer is running. You can now request or download the file.
 * Serving Flask app 'seeder'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:6881
 * Running on http://10.28.130.158:6881
Press CTRL+C to quit
```

After the above operations, `.torrent` file generates.

Execute requester.py

```
python3 requester.py
#python requester.py
```

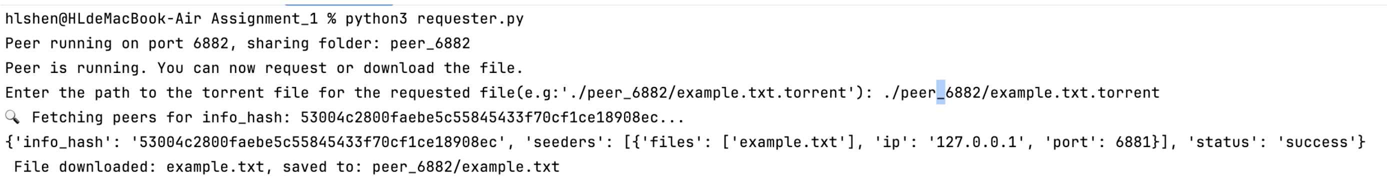
Expected output:



```
hlshen@HLdeMacBook-Air Assignment_1 % python3 requester.py
Peer running on port 6882, sharing folder: peer_6882
Peer is running. You can now request or download the file.
Enter the path to the torrent file for the requested file(e.g:'./peer_6882/example.txt.torrent'): 
```

Then we need to enter the file path. (e.g ./peer_6882/example.txt.torrent)

Expected output:



```
hlshen@HLdeMacBook-Air Assignment_1 % python3 requester.py
Peer running on port 6882, sharing folder: peer_6882
Peer is running. You can now request or download the file.
Enter the path to the torrent file for the requested file(e.g:'./peer_6882/example.txt.torrent'): ./peer_6882/example.txt.torrent
🔍 Fetching peers for info_hash: 53004c2800faebe5c55845433f70cf1ce18908ec...
{'info_hash': '53004c2800faebe5c55845433f70cf1ce18908ec', 'seeders': [{'files': ['example.txt'], 'ip': '127.0.0.1', 'port': 6881}], 'status': 'success'}
 File downloaded: example.txt, saved to: peer_6882/example.txt
```

Then we have successfully downloaded the file of `example.txt`.