

ĐẠI HỌC QUỐC GIA TP.HCM
ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH



MẠNG MÁY TÍNH (CO3094)

BÁO CÁO BÀI TẬP LỚN 1

Develop a Network Application

GVHD:	Bùi Xuân Giang	
SV thực hiện:	Nguyễn Văn Đạt	2115397
	Nguyễn Trung Hiếu	2113358
	Hồ Chí Anh Khởi	2113791

Thành phố Hồ Chí Minh, tháng 11 năm 2024



Mục lục

1	Member list & Workload	2
2	Mô tả các chức năng	3
2.1	Tracker HTTP Protocol	3
2.2	Peer download	3
2.3	Peer upload	4
2.4	Mở rộng	5
3	Mô tả các giao thức truyền thông	7
3.1	Tracker HTTP Protocol	7
3.2	Peer download	8
3.3	Peer upload	8
4	Đặc tả các lớp và chức năng của ứng dụng	9
4.1	Class Peer (trong peer.py)	9
4.2	Class Tracker (trong tracker.py)	10
4.3	Hàm hỗ trợ trong transform.py	10
4.4	Luồng thực thi chính của ứng dụng	10
4.5	Thiết kế chi tiết ứng dụng	11
5	Kiểm thử và đánh giá	12
6	Hướng dẫn sử dụng	14
6.1	Ngôn ngữ và thiết	14
6.2	Khởi tạo môi trường	14
6.3	Khởi tạo Tracker và Peer	14
6.4	Cách chạy:	14
6.5	Chú ý	14



1 Member list & Workload

Lớp	Họ và tên	MSSV	Công việc	Đánh giá
L09	Nguyễn Văn Đạt	2115397	Thiết kế và hiện thực ứng dụng.	100%
L09	Nguyễn Trung Hiếu	2113358	Phát triển ứng dụng	100%
L09	Hồ Chí Anh Khôi	2113791	Đánh giá ứng dụng	100%

2 Mô tả các chức năng

2.1 Tracker HTTP Protocol

BitTorrent là một giao thức chia sẻ tệp tin, triển khai công nghệ ngang hàng (peer-to-peer). Ý tưởng chính là giảm áp lực cho một máy chủ duy nhất khi số lượng người dùng tăng lên, bằng cách chia tệp thành các phần nhỏ hơn. Một phần được tải xuống bởi một máy khách sau đó được chia sẻ để tải lên bởi máy khách đó. Các thành phần chính trong giao thức này bao gồm:

- **Tệp Metainfo:** Một tệp được sử dụng để mô tả dữ liệu về các tệp tin đang được chia sẻ và máy chủ theo dõi, giúp các máy khách và máy chủ theo dõi quản lý quá trình tải xuống. Nó còn được gọi là tệp torrent, với phần mở rộng .torrent.
- **Tracker:** Máy chủ trung gian theo dõi các peer và giúp điều phối quá trình chia sẻ tệp giữa các peer.
- **Peer:** Người dùng trong mạng P2P, vừa tải xuống tệp, vừa chia sẻ các phần đã tải xuống với các peer khác.
- **Seed:** Peer có bản sao hoàn chỉnh của tệp và chia sẻ nó với các peer khác. Ít nhất một seed cần thiết để duy trì khả năng chia sẻ tệp.
- **Client:** Phần mềm sử dụng giao thức BitTorrent để tải xuống và chia sẻ tệp từ các peer khác.

Quy trình chia sẻ một tệp tin được triển khai gần đúng như sau:

1. Một tệp Metainfo (torrent) chứa thông tin về tệp và máy chủ Tracker được tạo ra, thường trên trang web chia sẻ.
2. Người dùng tải tệp Metainfo về và mở bằng ứng dụng Client BitTorrent. Client kết nối với Tracker để thông báo sự hiện diện và nhận danh sách các Peer đang chia sẻ hoặc tải xuống tệp.
3. Các Peer trao đổi thông tin và tải xuống các phần tệp còn thiếu. Khi một phần tải xong, nó sẽ được chia sẻ lại với các Peer khác.
4. Khi tệp hoàn chỉnh đã tải xong, Peer có thể tiếp tục chia sẻ tệp hoặc ngừng kết nối.

2.2 Peer download

1. Máy chủ truy xuất file text magnet để truy xuất magnet link, hoặc file metainfo.
2. Từ magnet link hoặc metainfo, máy chủ truy xuất được đường dẫn URL tracker server.
3. Máy chủ sử dụng URL đó để kết nối với tracker, đăng ký với tracker để trở thành 1 peer trong hệ thống torrent của tracker đó.
 - Máy chủ sử dụng giao thức ở tầng ứng dụng (application-layer protocol) là HTTP để truy cập tracker server nhờ vào URL của tracker. Cụ thể hơn, HTTP này là loại non-persistent: kết nối TCP giữa tracker và máy chủ sẽ được đóng sau mỗi phiên yêu cầu (request) từ máy chủ và trả lời (response) từ tracker. Bởi vì, thông thường giao tiếp giữa tracker và các máy chủ sẽ chỉ diễn ra 1 lần trong mỗi phiên torrent (không tính yêu cầu đăng ký vào hệ thống torrent lúc đầu của máy chủ).

4. Máy chủ (hiện đã là 1 peer trong hệ thống torrent của tracker này) nhận phản hồi từ tracker là danh sách các peer nắm giữ các phần của file cần tải xuống.
 - Tracker sử dụng đúng giao thức HTTP được nêu ở trên để phản hồi đến máy chủ.
5. Máy chủ cố gắng kết nối với càng nhiều peer trong số đó càng tốt.
 - Máy chủ kết nối với các peer đó bằng giao thức ở tầng vận chuyển (transport-layer protocol) là TCP. Cụ thể hơn, đây là TCP bắt tay 2 chiều trước khi kết nối: máy chủ gửi yêu cầu kết nối tới các peer bằng gói tin SYN để yêu cầu kết nối và các peer phản hồi lại máy chủ bằng gói tin SYN-ACK nhằm xác nhận đã nhận được gói SYN và chấp nhận kết nối; lúc này kết nối TCP giữa máy chủ và các peer đôi một khác nhau được hình thành (máy chủ đó kết nối với mỗi peer một phiên TCP riêng biệt).
6. Máy chủ bắt đầu tải xuống các khối (piece) từ các peer đã kết nối cùng lúc.
 - Việc này được thực hiện trên giao thức TCP được nêu ở trên.
 - Yêu cầu thêm:
 - Máy chủ đã có được danh sách các peer trong torrent hiện tại và biết được các peer nào nắm giữ khối nào của file. Ngoài ra, máy chủ cũng duy trì một danh sách gồm các khối “đã được gửi yêu cầu tải xuống” nhằm tránh việc gửi yêu cầu quá nhiều một cách không cần thiết đến các peer cho cùng 1 khối.
 - Lí do cần danh sách này: do máy chủ tiến hành gửi yêu cầu liên tục cho từng khối tới các peer bằng những kết nối TCP độc lập nhau, một khối bắt đầu tải xuống mà không cần các khối trước (theo index) hoàn thành tải xuống, dẫn đến thiếu sự đồng bộ giữa các phiên TCP và có thể tạo ra nhiều yêu cầu cho cùng 1 khối.
 - Cách hoạt động của danh sách này: máy chủ sẽ kiểm tra danh sách xem khối đó có trong đó hay không. Nếu có tức là yêu cầu cho khối đó đã được gửi nên sẽ không gửi nữa mà chờ cho khối đó được tải xuống. Nếu không thì khối đó chưa được gửi yêu cầu, nên máy chủ sẽ gửi yêu cầu tải xuống khối đó tới peer đang nắm giữ.
7. Máy chủ đóng kết nối TCP với các peer nếu như nó đã tải xong phần được các peer đó chia sẻ.
8. Máy chủ sắp xếp lại các phần theo đúng thứ tự.
9. Sau khi đã sắp xếp, máy chủ lưu vào 1 file.
10. Máy chủ cập nhật trạng thái đã hoàn thành tải file cho tracker.
 - Việc cập nhật trạng thái này được tiến hành trên giao thức HTTP non-persistent như đã nêu ở trên, vì phản hồi từ tracker về lại máy chủ là không cần thiết, hoặc kể cả khi có thì sau lời phản hồi đó cũng không có phiên yêu cầu - phản hồi nào khác.

2.3 Peer upload

1. Chuẩn bị dữ liệu file và metadata:
 - Máy chủ (uploading peer) cần chuẩn bị dữ liệu của file cần chia sẻ và tạo ra metadata (ví dụ: metainfo file hoặc magnet link) để mô tả thông tin về file.
2. Kết nối với tracker server:

- Sử dụng thông tin từ metadata, máy chủ gửi một HTTP request đến tracker server để đăng ký và trở thành một peer trong hệ thống torrent.
 - Quá trình này sử dụng giao thức HTTP non-persistent, với các kết nối TCP được đóng sau mỗi phiên yêu cầu và phản hồi.
3. Đăng ký và cập nhật trạng thái với tracker:
- Máy chủ gửi yêu cầu đăng ký (register request) tới tracker server, cung cấp thông tin về file mà nó sẵn sàng chia sẻ.
 - Tracker server ghi nhận máy chủ là một peer chia sẻ và cập nhật danh sách các file mà peer này đang chia sẻ.
4. Nhận yêu cầu tải xuống từ peer tải:
- Máy chủ nhận được yêu cầu tải xuống (download request) từ các peer tải về thông qua tracker.
 - Yêu cầu này bao gồm thông tin về file mà peer tải về muốn nhận từ máy chủ.
5. Chấp nhận kết nối với peer tải
- Máy chủ chấp nhận kết nối từ peer tải về và bắt đầu gửi các phần của file được yêu cầu tới peer này.
 - Quá trình này diễn ra qua giao thức TCP, với các kết nối TCP được thiết lập qua quá trình bắt tay 3 bước.
6. Truyền dữ liệu file cho peer tải:
- Máy chủ bắt đầu truyền (uploading) các phần của file được yêu cầu tới peer tải về thông qua kết nối TCP đã thiết lập.
 - Mỗi phần của file được chuyển giao từ máy chủ tới peer tải về theo yêu cầu của peer.
7. Đóng kết nối TCP với peer tải:
- Sau khi máy chủ đã truyền xong các phần của file cho peer tải về, đóng kết nối TCP với peer này.
 - Quá trình này diễn ra sau khi peer tải về đã nhận đủ dữ liệu cần thiết từ máy chủ.
8. Cập nhật trạng thái và tiếp tục chia sẻ:
- Máy chủ cập nhật trạng thái đã chia sẻ xong file với tracker thông qua một yêu cầu HTTP.
 - Sau khi cập nhật trạng thái, máy chủ tiếp tục lắng nghe và chấp nhận kết nối từ các peer tải về khác để chia sẻ dữ liệu.

2.4 Mở rộng

Cách tiếp cận downloading/seeding: Rarest-first

- Máy chủ sẽ xem trong các khối chưa được gửi yêu cầu (các khối không có trong danh sách khối “đã gửi yêu cầu”), xác định khối có ít peer trong torrent đang giữ nhất và tiến hành gửi yêu cầu cho khối đó tới các peer giữ nó. Thực hiện đồng thời với việc gửi yêu cầu cho khối đó, máy chủ thêm khối đó vào danh sách đã nêu trên và tiếp tục thực hiện như vậy. Về mặt tổng quát, khối càng có ít peer trong torrent đang giữ thì sẽ càng được ưu tiên gửi yêu cầu nhất.



- Cách tiếp cận này làm tăng hiệu suất trong tải xuống toàn bộ file vì nó làm giảm lượng tải xuống của các khối được yêu cầu đầu tiên (do càng ít peer đang giữ khối đó thì càng ít peer phản hồi khối đó về lại máy chủ). Các khối đầu tiên sẽ không bị các khối sau cùng (có nhiều peer phản hồi) cản trở việc tải xuống.

3 Mô tả các giao thức truyền thông

3.1 Tracker HTTP Protocol

Giao thức theo dõi (tracker protocol) được triển khai trên giao thức HTTP/HTTPS. Điều này có nghĩa là tracker sẽ chạy một máy chủ HTTP hoặc HTTPS và có hành vi được mô tả dưới đây:

1. Client gửi một yêu cầu GET đến URL của Tracker, với một số biến cụ thể và các giá trị được thêm vào URL.
2. Client nhận trả lời từ Tracker là một tài liệu dạng "text/plain".
3. Client sau đó gửi các yêu cầu request đến Tracker và Tracker sẽ phản hồi.

Các biến và giá trị được thêm vào URL cơ sở bởi Client gửi yêu cầu GET là:

- **info_hash**: Giá trị hash SHA1 20 byte được tính từ giá trị mà khóa info ánh xạ đến trong tệp metainfo.
- **peer_id**: Một id dài 20 ký tự của Client đang tải xuống, được tạo ngẫu nhiên ở đầu mỗi lần tải xuống.
- **ip**: Đây là một biến tùy chọn, cung cấp địa chỉ IP của Client.
- **port**: Port number mà Client đang liên kết.
- **uploaded**: Lượng dữ liệu đã tải lên cho đến nay bởi Client.
- **left**: Số byte còn lại phải tải xuống để hoàn thành.
- **event**: Một biến tùy chọn, tương ứng với một trong bốn khả năng:
 - **started**: Gửi khi Client bắt đầu tải xuống.
 - **stopped**: Gửi khi Client dừng tải xuống.
 - **completed**: Gửi khi việc tải xuống hoàn thành. Nếu việc tải xuống đã hoàn thành ngay từ lúc khởi đầu, thì không nên gửi thông điệp này.

Như đã đề cập trước đó, phản hồi là một phản hồi "text/plain":

- **failure reason**: Nếu khóa này tồn tại, không có khóa nào khác được bao gồm. Giá trị được ánh xạ với khóa này là một chuỗi dễ đọc giải thích lý do tại sao kết nối thất bại.
- **interval**: Số giây mà Client nên chờ giữa các lần yêu cầu rerequest.
- **peers**: Ánh xạ sang một danh sách các từ điển, mỗi từ điển đại diện cho một Peer tham gia, trong đó mỗi từ điển có các khóa sau:
 - **peer_id**: Id của Peer tham gia đang được xem xét. Tracker đã thu được thông tin này thông qua biến peer_id trong yêu cầu GET được gửi đến theo dõi.
 - **ip**: Địa chỉ của Peer tham gia, là địa chỉ IP hoặc tên miền DNS.
 - **port**: Port number mà Peer tham gia đang liên kết trên đó.

Thông điệp	Chức năng
RESPONSE_REQUEST <status>	Sau khi nhận yêu cầu kết nối từ các peer thì gửi trả lại peer đó trạng thái đã kết nối thành công chưa.
RESPONSE_DOWNLOAD <list các port và ip>	Sau khi nhận yêu cầu tải file, tracker trả về danh sách peer có thể kết nối để gửi file.

Bảng 1: Protocol cho Tracker HTTP Protocol

3.2 Peer download

Thông điệp	Chức năng
REGISTER_REQUEST <file metadata>	Yêu cầu đăng ký và trở thành một peer trong hệ thống torrent với thông tin về file metadata.
DOWNLOAD_REQUEST <tên file>	Yêu cầu tải file đến tracker.
GET_FILE <địa chỉ của peer> <tên file>	Yêu cầu kết nối và tải file đến peer mà tracker trả địa chỉ.

Bảng 2: Protocol cho Peer Download

3.3 Peer upload

Thông điệp	Chức năng
REGISTER_REQUEST <file metadata>	Yêu cầu đăng ký và trở thành một peer trong hệ thống torrent với thông tin về file metadata.
RESPONSE <piece data>	Phản hồi chấp nhận kết nối với peer tải và gửi đi dữ liệu khối của file được yêu cầu bởi peer tải.

Bảng 3: Protocol cho Peer Upload

4 Đặc tả các lớp và chức năng của ứng dụng

4.1 Class Peer (trong peer.py)

Đại diện cho Peer trong kiến trúc mạng ngang hàng P2P, có khả năng kết nối tới Tracker Server, tải lên cái tệp tin, lắng nghe và kết nối với các Peer khác, xử lý các yêu cầu từ các Peer khác.

1. **find_empty_port:** Hàm này tìm một cổng trống trong một phạm vi cụ thể mà peer có thể sử dụng cho giao tiếp.
2. **write_string_to_file:** Nó ghi một chuỗi vào một tệp trong một thư mục cụ thể, đảm bảo mỗi chuỗi được ghi trên một dòng riêng biệt và tránh trùng lặp.
3. **create_torrent_file:** Hàm này tạo một tệp torrent cho một đường dẫn tệp đã cho bằng cách sử dụng URL tracker đã chỉ định và lưu nó trong một thư mục.
4. **upload_torrent_file:** Nó tải một tệp torrent lên một tracker bằng cách sử dụng URL tracker đã chỉ định.
5. **download_torrent_file:** Hàm này tải một tệp torrent từ một tracker bằng cách sử dụng đường dẫn tệp torrent đã cung cấp và lưu tệp đã tải xuống vào đích đã chỉ định.
6. **download_range:** Nó tải một phạm vi các phần của một tệp torrent từ các địa chỉ IP đã chỉ định.
7. **download_piece:** Hàm này tải một phần cụ thể của một tệp torrent từ một peer.
8. **parse_peer_message:** Nó phân tích các tin nhắn nhận được từ một peer trong quá trình giao tiếp.
9. **handle_peer_request:** Hàm này xử lý các yêu cầu nhận được từ các peer trong quá trình giao tiếp.
10. **read_strings_from_file:** Nó đọc các chuỗi từ một tệp trong một thư mục cụ thể.
11. **find_file_by_infohash:** Hàm này tìm một tệp dựa trên info hash của nó nhận được từ một peer.
12. **create_unchoke_message:** Nó tạo một tin nhắn để mở chặn một peer trong quá trình giao tiếp.
13. **process_request:** Hàm này xử lý các yêu cầu nhận được từ các peer trong quá trình giao tiếp.
14. **merge_temp_files:** Nó hợp nhất các tệp tạm thời tạo ra trong quá trình tải xuống thành một tệp duy nhất.
15. **get_local_ip:** Hàm này lấy địa chỉ IP cục bộ của peer.

4.2 Class Tracker (trong tracker.py)

Đại diện cho Tracker Server trung tâm, lắng nghe và chấp nhận kết nối từ các Peer, có cơ chế phân phối các tệp tin cho các Peer, duy trì thông tin về các Peer đang kết nối và các tệp tin đã được chia sẻ, cung cấp cho Peer thông tin về tệp tin mà Peer muốn tải xuống

- **Đại diện cho Tracker Server trung tâm:** Lắng nghe và chấp nhận kết nối từ các Peer, có cơ chế phân phối các tệp tin cho các Peer, duy trì thông tin về các Peer đang kết nối và các tệp tin đã được chia sẻ, cung cấp cho Peer thông tin về tệp tin mà Peer muốn tải xuống.
- **TrackerRequestHandler:** Lớp này xử lý các yêu cầu gửi đến máy chủ tracker, bao gồm yêu cầu `/announce/upload` và `/announce/download`.
- **`_update_seeder()`:** Phương thức này cập nhật thông tin về các seeder cho một tệp torrent cụ thể.
- **`find_and_print_line()`:** Phương thức này tìm kiếm và in ra dòng chứa chuỗi mục tiêu trong một tệp văn bản.
- **`get_local_ip()`:** Hàm này trả về địa chỉ IP cục bộ của máy tính.
- **`start_tracker()`:** Hàm này khởi động máy chủ tracker trên một cổng được chỉ định và lắng nghe yêu cầu.

4.3 Hàm hỗ trợ trong transform.py

- **`create_torrent`:** Hàm này tạo ra một tệp torrent từ một tệp tin cục bộ. Nó tính toán các giá trị hash của từng phần của tệp và tạo ra thông tin torrent dưới dạng một dictionary, sau đó mã hóa dictionary đó bằng bencode và ghi vào một tệp mới.
- **`get_info_hash`:** Hàm này tính toán và trả về giá trị hash (info hash) của một tệp torrent mà không cần tạo tệp torrent thực sự. Nó cũng tính toán các giá trị hash của từng phần của tệp và sử dụng chúng để tạo thông tin torrent dưới dạng một dictionary, sau đó mã hóa dictionary đó và trích xuất giá trị hash từ nó.
- **Các hàm phụ trợ decode và `bytes_to_str`:** Các hàm này chuyển đổi dữ liệu từ dạng bytes sang dạng chuỗi và giải mã dữ liệu sử dụng bencode.

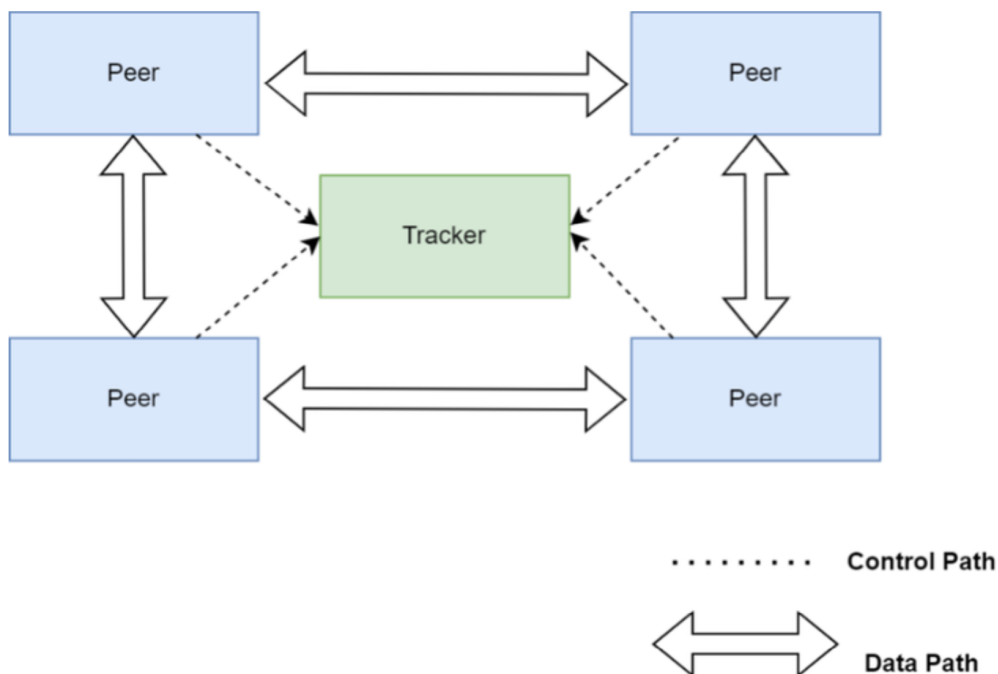
4.4 Luồng thực thi chính của ứng dụng

1. Khởi tạo Tracker và các Peer.
2. Tracker được bắt đầu và lắng nghe các kết nối từ Peer.
3. Peer tạo file torrent cho file muốn chia sẻ.
4. Peer gửi yêu cầu được chia sẻ file bằng cách gửi file torrent đến địa chỉ HTTP của Tracker.
5. Tracker chấp nhận kết nối từ các Peer.
6. Tracker nhận yêu cầu được tải về file từ một file torrent do một Peer nào đó gửi đến.
7. Tracker gửi về cho Peer đó các địa chỉ của Peer khác chia sẻ file.

8. Peer lắng nghe các kết nối từ Peer khác.
9. Xử lý các lệnh người dùng hoặc yêu cầu đến từ các Peer.
10. Thoát chương trình.

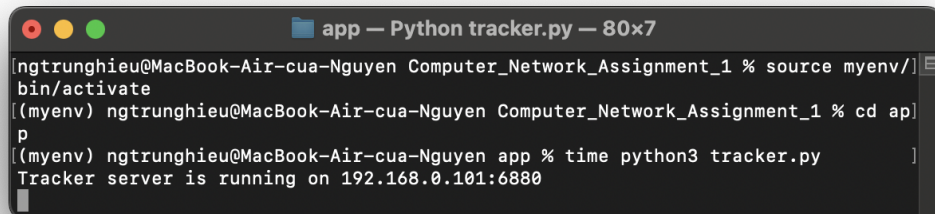
4.5 Thiết kế chi tiết ứng dụng

Ứng dụng tuân theo kiến trúc chia sẻ tệp ngang hàng (P2P) với Tracker Server trung tâm. Các Peer kết nối với Tracker để thông báo và tải lên thông tin các tệp được chia sẻ cũng như tải xuống các tệp từ các Peer khác. Tracker phân phối các thông tin về Peer giữ file khi nhận được yêu cầu từ một Peer nào đó. Peer và các Peer giữ file được kết nối và các mẫu của file được chia sẻ.



Hình 1: Kiến trúc ứng dụng

5 Kiểm thử và đánh giá



Hình 2: Khởi tạo tracker thành công

```
1 source myenv/bin/activate
```

`source myenv/bin/activate`: Kích hoạt môi trường ảo Python (virtual environment) có tên `myenv`. Môi trường ảo giúp cô lập các thư viện và phụ thuộc của dự án, tránh xung đột với các dự án khác.

```
1 cd app
```

`cd app`: Chuyển đến thư mục `app` trong hệ thống tệp. Thao tác này giúp di chuyển vào thư mục chứa mã nguồn hoặc dự án cần làm việc.

```
1 time python3 tracker.py
```

`time python3 tracker.py`: Chạy script Python `tracker.py` với phiên bản Python 3 và đo thời gian thực thi của lệnh. Lệnh `time` sẽ ghi lại thời gian bắt đầu, kết thúc và tổng thời gian chạy của chương trình.

Sau đó tạo một terminal mới để chạy peer

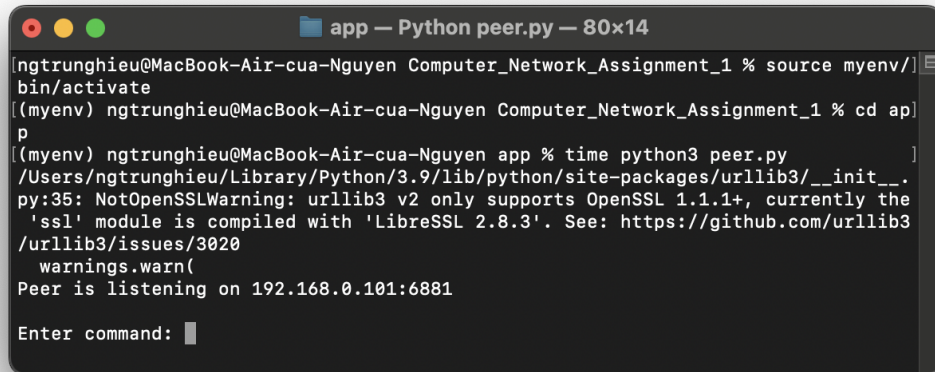
```
1 time python3 peer.py
```

`time python3 peer.py`: Chạy script Python `peer.py` với phiên bản Python 3 và đo thời gian thực thi của lệnh. Lệnh `time` sẽ hiển thị thời gian bắt đầu, kết thúc và tổng thời gian chạy của chương trình.

```
1 create /Users/ngtrunghieu/Downloads/Computer_Network_Assignment_1/video.mp4 /Users
/ngtrunghieu/Downloads/Computer_Network_Assignment_1/app/metainfo http://
192.168.0.101:6880/announce
```

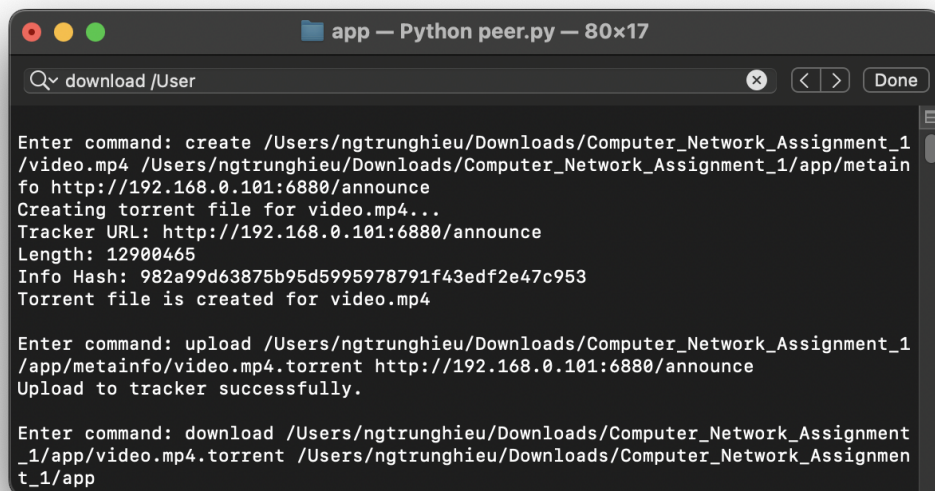
```
1 upload /Users/ngtrunghieu/Downloads/Computer_Network_Assignment_1/app/metainfo/
video.mp4.torrent http://192.168.0.101:6880/announce
```

```
1 download /Users/ngtrunghieu/Downloads/Computer_Network_Assignment_1/app/metainfo/
video.mp4.torrent /Users/ngtrunghieu/Downloads/Computer_Network_Assignment_1/
app
```



```
app — Python peer.py — 80x14
ngtrunghieu@MacBook-Air-cua-Nguyen Computer_Network_Assignment_1 % source myenv/bin/activate
(myenv) ngtrunghieu@MacBook-Air-cua-Nguyen Computer_Network_Assignment_1 % cd app
(myenv) ngtrunghieu@MacBook-Air-cua-Nguyen app % time python3 peer.py
/Users/ngtrunghieu/Library/Python/3.9/lib/python/site-packages/urllib3/__init__.py:35: NotOpenSSLWarning: urllib3 v2 only supports OpenSSL 1.1.1+, currently the 'ssl' module is compiled with 'LibreSSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
Peer is listening on 192.168.0.101:6881
Enter command: 
```

Hình 3: Khởi tạo peer thành công



```
app — Python peer.py — 80x17
download /User
Enter command: create /Users/ngtrunghieu/Downloads/Computer_Network_Assignment_1/video.mp4 /Users/ngtrunghieu/Downloads/Computer_Network_Assignment_1/app/metainfo http://192.168.0.101:6880/announce
Creating torrent file for video.mp4...
Tracker URL: http://192.168.0.101:6880/announce
Length: 12900465
Info Hash: 982a99d63875b95d5995978791f43edf2e47c953
Torrent file is created for video.mp4

Enter command: upload /Users/ngtrunghieu/Downloads/Computer_Network_Assignment_1/app/metainfo/video.mp4.torrent http://192.168.0.101:6880/announce
Upload to tracker successfully.

Enter command: download /Users/ngtrunghieu/Downloads/Computer_Network_Assignment_1/app/video.mp4.torrent /Users/ngtrunghieu/Downloads/Computer_Network_Assignment_1/app
```

Hình 4: upload và download peer

6 Hướng dẫn sử dụng

6.1 Ngôn ngữ và thiết

Nhóm sử dụng ngôn ngữ lập trình Python, phiên bản python3, sử dụng máy ảo myenv để hiện thực chương trình.

6.2 Khởi tạo môi trường

Trên máy ảo, trong folder chứa source code, chọn Open in terminal. Hoặc có thể mở terminal và cd tới folder chứa source code. Với mỗi cửa sổ command để khởi tạo Tracker và mỗi Peer, chạy các dòng lệnh theo thứ tự: 'python3 -m venv myenv' 'source <path to /myenv/bin/activate>'

6.3 Khởi tạo Tracker và Peer

Lưu ý: Khởi tạo Tracker trước Peer Để khởi tạo Tracker, chạy 'python3 tracker.py' để khởi tạo Tracker Để khởi tạo từng Peer, chạy 'python3 peer.py' để khởi tạo Peer

6.4 Cách chạy:

- Đầu tiên chạy file `tracker.py`, sau đó tới các file `Peer.py`.
- Với Peer upload nhập lệnh "create <đường dẫn đến file muốn thành file torrent> <đường dẫn đến thư mục lưu file torrent đó> <địa chỉ http của tracker>" để tạo file torrent.
- Peer upload tiếp tục nhập lệnh "upload <đường dẫn đến file muốn thành file torrent> <địa chỉ http của tracker>" để kết nối đến địa chỉ Tracker, cho phép các Peer khác tải về từ nó.
- Gửi file torrent cho Peer Download.
- Peer download nhập "download <đường dẫn đến file torrent> <đường dẫn đến thư mục muốn lưu file tải về>" để liên hệ với tracker và nhận danh sách IP và port của các peer giữ file, tiến hành handshake với các peer đó và tải file về.
- Peer upload nhập "stop" sẽ hiện ra số byte đã tải.

6.5 Chú ý

- Peer upload không được offline khi có một Peer khác muốn download file.
- Nếu máy sử dụng hệ điều hành Windows thì sửa lại `ifconfig` trong hàm `get_local_ip()` thành `ipconfig`.