

**TRƯỜNG ĐẠI HỌC BÁCH KHOA TP HCM**

-----🙞🙜🕮🙞🙜-----



**ASSIGNMENT 1**

**Computer Network**

**Nguyễn Duy Thanh - 1813967**

**Nguyễn Quang Tùng - 1814701**

**Nguyễn Ngọc Tân – 1813942**

**GVHD: Hoàng Nguyễn Minh Đức**

Contents

[**1.Requirement analysis** 3](#_Toc56373098)

[**2. Function Description** 5](#_Toc56373099)

[**3. Class diagram** 11](#_Toc56373100)

[**4. Achiedved results:** 12](#_Toc56373101)

[**5 .User manual** 17](#_Toc56373102)

# **1.Requirement analysis**

**SEVER**

* Cho phép máy khách(client) kết nối, giao tiếp và truyền dữ liệu video.

**CLIENT**

* Kết nối, giao tiếp với máy chủ và nhận dữ liệu video với tên video đã biết trước.

**def sendRtspRequest(requestCode)**

If requestCode == SETUP && state == INIT

* Client thiết lập kết nối TCP với máy chủ, thường là trên cổng TCP 554 còn được biết là cổng cho RTSP.
* Máy khách sau đó sẽ bắt đầu phát hành một loạt lệnh tiêu đề RTSP có định dạng tương tự như HTTP.

> Mô tả chi tiết các yêu cầu phiên bản đối với máy chủ.

> Ví dụ: Phiên bản RTSP mà nó hỗ trợ.

> Phương tiện được sử dụng cho luồng dữ liệu.

> Mọi thông tin cổng UDP hoặc TCP được liên kết.

> Những thông tin này được chuyển bẳng cách sử dụng tiêu đề DESCRIBE và SETUP và được bổ sung trên phản hồi của máy chủ với phiên bản ID mà máy khách và bất kỳ thiết bị proxy tạm thời nào có thể sử dụng để xác định luồng trong các trao đổi trong tương lai.

Sau khi các thông số thương lượng đã hoàn thành

> Máy khách đưa ra lệnh PLAY để hướng dẫn máy chủ bắt đầu phân phối luồng dữ liệu RTP.

Sau khi máy khách quyết định đóng luồn, lệnh TEARDOWN được đưa ra cùng với phiên bản ID hướng dẫn máy chủ ngừng phân phối RTP được liên kết với ID đó.

**SETUP**

**\*** Gửi yêu cầu SETUP đến máy chủ. Sinh viên cần chèn “Transport header” trong đó sinh viên chỉ định cổng ổ cắm dữ liệu RTP mà sinh viên vừa tạo.

\* Đọc phản hồi của máy chủ và phân tích cú pháp tiêu đề Session (từ phản hồi) để nhận phiên bản ID.

\* Tạo một ổ cắm datagram để nhận dữ liệu RTP và đặt thời gian chời trên ổ cắm là 0.5 giây.

**PLAY**

\* Gửi yêu cầu PLAY. Sinh viên phải cần chèn “Session header” và sử dụng ID phiên bản được trả về trong phản hồi SETUP. Sinh viên không được đặt “Transport header” trong yêu cầu này

\*Đọc phản hồi của máy chủ.

**PAUSE**

\*Gửi yêu cầu PAUSE. Sinh viên phải chèn “Session header” và sử dụng ID phiên bản được trả về trong phản hồi SETUP. Sinh viên không được đặt “Transport header” trong yêu cầu này

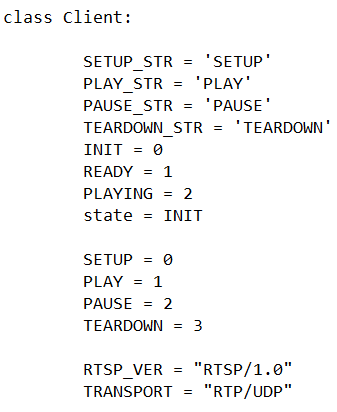
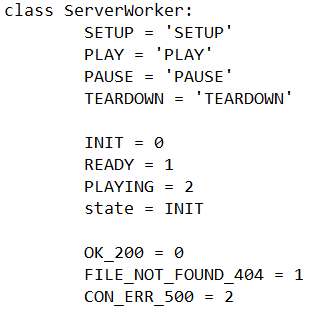
\* Đọc phản hồi của máy chủ.

**TEARDOWN**

\*Gửi yêu cầu TEARDOWN. Sinh viên phải chèn “Session header” và sử dụng ID phiên bản được trả về trong phản hồi SETUP. Sinh viên không được đặt “Transport header” trong yêu cầu này.

\* Đọc phản hồi của máy chủ. Lưu ý: Sinh viên phải chèn tiêu đề CSeq trong mọi yêu cầu sinh viên gửi. Giá trị của tiêu đề CSeq là một số bắt đầu từ 1 và được tăng lên một cho mỗi yêu cầu sinh viên gửi

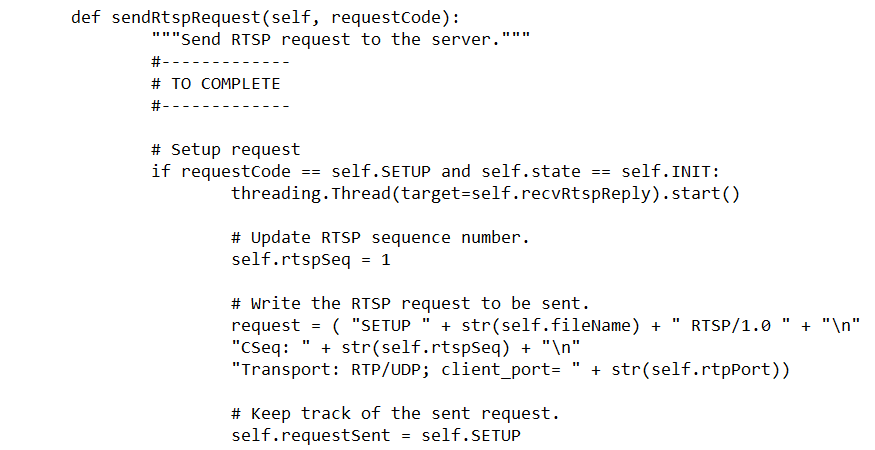
# **2. Function Description**



Những gì sẽ được gửi từ máy khách đến máy chủ thông qua giao thức RTSP là các lệnh như SETUP, PLAY, PAUSE, TEARDOWN. Các lệnh này sẽ cho phía máy chủ biết hành động tiếp theo mà nó sẽ hoàn thành.

Những gì sẽ được trả lời từ máy chủ đến máy khách thông qua giao thức RTSP là các tham số như: OK\_200, FILE\_NOT\_FOUND\_404, CON\_ERR để thông báo cho client nếu server nhận lệnh chính xác.

Sau khi máy khách nhận được câu trả lời của máy chủ, nó sẽ thay đổi trạng thái tương ứng thành READY, nếu lệnh SETUP được gửi từ máy khách đến máy chủ.



Gói “SETUP” RTSP sẽ bao gồm:

> Lệnh SETUP

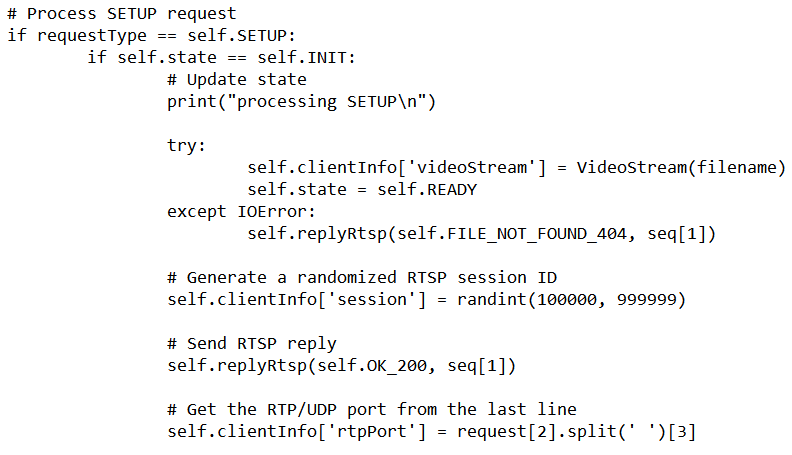
> Tên tệp video sẽ phát

> RTSP Pakcet Sequence Number bắt đầu từ 1

> Loại giao thức: RTSP/1.0 RTP

> Giao thức truyền: UDP

> Cổng RTP để truyền dòng video

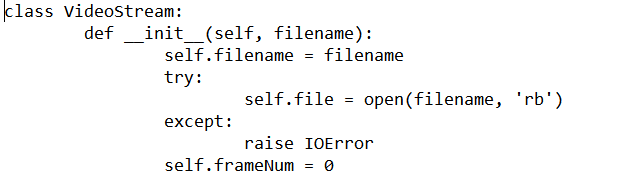


Khi phía Server nhận được lệnh “SETUP” nó sẽ

> Gán cho khách hàng một số Session cụ thể ngẫu nhiên

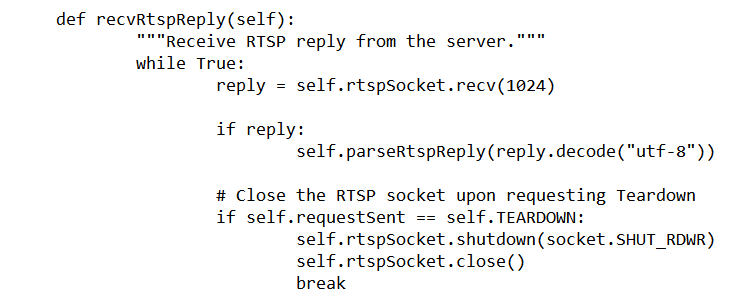
> Nếu có gì đó sai với lệnh này hoặc trạng thái của máy chủ, nó sẽ trả lười gói Lỗi trở lại máy khách

> Nếu lệnh xử lý chính xác, nó sẽ trả lời lại OK\_200 cho máy khách và đặt STATE của nó thành READY



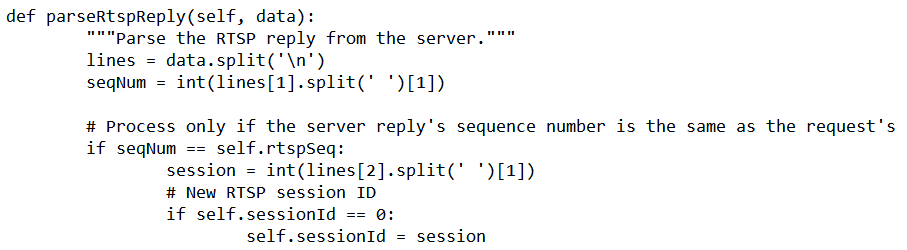
Máy chủ sẽ mở tệp video được chỉ định trong gói SETUP Packet và khởi tạo frame của video thành 0.

Phía Client sẽ lặp lại để nhận RTSP Reply của Server

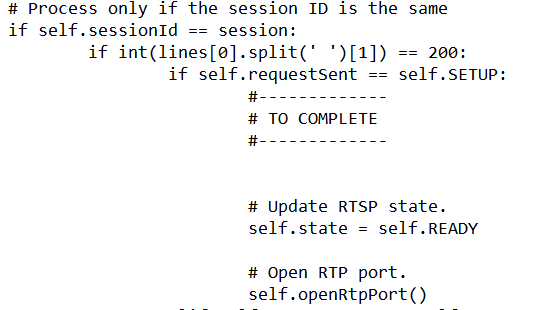


Sau đó phân tích cú pháp gói RTSP Reply

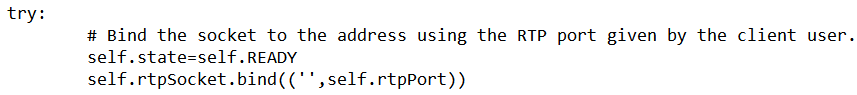
Lấy Session Number



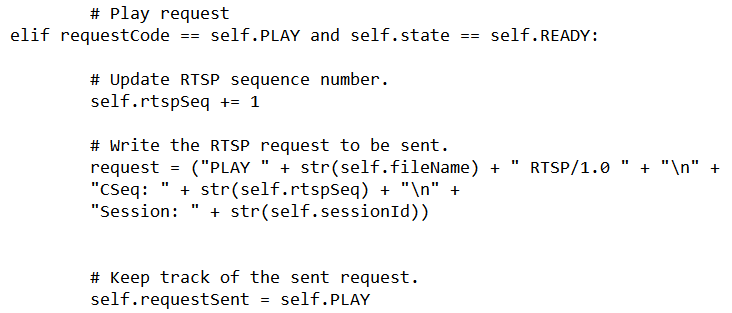
Và nếu Reply Packet phản hồi cho lệnh SETUP. Máy chủ sẽ chuyển trạng thái sang READY



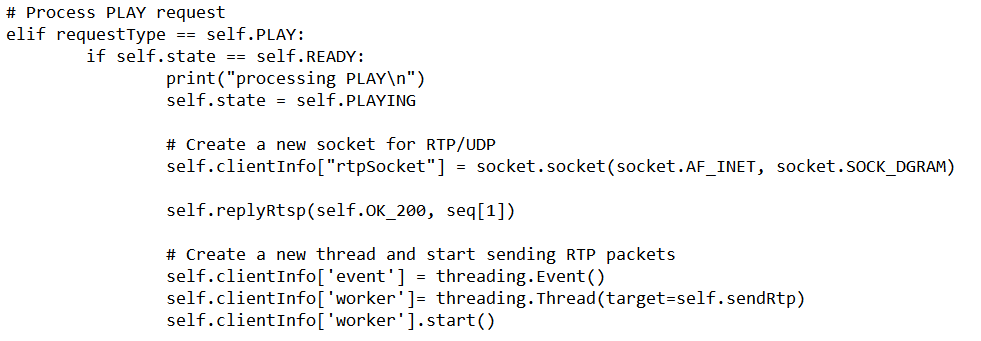
Sau đó openRTPPort() nhận luồng video đến



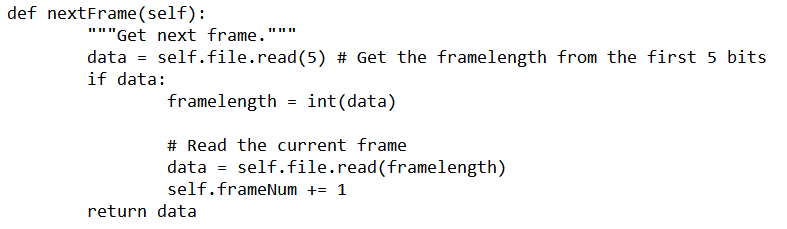
Sau đó nếu lệnh PLAY RTSP được gửi đến server:



Máy chủ sẽ tạo một Socket để truyền RTP qua UDP và bắt đầu một bước để gửi gói luồng video



VideoStream.py sẽ giúp cắt tệp video thành từng khung riêng biệt và đưa từng khung vào gói dữ liệu RTP



Mỗi gói dữ liệu cũng sẽ được mã hóa với một tiêu đề, tiêu đề sẽ bao gồm

> RTP-version

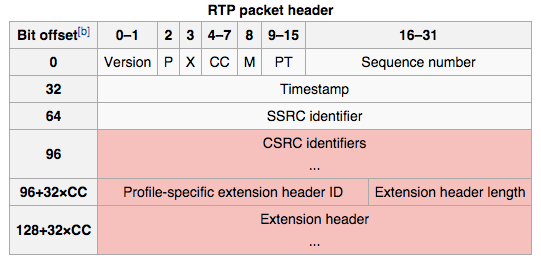
> Padding extension

> Contributing source

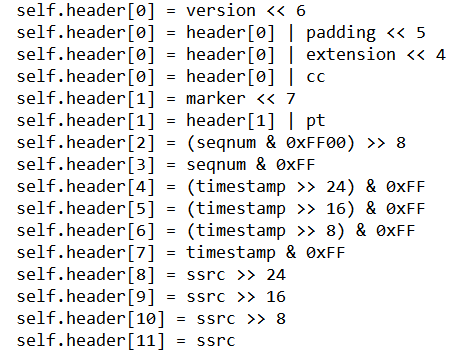
> Marker Type Field Sequence Number

> Timestamp

> SSRC



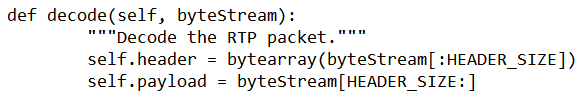
Chúng đã được chèn vào gói RTP thông qua các thao tác bitwise



Cuối cùng, RTP Packet sẽ bao gồm một tiêu đề và một khung video được gửi đến RTP Port ở phía client.



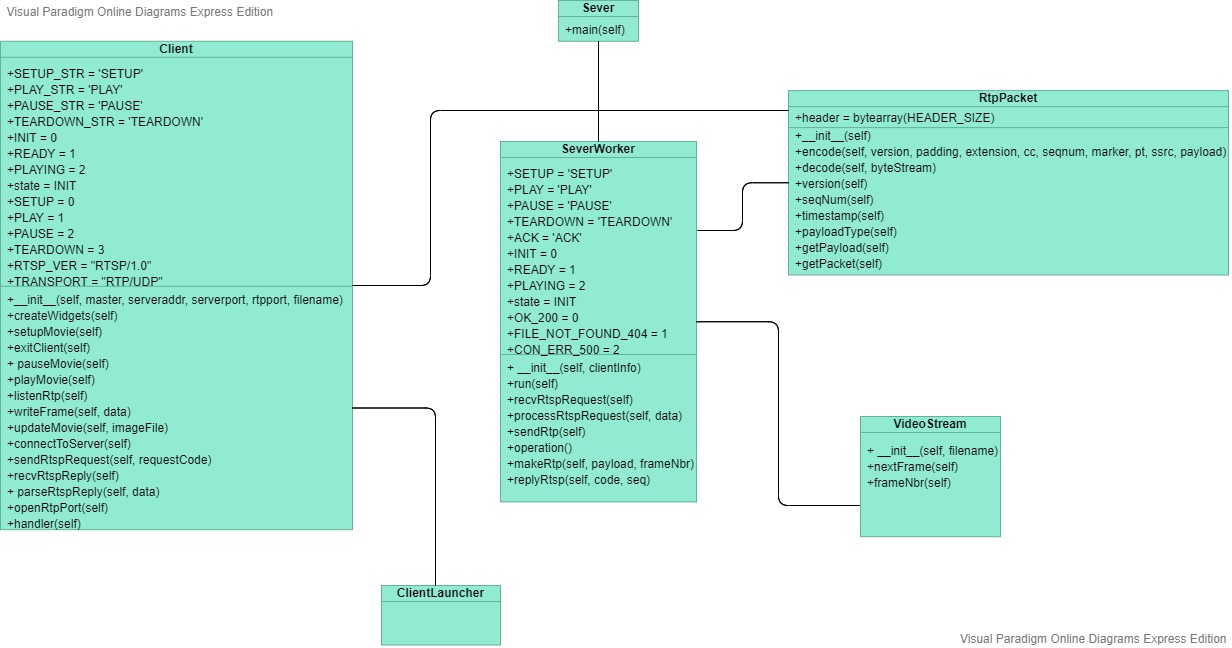
Sau đó, Client giải mã RTP Packet để lấy tiêu đề và frame, tổ chức lại cái fram và hiển thị trên giao diện người dùng



Nếu lệnh PAUSE được gửi từ máy khách đến máy chủ, nó sẽ ngăn máy chủ gửi các khung hình video đến máy khách

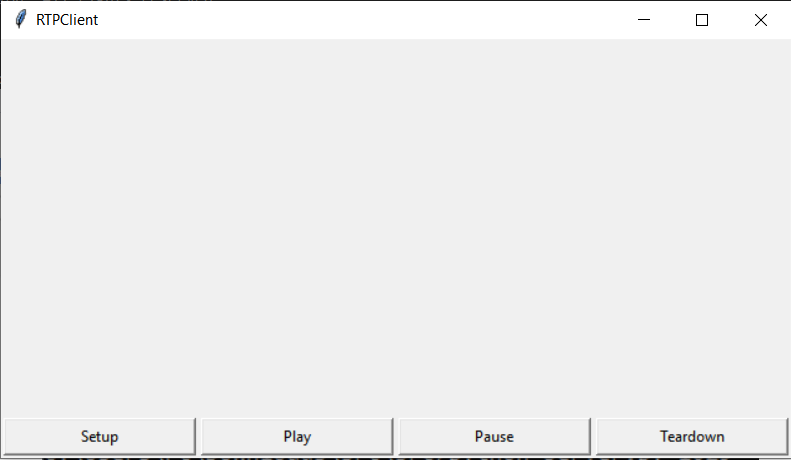
Nếu một lệnh TEARDOWN được gửi từ máy khách đến máy chủ, nó cũng sẽ ngăn máy chủ gửi các khung hình video đến máy khách và đóng cả thiết bị đầu cuối của máy khách

# **3. Class diagram**

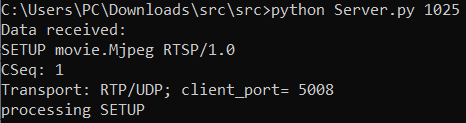


# **4. Achiedved results:**

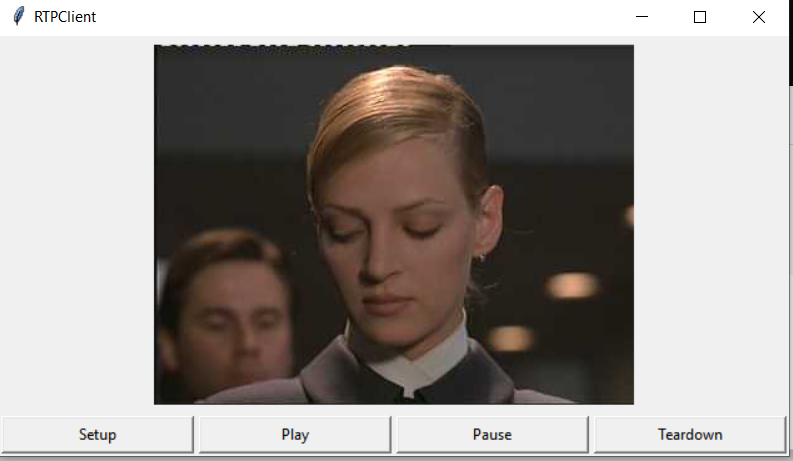
Sau khi chạy dòng lệnh “python ClientLauncher.py 127.0.0.1 1025 5008 movie.Mjpeg”, ta thu được giao diện sau:



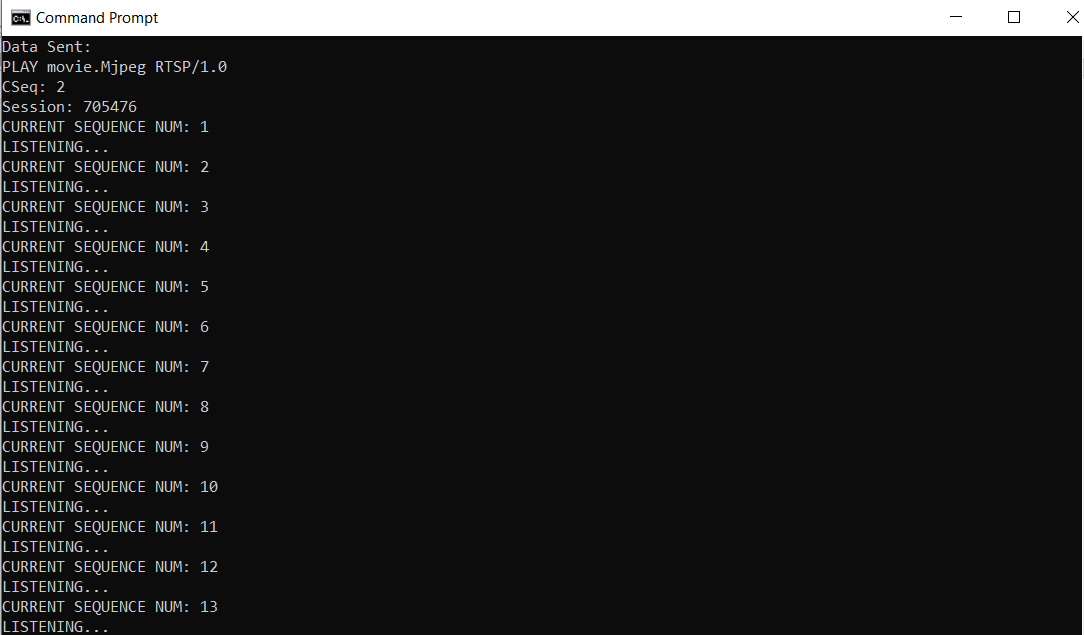
Ta kích hoạt nút Setup:



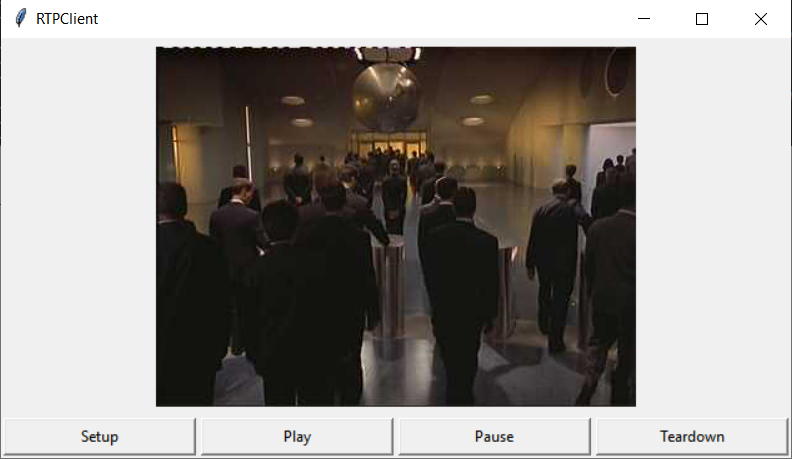
Tiếp đó ta kích nút PLAY:

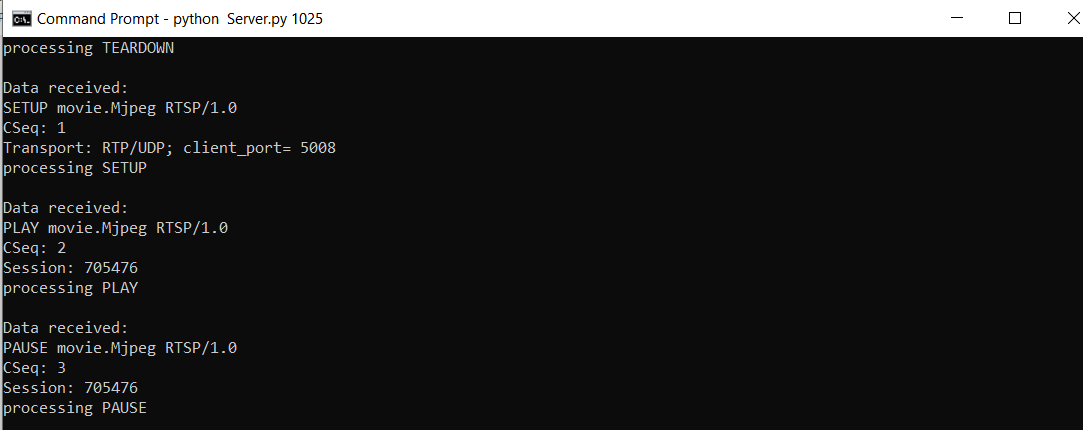






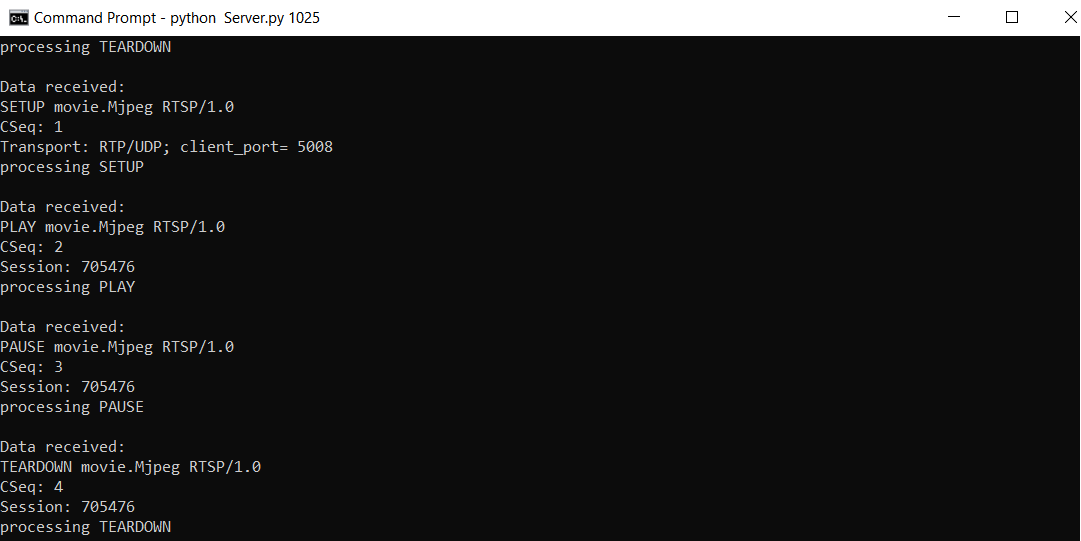
Sau đó ta bấm nút PAUSE:

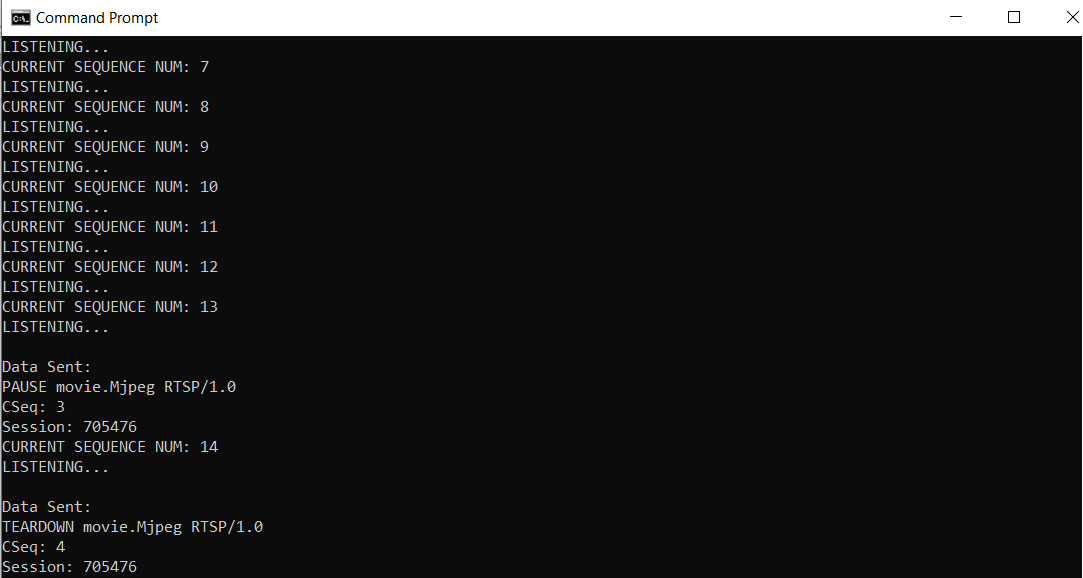






Cuối cùng ta nhấn nút TEARDOWN để ngắt kết nối:



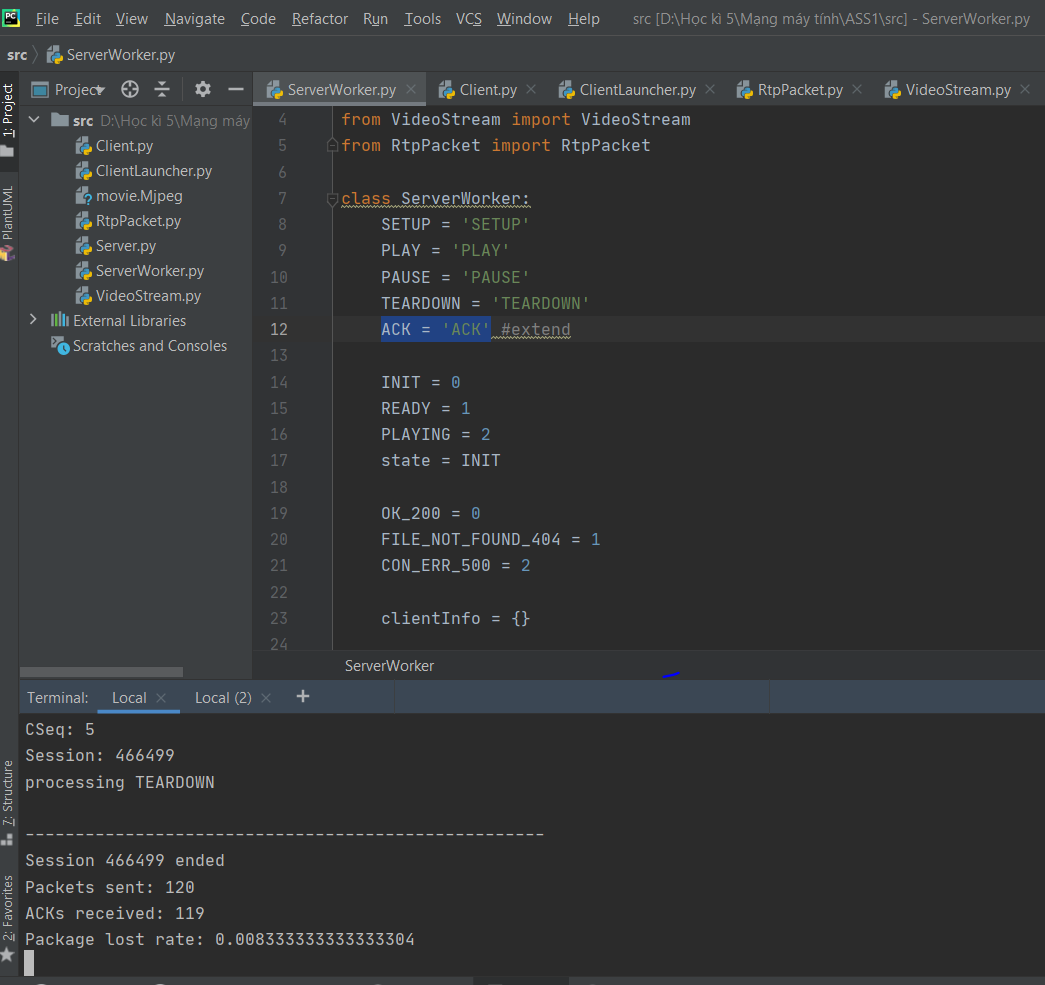


Chương trình được thiết kế để khi chưa nhấn nút SETUP hoặc PAUSE thì người sử dụng không thể PLAY video

Và chương trình sẽ báo lỗi nếu TEARDOWN chương trình khi chưa nhấn PLAY

Extend:

Nhóm đã tính được tỉ lệ mất gói (lost rate):



Khi ta làm ngắt quãng việc truyền dữ liệu có khả năng gây mất gói. Sau khi người dừng nhất TEARDOWN thì chương trình sẽ thông báo số gói gửi và nhận được cùng với tỉ lệ mất gói.

# **5 .User manual**

Chương trình gồm 6 files:

> Client.py

> ClientLauncher.py

> Server.py

> ServerWorker.py

> RtpPacket.py

Để thực thi chương trình ta làm các bước sau

1. Chạy chương trình Server.py trên Terminal để bắt đầu server:

E.g: python Server.py server\_port

Server\_port là cổng mà máy chủ của bạn nghe các kết nối RTSP đến.

> Trong project này chúng ta sẽ cho giá trị > 1024

> Ở phần demo, nhóm sẽ cho port có giá trị là 1025

> Standard RTSP port là 554



1. Chạy chương trình ClientLauncher.py trên Client Terminal để bắt đầu client:

E.g: python ClientLauncher.py server\_host server\_port PRT\_port video\_file

Server\_host là IP của địa chỉ máy cục bộ (ta có thể sử dụng “127.0.0.1”)

RTP\_port là cổng mà RTP packet được nhận ( ở đây là ta sử dụng cổng “5008”)

Video\_file là tên của video file mà ta muốn chạy ( ở đây ta sử dụng “video.Mjpeg”)

